Public key cryptosystems based on diophantine equations

奥村, 伸也

https://doi.org/10.15017/1500515

Public key cryptosystems based on diophantine equations

Shinya Okumura

A dissertation submitted to
Kyushu University
for the degree of
Doctor of Philosophy (Mathematics)
February 2015

Introduction

The purpose of this thesis is two-fold: one is to study the semi-regularity of the equation systems arising from the section finding problem on algebraic surfaces and the other is to propose a new public key cryptosystem based on the difficulty of finding integral or rational solutions to diophantine equations.

After Diffie and Hellman proposed the concept of public key cryptography ([16]), the theory of cryptography has been developed rapidly and has contributed to the security of networks. This cryptosystem is based on computationally hard problems, like factorization of large integers and computation of discrete logarithm in large finite groups. The most famous public key cryptosystems are the RSA cryptosystem ([43]) and elliptic curve cryptosystem ([31], [38]). Although these cryptosystems are studied by many researchers, efficient attacks have not been found in general. However, Shor showed that factorization of integers and computation of discrete logarithm can be done efficiently by using quantum computers ([45]). So it is important to find new computationally hard problems which are intractable even with quantum computers and can be used to construct cryptosystems. We expect that the diophantine problem is one of such problems. This problem is to find integral or rational solutions of a given multivariate polynomial with integer coefficients. Despite many researchers' endeavor (see e.g. [29]), this problem is usually a very difficult problem. Moreover Matijasevič showed that there is no general method which determines the solvability of an arbitrary diophantine equation ([15]). On the other hand, for any integers $a_1, a_2, \cdots, a_n$, it is easy to find a polynomial $X(x_1, \cdots, x_n) \in \mathbb{Z}[x_1, \cdots, x_n]$ with $X(a_1, a_2, \cdots, a_n) = 0$ (see §4.6.1). So we can expect that diophantine equations can be used to construct new public key cryptosystems. Indeed some cryptosystems based on this problem have already been proposed ([50], [30], [35]). But the one-wayness of the cryptosystem proposed in [35] was broken ([14]). On the other hand, cryptosystems in [50] and [30] are interesting in theory, but these cryptosystems can be used only a few times with the same key ([30], Proposition 2).

We can also consider the diophantine problem over global function fields.

This problem is also difficult and it is proved that there is no general method which determines the solvability of an arbitrary diophantine equation ([41]). The Algebraic Surface Cryptosystem (ASC) proposed in [1] is based on the difficulty of the section finding problem (SFP), which can be viewed as a diophantine problem with 2 variables over $\mathbb{F}_p[t]$ (or $\mathbb{F}_p(t)$). In Chapter 1, we give a brief rivew of ASC and some attacks against its one-wayness.

In Chapter 2, we show that solving SFP for $X \in \mathbb{F}_p[x, y, t]$ can be reduced to solving a certain multivariate equation system of high degree depending on X and its section. We call this system the *section equation system* for X. In order to make SFP infeasible for the security of ASC, we need to estimate the complexity of solving the section equation system. One of the most famous methods to solve the multivariate equation systems is the Gröbner basis method. The semi-regularity is important to estimating the complexity of a Gröbner basis computation. In Chapter 2, we describe the importance of the semi-regularity and derive the criterion for semi-regularity to judge whether the section equation system for a given X is semi-regular or not. We show some experimental results on the semi-regularity of the section equation by using this criterion.

Chapter 3 used to be a survey on diophantine problems. In particular, there were recalled some known results on diophantine approximation and integral/rational points on varieties over number fields. Recalled also was Baker's theorem on lower bounds for linear forms in logarithms, which provides effective upper bounds for integral solutions to certain types of diophantine equations with two variables. However, as the contents are well-known and can be found easily in the existing literature, we have decided not to include them in the final version of this thesis.

In number theory, there are many analogous problems between number fields and function fields. There are many cases where problems over function fields have been solved while the corresponding problems over number fields have hardly been solved. For example, there is an algorithm to factorize elements of $\mathbb{F}_p[t]$ in probabilistic polynomial time ([7], [12]), while the best known algorithm (the general number field sieve) for factorization in $\mathbb{Z}$ takes subexponential time $O(e^{(c+o(1))(\log N)^{\frac{1}{3}}(\log\log N)^{\frac{2}{3}}})$, where $c = (\frac{9}{64})^{\frac{1}{3}}$ and N is an integer which we want to factorize ([34]). The Riemann Hypothesis for function fields was proved by André Weil ([49]), while the Riemann Hypothesis for $\mathbb{Z}$ still seems far beyond our reach. The *abc* conjecture for function fields (the Mason-Stothers Theorem) was proved in [47] and [37], while a proof of the *abc* conjecture for $\mathbb{Z}$ was announced just a few years ago by Shinichi Mochizuki ([39]).

In Chapter 4, we propose four public key cryptosystems based on the dif-

ficulty of finding integral or rational solutions to diophantine equations over integers. Our method is similar to ASC. Our first cryptosystem is essentially the same as ASC but we use diophantine equations with $n \geq 2$ variables. In this cryptosystem, a secret key is $\underline{a} := (a_1, \ldots, a_n) \in \mathbb{Z}^n$ and a public key is a polynomial $X \in \mathbb{Z}[\underline{x}] := \mathbb{Z}[x_1, \ldots, x_n]$ such that $X(\underline{a}) := X(a_1, \ldots, a_n) = 0$. In the encryption process, the sender constructs the following polynomials:

$$F_i = m + s_i f + r_i X \ (i = 1, 2),$$

where $m \in \mathbb{Z}[\underline{x}]$ is the plaintext polynomial and s_i, f and r_i are polynomials in $\mathbb{Z}[\underline{x}]$ satisfying some conditions. Then the sender sends (F_1, F_2). In decryption process, we use the secret key and some arithmetics to get $m(\underline{a})$ and recover m from it. This is our basic cryptosystem.

The one-wayness of ASC can be broken by the ideal decomposition attack ([23]). In this attack, one constructs an ideal which contains a few polynomials with the same form as the plaintext polynomial except for the plaintext polynomial by using a decomposition of a certain ideal (essentially, a resultant or a Gröbner basis is used). The similar method is applicable against our cryptosystem. So, we test the effectiveness of the ideal decomposition attack for our cryptosystem by experiments. According to our experiments, this attack works well if $n = 2$. On the other hand if $n \geq 3$, then this attack succeeds in constructing the proper ideal containing a plaintext polynomial in low probability.

Our first idea to avoid the ideal decomposition attack is to translate the public key into another polynomial that we can eliminate by using a secret key and some simple arithmetics. Our second and third cryptosystems are based on this idea. More precisely, the sender chooses an integer ℓ and constructs $X'(\underline{x}, t)$ so that $X = X'(\underline{x}, \ell)$. In the decryption process, we get ℓ by using the secret key and decrypt the ciphertext. However, we cannot say that these cryptosystems are safe because one can also get ℓ without finding a secret key.

Our second idea is to twist the plaintext polynomial by using some modular arithmetic and use a polynomial of degree increasing type (see Definition 4.5.1) as a public key. (We think that using this type of special polynomials does not affect the security of our cryptosystem; see Theorem 4.7.1). If one can construct the ideal containing the twisted plaintext polynomial by a method similar to the ideal decomposition attack, then this ideal contains many polynomials having the same form as the twisted plaintext polynomial and those coefficients are as large as those of the twisted plaintext polynomial. Our fourth cryptosystem, which is the main result of Chapter 4, is based on these ideas. In this cryptosystem, we use three polynomials as

ciphertext polynomials and the GCD computation to avoid using the factorization of integers. There are some known attacks against the one-wayness of ASC other than the ideal decomposition attack. We also analyze the effectiveness of attacks similar to them (§4.7). Although we have not been able to give a security proof, our analysis shows that our fourth cryptosystem has resistance against the above possible attacks including the ideal decomposition attack. We have not been able to propose an effective attack against it. Finally, we estimate the size of keys of our fourth cryptosystem for providing 128-bit security. Our estimation shows that if we use integers d, e and a diophantine equation with n variables and total degree w as the public key, then the size of the secret key is at most $(\lceil \frac{128}{n-1} \rceil + 1)n + \lceil \log_2 d - \log_2 \varphi(d) \rceil$ bits and the size of the public key is at most $(\lceil \frac{128}{n-1} \rceil + 76 + \lceil \log_2 d - \log_2 \varphi(d) \rceil)w + 65 + \lceil \log_2 e \rceil$ bits. We also estimate the size of ciphertexts to be at most $\frac{3}{2}(w^2 + w)(129 + 130w + \lceil \log_2 w \rceil) + 129 + 65(w - 1)$ bits. In §4.9 we give some examples of the size of keys and ciphertexts together with the time it took to encrypt and decrypt.

Acknowledgments

I am grateful to my supervisor Yuichiro Taguchi for comments, corrections, and suggestions on this research. I am also grateful to Koichiro Akiyama, Noriko Hirata-Kohno, Attila Pethő, Takakazu Sato and Tsuyoshi Takagi for useful comments, suggestions and discussions.

Contents

Chapter 1

The Algebraic Surface Cryptosystem

In this chapter, we give a review of the Algebraic Surface Cryptosystem (ASC) and known attacks against the one-wayness of ASC. The ASC was proposed by Akiyama, Goto and Miyake ([1]). This cryptosystem is based on computationally hard problem in algebraic geometry. In [1], it is called the finding section problem (SFP) on algebraic surface fibered on an affine line. In general, there is no algorithm to solve it in polynomial time even if a quantum computer is built. Solving SFP is reduced to solving a certain multivariate polynomial system of high degree over a finite field (see Chapter 2). It is known to be NP-complete ([25]). We can also view SFP as a diophantine problem over global function fields and it is proved that there is no general method which determines the solvability of an arbitrary diophantine equation ([41]). So, ASC is expected to have resistance against quantum computers (such cryptosystems are called post-quantum public key cryptosystems). In [1], it is pointed out that one of the advantages of ASC is the small key size. It is one of the shortest keys of the known post-quantum public key cryptosystems when ASC was announced (see §1.3.6).

1.1 Notation

Let p be a prime number and $\mathbb{F}_p$ a finite field with p elements. For a polynomial $g = \sum_{i,j} g_{ij}(t)x^i y^j = \sum_{i,j,k} g_{ijk}x^i y^j t^k \in \mathbb{F}_p[x, y, t]$ we define

$$\begin{aligned} \Lambda_g^{(p)} &:= \{(i, j) \in \mathbb{Z}^2 \mid g_{ij}(t) \neq 0\}, \\ \Gamma_g^{(p)} &:= \{(i, j, k) \in \mathbb{Z}^3 \mid g_{ijk} \neq 0\}. \end{aligned}$$

For two subsets Λ_1, $\Lambda_2 \subset (\mathbb{Z}_{\geq 0})^2$ we define

$$\Lambda_1\Lambda_2 := \{(i_1 + i_2, j_1 + j_2) \mid (i_1, j_1) \in \Lambda_1, (i_2, j_2) \in \Lambda_2\}.$$

This means that if $\Lambda_i^{(p)} = \Lambda_{f_i}^{(p)}$ for some $f_i \in \mathbb{F}_p[x, y, t]$, then $\Lambda_1\Lambda_2 = \Lambda_{f_1 f_2}^{(p)}$. For each ideal $J = (f_1, \ldots, f_n) \subset \mathbb{F}_p[x, y, t]$, each polynomial $g \in \mathbb{F}_p[x, y, t]$ and each monomial ordering $<$, there are polynomials h, $r \in \mathbb{F}_p[x, y, t]$ such that $h \in J$, $g = h + r$ and that no monomial of r is in the ideal generated by the leading monomials of f_i for $i = 1, \ldots, n$. The r may depend on the choice of a system of generators of J, but is uniquely determined if we calculate it using a Gröbner basis of J. Then this unique r is called the normal form of g with respect to J and $<$, and we denote it by $NF_J(g)$.

1.2 The section finding problem on algebraic surfaces

Let p be a prime number. The ASC makes use of a section to a fibration of an algebraic surface to the afine line over $\mathbb{F}_p$. More precisely, let $X \in \mathbb{F}_p[x, y, t]$ be a polynomial which defines a surface S with a fibration $\sigma_S : S \to \mathbb{A}^1_{\mathbb{F}_p}$ over the affine t-line. Then a map $\tau : \mathbb{A}^1_{\mathbb{F}_p} \to S$ over $\mathbb{F}_p$ satisfying $\sigma_S \circ \tau = \mathrm{id}_{\mathbb{A}^1_{\mathbb{F}_p}}$ is called a section of σ_S. The section finding problem (SFP) on the surface S is to find a section of σ_S.

On the other hand, X also defines a curve over $\mathbb{F}_p(t)$. We denote it by C. There is a bijection between the set of sections of σ_S and the set of $\mathbb{F}_p(t)$-rational point on C. Thus, the SFP is restated as follows: the SFP on the surface S is to find $u_x(t)$, $u_y(t) \in \mathbb{F}_p[t]$ such that $X(u_x(t), u_y(t), t) = 0$. It implies that we can view the SFP as the diophantine problem over global function fields.

1.3 An algorithm of ASC

1.3.1 System parameters

The system parameters in ASC are as follows:

1. p : size of a finite field $\mathbb{F}_p$;
2. $\Lambda_1^{(p)}$: a finite subset of $(\mathbb{Z}_{\geq 0})^2$;
3. $D_1 := \{d_{ij}^{(1)} \mid (i, j) \in \Lambda_1^{(p)}\} \subset \mathbb{Z}_{\geq 0}$.

1.3.2 Key generation

1. Secret key
   Choose two random polynomials $u_x(t)$, $u_y(t) \in \mathbb{F}_p[t]$ of degree d.

2. Public key
   For $k = 2, 3$, choose finite subsets $\Lambda_k^{(p)} \subset (\mathbb{Z}_{\geq 0})^2$ and $D_k = \{d_{ij}^{(k)} \mid (i,j) \in \Lambda_k^{(p)}\} \subset \mathbb{Z}_{\geq 0}$ so that the following hold:

   (i) $\Lambda_2^{(p)} \subset \Lambda_1^{(p)} \Lambda_3^{(p)}$.

   (ii) For any polynomial $f_k = \sum_{(i,j)\in\Lambda_k^{(p)}} f_{ij}^{(k)}(t) x^i y^j \in \mathbb{F}_p[x,y,t]$ $(k = 1,2,3)$ with $\Lambda_{f_k}^{(p)} = \Lambda_k^{(p)}$ and $\deg f_{ij}^{(k)}(t) = d_{ij}^{(k)}$, we have

$$\begin{cases} \deg_x f_1 < \deg_x f_2 < \deg_x f_3, \\ \deg_y f_1 < \deg_y f_2 < \deg_y f_3, \\ \deg_t f_1 < \deg_t f_2 < \deg_t f_3, \\ (\deg_x f_2, \deg_y f_2, \deg_t f_2) \in \Gamma_{f_2}^{(p)}, \\ (\deg_x f_3, \deg_y f_3, \deg_t f_3) \in \Gamma_{f_3}^{(p)}. \end{cases} \tag{1.1}$$

   Construct an $X(x,y,t) = \sum_{(i,j)\in\Lambda_1^{(p)}} c_{ij}(t) x^i y^j \in \mathbb{F}_p[x,y,t]$ such that $X(u_x(t), u_y(t), t) = 0$, $\deg c_{ij}(t) = d_{ij}^{(1)}$ and $c_{ij}(t) \neq 0$ for $(i,j) \in \Lambda_1^{(p)}$. In §1.3.5 we give a method to construct such a polynomial. For $i = 2, 3$, make X, $\Lambda_i^{(p)}$ and D_i public.

1.3.3 Encryption

Assume that the sender wants to send a polynomial $m(x,y,t) = \sum_{(i,j)\in\Lambda_2^{(p)}} m_{ij}(t) x^i y^j \in \mathbb{F}_p[x,y,t]$ with $\deg m_{ij}(t) = d_{ij}^{(2)}$ for $(i,j) \in \Lambda_2^{(p)}$.

1. For $k = 1, 2$, choose random polynomials in $\mathbb{F}_p[x,y,t]$:

$$\begin{aligned} s_k &= \sum_{(i,j)\in\Lambda_1^{(p)}} s_{ij}^{(k)}(t) x^i y^j, \\ r_k &= \sum_{(i,j)\in\Lambda_3^{(p)}} r_{ij}^{(k)}(t) x^i y^j, \\ f &= \sum_{(i,j)\in\Lambda_3^{(p)}} f_{ij}(t) x^i y^j, \end{aligned}$$

such that $\deg s_{ij}^{(k)}(t) = d_{ij}^{(1)}$ and $\deg r_{ij}^{(k)}(t) = \deg f_{ij}(t) = d_{ij}^{(3)}$. Note that from (1.1), we have

$$\begin{cases} \deg_x X < \deg_x m < \deg_x f, \\ \deg_y X < \deg_y m < \deg_y f, \\ \deg_t X < \deg_t m < \deg_t f, \\ (\deg_x m, \deg_y m, \deg_t m) \in \Gamma_m^{(p)}, \\ (\deg_x f, \deg_y f, \deg_t f) \in \Gamma_f^{(p)}. \end{cases} \tag{1.2}$$

2. Put $F_i := m + s_i f + r_i X$ for $i = 1, 2$, and send (F_1, F_2).

1.3.4 Decryption

1. For $i = 1, 2$, compute

$$\begin{aligned} h_i(t) &:= F_i(u_x(t), u_y(t), t) \\ &= m(u_x(t), u_y(t), t) \\ &\quad + s_i(u_x(t), u_y(t), t) f(u_x(t), u_y(t), t). \end{aligned}$$

2. Factorize $h_1 - h_2$ and find a factor h_3 of it whose degree is equal to $\deg f(u_x(t), u_y(t), t)$. Note that from (1.2), we have

$$\deg f(u_x(t), u_y(t), t) = \deg h_3 > \deg m(u_x(t), u_y(t), t).$$

3. Compute $h_4 := h_1 \pmod{h_3}$. Note that if h_3 divides $s_1(u_x(t), u_y(t), t) f(u_x(t), u_y(t), t)$, then $h_4 = m(u_x(t), u_y(t), t)$.

4. Extract $m(x, y, t)$ from h_4 by solving the following linear equation

$$h_4 = \sum_{(i,j,k) \in \Gamma_m^{(p)}} m_{ijk} u_x^i u_y^j t^k,$$

in variables m_{ijk} for $(i, j, k) \in \Gamma_m^{(p)}$, and put

$$m'(x, y, t) := \sum_{(i,j,k) \in \Gamma_m^{(p)}} m_{ijk} x^i y^j t^k.$$

5. We can verify whether $m' = m$ or not by a MAC (message authentication code) of m. If the verification fails, then go back to step 2 and choose another factor of $h_1 - h_2$.

1.3.5 Construction of $X(x, y, t)$

We describe a method to construct a polynomial $X(x, y, t) \in \mathbb{F}_p[x, y, t]$ such that $X(u_x(t), u_y(t), t) = 0$ for given polynomials $u_x(t)$, $u_y(t) \in \mathbb{F}_p[t]$.

1. Choose a finite subset $(0, 0) \in \Lambda^{(p)} \subset (\mathbb{Z}_{\geq 0})^2$ and $D := \{(d_{ij} \mid (i, j) \in \Lambda^{(p)}\} \subset \mathbb{Z}_{\geq 0}$.

2. Choose random non-zero polynomials $c_{ij}(t)$ of degree d_{ij} for $(i, j) \in \Lambda^{(p)} \setminus \{(0, 0)\}$.

3. Compute $c_{00}(t) := -\sum_{(i,j)\in\Lambda^{(p)}\setminus\{(0,0)\}} c_{ij}(t)u_x^i u_y^j$.

4. Define
$$X := \sum_{(i,j)\in\Lambda^{(p)}} c_{ij}(t)x^i y^j.$$

1.3.6 Recommended parameters

The designers of ASC proposed the following parameters:

- $p = 2$;
- d is at least 50;
- $\deg_{x,y} X := \max\{i + j \mid (i, j) \in \Lambda_X^{(p)}\}$ is at least 5;
- $\#\Lambda_X^{(p)}$ is at least 3.

For these parameters the size of public key is about 500 bits (cf. [1], §6).

1.3.7 Known attacks

We describe four possible attacks against ASC. For more details, see [1], §5 and [23].

Reduction to solving a multivariate equation system

Let

$$\begin{aligned} f'(x,y,t) &= \sum_{(i,j,k)\in\Gamma_f^{(p)}} f'_{ijk}x^iy^jt^k, \\ s'(x,y,t) &= \sum_{(i,j,k)\in\Gamma_{s_1}^{(p)}} s'_{ijk}x^iy^jt^k, \\ r'(x,y,t) &= \sum_{(i,j,k)\in\Gamma_{r_1}^{(p)}} r'_{ijk}x^iy^jt^k, \\ m'(x,y,t) &= \sum_{(i,j,k)\in\Gamma_m^{(p)}} m'_{ijk}x^iy^jt^k, \end{aligned} \tag{1.3}$$

where f'_{ijk}, s'_{ijk}, r'_{ijk} and m'_{ijk} are variables. If one can get f by solving the following quadratic equation system

$$F_1 - F_2 = (s_1 - s_2)f + (r_1 - r_2)X = s'f' + r'X,$$

then one may get m by solving $NF_I(m') = 0$, where $I = (f, X) \subset \mathbb{F}_p[x, y, t]$. In [1], it is pointed out that if $\#\Gamma_f^{(p)} > 50$ and $\#\Gamma_{s_1}^{(p)} > 50$, then finding solutions of this system becomes computationally intractable, even if the r'_{ijk}'s are eliminated by substituting rational points of X over a finite extension of $\mathbb{F}_p$.

Reduction by X

Since X is made public, one can try to divide $F_1 - F_2$ by X to find f in the remainder. But f does not appear in the remainder because of (1.2).

Rational point attack

Let $F(x, y, t) := F_1 - F_2$. Let $f'(x, y, t)$ and $s'(x, y, t)$ be as above. Let $g(x, y, t) := s'(x, y, t)f'(x, y, t)$. We write

$$g(x,y,t) = \sum_{(i,j,k)\in\Gamma_g^{(p)}} g_{ijk}x^iy^jt^k,$$

where g_{ijk} are polynomials in the coefficients of s' and f' for $(i, j, k) \in \Gamma_g^{(p)}$. For a large positive integer L and $\ell = 1, \ldots, L$, if one can find rational points

(x_ℓ, y_ℓ, t_ℓ) on $X(x, y, t) = 0$ over a certain extension field of $\mathbb{F}_p$, then one may be able to get $(s_1 - s_2)f$ by solving the following linear equation system

$$g(x_\ell, y_\ell, t_\ell) = F(x_\ell, y_\ell, t_\ell) \ (\ell = 1, \ldots, L). \tag{1.4}$$

Then one can find f by factorization. Then one may get m by solving $NF_{(f,X,F_1)}(m') = 0$. However, one cannot determine f and m uniquely. If $g_0(x, y, t) \in \mathbb{F}_p[x, y, t]$ satisfies (1.4), then $g_0 + rX$ also satisfies (1.4) and has the same form as g for any polynomial $r(x, y, t) \in \mathbb{F}_p[x, y, t]$ having the same form as f. In [1], it is pointed out that if $p^{\#\Gamma_r^{(p)}} = p^{\Gamma_f^{(p)}} > 2^{100}$, then we may avoid this attack.

Ideal decomposition attack

As mentioned above, we can design ASC to avoid the above three attacks. However, in [23] Faugére and Spaenlehauer proposed a new attack called the ideal decomposition attack and claimed that this attack can fully break ASC. The idea of this attack is to reconstruct the ideal $I := (m, f, X)$ in $\mathbb{F}_p[x, y, t]$ or the ideal $J := (m + z, f, X)$ in $\mathbb{F}_p(t)[x, y, z]$ or $(\mathbb{F}_p[t]/(P(t)))[x, y, z]$ from the data (F_1, F_2, X) by using the ideal decomposition $(F_1 - F_2, X) = ((s_1 - s_2)f, X) = I_1 \bigcap I_2$ for some ideals $(f, X) \subset I_1$ and I_2. Then the following equality holds:

$$(F_1 + z, F_2 + z, X) + I_1 = (m + z, f, X).$$

(Note that essentially, a resultant was used to reconstruct J in [23]). Let m' be as in (1.3). Then one can get m by solving $NF_I(m') = 0$ or $NF_J(m' + z) = 0$, where z is a new variable and P is an irreducible polynomial in $\mathbb{F}_p[t]$. There are three versions of this attack called the Level 1, the Level 2 and the Level 3 attack, respectively. The largest difference between these attacks is the polynomial ring under consideration. In the Level 1 attack, the polynomial ring $\mathbb{F}_p[x, y, t]$ is used, and they gave an algorithm to reconstruct the ideal $I \subset \mathbb{F}_p[x, y, t]$. The most time consuming computation in this attack is to compute a Gröbner basis of I to solve $NF_I(m') = 0$. In [23], it is pointed out that the Level 1 attack is not efficient and cannot break ASC for the recommended parameters. In the Level 2 attack, the polynomial ring $\mathbb{F}_p(t)[x, y, z]$ is used. In this case they gave an algorithm (which is similar to the Level 1 attack) to reconstruct the ideal $J \subset \mathbb{F}_p(t)[x, y, z]$. Note that the new variable z is necessary because the ideal (m, f, X) is generically equal to $\mathbb{F}_p(t)[x, y]$ (see [23] §3.2). The key which accelerates the computation of Gröbner basis is the following observation: the polynomials occuring in ASC have a high degree in t and a low degree in x and y. Thus, it is natural

to regard these polynomials as elements of $\mathbb{F}_p(t)[x, y]$ rather than elements of $\mathbb{F}_p[x, y, t]$. To make this attack more practical, in the Level 3 attack a modular arithmetic was used, i.e., the polynomial ring $(\mathbb{F}_p[t]/(P(t)))[x, y, z]$ is used for an irreducible polynomial $P(t)$ with $\deg P > \deg_t m$. The degree in t of the polynomials appearing in the computation of Gröbner basis is bounded by $\deg P(t)$ and so using a polynomial P of small degree, for example $\deg P = \deg_t m + 1$, makes this attack becomes more efficient than the Level 2 attack. Moreover, it is also possible to use a polynomial $P(t) = \prod_i P_i(t)$ such that $P_i(t)$'s are distinct irreducible polynomials and $\sum_i \deg P_i > \deg_t m$. In this case for each i we compute $m \pmod{P_i}$ and get m by the Chinese Remainder Theorem. Since $\deg P_i < \deg P$, we may have more efficient attack by using P having an appropriate number of irreducible factors and degree if $\deg_t m$ is large. Now, we give an algorithm of the Level 3 attack.

1. Choose a constant C and an integer $n \approx \deg_t(m) \cdot \log p/C$. Choose n irreducible polynomials $P_1, \ldots, P_n$ of degree $\approx C/\log p$ such that $\sum_{1 \le i \le n} \deg P_i > \deg_t m$. Set $i = 1$.

2. Let $K_i := \mathbb{F}_p[t]/(P_i)$.

3. Let $F_k^{(P_i)} := F_k \pmod{P_i}$ and $X^{(P_i)} := X \pmod{P_i}$. Compute $Q(y) := \mathrm{Res}_x(F_1^{(P_i)} - F_2^{(P_i)}, X^{(P_i)}) \in K_i[y]$, the resultant of $F_1^{(P_i)} - F_2^{(P_i)}$ and $X^{(P_i)}$ with respect to x.

4. Factor $Q(y)$ and let $Q_0(y)$ be an irreducible factor of highest degree.

5. Compute a Gröbner basis of the ideal $J := (F_1^{(P_i)} + z, F_2^{(P_i)} + z, X^{(P_i)}, Q_0) \subset K_i[x, y, z]$ with respect to the graded reverse lexicographical ordering.

6. Using the above Gröbner basis solve the following linear equation system over K_i to get $m^{(P_i)} := m \pmod{P_i}$

$$NF_J(m' + z) = 0,$$

   where m' is as above. If the system has no solution, then go back to step 4 and choose another factor of Q.

7. If $i < n$, then replace i by $i + 1$ and go back to step 2.

8. Recover m from $m^{(P_i)}$ by using the Chinese Remainder Theorem.

In [40], Ogura studied the ideal decomposition attack and proved the following two lemmas.

Lemma 1.3.1 ([40], Lemma 7.3.3)**.** *Let X, f, F_1 and F_2 be as above. Then*

$$\mathrm{Res}_x(f, X) \mid \mathrm{Res}_x(F_1 - F_2, X).$$

Lemma 1.3.2 ([40], Lemma 7.3.4)**.** *Let K be either $\mathbb{F}_p(t)$ or $\mathbb{F}_p[t]/(P(t))$ for some irreducible polynomial $P(t) \in \mathbb{F}_p[t]$. Let X, f, s_1, s_2, F_1 and F_2 be as above. We consider these polynomials as elements of K[x,y,z]. Let A and B be elements of $K[x, y, z]$ satisfying $Af + BX = \mathrm{Res}_x(f, X)$. If the ideal $(A, s_1 - s_2, X)$ coincides with K[x,y,z], then the following equality holds:*

$$(\mathrm{Res}_x(f, X), X, F_1 + z, F_2 + z) = (m + z, f, X).$$

Note that the existence of the polynomials A and B in Lemma 4.2.4 is ensured by the property of the resultant ([13]). Geometrically, the condition $(A, s_1 - s_2, X) = K[x, y, z]$ means that there are no common solutions to $A = 0$, $s_1 - s_2 = 0$ and $X = 0$ in the algebraic closure of K. Instinctively, in many cases it seems to be true and the experimental result in [40] shows that this attack is efficient against ASC.

Chapter 2

Semi-regularity of section equation systems

The results in this chapter are joint works with Koichiro Akiyama. In this section, we study the semi-regularity of equation systems arising from SFP. In §2.2, we show that solving SFP is reduced to solving a certain multivariate polynomial system of high degree over a finite field depending on X. When we compute a Gröbner basis to solve this system by using F_4, F_5 or matrix version of F_5 algorithm ([21], [22], [3]) and estimate its complexity, it is important to estimate the maximal degree of the polynomials appearing in a Gröbner basis computation with these algorithms. If a polynomial system which we want to solve is a semi-regular system, then it is easy to estimate the upper bounds of the maximum degree in matrix version of F_5 algorithm (cf. §2.3).

2.1 Notation

Let p be a prime number and $\mathbb{F}_p$ a finite field with p elements. Let R be a polynomial ring over $\mathbb{F}_p$. Let

$$R_d \ := \ \{f \in R \mid f \text{ is homogeneous of total degree } d\} \cup \{0\}.$$

For a homogeneous ideal $I \subset R$, we denote $R_d \cap I$ by I_d. For a polynomial $f \in R$ we denote by f^h the homogeneous part of highest degree of f.

2.2 General method of solving SFP

Let p be a prime number and $X \in \mathbb{F}_p[x, y, t]$ a polynomial which defines a surface S with a fibration $\sigma_S : S \to \mathbb{A}^1_{\mathbb{F}_p}$ over the affine t-line. Recall

that the SFP on the surface S is to find $u_x(t),\ u_y(t) \in \mathbb{F}_p[t]$ such that $X(u_x(t), u_y(t), t) = 0$. The general method of solving SFP is as follows: Let

$$u_x(t) = \sum_{1 \le i \le d_x} \alpha_i t^i,$$
$$u_y(t) = \sum_{1 \le i \le d_y} \beta_i t^i,$$

where α_i and β_i are varibles. We assume $d_x \ge d$ and $d_y \ge d$. By substituting them into X, we have

$$X(u_x(t), u_y(t), t) = \sum_{0 \le i \le r} f_i t^i,$$

where $r = \max\{id_x + jd_y + k \mid (i,j,k) \in \Lambda_X^{(p)}\}$ and $f_i \in \mathbb{F}_p[\alpha_0, \ldots, \alpha_{d_x}, \beta_0, \ldots, \beta_{d_y}]$. Thus, solving SFP is reduced to solving the following multivariate equation system:

$$\begin{cases} f_0(\alpha_0, \ldots, \alpha_{d_x}, \beta_0, \ldots, \beta_{d_y}) = 0, \\ \vdots \\ f_r(\alpha_0, \ldots, \alpha_{d_x}, \beta_0, \ldots, \beta_{d_y}) = 0, \\ \alpha_0^p - \alpha_0 = 0, \\ \vdots \\ \beta_{d_y}^p - \beta_{d_y} = 0. \end{cases}$$

Note that we need to add the equations called the field equations $\alpha_0^p - \alpha_0 = 0, \ldots, \beta_{d_y}^p - \beta_{d_y} = 0$ because we only need a solution in $\mathbb{F}_p$ rather than a solution in the algebraic closure of $\mathbb{F}_p$ for cryptographic applications. We call this equation system the *section equation system* for X. There are several methods to solve multivariate equation systems over finite fields, for example the XL algorithm ([44]), the Zhuang-Zi algorithm ([17]) and the Gröbner basis method ([33]). In this paper, we only consider the Gröbner basis method.

2.3 Regular systems and semi-regular systems

Let $R := \mathbb{F}_p[x_1, \ldots, x_n]$ be a polynomial ring with n variables over $\mathbb{F}_p$. Suppose that p is an odd prime number. In this section we give the definition of a regular sequence and a semi-regular sequence of R, and some properties which relate to a Gröbner basis computation.

2.3.1 Definition and properties

First, we define a regular sequence of R.

Definition 2.3.1. Let $\{f_i\}_{1\leq i\leq m}$ be a sequence of homogeneous elements of R. This sequence is *regular* if f_i is not a zero-divisor of $R/(f_1,\ldots,f_{i-1})$ for $i=1,\ldots,m$.

In order to characterize regular sequences and semi-regular sequences of R, we need to define the Hilbert function and the Hilbert series.

Definition 2.3.2. Let $I\subset R$ be a homogeneous ideal of R. The *Hilbert function* $HF_I(\cdot)$ and the *Hilbert series* $HS_I(z)$ of I are defined as follows:

$$HF_I(d) := \dim_{\mathbb{F}_p} R_d - \dim_{\mathbb{F}_p} I_d, \tag{2.1}$$

$$HS_I(z) := \sum_{d\geq 0} HF_I(d)z^d. \tag{2.2}$$

For $d<0$, we define $HF_I(d):=0$.

The next proposition shows the characterization of regular sequences by the Hilbert series and the importance of regular sequences for a Gröbner basis computation.

Proposition 2.3.3. *Let $\{f_i\}_{1\leq i\leq m}$ be a sequence of homogeneous elements of R. Let d_i be the total degree of f_i. Then, the following holds:*

(i) *The sequence $\{f_i\}_{1\leq i\leq m}$ is regular if and only if the Hilbert series of the ideal $(f_1,\ldots,f_m)$ is given by*

$$\frac{\prod_{1\leq i\leq m}(1-z^{d_i})}{(1-z)^n}.$$

(ii) *After a generic linear change of variables, the highest degree of elements of a Gröbner basis for the degree reverse lexicographic ordering is bounded by the index of regularity*

$$\sum_{1\leq i\leq m}(d_i-1)+1.$$

(iii) *The sequence $\{f_i\}_{1\leq i\leq m}$ is regular if and only if there are no reduction to 0 in F_5 algorithm.*

Proof. For proof of the property (i), see [32], Theorem 6.6 (pp. 436) and [24], pp. 137. The property (ii) is given by [33] and [26]. The property (iii) is proved in [22]. □

Geometrically, the sequence $\{f_i\}_{1\leq i\leq m}$ of homogeneous elements of R is regular when the algebraic set defined by the ideal $(f_1,\ldots,f_i)$ has codimension i for $i=1,\ldots,m$. This shows that if $m>n$, then $\{f_i\}_{1\leq i\leq m}$ is not regular for each sequence $\{f_i\}_{1\leq i\leq m}$ of homogeneous elements of R since the Krull dimension of R is n. The number of equations of a section equation system is larger than the number of variables because we add the field equations. So, we need a concept of semi-regularity.

Definition 2.3.4. Let $\{f_i\}_{1\leq i\leq m}$ be a sequence of homogeneous elements of R. This sequence is *d-regular* if $gf_i\in(f_1,\ldots,f_{i-1})$ and $\deg gf_i<d$ implies $g\in(f_1,\ldots,f_{i-1})$ for $i=1,\ldots,m$ and $g\in R$.

Definition 2.3.5. Let I be a homogeneous ideal of R. Suppose that the dimension of I is 0, i.e., the Krull dimension of R/I is zero. Then the *degree of regularity* of I is defined by

$$d_{\mathrm{reg}}:=\min\{d\mid \dim_{\mathbb{F}_p} I_d=\#\{\text{monomials of } R \text{ with degree } d\}\}.$$

In other words, the degree of regularity of I is the minimal integer d such that $R_d=I_d$.

The next lemma ([6], Theorem 6.54) ensures the existence of the degree of regularity of the zero-dimensional ideal.

Lemma 2.3.6. *Let $I\subset R$ be a homogeneous ideal. Then the dimension of I is 0 if and only if $\dim_{\mathbb{F}_p} R/I=\sum_{d\geq 0}\dim_{\mathbb{F}_p} R_d/I_d$ is finite.*

For a homogeneous ideal $I\subset R$, $R_d=I_d$ implies $R_k=I_k$ for $k>d$, i.e., $HF_I(d)=0$ implies $HF_I(k)=0$ for $k>d$. So, the degree of regularity of I is $\deg HS_I(z)+1$ if the dimension of I is 0. Now we can define the semi-regular sequences of R.

Definition 2.3.7. Let $\{f_i\}_{1\leq i\leq m}$ be a sequence of homogeneous elements of R. Suppose that the dimension of the ideal $(f_1,\ldots,f_m)$ is 0. Let d_{reg} be the degree of regularity of the ideal $(f_1,\ldots,f_m)\subset R$. Then $\{f_i\}_{1\leq i\leq m}$ is *semi-regular* if it is d_{reg}-regular.

Definition 2.3.8. Let $\{f_i\}_{1\leq i\leq m}$ be a sequence of elements of R. The sequence $\{f_i\}_{1\leq i\leq m}$ is *semi-regular* if $\{f_i^h\}_{1\leq i\leq m}$ is semi-regular.

The next theorem shows the importance of semi-regularity to estimate the complexity of a Gröbner basis computation.

Theorem 2.3.9. *Let $\{f_i\}_{1\leq i\leq m}$ be a semi-regular sequence of homogeneous elements of R. Suppose that the dimension of the ideal $(f_1,\ldots,f_m)$ is 0. Let d_i be the total degree of f_i. Let*

$$\frac{\prod_{1\leq i\leq m}(1-z^{d_i})}{(1-z)^n}=\sum_{i\geq 0}a_iz^i.$$

Then, the following statements hold:

(i) *For $m\leq n$, the sequence $\{f_i\}_{1\leq i\leq m}$ is regular if and only if it is semi-regular.*

(ii) *The degree of regularity of the ideal $(f_1,\ldots,f_m)$ is given by*

$$d_{\mathrm{reg}}=\min\{i\mid a_i\leq 0\}.$$

(iii) *Let d_{reg} be the degree of regularity of the ideal $(f_1,\ldots,f_m)$. We assume that the sequence $\{f_i\}_{1\leq i\leq m}$ is semi-regular. Then there is no reduction to 0 in matrix version of F_5 algorithm for degrees smaller than d_{reg} during computations of a Gröbner basis of the ideal $(f_1,\ldots,f_m)$. Moreover, the total number of arithmetic operations in $\mathbb{F}_p$ performed by matrix version of F_5 algorithm is bounded by*

$$O\left(\binom{n+d_{\mathrm{reg}}}{n}^{\omega}\right),$$

where the exponent $\omega<2.39$ is the exponent in the complexity of matrix multiplication.

Proof. See [4] for the proof of statements (i) and (ii). For statement (iii) see [3], Theorem 3.2.10. □

2.3.2 Criterion for semi-regularity

We derive the criterion for semi-regularity to judge the semi-regularity of the section equation systems. Our criterion is the same as the criterion proposed in [4] or [3] but we give more detailed description.

Theorem 2.3.10. *Let $\{f_i\}_{1\leq i\leq m}$ be a sequence of homogeneous elements of R and $I^{(i)}:=(f_1,\ldots,f_i)$ (we define $I^{(0)}$ as the zero ideal (0)). Let d_i be*

the total degree of f_i. Suppose that the dimension of $I^{(m)}$ is 0. Then, the sequence $\{f_i\}_{1\leq i\leq m}$ is d-regular if and only if the following holds:

$HF_{I^{(i)}}(d') = HF_{I^{(i-1)}}(d') - HF_{I^{(i-1)}}(d'-d_i)$ *(for $i = 1, \ldots, m$ and $0 \leq d' < d$).*

In particular, the sequence $\{f_i\}_{1\leq i\leq m}$ is semi-regular if and only if the following holds:

$HF_{I^{(i)}}(d') = HF_{I^{(i-1)}}(d') - HF_{I^{(i-1)}}(d'-d_i)$ *(for $i = 1, \ldots, m$ and $0 \leq d' < d_{\mathrm{reg}}$),*

where d_{reg} is the degree of regularity of $I^{(m)}$.

Proof. For an integer $d' \geq d_i$ we consider the linear map of $\mathbb{F}_p$-vector spaces

$$\phi_{d',i} : R_{d'-d_i}/I^{(i-1)}_{d'-d_i} \longrightarrow R_{d'}/I^{(i-1)}_{d'}$$

given by $g \pmod{I^{(i-1)}_{d'-d_i}} \mapsto gf_i \pmod{I^{(i-1)}_{d'}}$. Then we have

$$\begin{aligned}(R_{d'}/I^{(i-1)}_{d'})/\mathrm{Im}\phi_{d',i} &\cong (R_{d'}/I^{(i-1)}_{d'})/((R_{d'-d_i}f_i + I^{(i-1)}_{d'})/I^{(i-1)}_{d'}) \\ &\cong R_{d'}/(R_{d'-d_i}f_i + I^{(i-1)}_{d'}) = R_{d'}/I^{(i)}_{d'}.\end{aligned}$$

It implies

$$\begin{aligned}\dim_{\mathbb{F}_p} R_{d'}/I^{(i)}_{d'} &= HF_{I^{(i)}}(d') = \dim_{\mathbb{F}_p} R_{d'}/I^{(i-1)}_{d'} - \dim_{\mathbb{F}_p} \mathrm{Im}\phi_{d',i} \\ &= HF_{I^{(i-1)}}(d') - \dim_{\mathbb{F}_p} \mathrm{Im}\phi_{d',i}.\end{aligned}$$

For $i = 1, \ldots, m$, because $\mathrm{Im}\phi_{d',i} \leq \dim_{\mathbb{F}_p} R_{d'-d_i}/I^{(i-1)}_{d'-d_i}$, we have

$$HF_{I^{(i)}}(d') \geq HF_{I^{(i-1)}}(d') - HF_{I^{(i-1)}}(d'-d_i). \tag{2.3}$$

The sequence $\{f_i\}_{1\leq i\leq m}$ is d-regular ($d > d_i$) if and only if $\phi_{d',i}$ is injective for $i = 1, \ldots, m$ and $d_i \leq d' < d$, i.e.,

$$\dim_{\mathbb{F}_p} \phi_{d',i} = \dim_{\mathbb{F}_p} R_{d'-d_i}/I^{(i-1)}_{d'-d_i} = HF_{I^{(i-1)}}(d'-d_i).$$

Thus, the sequence $\{f_i\}_{1\leq i\leq m}$ is d-regular ($d > d_i$) if and only if the following equality holds:

$$HF_{I^{(i)}}(d') = HF_{I^{(i-1)}}(d') - HF_{I^{(i-1)}}(d'-d_i),$$

for $i = 1, \ldots, m$ and $d_i \leq d' < d$. This equation is satisfied for $0 \leq d' < d_i$ because $HF_{I^{(i-1)}}(d'-d_i) = 0$ and $I^{(i)}_{d'} = I^{(i-1)}_{d'}$ are satisfied for $d' < d_i$. Thus, we have proved this theorem. □

This theorem implies that we have to compute various Hilbert functions to judge the semi-regularity of the sequence $\{f_i\}_{1\leq i\leq m}$. If we compute Hilbert functions of $I^{(i)}$ by using the method proposed in [5], we have to compute a Gröbner basis of $I^{(i)}$.

Next, we derive the more efficient criterion for semi-regularity from this theorem.

Theorem 2.3.11. *Let $\{f_i\}_{1\leq i\leq m}$, d_i and $I^{(i)}$ be as above. Suppose that the dimension of $I^{(m)}$ is 0. Let*

$$S_{m,n}(z) := \frac{\prod_{1\leq i\leq m}(1-z^{d_i})}{(1-z)^n}.$$

Then, the sequence $\{f_i\}_{1\leq i\leq m}$ is d-regular if and only if the following holds:

$$\sum_{0\leq d'<d} HF_{I^{(m)}}(d')z^{d'} = \lceil S_{m,n}(z)\rceil_d, \tag{2.4}$$

where $\lceil \sum_{i\geq 0} a_i z^i\rceil_d = \sum_{0\leq i<d} a_i z^i$. In particular, the sequence $\{f_i\}_{1\leq i\leq m}$ is semi-regular if and only if the following holds:

$$HS_{I^{(m)}}(z) = \lceil S_{m,n}(z)\rceil_{d_{reg}} = [S_{m,n}(z)], \tag{2.5}$$

where d_{reg} is the degree of regularity of $I^{(m)}$ and $\left[\sum_{i\geq 0} a_i z^i\right] = \sum_{i\geq 0} b_i z^i$ with $b_i = a_i$ if $a_j > 0$ for $0 \leq j \leq i$ and $b_i = 0$ otherwise.

Proof. The inequality (2.3) implies that there are integers $a_{i,d'} \geq 0$ such that

$$HF_{I^{(i)}}(d') = HF_{I^{(i-1)}}(d') - HF_{I^{(i-1)}}(d'-d_i) + a_{i,d'},$$

for $i = 1,\ldots,m$ and $d' \geq 0$. Multiplying the above equality by $z^{d'}$ and summing over $d' \geq 0$ leads to

$$\sum_{d'\geq 0} HF_{I^{(i)}}(d')z^{d'} = \sum_{d'\geq 0} HF_{I^{(i-1)}}(d')z^{d'} - \sum_{d'\geq 0} HF_{I^{(i-1)}}(d'-d_i)z^{d'} + \sum_{d'\geq 0} a_{i,d'}z^{d'},$$

for $i = 1, \ldots, m$. So, we have

$$\begin{aligned}
HS_{I^{(m)}}(z) &= (1 - z^{d_m}) HS_{I^{(m-1)}}(z) + \sum_{d' \geq 0} a_{m,d'} z^{d'} \\
&= (1 - z^{d_m}) \left((1 - z^{d_{m-1}}) HS_{I^{(m-2)}}(z) + \sum_{d' \geq 0} a_{m-1,d'} z^{d'} \right) + \sum_{d' \geq 0} a_{m,d'} z^{d'} \\
&\vdots \\
&= \prod_{1 \leq i \leq m} (1 - z^{d_i}) HS_{(0)}(z) + \sum_{2 \leq i \leq m} \left(\prod_{i \leq j \leq m} (1 - z^{d_j}) \sum_{d' \geq 0} a_{i-1,d'} z^{d'} \right) \\
&\quad + \sum_{d' \geq 0} a_{m,d'} z^{d'} \\
&= S_{m,n}(z) + \sum_{2 \leq i \leq m} \left(\prod_{i \leq j \leq m} (1 - z^{d_j}) \sum_{d' \geq 0} a_{i-1,d'} z^{d'} \right) + \sum_{d' \geq 0} a_{m,d'} z^{d'}.
\end{aligned}$$

Note that $HS_{(0)}(z) = (\sum_{i \geq 0} z^i)^n = \frac{1}{(1-z)^n}$ because by definition, we have

$$HF_{(0)}(d) = \dim_{\mathbb{F}_p} R_d = \left(\begin{array}{c} n + d - 1 \\ d \end{array} \right).$$

Proposition 2.3.10 implies that the sequence $\{f_i\}_{1 \leq i \leq m}$ is d-regular if and only if $a_{i,d'} = 0$ for $i = 1, \ldots, m$ and $0 \leq d' < d$. This proves (2.4) and the first equality of (2.5). The second equality of (2.5) is obtained from $HF_{I^{(m)}}(d_{\mathrm{reg}}) = 0$ and $a_{i,d'} \geq 0$. Thus, we have proved this theorem. □

The second criterion (Theorem 2.3.11) implies that we only have to compute the Hilbert function of $I^{(m)}$ to determine the d-regularity or semi-regularity of the sequence $\{f_i\}_{1 \leq i \leq m}$.

2.4 Semi-regularity of section equation systems

In this section we give the experimental results on the semi-regularity of the section equation systems for $X(x, y, t) \in \mathbb{F}_p[x, y, t]$.

2.4.1 Algorithm

In order to determine the semi-regularity of the section equation systems for X, we take the following steps:

1. For a given $X \in \mathbb{F}_p[x, y, t]$, construct the section equation system for X:

$$\begin{cases} f_0(\alpha_0, \ldots, \alpha_{d_x}, \beta_0, \ldots, \beta_{d_y}) = 0, \\ \vdots \\ f_r(\alpha_0, \ldots, \alpha_{d_x}, \beta_0, \ldots, \beta_{d_y}) = 0, \\ \alpha_0^p - \alpha_0 = 0, \\ \vdots \\ \beta_{d_y}^p - \beta_{d_y} = 0. \end{cases} \tag{2.6}$$

   Note that $\deg f_i = \deg_{x,y} X$ for $i = 1, \ldots, r$.

2. Let $I := (f_1^h, \ldots, f_r^h, \alpha_0^p, \ldots, \beta_{d_y}^p)$. Note that the dimension of I is 0 because any prime ideal of $\mathbb{F}_p[\alpha_0, \ldots, \alpha_{d_x}, \beta_0, \ldots, \beta_{d_y}]$ cantaining I contains the maximal ideal $(\alpha_0, \ldots, \beta_{d_y})$.

3. Let

$$\begin{aligned} S_{m,n}(z) &:= \prod_{1 \le i \le r} (1 - z^{\deg_{x,y} X})(1 - z^p)^{2+d_x+d_y}/(1 - z)^n \\ &= (1 - z^{\deg_{x,y} X})^r (1 - z^p)^{2+d_x+d_y}/(1 - z)^n. \end{aligned}$$

4. Compute $HS_I(z)$ and compare it with $[S_{m,n}(z)]$.

2.4.2 Parameters

We use the following parameters:

- p : the number of elements of a base finite field;
- w : $\deg_{x,y} X$; degree of X with respect to x and y;
- NonRed-Monos (nrm) : the number of distinct monomials in (2.6) other than the field equations;
- d_c : $\deg c_{ij}(t)$ for $(i, j) \in \Lambda_X^{(p)} \setminus \{(0, 0)\}$, where $c_{ij}(t)$ is a coefficient of a monomial $x^i y^j$ of X, i.e., $X = \sum_{(i,j) \in \Lambda_X^{(p)}} c_{ij}(t) x^i y^j$;
- d_x : $\deg u_x(t)$, where $u_x(t) = \sum_{0 \le i \le d_x} \alpha_i t^i$;
- d_y : $\deg u_y(t)$, where $u_y(t) = \sum_{0 \le i \le d_y} \beta_i t^i$;
- d_{reg} : the degree of regularity of the ideal $(f_1^h, \ldots, f_r^h, \alpha_0^p, \ldots, \beta_{d_y}^p)$;

- d_{semi} : the degree of regularity of the ideal $(f_1^h, \ldots, f_r^h, \alpha_0^p, \ldots, \beta_{d_y}^p)$ when the section equation system is semi-regular, i.e., $d_{\text{semi}} = \deg([S_{m,n}(z)])+1$;

- $d_{\max}$: the maximal degree of polynomials appearing in the Gröbner basis computation except for the polynomials arising from section equation systems.

2.4.3 Experimental results

For each set of parameters, we construct a random $X \in \mathbb{F}_p[x, y, t]$ with its section and the section equation system for X. Then we determine the semi-regularity of the section equation system for X by using the above algorithm. We use the computer algebra system Magma ([10]) to perform those processes. We also collect the maximal degree reached during the Gröbner basis computation by applying Magma function *Groebner* and its *Verbose* output. We repeat each experiment 1000 times for each parameter.

In the following we show the experimental results on the semi-regularity of the section equation system for X. We also show the values of d_{reg}, d_{semi} and $d_{\max}$ together with their frequencies. Our result shows that in many cases, the section equation systems are not semi-regular for small w and d. As nrm and d_c increase, d_{semi} is a good approximation to $d_{\max}$ and the variations of d_{reg} and $d_{\max}$ decrease.

We think that these results are interesting and it is necessary to experiment on the semi-regularity of the section equation systems for large parameters. It is also important to find parameters such that there is at least one section equation system which is semi-regular, or prove that there are infinitely many parameters or only finite parameters such that there is at least one section equation system which is semi-regular.

Table 2.4.1: Semi-regularity of section equation systems

p	d	w	nrm	d_c	frequency (times)	d_{reg}	d_{semi}	d_{max}
11	2	3	84	0	0	31 (916 times) 33 (76 times) 35 (8 times)	8	2 (1 times) 3 (1 times) 4 (4 times) 5 (74 times) 6 (920 times)
11	2	3	84	1	838	7 (838 times) 8 (71 times) 9 (10 times) 31 (81 times)	7	5 (2 times) 6 (160 times) 7 (838 times)
11	2	3	84	2	0	7 (987 times) 21 (2 times) 31 (11 times)	6	5 (5 times) 6 (995 times)
11	2	3	84	3	824	6 (896 times) 7 (104 times)	6	5 (4 times) 6 (996 times)

Table 2.4.2: Semi-regularity of section equation systems (continuation)

p	d	w	nrm	d_c	frequency (times)	d_{reg}	d_{semi}	$d_{\max}$
11	2	4	100	2	0	9 (134 times) 11 (130 times) 31 (336 times) 33 (329 times) 35 (71 times)	8	5 (3 times) 6 (366 times) 7 (353 times) 8 (278 times)
11	2	4	155	2	0	9 (416 times) 10 (2 times) 31 (558 times) 33 (24 times)	8	6 (14 times) 7 (240 times) 8 (746 times)
11	2	4	210	2	0	9 (992 times) 31 (8 times)	8	7 (3 times) 8 (997 times)
11	2	4	100	3	0	9 (134 times) 11 (131 times) 31 (336 times) 33 (329 times) 35 (70 times)	8	5 (3 times) 6 (376 times) 7 (367 times) 8 (254 times)
11	2	4	155	3	0	9 (413 times) 31 (539 times) 33 (48 times)	8	5 (2 times) 6 (30 times) 7 (390 times) 8 (578 times)
11	2	4	210	3	0	9 (997 times) 31 (3 times)	8	7 (5 times) 8 (995 times)
11	2	4	100	4	0	9 (130 times) 11 (119 times) 31 (351 times) 33 (335 times) 35 (65 times)	8	5 (11 times) 6 (387 times) 7 (602 times)
11	2	4	155	4	0	8 (308 times) 9 (130 times) 31 (536 times) 33 (36 times)	8	6 (14 times) 7 (986 times)
11	2	4	210	4	0	8 (921 times) 9 (79 times)	8	7

Chapter 3

A Survey on diophantine problems

This chapter used to be a survey on diophantine problems. However, as the contents are well-known and can be found easily in the existing literature, we have decided not to include them in the final version of this thesis.

Chapter 4

Our cryptosystem

In this chapter we give four cryptosystems based on the difficulty of finding solutions of diophantine equations.

4.1 Notation

We denote by $\mathbb{Z}[\underline{x}] := \mathbb{Z}[x_1, \ldots, x_n]$ the polynomial ring with n variables. For a vector $\underline{i} := (i_1, \ldots, i_n) \in (\mathbb{Z}_{\geq 0})^n$ we write $\underline{x}^{\underline{i}} := x_1^{i_1} \cdots x_n^{i_n}$ and $\sum \underline{i} := \sum_{1 \leq j \leq n} i_j$. For a finite subset $\Lambda \subset (\mathbb{Z}_{\geq 0})^n$ and a polynomial $f = \sum_{(i_1,\ldots,i_n)\in\Lambda} f_{i_1 \cdots i_n} x_1^{i_1} \cdots x_n^{i_n} = \sum_{\underline{i}} f_{\underline{i}} \underline{x}^{\underline{i}} \in \mathbb{Z}[\underline{x}]$ we define

$$\begin{aligned} \Lambda_f &:= \{\underline{i} \in (\mathbb{Z}_{\geq 0})^n \mid f_{\underline{i}} \neq 0\}, \\ \Gamma_f &:= \{(\underline{i}, b_{\underline{i}}) \in \Lambda_f \times \mathbb{Z}_{>0} \mid 2^{b_{\underline{i}}-1} \leq |f_{\underline{i}}| < 2^{b_{\underline{i}}}\}. \end{aligned}$$

We call Λ_f the support of f. For two finite subsets Λ_1, $\Lambda_2 \subset (\mathbb{Z}_{\geq 0})^n$, we define the product of Λ_1 and Λ_2 as follows:

$$\Lambda_1 \Lambda_2 := \{\underline{i}_1 + \underline{i}_2 \mid \underline{i}_1 \in \Lambda_1, \underline{i}_2 \in \Lambda_2\}.$$

This means that if $\Lambda_i = \Lambda_{f_i}$ for some polynomials $f_i \in \mathbb{Z}[\underline{x}]$, then $\Lambda_1 \Lambda_2 = \Lambda_{f_1 f_2}$. For example, for $f_1 = 5x_1^4 x_2^2 x_3 - 13x_1^2 x_2 + 7x_3 + 2$ and $f_2 = 8x_1^2 x_2^2 x_3 - 9x_1 x_2^2 + 6x_3 - 11$, we have

$$\begin{aligned} \Lambda_{f_1} &:= \{(4,2,1), (2,1,0), (0,0,1), (0,0,0)\}, \\ \Gamma_{f_1} &:= \{(4,2,1,3), (2,1,0,4), (0,0,1,3), (0,0,0,2)\}, \\ \Lambda_{f_2} &:= \{(2,2,1), (1,2,0), (0,0,1), (0,0,0)\}, \\ \Gamma_{f_2} &:= \{(2,2,1,4), (1,2,0,4), (0,0,1,3), (0,0,0,4)\}. \end{aligned}$$

We also have

$$\begin{aligned}\Lambda_{f_1}\Lambda_{f_2} &= \{(6,4,2),(5,4,1),(4,3,1),(4,2,2),(4,2,1),(3,3,0),(2,2,2),(2,2,1),\\ &\quad (2,1,1),(2,1,0),(1,2,1),(1,2,0),(0,0,2),(0,0,1),(0,0,0)\} = \Lambda_{f_1 f_2}.\end{aligned}$$

We denote by w_f the total degree of f. Define

$$\begin{aligned}C_n(f) &:= \{f_{\underline{i}} \mid \underline{i} \in \Lambda_f, \sum \underline{i} = n\},\\ H(f) &:= \max\{|f_{\underline{i}}| \mid \underline{i} \in \Lambda_f\}.\end{aligned}$$

We call the elements of $C_n(f)$ the coefficients of f of degree n. For a vector $\underline{b} := (b_1, \ldots, b_n) \in \mathbb{Q}^n$, we denote by $f(\underline{b})$ the value of f at $\underline{b}$. For an integer d, we denote by $\underline{b}/d$ the vector $(\frac{b_1}{d}, \ldots, \frac{b_n}{d})$. For each ideal $J \subset \mathbb{Q}[\underline{x}]$, each polynomial $f \in \mathbb{Q}[\underline{x}]$ and each monomial ordering $<$, we denote by $NF_J(f)$ a normal form of f with respect to J and $<$. For a polynomial $f \in \mathbb{Z}[\underline{x}]$ and an integer m, we denote by $\overline{f}^{(m)}$ the polynomial $f \pmod m \in (\mathbb{Z}/m\mathbb{Z})[\underline{x}]$.

4.2 Our cryptosystem 1

First, we try to construct a similar cryptosystem to ASC. However, there are no algorithms to factorize integers in polynomial time. So this cryptosystem is not efficient. Moreover, we show that if $n = 2, 3$, the attack which is similar to the ideal decomposition attack can break the one-wayness of this cryptosystem.

4.2.1 Algorithm

System parameters

The system parameters in our cryptosystem are as follows:

1. n : the number of variables of the polynomials in our cryptosystem;
2. b : bit length of a secret key;
3. w : the total degree of a public key;
4. Λ_1 : a finite subset of $(\mathbb{Z}_{\geq 0})^n$;
5. $D_1 := \{d_{\underline{i}}^{(1)} \mid \underline{i} \in \Lambda_1\} \subset \mathbb{Z}_{\geq 0}$.

For the size of b, see §4.2.2.

Key generation

1. Secret key
   Choose a vector $\underline{a} = (a_1, \ldots, a_n) \in \mathbb{Z}^n$ such that $2^{b-1} \leq |a_i| < 2^b$ for $i = 1, \ldots, n$. Make them secret.

2. Public key
   For $k = 2, 3$, choose finite subsets $\Lambda_k \subset (\mathbb{Z}_{\geq 0})^n$ and $D_k = \{d_{\underline{i}}^{(k)} \mid \underline{i} \in \Lambda_k\} \subset \mathbb{Z}_{\geq 0}$ so that the following holds:

   (i) $\Lambda_2 \subset \Lambda_1 \Lambda_3$.

   (ii) For any polynomial $f_k = \sum_{\underline{i} \in \Lambda_k} f_{\underline{i}}^{(k)} \underline{x}^{\underline{i}} \in \mathbb{Z}[\underline{x}]$ $(k = 1, 2, 3)$ with $\Lambda_{f_k} = \Lambda_k$ and $2^{d_{\underline{i}}^{(k)}-1} \leq f_{\underline{i}}^{(k)} < 2^{d_{\underline{i}}^{(k)}}$, we have

$$\begin{cases} \deg_{x_1} f_1 < \deg_{x_1} f_2 < \deg_{x_1} f_3, \\ \vdots \\ \deg_{x_n} f_1 < \deg_{x_n} f_2 < \deg_{x_n} f_3, \\ \max\{d_{\underline{i}}^{(1)} \mid \underline{i} \in \Lambda_1\} < \max\{d_{\underline{i}}^{(2)} \mid \underline{i} \in \Lambda_2\} < \max\{d_{\underline{i}}^{(3)} \mid \underline{i} \in \Lambda_3\}, \\ (\deg_{x_1} f_2, \ldots, \deg_{x_n} f_2, \max\{d_{\underline{i}}^{(2)} \mid \underline{i} \in \Lambda_2\}) \in \Gamma_{f_2}, \\ (\deg_{x_1} f_3, \ldots, \deg_{x_n} f_3, \max\{d_{\underline{i}}^{(3)} \mid \underline{i} \in \Lambda_3\}) \in \Gamma_{f_3}. \end{cases} \tag{4.1}$$

   Construct an irreducible polynomial $X(\underline{x}) \in \mathbb{Z}[\underline{x}]$ such that $X(\underline{a}) = 0$ and $\Gamma_X = \{(\underline{i}, d_{\underline{i}}^{(1)}) \mid \underline{i} \in \Lambda_1\}$. For $i = 2, 3$, make X, Λ_i and D_i public.

We give a method to construct a public key X with $X(\underline{a}) = 0$.

1. Choose a finite subset $\Lambda \subset (\mathbb{Z}_{\geq 0})^n$ containing $\underline{0}$.

2. For $\underline{i} \in \Lambda' := \Lambda \setminus \{\underline{0}\}$, choose random non-zero integers $c_{\underline{i}}$.

3. Let $c_{\underline{0}} := -\sum_{\underline{i} \in \Lambda'} c_{\underline{i}} \underline{a}^{\underline{i}}$ and define

$$X := \sum_{\underline{i} \in \Lambda} c_{\underline{i}} \underline{x}^{\underline{i}}.$$

Encryption

Assume that the sender wants to send a polynomial $m(\underline{x}) = \sum_{\underline{i} \in \Lambda_2} m_{\underline{i}} x^{\underline{i}} \in \mathbb{Z}[\underline{x}]$ with $\Gamma_m = \{(\underline{i}, d_{\underline{i}}^{(2)}) \mid \underline{i} \in \Lambda_2\}$.

1. For $k = 1, 2$, choose random polynomials

$$\begin{aligned} s_k &= \sum_{\underline{i} \in \Lambda_1} s_{\underline{i}}^{(k)} \underline{x}^{\underline{i}}, \\ r_k &= \sum_{\underline{i} \in \Lambda_3} r_{\underline{i}}^{(k)} \underline{x}^{\underline{i}}, \\ f &= \sum_{\underline{i} \in \Lambda_3} f_{\underline{i}} \underline{x}^{\underline{i}}, \end{aligned}$$

in $\mathbb{Z}[\underline{x}]$ such that $\Gamma_{s_k} = \Gamma_X$ and $\Gamma_{r_k} = \Gamma_f = \{(\underline{i}, d_{\underline{i}}^{(3)}) \mid \underline{i} \in \Lambda_3\}$. Note that from (4.1), we have

$$\begin{cases} \deg_{x_1} X < \deg_{x_1} m < \deg_{x_1} f, \\ \vdots \\ \deg_{x_n} X < \deg_{x_n} m < \deg_{x_n} f, \\ \max\{d_{\underline{i}}^{(1)} \mid \underline{i} \in \Lambda_X\} < \max\{d_{\underline{i}}^{(2)} \mid \underline{i} \in \Lambda_m\} < \max\{d_{\underline{i}}^{(3)} \mid \underline{i} \in \Lambda_f\}, \\ (\deg_{x_1} m, \ldots, \deg_{x_n} m, \max\{d_{\underline{i}}^{(2)} \mid \underline{i} \in \Lambda_m\}) \in \Gamma_m, \\ (\deg_{x_1} f, \ldots, \deg_{x_n} f, \max\{d_{\underline{i}}^{(3)} \mid \underline{i} \in \Lambda_f\}) \in \Gamma_f. \end{cases} \tag{4.2}$$

2. Put $F_i := m + s_i f + r_i X$ for $i = 1, 2$, and send (F_1, F_2).

Decryption

1. For $i = 1, 2$, compute

$$h_i := F_i(\underline{a}) = m(\underline{a}) + s_i(\underline{a}) f(\underline{a}).$$

2. Factorize $h_1 - h_2$. Find a factor h_3 of it whose bit length is equal to $\mathrm{bit}(f) := \max\{d_{\underline{i}}^{(3)} \mid \underline{i} \in \Lambda_3\} + \log_2 \underline{a}^{\underline{k}}$ bits, where $\underline{k}$ is the element of Λ_f such that $\sum \underline{k} = w_f$. If there is no such a factor, then we let h_3 be the largest factor having bit length close to $\mathrm{bit}(f)$. Note that from (4.2), we can expect that h_3 and $f(\underline{a})$ have the same bits and $h_3 > m(\underline{a})$.

3. Compute $h_4 := h_1 \pmod{h_3}$. Note that if h_3 divides $s_1(\underline{a}) f(\underline{a})$, then $h_4 = m(\underline{a})$.

4. Extract $m(\underline{x})$ from h_4. Then we assume that we obtain $m'(\underline{x})$ (cf. §4.2.1).

5. We can verify whether $m' = m$ or not by a MAC (message authentication code) of m. If the verification fails, then go back to step 2 and choose another factor of $h_1 - h_2$.

Recovering m from $m(\underline{a})$

We consider the conditions to recover m from $m(\underline{a})$ uniquely. More precisely, we consider the following problem:

Problem : For a given vector $\underline{b} \in \mathbb{Z}^n$ and a map $\sigma_{\underline{b}} : \mathbb{Z}[\underline{x}] \longrightarrow \mathbb{Z}$ by $g \mapsto g(\underline{b})$, find a subset $S \subset \mathbb{Z}[\underline{x}]$ such that $\sigma_{\underline{b}}|_S$ is injective.

Let S be a subset of $(\mathbb{Z}[\underline{x}]$ such that $\sigma_{\underline{a}}|_S$ is injective. If $m \in S$, then we can recover m from $m(\underline{a})$ uniquely. We give some examples of such subsets.

Proposition 4.2.1. *Let $\underline{a}$ and $\sigma_{\underline{a}}$ be as above and $a_{\min} := \min\{|a_1|, \ldots, |a_n|\}$. Define*

$$\begin{aligned} \Lambda &= \{\underline{i}_1, \ldots, \underline{i}_k\} \subset (\mathbb{Z}_{\geq 0})^n \left(\sum \underline{i}_j \leq \sum \underline{i}_{j+1}\right), \\ S_{\Lambda,1} &:= \left\{ g \in \mathbb{Z}_{\geq 0}[\underline{x}] \mid \Lambda_g = \Lambda, |\underline{a}^{\underline{i}_j}| > H(g) \sum_{\underline{i} \in \{\underline{i}_1, \ldots, \underline{i}_{j-1}\}} |\underline{a}^{\underline{i}}| \text{ for } j = 1, \ldots, k \right\}. \end{aligned}$$

If $i_{j,\ell} \leq i_{j+1,\ell}$ for $j = 1, \ldots, k-1$ and $\ell = 1, \ldots, n$, then we define

$$S_{\Lambda,2} := \{g \in \mathbb{Z}_{\geq 0}[\underline{x}] \mid \Lambda_g = \Lambda, H(g) < a_{\min}\},$$

where $\underline{i}_j = (i_{j,1}, \ldots, i_{j,n})$. If $d := \gcd(a_1, \ldots, a_n) > 1$, $\gcd\left(\frac{\prod_{1 \leq i \leq n} a_i}{d^n}, d\right) = 1$ and $\sum \underline{i}_j < \sum \underline{i}_{j+1}$ for $j = 1, \ldots, k-1$, then we define

$$S_{\Lambda,3} := \{g \in \mathbb{Z}_{\geq 0}[\underline{x}] \mid \Lambda_g = \Lambda, H(g) < d\}.$$

Then $\sigma_{\underline{a}}|_{S_{\Lambda,i}}$ is injective for $i = 1, 2, 3$.

Proof. We assume that $\sigma_{\underline{a}}|_{S_{\Lambda,1}}$ is not injective. Let $g_1^{(1)} = \sum g_{1,\underline{i}}^{(1)} \underline{x}^{\underline{i}}$ and $g_2^{(1)} = \sum g_{2,\underline{i}}^{(1)} \underline{x}^{\underline{i}}$ be distinct elements of $S_{\Lambda,1}$ such that $g_1^{(1)}(\underline{a}) = g_2^{(1)}(\underline{a})$. Without loss of generality, we may assume $H\left(g_1^{(1)}\right) \geq H\left(g_2^{(1)}\right)$. Let $j_1 := \max\{j \mid g_{1,\underline{i}_j}^{(1)} \neq g_{2,\underline{i}_j}^{(1)}\}$. Then

$$g_1^{(1)}(\underline{a}) - g_2^{(1)}(\underline{a}) = \sum_{1 \leq j \leq j_1} \left(g_{1,\underline{i}_j}^{(1)} - g_{2,\underline{i}_j}^{(1)}\right) \underline{a}^{\underline{i}_j} = 0.$$

Thus we have

$$\begin{aligned} \left| \sum_{1 \leq j \leq j_1 - 1} \left(g_{1,\underline{i}_j}^{(1)} - g_{2,\underline{i}_j}^{(1)}\right) \underline{a}^{\underline{i}_j} \right| &= \left| \left(g_{2,\underline{i}_{j_1}}^{(1)} - g_{1,\underline{i}_{j_1}}^{(1)}\right) \underline{a}^{\underline{i}_{j_1}} \right| \\ &> \left| \left(g_{2,\underline{i}_{j_1}}^{(1)} - g_{1,\underline{i}_{j_1}}^{(1)}\right) \right| H\left(g_1^{(1)}\right) \sum_{1 \leq j \leq j_1 - 1} \left|\underline{a}^{\underline{i}_j}\right|. \end{aligned}$$

It implies

$$\sum_{1\le j\le j_1-1}\left|(g^{(1)}_{1,\underline{i}_j}-g^{(1)}_{2,\underline{i}_j})\underline{a}^{\underline{i}_j}\right| > \left|\left(g^{(1)}_{2,\underline{i}_{j_1}}-g^{(1)}_{1,\underline{i}_{j_1}}\right)\right| H\left(g^{(1)}_1\right)\sum_{1\le j\le j_1-1}\left|\underline{a}^{\underline{i}_j}\right|.$$

This is a contradiction because $\left|g^{(1)}_{1,\underline{i}_j}-g^{(1)}_{2,\underline{i}_j}\right| \le \left|\left(g^{(1)}_{2,\underline{i}_{j_1}}-g^{(1)}_{1,\underline{i}_{j_1}}\right)\right| H\left(g^{(1)}_1\right)$.

We assume that $\sigma_{\underline{a}}|_{S_{\Lambda,2}}$ is not injective even if $i_{j,\ell}\le i_{j+1,\ell}$ for $j=1,\dots,k-1$ and $\ell=1,\dots,n$. Let $g^{(2)}_1=\sum g^{(2)}_{1,\underline{i}}\underline{x}^{\underline{i}}$ and $g^{(2)}_2=\sum g^{(2)}_{2,\underline{i}}\underline{x}^{\underline{i}}$ be distinct elements of $S_{\Lambda,2}$ such that $g^{(2)}_1(\underline{a})=g^{(2)}_2(\underline{a})$. Let $j_2:=\min\{j\mid g^{(2)}_{1,\underline{i}_j}\neq g^{(2)}_{2,\underline{i}_j}\}$. Then we have

$$g^{(2)}_1(\underline{a})-g^{(2)}_2(\underline{a})=\sum_{j_2\le j\le k}\left(g^{(2)}_{1,\underline{i}_j}-g^{(2)}_{2,\underline{i}_j}\right)\underline{a}^{\underline{i}_j}=0.$$

Our assumption implies that $\underline{a}^{\underline{i}_j}$ is divisible by $\underline{a}^{\underline{i}_{j_2}}$ for $j=j_2+1,\dots,k$. Thus we have

$$g^{(2)}_{1,\underline{i}_{j_2}}-g^{(2)}_{2,\underline{i}_{j_2}}=-\sum_{j_2+1\le j\le k}\left(g^{(2)}_{1,\underline{i}_j}-g^{(2)}_{2,\underline{i}_j}\right)\underline{a}^{\underline{i}_j-\underline{i}_{j_2}}.$$

The right hand side of the above equation is divisible by $a_\ell^{i_{j,\ell}-i_{j_2,\ell}}$ for some ℓ because of our assumption. On the other hand, we have $\left|g^{(2)}_{1,\underline{i}_{j_2}}-g^{(2)}_{2,\underline{i}_{j_2}}\right| < |a_{\min}|$ by definition of S_2. Thus we have a contradiction.

Finally, we assume that $\sigma_{\underline{a}}|_{S_{\Lambda,3}}$ is not injective even if $d:=\gcd(a_1,\dots,a_n)>1$, $\gcd\left(\frac{\prod_{1\le i\le n}a_i}{d^n},d\right)=1$ and $\sum\underline{i}_j<\sum\underline{i}_{j+1}$ for $j=1,\dots,k-1$. Let $g^{(3)}_1=\sum g^{(3)}_{1,\underline{i}}\underline{x}^{\underline{i}}$ and $g^{(3)}_2=\sum g^{(3)}_{2,\underline{i}}\underline{x}^{\underline{i}}$ be distinct elements of $S_{\Lambda,3}$ such that $g^{(3)}_1(\underline{a})=g^{(3)}_2(\underline{a})$. Let $j_3:=\min\{j\mid g^{(3)}_{1,\underline{i}_j}\neq g^{(3)}_{2,\underline{i}_j}\}$. Then we have

$$g^{(3)}_1(\underline{a})-g^{(3)}_2(\underline{a})=\sum_{j_3\le j\le k}\left(g^{(3)}_{1,\underline{i}_j}-g^{(3)}_{2,\underline{i}_j}\right)\underline{a}^{\underline{i}_j}=0.$$

It implies

$$\left(g^{(3)}_{1,\underline{i}_{j_3}}-g^{(3)}_{2,\underline{i}_{j_3}}\right)\underline{a}^{\underline{i}_{j_3}}=-\sum_{j_3+1\le j\le k}\left(g^{(3)}_{1,\underline{i}_j}-g^{(3)}_{2,\underline{i}_j}\right)\underline{a}^{\underline{i}_j}.$$

The right hand side of the above equation is divisible by $d^{\sum\underline{i}_{j_3+1}}$. On the other hand, the left hand side is not divisible by $d^{\sum\underline{i}_{j_3+1}}$ because of $\sum\underline{i}_{j_3}<\sum\underline{i}_{j_3+1}$ and $\left|g^{(3)}_{1,\underline{i}_{j_3}}-g^{(3)}_{2,\underline{i}_{j_3}}\right|<d$. This is a contradiction. Thus we have proved this proposition. □

Algorithms

Let Λ, $S_{\Lambda,1}$, $S_{\Lambda,2}$ and $S_{\Lambda,3}$ be as in Proposition 4.2.1. For $g \in S_{\Lambda,i}$, we can recover g from $g(\underline{a})$ by solving the following linear diophantine equation:

$$\sum_{\underline{i}\in\Lambda} g_{\underline{i}}\underline{a}^{\underline{i}} - g(\underline{a}) = 0,$$

where $g_{\underline{i}}$'s are variables. This equation has infinitely many solutions. However, Proposition 4.2.1 implies that we can recover g uniquely if an upper bound of $H(g)$ is given. In the following, we also give another algorithm to recover $g \in S_{\Lambda,i}$ from $g(\underline{a})$.

An algorithm for $S_{\Lambda,1}$: We assume $\underline{a}^{\underline{i}} > 0$ for $\underline{i} \in \Lambda$ and an upper bound of $H(g)$ is given by $M \in \mathbb{Z}$. Let $g = \sum_{\underline{i}\in\Lambda} g_{\underline{i}}\underline{x}^{\underline{i}} \in S_{\Lambda,1}$.

1. First, we find $g_{\underline{i}_k}$ by using the following formula:

$$g_{\underline{i}_k} = \begin{cases} M & \text{if } M\underline{a}^{\underline{i}_k} \le g(\underline{a}), \\ g'_{\underline{i}_k} & \text{if } g'_{\underline{i}_k}\underline{a}^{\underline{i}_k} \le g(\underline{a}) < (g'_{\underline{i}_k}+1)\underline{a}^{\underline{i}_k}. \end{cases}$$

2. For $1 \le j < k$ we find $g_{\underline{i}_j}$ by using the following inductive formula:

$$g_{\underline{i}_j} = \begin{cases} M & \text{if } M\underline{a}^{\underline{i}_j} \le g(\underline{a}) - \sum_{j+1\le\ell\le k} g_{\underline{i}_\ell}\underline{a}^{\underline{i}_\ell}, \\ g'_{\underline{i}_j} & \text{if } g'_{\underline{i}_j}\underline{a}^{\underline{i}_j} \le g(\underline{a}) - \sum_{j+1\le\ell\le k} g_{\underline{i}_\ell}\underline{a}^{\underline{i}_\ell} < (g'_{\underline{i}_j}+1)\underline{a}^{\underline{i}_j}. \end{cases}$$

An algorithm for $S_{\Lambda,2}$: We assume $i_{j,\ell} \le i_{j+1,\ell}$ for $j = 1, \ldots, k-1$ and $\ell = 1, \ldots, n$, where $\underline{i}_j = (i_{j,1}, \ldots, i_{j,n})$ for $j = 1, \ldots, k$. Let $g = \sum_{\underline{i}\in\Lambda} g_{\underline{i}}\underline{x}^{\underline{i}} \in S_{\Lambda,2}$.

1. First, we find $g_{\underline{i}_1}$ by using the following formula:

$$\begin{aligned} g_1 &:= \frac{g(\underline{a})}{\underline{a}^{\underline{i}_1}} = g_{\underline{i}_1} + \sum_{2\le j\le k} g_{\underline{i}_j}\underline{a}^{\underline{i}_j-\underline{i}_1}, \\ g_{\underline{i}_1} &= g_1 \pmod{\underline{a}^{\underline{i}_2-\underline{i}_1}}. \end{aligned}$$

2. For $2 \le j \le k$ we find $g_{\underline{i}_j}$ by using the following inductive formula:

$$\begin{aligned} g_j &:= \frac{g(\underline{a}) - \sum_{1\le\ell\le j-1} g_{\underline{i}_\ell}\underline{a}^{\underline{i}_\ell}}{\underline{a}^{\underline{i}_j}} = g_{\underline{i}_j} + \sum_{j+1\le\ell\le k} g_{\underline{i}_\ell}\underline{a}^{\underline{i}_\ell-\underline{i}_j}, \\ g_{\underline{i}_j} &= g_j \pmod{\underline{a}^{\underline{i}_{j+1}-\underline{i}_j}}. \end{aligned}$$

An algorithm for $S_{\Lambda,3}$: We assume $d := \gcd(a_1, \ldots, a_n) > 1$, $\gcd\left(\frac{\prod_{1\le i\le n} a_i}{d^n}, d\right) = 1$ and $\sum \underline{i}_j < \sum \underline{i}_{j+1}$ for $j = 1, \ldots, k-1$. Let $b_i := \frac{a_i}{d}$ for $i = 1, \ldots, n$ and $\underline{b} := (b_1, \ldots, b_n)$. Let $g = \sum_{\underline{i}\in\Lambda} g_{\underline{i}} \underline{x}^{\underline{i}} \in S_{\Lambda,3}$.

1. First, we find $g_{\underline{i}_1}$ by using the following formula:

$$\begin{aligned} g_1 &:= \frac{g(\underline{a})}{d^{\sum \underline{i}_1}} = g_{\underline{i}_1} \underline{b}^{\underline{i}_1} + \sum_{2\le j\le k} g_{\underline{i}_j} \underline{b}^{\underline{i}_j} d^{\sum(\underline{i}_j - \underline{i}_1)}, \\ g_{\underline{i}_1} &= g_1 \underline{b}^{-\underline{i}_1} \pmod d. \end{aligned}$$

2. For $2 \le j \le k$ we find $g_{\underline{i}_j}$ by using the following inductive formula:

$$\begin{aligned} g_j &:= \frac{g(\underline{a}) - \sum_{1\le \ell\le j-1} g_{\underline{i}_\ell} \underline{a}^{\underline{i}_\ell}}{d^{\sum \underline{i}_j}} = g_{\underline{i}_j} \underline{b}^{\underline{i}_j} + \sum_{j+1\le \ell\le k} g_{\underline{i}_\ell} \underline{b}^{\underline{i}_\ell} d^{\sum(\underline{i}_\ell - \underline{i}_j)}, \\ g_{\underline{i}_j} &= g_j \underline{b}^{-\underline{i}_j} \pmod d. \end{aligned}$$

Proposition 4.2.2. *Let Λ, $S_{\Lambda,1}$, $S_{\Lambda,2}$ and $S_{\Lambda,3}$ be as above. The above algorithms for $S_{\Lambda,i}$ are correct.*

Proof. The correctness of the algorithms for $S_{\Lambda,2}$ and $S_{\Lambda,3}$ are clear. We prove the correctness of the algorithm for $S_{\Lambda,1}$. Let $g = \sum_{\underline{i}\in\Lambda} g_{\underline{i}} \underline{x}^{\underline{i}} \in S_{\Lambda,1}$. We assume $g_{\underline{i}_j} \ne M, g'_{\underline{i}_j}$ for some j. Let $j_1 := \max\{j \mid g_{\underline{i}_j} \ne M, g'_{\underline{i}_j}\}$. We note that $g_{\underline{i}_{j_1}}$ is smaller than M and $g'_{\underline{i}_{j_1}}$. We assume $M\underline{a}^{\underline{i}_{j_1}} \le g(\underline{a}) - \sum_{j_1+1\le j\le k} g_{\underline{i}_j} \underline{a}^{\underline{i}_j}$. Then we have

$$g(\underline{a}) - \sum_{j_1+1\le j\le k} g_{\underline{i}_j} \underline{a}^{\underline{i}_j} - M\underline{a}^{\underline{i}_{j_1}} = (g_{\underline{i}_{j_1}} - M)\underline{a}^{\underline{i}_{j_1}} + \sum_{1\le j\le j_1-1} g_{\underline{i}_j} \underline{a}^{\underline{i}_j} > 0.$$

This is a contradiction because of $\underline{a}^{\underline{i}_{j_1}} > H(g) \sum_{1\le j\le j_1-1} \underline{a}^{\underline{i}_j}$. Similarly, we can derive a contradiction when

$$g'_{\underline{i}_{j_1}} \underline{a}^{\underline{i}_{j_1}} \le g(\underline{a}) - \sum_{j_1+1\le j\le k} g_{\underline{i}_j} \underline{a}^{\underline{i}_j} < (g'_{\underline{i}_{j_1}} + 1)\underline{a}^{\underline{i}_{j_1}}.$$

Thus we have proved this proposition. □

We return to recovering the plaintext m. If we choose $\Lambda_2 = \Lambda_m$ so that $S_{\Lambda_m,i} \ne \emptyset$ and choose $m \in S_{\Lambda_m,i}$ for some i, we can recover m uniquely. However, in many cases $a_{\min}$ and d $(= \gcd(a_1, \ldots, a_n))$ are smaller than $H(X)$. Then any elements of $S_{\Lambda_m,2}$ and $S_{\Lambda_m,3}$ do not satisfy the condition

$H(m) > H(X)$. On the other hand, it is easy to choose Λ_i for $i = 1, 2, 3$ such that $\Lambda_2 \subset \Lambda_1\Lambda_3$ and $\{g \in S_{\Lambda_2,1} \mid H(g) > H(X)\} \neq \emptyset$. For example, choose $\underline{k}_1 := (k_{1,1}, \ldots, k_{1,n}) \in (\mathbb{Z}_{\geq 0})^n$ so that $k_{1,i} > \deg_{x_i} X$ and $|\underline{a}^{\underline{k}_1}| > H(X) + 1$. Let $\underline{k}_2 := (k_{1,1} + 1, \ldots, k_{1,n} + 1)$, $(\Lambda_m =)\Lambda_2 := \{\underline{k}_1, \underline{0}\}$ and $(\Lambda_f =)\Lambda_3 := \{\underline{k}_1, \underline{k}_2, \underline{0}\}$. Then we have the desired subsets of $(\mathbb{Z}_{\geq 0})^n$.

4.2.2 Sizes of the system parameters

For the sizes of n and w, see §4.7.5. We assume 128-bit security. We estimate the size of b to achieve 128-bit security. A typical brute force attack is as follows: One chooses a random vector $(b_1, \ldots, b_{n-1})$ and factorize the polynomial $X(b_1, \ldots, b_{n-1}, x_n)$ in x_n. If $X(b_1, \ldots, b_{n-1}, x_n)$ has a factor of the form $(x_n - b_n)$ for some integer b_n, then $(b_1, \ldots, b_n)$ is a solution to $X = 0$. So we should choose a secret key $\underline{a} = (a_1, \ldots, a_n)$ such that $|a_i|$ is sufficiently large for $i = 1, \ldots, n$ to avoid the brute force attack. Since the number of choices of the vector $(b_1, \ldots, b_{n-1})$ which satisfies $2^{b-1} \leq |b_i| < 2^b$ is $2^{(n-1)(b-1)+1}$ (we may use $b_i < 0$). Thus we should choose the system parameter b so that

$$(n-1)(b-1) + 1 \geq 128, \tag{4.3}$$

to achieve 128-bit security.

4.2.3 Security analysis

The ideal decomposition attack is the attack against ASC. A similar attack is applicable to our cryptosystem 1. So we need to test the effectiveness of this attack. We give an algorithm of the ideal decomposition attack of Level 3.

1. Choose k prime numbers $p_1, \ldots, p_k$ such that $\prod_{1 \leq i \leq k} p_i \geq 2^{\max\{d_{\underline{i}}^{(2)} | \underline{i} \in \Lambda_2\}} > H(m)$. Set $i = 1$.

2. Let $K_i := \mathbb{Z}/p_i\mathbb{Z}$.

3. Compute $Q(x_2, \ldots, x_n) := \mathrm{Res}_{x_1}\left(\overline{F}_1^{(p_i)} - \overline{F}_2^{(p_i)}, \overline{X}^{(p_i)}\right) \in K_i[x_2, \ldots, x_n]$ the resultant of $\overline{F}_1^{(p_i)} - \overline{F}_2^{(p_i)}$ and $\overline{X}^{(p_i)}$ with respect to x_1. (Recall that $\overline{F}_j^{(p_i)} := F_j \pmod{p_i}$).

4. Factor $Q(x_2, \ldots, x_n)$ and let $Q_0(x_2, \ldots, x_n)$ be an irreducible factor of highest degree.

5. Compute a Gröbner basis of the ideal $J := (\overline{F}_1^{(p_i)}+z, \overline{F}_2^{(p_i)}+z, \overline{X}^{(p_i)}, Q_0) \subset K_i[\underline{x}]$ with respect to the graded reverse lexicographical ordering.

6. Using the above Gröbner basis, solve the following linear equation system over K_i to get $\overline{m}^{(p_i)}$:

$$NF_J\left(\sum_{\underline{i}\in\Lambda_m} m'_{\underline{i}}\underline{x}^{\underline{i}} + z\right) = 0,$$

   where $m'_{\underline{i}}$'s are variables. If the system has no solution, then go back to step 4 and choose another factor of Q.

7. If $i < n$, then replace i by $i+1$ and go back to step 2.

8. Recover m from $\overline{m}^{(p_i)}$ by using the Chinese Remainder Theorem.

We give the analogue of Lemma 1.3.1 and Lemma 1.3.2 for our case. Essensially, their proofs are the same as Lemma 7.3.3 and Lemma 7.3.4 in [40], respectively.

Lemma 4.2.3. *Let X, f, F_1 and F_2 be as above. Then for $i = 1, \ldots, n$, we have*

$$\mathrm{Res}_{x_i}(f, X) \mid \mathrm{Res}_{x_i}(F_1 - F_2, X).$$

Lemma 4.2.4. *Let K be either $\mathbb{Q}$ or $\mathbb{Z}/p\mathbb{Z}$ for some prime number p. Let X, f, s_1, s_2, F_1 and F_2 be as above. We consider these polynomials as elements of $K[\underline{x},z]$. Let A_i and B_i be elements of $K[\underline{x}]$ satisfying $A_i f + B_i X = \mathrm{Res}_{x_i}(f, X)$. If the ideal $(A_i, s_1 - s_2, X)$ coincides with $K[\underline{x}]$, then the following equality holds:*

$$(\mathrm{Res}_{x_i}(f, X), X, F_1 + z, F_2 + z) = (m + z, f, X).$$

These lemmas justify taking step 4, step 5 and step 6 in the above algorithm.

Experimental results

To test the effectiveness of the ideal decomposition attack against our cryptosystem, we take following steps.

1. Choose positive integers n, w, b, b_c, d_f and d_m. We assume $n \geq 2$ and $d_m < d_f$.

2. Choose a secret key $\underline{a} := (a_1, \ldots, a_n) \in \mathbb{Z}$ such that $2^{b-1} \leq |a_i| < 2^b$ for $i = 1, \ldots, n$.

3. Let $\Lambda_1 := \{\underline{i} \in (\mathbb{Z}_{\geq 0})^n \mid \sum \underline{i} \leq w\}$.

4. Choose non-zero random integers $c_{\underline{i}}$ for $\underline{i} \in \Lambda_1' := \Lambda_1 \setminus \{\underline{0}\}$ such that $2^{b_c-1} \leq |c_{\underline{i}}| < 2^{b_c}$.

5. Let $c_{\underline{0}} := -\sum_{\underline{i} \in \Lambda_1'} c_{\underline{i}} \underline{a}^{\underline{i}}$. Define
$$X(\underline{x}) := \sum_{\underline{i} \in \Lambda_1} c_{\underline{i}} \underline{x}^{\underline{i}}.$$

6. Let $k_{i,m} = \deg_{x_i} X + d_m$ for $i = 1, \ldots, n$ and $\underline{k}_m := (k_{1,m}, \ldots, k_{n,m})$.

7. Let $\Lambda_2 := \{\underline{x}^{\underline{k}_m}, \underline{0}\}$. If $\{g \in S_{\Lambda_2,1} \mid H(X) < H(g)\} \neq \emptyset$, then construct $m := \sum_{\underline{i} \in \Lambda_2} m_{\underline{i}} \underline{x}^{\underline{i}}$ so that $m \in S_{\Lambda_2,1}$, $m_{\underline{k}_m} = H(m) \leq 2H(X)$ and $H(m) < \underline{a}^{\underline{k}_m}$. Otherwise, go back to the step 1 and choose other parameters.

8. Let $k_{i,f} = \deg_{x_i} X + d_f$ for $i = 1, \ldots, n$ and $\underline{k}_f := (k_{1,f}, \ldots, k_{n,f})$.

9. Let $\Lambda_3 := \{(i_1, \ldots, i_n) \in (\mathbb{Z}_{\geq 0})^n \mid i_j \leq k_{j,f} \ (j = 1, \ldots, n)\}$.

10. Let $f_{\underline{k}_f}$ be a random integer such that $2H(X) < |f_{\underline{k}_f}| \leq 4H(X)$.

11. Choose non-zero random integers $f_{\underline{i}}$ for $\underline{i} \in \Lambda_2 \setminus \{\underline{k}\}$ such that $2H(X) < |f_{\underline{i}}| \leq |f_{\underline{k}_f}|$. Define
$$f(\underline{x}) := \sum_{\underline{i} \in \Lambda_2} f_{\underline{i}} \underline{x}^{\underline{i}}.$$
Note that the above f, m and X satisfy the condition (4.2).

12. Put $F_i := m + s_i f + r_i X$ for $i = 1, 2$, where s_i and r_i are random polynomials in $\mathbb{Z}[\underline{x}]$ such that $\Gamma_{s_i} = \Gamma_X$ and $\Gamma_{r_i} = \Gamma_f$.

13. Choose an integer e. Let $p_1, \ldots, p_k$ be prime numbers such that the following hold:

    (i)
    $$p_1 \geq \begin{cases} \lfloor (2H(X))^{\frac{1}{e}} \rfloor & \text{if } 2H(X) < \underline{a}^{\underline{k}_m}, \\ \lfloor \underline{a}^{\underline{k}_m/e} \rfloor & \text{if } 2H(X) \geq \underline{a}^{\underline{k}_m}. \end{cases}$$

    (ii) If $2H(X) < \underline{a}^{\underline{k}_m}$, then $\prod_{1 \leq i \leq k-1} p_i \leq 2H(X) < \prod_{1 \leq i \leq k} p_i$. Otherwise, $\prod_{1 \leq i \leq k-1} p_i \leq \underline{a}^{\underline{k}_m} < \prod_{1 \leq i \leq k} p_i$.

(iii) For $i = 1, \ldots, k-1$, p_{i+1} is the next prime number of p_i

14. Apply the ideal decomposition attack described above with F_1 and F_2 by using $p_1, \ldots, p_k$.

We use a computer Windows 8.1 Pro 64 bit with Intel(R) Core(TM) i7-3840QM CPU 2.80 GHz, with 8 GB of RAM. We implemented in Magma V2.19-7 ([10]). For each parameter, we experiment 10 times. In Table 4.2.1, we show their averages when $n = 2$. We see that if $n = 2$, then this attack is efficient. On the other hand, experimentally, if $n \geq 3$, then this attack failed in constructing the ideal $\left(\overline{m}^{(p_i)} + z, \overline{f}^{(p_i)}, \overline{X}^{(p_i)}\right)$ with high probability. Moreover, even if this attack succeeded in constructing the ideal $\left(\overline{m}^{(p_i)} + z, \overline{f}^{(p_i)}, \overline{X}^{(p_i)}\right)$, it coincides with the whole ring $\mathbb{F}_{p_i}[\underline{x}, z]$. Then any polynomials having the same form as $\overline{m}^{(p_i)} + z$ are contained in the ideal. So, there is no meaning for this attack.

The next Proposition and Lemma 4.2.4 may show the reason why it happens.

Proposition 4.2.5. *Let K be either $\mathbb{Q}$ or $\mathbb{Z}/p\mathbb{Z}$ for some prime number p. Let X, f, s_1, s_2, F_1 and F_2 be as above. We consider these polynomials as elements of $K[\underline{x},z]$. Let A_i and B_i be the elements of $K[\underline{x}]$ satisfying $A_i f + B_i X = \mathrm{Res}_{x_i}(f, X)$. Let*

$$\begin{aligned} V(A_i, s_1 - s_2, X) &:= \{\underline{c} \in \overline{K}^n \mid A_i(\underline{c}) = s_1(\underline{c}) - s_2(\underline{c}) = X(\underline{c}) = 0\}, \\ V(f) &:= \{\underline{c} \in \overline{K}^n \mid f(\underline{c}) = 0\}. \end{aligned}$$

Assume that $V(A_i, s_1 - s_2, X) \not\subset V(f)$. Then the following equality

$$(\mathrm{Res}_{x_i}(f, X), X, F_1 + z, F_2 + z) = (m + z, f, X),$$

implies $(A_i, s_1 - s_2, X) = \overline{K}[\underline{x}]$, where $\overline{K}$ is the algebraic closure of K.

Proof. The equality $(\mathrm{Res}_{x_i}(f, X), X, F_1 + z, F_2 + z) = (m + z, f, X)$ implies that there are four polynomials g_1, g_2, g_3 and g_4 in $K[\underline{x}, z]$ satisfying

$$\begin{aligned} f &= g_1(A_i f + B_i X) + g_2(F_1 + z) + g_3(F_2 + z) + g_4 X \\ &= (A_i g_1 + s_1 g_2 + s_2 g_3) f + (g_2 + g_3)(m + z) \\ &\quad + (B_i g_1 + r_1 g_2 + r_2 g_3 + g_4) X. \end{aligned} \tag{4.4}$$

We assume $(A_i, s_1 - s_2, X) \neq \overline{K}[\underline{x}]$. Then $V(A_i, s_1 - s_2, X) \not\subset V(f)$ implies that there is at least one element $\underline{c} \in V(A_i, s_1 - s_2, X)$ such that $f(\underline{c}) \neq 0$. By substituting $\underline{c}$ for (4.4) into $\underline{x}$, we have

$$\begin{aligned} f(\underline{c}) &= s_1(\underline{c})(g_2(\underline{c}, z) + g_3(\underline{c}, z)) f(\underline{c}) + (g_2(\underline{c}, z) + g_3(\underline{c}, z))(m(\underline{c}) + z) \\ &= (g_2(\underline{c}, z) + g_3(\underline{c}, z))(m(\underline{c}) + s_1(\underline{c}) f(\underline{c}) + z). \end{aligned}$$

It implies that $m(\underline{c}) + s_1(\underline{c})f(\underline{c}) + z$ divides $f(\underline{c})$. This is a contradiction because $\deg_z(m(\underline{c})+s_1(\underline{c})f(\underline{c})+z) = 1$. Thus we have proved this proposition. □

In order to improve this attack, we consider the polynomial ring $\mathbb{F}_p(x_{i_1})[x_{i_2}, x_{i_3}]$ or $(\mathbb{F}_p[x_{i_1}]/(P))[x_{i_2}, x_{i_3}]$ for some irreducible polynomial $P(x_{i_1}) \in \mathbb{F}_p[x_{i_1}]$ ($\{i_1, i_2, i_3\} = \{1, 2, 3\}$) as in the Level 2 or the Level 3 attack described in §1.3.7 when $n = 3$. Then this attack works well. In Table 4.2.2, we show the experimental results when $n = 3$ and we use the ring $(\mathbb{F}_p[x_1]/(P(x_1)))[x_2, x_3]$ for some irreducible polynomial $P \in \mathbb{F}_p[x_1]$. When $w = 3$, 5, we use a random irreducible polynomial of degree 5, two random irreducible polynomials of degree 3 and degree 4, respectively. Note that in this case, we compute $\mathrm{Res}_{x_2}(\overline{F}_1^{(p_i)} - \overline{F}_2^{(p_i)}, \overline{X}^{(p_i)})$ instead of $\mathrm{Res}_{x_1}(\overline{F}_1^{(p_i)} - \overline{F}_2^{(p_i)}, \overline{X}^{(p_i)})$ in the step 3 of the ideal decomposition attack algorithm.

Table 4.2.1: The effectiveness of the ideal decomposition attack for 2 variables

n	w_X	b	b_c	d_f	d_m	Time(sec)
2	5	129	10	2	1	3.114
2	5	129	100	2	1	4.176
2	10	129	100	2	1	243.764

Table 4.2.2: The effectiveness of the ideal decomposition attack for 3 variables

n	w_X	b	b_c	d_f	d_m	Time(sec)
3	3	65	10	2	1	106.1796
3	3	65	100	2	1	152.8076
3	5	65	10	2	1	11740.008
3	5	65	100	2	1	13853.562

4.3 Our cryptosystem 2

To avoid the ideal decomposition attack, our first idea is to transform X and m into another polynomial X' and m', respectively so that if we know a solution $\underline{a}$ to $X = 0$, then we can recover X and m from X' and m', respectively. For example, the sender chooses an integer ℓ and constructs $X'(\underline{x}, t)$ and $m'(\underline{x}, t)$ so that $X = X'(\underline{x}, \ell)$ and $m = m'(\underline{x}, \ell)$ are satisfied. For any polynomial $X_2 \in \mathbb{Z}[\underline{x}]$ the sender computes $F(x_2, \ldots, x_n, t, z) := \mathrm{Res}_{x_1}(X', X_2 - z)$. Then we can get ℓ by factorizing $F(a_2, \ldots, a_n, t, X_2(\underline{a}))$ if we know X_2.

4.3.1 Algorithm

We propose the following cryptosystem based on the above idea.

System parameters

The system parameters in our cryptosystem are as follows:

1. n : the number of variables of the polynomials in our cryptosystem;
2. b : bit length of a secret key;
3. w : the total degree of a public key;
4. Λ_1 : a finite subset of $(\mathbb{Z}_{\geq 0})^n$;
5. $D_1 := \{d_{\underline{i}}^{(1)} \mid \underline{i} \in \Lambda_1\} \subset \mathbb{Z}_{\geq 0}$.

For the size of b, see §4.2.2.

Key generation

1. Secret key
   Choose a vector $\underline{a} = (a_1, \ldots, a_n) \in \mathbb{Z}^n$ such that $2^{b-1} \leq |a_i| < 2^b$ for $i = 1, \ldots, n$. Make them secret.

2. Public key
   For $k = 2, 3$, choose finite subsets $\Lambda_k \subset (\mathbb{Z}_{\geq 0})^n$ and $D_k = \{d_{\underline{i}}^{(k)} \mid \underline{i} \in \Lambda_k\} \subset \mathbb{Z}_{\geq 0}$ such that $\Lambda_2 \subset \Lambda_1\Lambda_3$. Choose an irreducible polynomial $X_1(\underline{x}) \in \mathbb{Z}[\underline{x}]$ such that $X_1(\underline{a}) = 0$ and $\Gamma_{X_1} = \{(\underline{i}, d_{\underline{i}}^{(1)}) \mid \underline{i} \in \Lambda_1\}$. Choose an arbitrary polynomial $X_2(\underline{x}) \in \mathbb{Z}[\underline{x}]$. For $i = 2, 3$, make X_1, X_2, Λ_i and D_i public.

Encryption

Assume that the sender wants to send a polynomial $m(\underline{x}) = \sum_{\underline{i} \in \Lambda_2} m_{\underline{i}} x^{\underline{i}} \in \mathbb{Z}[\underline{x}]$ with $\Gamma_m = \{(\underline{i}, d_{\underline{i}}^{(2)}) \mid \underline{i} \in \Lambda_2\}$.

1. Choose an integer ℓ and construct polynomials $X'(\underline{x}, t)$ and $m'(\underline{x}, t)$ so that $X_1(\underline{x}) = X'(\underline{x}, \ell)$ and $m(\underline{x}) = m'(\underline{x}, \ell)$.

2. Choose a random polynomial $r \in \mathbb{Z}[\underline{x}]$ such that $\Gamma_r = \{(\underline{i}, d_{\underline{i}}^{(3)}) \mid \underline{i} \in \Lambda_3\}$.

3. Put $F_1 := m' + rX'$ and $F_2 := \mathrm{Res}_{x_1}(X', X_2 - z)$, where z is a new variable. Send (F_1, F_2).

We give a method to construct X' by using X_1 and ℓ.

1. Let $X_1 = \sum_{\underline{i} \in \Lambda_{X_1}} c_{\underline{i}} \underline{x}^{\underline{i}}$.
2. For $\underline{i} \in \Lambda_{X_1}$, choose random polynomials $g_{\underline{i}}(t) \in \mathbb{Z}[t]$.
3. Let $c'_{\underline{i}}(t) := g_{\underline{i}}(t - \ell) + c_{\underline{i}}$ for $\underline{i} \in \Lambda_{X_1}$.
4. Define
$$X' := \sum_{\underline{i} \in \Lambda_{X_1}} c'_{\underline{i}}(t) \underline{x}^{\underline{i}}.$$

The step 3 implies $X_1(\underline{x}) = X'(\underline{x}, \ell)$. Similarly, we construct $m'(\underline{x}, t)$.

Decryption

1. Compute $h_2(t) := F_2(\underline{a}, t, X_2(\underline{a}))$.
2. Factorize $h_2(t)$ and find its factors of degree 1; $\{(t - \ell_1), \ldots, (t - \ell_k)\}$. Because $X_1 = 0$ and $X_2 - X_2(\underline{a}) = 0$ have the same solution $\underline{a}$, we have $h_2(\ell) = 0$. So $\ell = \ell_i$ for some i.
3. Compute $h_{1,i} := F(\underline{a}, \ell_i)$ for $i = 1, \ldots, k$.
4. Extract $m(\underline{x})$ from $h_{1,i}$ and assume that we obtain $m'_i(\underline{x})$ for $i = 1, \ldots, k$.
5. We can verify whether $m'_i = m$ or not by a MAC (message authentication code) of m for $i = 1, \ldots, k$. Note that $\ell = \ell_j$ implies $m(\underline{a}) = h_{1,j}$ and $m'_j = m$.

4.3.2 Security analysis

We show that the one-wayness of this cryptosystem can be broken because one can get ℓ without solving $X_1 = 0$. Because X_1 and X_2 are made public, one can compute $F_3 := \mathrm{Res}_{x_1}(X_1, X_2 - z)$. Let $F := F_2 - F_3$. The condition $X_1(\underline{x}) = X'(\underline{x}, \ell)$ implies $F(\underline{x}, \ell, z) = 0$. It means that F is divisible by $(t - \ell)$. So one can find ℓ by factorizing F. If one gets ℓ, one may get m by using the ideal $I := (F_1(\underline{x}, \ell), X_1)$ and the normal form $NF_I(\cdot)$.

Remark 4.3.1. To avoid this attack, let $F_2 := \mathrm{Res}_{x_1}((sX_1)', X_2 - z)$, where $sX_1 = (sX_1)'(\underline{x}, \ell)$ for some $s \in \mathbb{Z}[\underline{x}]$. Then $F := F_2 - F_3$ is not divisible by $(t - \ell)$. However, in this case one can also get ℓ in probabilistic polynomial time. We assume that an upper bound of ℓ is given by L. Let $p_1, \ldots, p_k$ be prime numbers with $\prod_{1 \leq i \leq k} p_i > L$ and $\overline{\ell}_i$ an integer such that $\overline{\ell}_i \leq p_i$ and $\overline{\ell}_i \equiv \ell \pmod{p_i}$ for $i = 1, \ldots, k$. If we find solutions $\underline{a}_i \in \mathbb{Z}^n$ to $X_1 \equiv 0 \pmod{p_i}$ for $i = 1, \ldots, k$, then $sX_1 \equiv 0 \pmod{p_i}$ and $X_2 - X_2(\underline{a}_i) \equiv 0 \pmod{p_i}$ have the common solution $\underline{a}_i$ for $i = 1, \ldots, k$. Thus $\overline{\ell}_i$ is the solution to $h(t) := F_2(\underline{a}_i, t, X_2(\underline{a}_i)) \equiv 0 \pmod{p_i}$. This means that $\overline{h}^{(p_i)}(t)$ is divisible by $(t - \overline{\ell}_i)$. If one can find $\overline{\ell}_i$ for $i = 1, \ldots, k$, then one can get ℓ by the Chinese Remainder Theorem.

4.4 Our cryptosystem 3

Our next cryptosystem is based on the following idea: let X be a polynomial in $\mathbb{Z}[\underline{x}]$ and $\underline{a} \in \mathbb{Z}^n$ a solution to $X = 0$. Let X' and ℓ be as above. The condition $X_1(\underline{x}) = X'(\underline{x}, \ell)$ implies that $X'(\underline{a}, t)$ is divisible by $(t - \ell)$. Let $F := m'(t - \ell) + rX'$. Then $F(\underline{a}, t)$ is divisible by $(t - \ell)$ and so we can get ℓ without using resultants.

4.4.1 Algorithm

We propose the following cryptosystem based on the above idea.

System parameters

The system parameters in our cryptosystem are as follows:

1. n : the number of variables of the polynomials in our cryptosystem;
2. b : bit length of a secret key;
3. w : the total degree of a public key;
4. Λ_1 : a finite subset of $(\mathbb{Z}_{\geq 0})^n$;
5. $D_1 := \{d_{\underline{i}}^{(1)} \mid \underline{i} \in \Lambda_1\} \subset \mathbb{Z}_{\geq 0}$.

For the size of b, see §4.2.2.

Key generation

1. Secret key
   Choose a vector $\underline{a} = (a_1, \ldots, a_n) \in \mathbb{Z}^n$ such that $2^{b-1} \leq |a_i| < 2^b$ for $i = 1, \ldots, n$. Make them secret.

2. Public key
   For $k = 2, 3$, choose finite subsets $\Lambda_k \subset (\mathbb{Z}_{\geq 0})^n$ and $D_k = \{d_{\underline{i}}^{(k)} \mid \underline{i} \in \Lambda_k\} \subset \mathbb{Z}_{\geq 0}$ such that $\Lambda_2 \subset \Lambda_1\Lambda_3$. Choose an irreducible polynomial $X(\underline{x}) \in \mathbb{Z}[\underline{x}]$ such that $X(\underline{a}) = 0$ and $\Gamma_X = \{(\underline{i}, d_{\underline{i}}^{(1)}) \mid \underline{i} \in \Lambda_1\}$. For $i = 2, 3$, make X, Λ_i and D_i public.

Encryption

Assume that the sender wants to send a polynomial $m(\underline{x}) = \sum_{\underline{i} \in \Lambda_2} m_{\underline{i}} \underline{x}^{\underline{i}} \in \mathbb{Z}[\underline{x}]$ with $\Gamma_m = \{(\underline{i}, d_{\underline{i}}^{(2)}) \mid \underline{i} \in \Lambda_2\}$.

1. Choose an integer ℓ and construct polynomials $X'(\underline{x}, t)$ and $m'(\underline{x}, t)$ so that $X_1(\underline{x}) = X'(\underline{x}, \ell)$, $X'(\underline{a}, t)$ is divisible by $(t-\ell)^2$ and $m = m'(\underline{x}, \ell)$.

2. Choose a random polynomial $r = \sum_{\underline{i} \in \Lambda_3} r_{\underline{i}}(t) \underline{x}^{\underline{i}} \in \mathbb{Z}[\underline{x}, t]$ such that $r_{\underline{i}}(t) \neq 0$ and $\deg r_{\underline{i}} = d_{\underline{i}}^{(3)}$ for $\underline{i} \in \Lambda_3$.

3. Put $F := m'(t - \ell) + rX'$. Send F.

We give a method to construct X' by using X_1 and ℓ.

1. Let $X = \sum_{\underline{i} \in \Lambda_{X_1}} c_{\underline{i}} \underline{x}^{\underline{i}}$.

2. For $\underline{i} \in \Lambda_X$, choose random non-zero polynomials $g_{\underline{i}}(t) \in \mathbb{Z}[t]$.

3. Let $c'_{\underline{i}}(t) := g_{\underline{i}}(t - \ell)^2 + c_{\underline{i}}$ for $\underline{i} \in \Lambda_X$.

4. Define
$$X' := \sum_{\underline{i} \in \Lambda_X} c'_{\underline{i}}(t) \underline{x}^{\underline{i}}.$$

The step 3 implies that $X_1(\underline{x}) = X'(\underline{x}, \ell)$ and $X'(\underline{a}, t)$ is divisible by $(t - \ell)^2$. Note that we may construct $m'(\underline{x}, t)$ by the method described in §4.3.1 because the condition that $m'(\underline{a}, t) - m(\underline{a})$ is divisible by $(t - \ell)^2$ is not necessary for this cryptosystem.

Decryption

1. Compute $h(t) := F(\underline{a}, t)$.
2. Factorize $h(t)$ and find its factors of degree 1; $\{(t-\ell_1), \ldots, (t-\ell_k)\}$. Because $X'(\underline{a}, t)$ is divisible by $(t-\ell)^2$, $\ell = \ell_i$ for some i.
3. Let $h_i(t) := h(t)/(t-\ell_i)$ and compute $h_i(\ell_i)$ for $i = 1, \ldots, k$. Note that if $\ell_i = \ell$, then $h_i(\ell_i) = m(\underline{a})$.
4. Extract $m(\underline{x})$ from $h_i(\ell_i)$ and assume that we obtain $m'_i(\underline{x})$ for $i = 1, \ldots, k$.
5. We can verify whether $m'_i = m$ or not by a MAC (message authentication code) of m for $i = 1, \ldots, k$.

4.4.2 Security analysis

We show that the one-wayness of this cryptosystem can be broken because one can get ℓ without solving $X = 0$ and construct an ideal containing $m(t-\ell)$. The construction of X' described in §4.4.1 implies $X' = X + \sum_{\underline{i} \in \Lambda_X} g_{\underline{i}}(t)(t-\ell)^2$. So $NF_{(X)}(X')$ is divisible by $(t-\ell)^2$. Because $NF_{(X)}(m'(t-\ell))$ is also divisible by $(t-\ell)$, one can get ℓ by factorizing $NF_{(X)}(F) = NF_{(X)}(m'(t-\ell)) + NF_{(X)}(rX')$. We show that the ideal $(F, X, (t-\ell)^2)$ contains $m(t-\ell)$. The construction of m' described in §4.3.1 implies $m' = m + m_t(t-\ell)$ for some $m_t \in \mathbb{Z}[\underline{x}, t]$. Thus, we have

$$m(t-\ell) = F - m_t(t-\ell)^2 - r\left(X + \sum_{\underline{i} \in \Lambda_X} g_{\underline{i}}(t)(t-\ell)^2\right) \in (F, X, (t-\ell)^2).$$

So one would get m by solving $NF_{(F,X,(t-\ell)^2)}(\sum_{\underline{i} \in \Lambda_m} m'_{\underline{i}} \underline{x}^{\underline{i}}(t-\ell)) = 0$, where $m'_{\underline{i}}$'s are variables.

4.5 Our cryptosystem 4

4.5.1 Polynomials of degree increasing type

Before we describe our cryptosystem, we define the following notion which is one of our key ideas to construct our cryptosystem.

Definition 4.5.1. Define a map $\sigma : \mathbb{Z}^n \longrightarrow \mathbb{Z}$ by $\underline{i} \mapsto \sum \underline{i}$. A polynomial $X \in \mathbb{Z}[\underline{x}]$ is *of degree increasing type* if $\sigma|_{\Lambda_X}$ is injective. In other words, X

is of *degree increasing type* if and only if for each $k \in \mathbb{Z}$, X has at most one term of degree k.

Example 4.5.2. *If* $X(x,y) := 5x^3y^2 + 12xy^2 + 7xy + 6x + 5$, *then* X *is of degree increasing type.*

Let $X \in \mathbb{Z}[\underline{x}]$ be a polynomial of degree increasing type. Then we can define the total order in Λ_f as follows: for $\underline{i}_1, \underline{i}_2 \in \Lambda_f$, we define $\underline{i}_1 \geq \underline{i}_2$ if $\sum \underline{i}_1 \geq \sum \underline{i}_2$. Since Λ_f is finite, there is a maximal element $\underline{k}$. We call the coefficient of degree $\sum \underline{k}$ of X the leading coefficient of X and denote it by $ld(X)$.

4.5.2 Outline of our cryptosystem

We use an analogous method to ASC. More precisely, we use a polynomial $X(\underline{x}) \in \mathbb{Z}[\underline{x}]$ of degree increasing type and a solution $\underline{a} = (\frac{a_1}{d}, \ldots, \frac{a_n}{d}) \in \mathbb{Q}^n$ to $X = 0$ as a public key and a secret key, respectively. A plaintext is given as a polynomial $m \in \mathbb{Z}[\underline{x}]$. We use the following polynomials in $\mathbb{Z}[\underline{x}]$ as cipher polynomials in our cryptosystem:

$$F_i(\underline{x}) := \tilde{m} + s_i f + r_i X \ (i = 1, 2, 3),$$

where $\tilde{m}$, s_i, f and r_i are polynomials in $\mathbb{Z}[\underline{x}]$ with $\Lambda_X = \Lambda_{\tilde{m}} = \Lambda_f = \Lambda_{s_i} = \Lambda_{r_i}$. The polynomial $\tilde{m}$ is constructed from a plaintext polynomial $m \in \mathbb{Z}[\underline{x}]$ and has large coefficients (see §4.6.2). We need to have $\Lambda_X = \Lambda_{\tilde{m}} = \Lambda_f = \Lambda_{s_i} = \Lambda_{r_i}$ and translate m into $\tilde{m}$ to avoid the ideal decomposition attack and other attacks (see §4.7). It is the lagest difference between ASC and our cryptosystem. Recall that w_X is the total degree of X. We compute $h_i := F_i(\underline{a})$, $H_1 := (F_1(\underline{a}) - F_2(\underline{a}))d^{2w_X}$, $H_2 := (F_1(\underline{a}) - F_3(\underline{a}))d^{2w_X}$ and $g := \gcd(H_1, H_2)$ to get $\tilde{m}(\underline{a})d^{w_X}$. Unlike factorizing a polynomial in $\mathbb{F}_p[t]$, it is hard to factorize integers and so we use three polynomials as cipher polynomials and a GCD computation to get $f(\underline{a})d^{w_X}$. If $g = f(\underline{a})d^{w_X}$ and $g > \tilde{m}(\underline{a})d^{w_X}$, then we can get $\tilde{m}(\underline{a})d^{w_X}$ by computing $H := h_1 d^{2w_X} \pmod{d}$ and $Hd^{-w_X} \pmod{d}$. If we can get $\tilde{m}(\underline{a})d^{w_X}$, then we can recover m by the Recovering Algorithm (RA) described in §4.6.4. In order to use RA, $\tilde{m}$ must be of degree increasing type (see §4.6.4) and for security reasons (§4.7), an X must have the same support as $\tilde{m}$. So we use X which is of degree increasing type.

4.6 Algorithm of our cryptosystem

Now, we describe our cryptosystem.

System parameters

The system parameters in our cryptosystem are as follows:

1. n : the number of variables of the polynomials in our cryptosystem;
2. w : the total degree of a public key;
3. d : a positive integer.

For the size of n and w, see §4.7.5. For the size of d, see §4.7.7.

4.6.1 Key generation

1. Secret key
   Choose a vector $\underline{a} = (a_1, \ldots, a_n) \in \mathbb{Z}^n$ of a suitable size[1] such that $\gcd(a_i, d) = 1$ for $i = 1, \ldots, n$. Make them secret.
2. Public key
   Choose an integer e of a suitable size[2] such that $\gcd(e, \varphi(d)) = 1$. Choose an irreducible polynomial $X(\underline{x}) \in \mathbb{Z}[\underline{x}]$ of degree increasing type such that $X(\underline{a}/d) = 0$ and $\#\Lambda_X \leq w = w_X$. Make e, X (and Λ_X) public.

We give a method to construct a public key X of degree increasing type with $X(\underline{a}/d) = 0$.

1. Choose a finite subset $\Lambda \subset (\mathbb{Z}_{\geq 0})^n$ such that $\#\{\sum \underline{i} \mid \underline{i} \in \Lambda\} = \#\Lambda$.
2. Let $\underline{k} = (k_1, \ldots, k_n)$ be the maximal element of Λ. For $\underline{i} \in \Lambda' := \Lambda \setminus \{\underline{0}, \underline{k}\}$, choose random non-zero integers $c_{\underline{i}}$.
3. Choose $c_{\underline{0}}$ and $c_{\underline{k}}$ so that

$$\frac{c_{\underline{k}}\underline{a}^{\underline{k}} + c_{\underline{0}}d^w}{d^w} = -\frac{\sum_{\underline{i}\in\Lambda'} c_{\underline{i}}\underline{a}^{\underline{i}}d^{w'-\sum \underline{i}}}{d^{w'}},$$

   where $w' = \max\{\sum \underline{i} \mid \underline{i} \in \Lambda'\}$, by solving the linear diophantine equation

$$c_{\underline{k}}\underline{a}^{\underline{k}} + c_{\underline{0}}d^w = -\sum_{\underline{i}\in\Lambda'} c_{\underline{i}}\underline{a}^{\underline{i}}d^{w-\sum \underline{i}}. \tag{4.5}$$

[1]The size of a_i should be $|a_i| \geq \frac{2^{\lceil \frac{128}{n-1} \rceil + 1} d}{\varphi(d)}$ for $i = 1, \ldots, n$, where $\varphi(\cdot)$ is the Euler function. (For the reason of this choice, see §4.8.)

[2]The size of e should be $e \geq 129 + 65w$. (For the reason of this choice, see §4.8.)

4. Define
$$X := \sum_{\underline{i} \in \Lambda} c_{\underline{i}} \underline{x}^{\underline{i}}.$$

The condition on Λ (step 1 above) means that X is of degree increasing type. The equation (4.5) means that $X(\underline{a}/d) = 0$.

Remark 4.6.1. Let $\underline{s} = (s_1, \ldots, s_n) \in \mathbb{Z}^n$ be a vector such that $s_i \neq 0$ for $i = 1, \ldots, n$ and $\gcd(\prod_i a_i, \prod_i s_i) = 1$. It is possible to construct X such that $X(\frac{a_1}{s_1 d}, \ldots, \frac{a_n}{s_n d}) = 0$. In this case, one must search for a secret key in $\mathbb{Q}$ instead of $\mathbb{Z} \subset \mathbb{Q}$. Thus, finding a secret key or a solution to $X = 0$ may be complicated. Let $\underline{k}' = (k'_1, \ldots, k'_n)$ be the maximal element of Λ'. We assume that $k'_i \leq k_i$ for $i = 1, \ldots, n$. In this case we must choose $c_{\underline{0}}$ and $c_{\underline{k}}$ so that

$$\frac{c_{\underline{k}} \underline{a}^{\underline{k}} + c_{\underline{0}} d^w \underline{s}^{\underline{k}}}{d^w \underline{s}^{\underline{k}}} = -\frac{\sum_{\underline{i} \in \Lambda'} c_{\underline{i}} \underline{a}^{\underline{i}} s_1^{k'_1 - i_1} \cdots s_n^{k'_n - i_n} d^{w' - \sum \underline{i}}}{d^{w'} \underline{s}^{\underline{k}'}},$$

where $\underline{i} = (i_1, \ldots, i_n)$, by solving the linear diophantine equation

$$c_{\underline{k}} \underline{a}^{\underline{k}} + c_{\underline{0}} d^w \underline{s}^{\underline{k}} = -\sum_{\underline{i} \in \Lambda'} c_{\underline{i}} \underline{a}^{\underline{i}} s_1^{k_1 - i_1} \cdots s_n^{k_n - i_n} d^{w - \sum \underline{i}}. \tag{4.6}$$

If we define
$$X := \sum_{\underline{i} \in \Lambda} c_{\underline{i}} \underline{x}^{\underline{i}},$$
then the equation (4.6) means that $X(\frac{a_1}{s_1 d}, \ldots, \frac{a_n}{s_n d}) = 0$.

4.6.2 Encryption

Assume that the sender wants to send a polynomial $m(\underline{x}) = \sum_{\underline{i} \in \Lambda_m} m_{\underline{i}} x^{\underline{i}} \in \mathbb{Z}[\underline{x}]$ ($1 < m_{\underline{i}} < d$ and $\gcd(m_{\underline{i}}, d) = 1$) with $\Lambda_m = \Lambda_X$.

1. Choose a positive integer N such that Nd is larger than the absolute value of each coefficient of X. We assume that an upper bound of N is given.

2. Construct a polynomial $\tilde{m}(\underline{x})$ with $\Lambda_{\tilde{m}} = \Lambda_m$ as follows:
Let $\tilde{m}_{\underline{i}}$ be an integer such that $0 < \tilde{m}_{\underline{i}} < Nd$ and $\tilde{m}_{\underline{i}} \equiv m_{\underline{i}}^e \pmod{Nd}$, and put $\tilde{m}(\underline{x}) = \sum_{\underline{i} \in \Lambda_m} \tilde{m}_{\underline{i}} x^{\underline{i}}$.

3. Choose a random polynomial $f \in \mathbb{Z}[\underline{x}]$ with $\Lambda_f = \Lambda_X$ such that $H(\tilde{m}) < ld(f) < Nd$ and $ld(f)$ is relatively prime to d. We also assume that all coefficients of f except $ld(f)$ are also as large as the coefficients of $\tilde{m}$.

4. Choose random polynomials s_i and r_i in $\mathbb{Z}[\underline{x}]$ with $\Gamma_{s_i} = \Gamma_X$ and $\Gamma_{r_i} = \Gamma_f$ for $1 \leq i \leq 3$.

5. Put $F_i := \tilde{m} + s_i f + r_i X$ for $1 \leq i \leq 3$ and send (F_1, F_2, F_3, N).

4.6.3 Decryption

1. Compute $h_i := F_i(\underline{a}/d) = \tilde{m}(\underline{a}/d) + s_i(\underline{a}/d) f(\underline{a}/d)$, $H_1 := (h_1 - h_2)d^{2w_X}$ and $H_2 := (h_1 - h_3)d^{2w_X}$. Note that $H_1, H_2 \in \mathbb{Z}$.

2. Compute $g := \gcd(H_1, H_2) > 0$, the greatest common divisor of H_1 and H_2. If $\gcd(g, d) > 1$ and $\gcd(g, d)^\alpha \mid\mid g$, then we replace g by $\frac{g}{\gcd(g,d)^\alpha}$. Note that if $g = f(\underline{a}/d)d^{w_X}$, then $\gcd(g, d) = 1$ (cf. Remark 4.6.4.3).

3. Compute $H := h_1 d^{2w_X} \pmod{g}$ and $\tilde{\mu} := H d^{-w_X} \pmod{g}$. Note that if $|g| > |\tilde{m}(\underline{a}/d)d^{w_X}|$ and g divides $s_1(\underline{a}/d) f(\underline{a}/d) d^{2w_X}$, then we have

$$\tilde{m}(\underline{a}/d)d^{w_X} = \begin{cases} \tilde{\mu} & \text{if } \tilde{m}(\underline{a}/d)d^{w_X} > 0, \\ \tilde{\mu} - g & \text{if } \tilde{m}(\underline{a}/d)d^{w_X} < 0. \end{cases}$$

   Note that $\tilde{m}(\underline{a}/d)d^{w_X} \neq 0$ (cf. Remark 4.6.4.4).

4. Recover $m(\underline{x})$ from $\tilde{\mu}$ or $\tilde{\mu} - g$ by RA which we will describe below.

4.6.4 Recovering Algorithm

We describe a method to recover $m(\underline{x})$ from $\tilde{\mu}$. Let N, d, e and Λ_X be as above.

Input : $\tilde{\mu}$, N, d, e and Λ_X.
Output : $m'(\underline{x}) \in \mathbb{Z}[\underline{x}]$ or "false".

1. Compute
$$e' := e^{-1} \pmod{\varphi(d)}.$$

2. Let $\underline{k}$ be the maximal element of Λ_X. Compute
$$\begin{aligned} m'_{\underline{k}} &:= (\tilde{\mu}\underline{a}^{-\underline{k}})^{e'} \pmod{d} \ (0 < m'_{\underline{k}} < d), \\ \tilde{m}'_{\underline{k}} &:= (m'_{\underline{k}})^e \pmod{Nd} \ (0 < \tilde{m}'_{\underline{k}} < Nd). \end{aligned}$$

3. If $\Lambda'_X := \Lambda_X \setminus \underline{k} = \emptyset$, then return $m'(\underline{x}) = \sum_{\underline{i} \in \Lambda_X} m'_{\underline{i}} \underline{x}^{\underline{i}}$. Otherwise, let $\underline{k}'$ be the maximal element of Λ'_X. Let $w'_X := \sum \underline{k}'$. Put $\tilde{\mu}' := \frac{\tilde{\mu} - \tilde{m}'_{\underline{k}} \underline{a}^{\underline{k}}}{d^{w_X - w'_X}}$. If $\tilde{\mu}' \in \mathbb{Z}$ then replace $\tilde{\mu}$, $\underline{k}$ and Λ_X by $\tilde{\mu}'$, $\underline{k}'$ and Λ'_X, respectively. Otherwise, return "false".

4. Go back to step 2.

Proposition 4.6.2. *If $\tilde{\mu} = \tilde{m}(\underline{a}/d)d^{w_{\tilde{m}}}$, then RA returns $m(\underline{x})$.*

Proof. We assume that $\tilde{\mu} = \tilde{m}(\underline{a}/d)d^{w_{\tilde{m}}} = ld(\tilde{m})\underline{a}^{\underline{k}} + \sum_{\underline{i}\in\Lambda_X\smallsetminus\{\underline{k}\}} \tilde{m}_{\underline{i}}\underline{a}^{\underline{i}}d^{\sum\underline{k}-\sum\underline{i}}$. Because $\tilde{m}$ is of degree increasing type, we have $\sum\underline{k} - \sum\underline{i} \geq 1$. It implies that

$$\begin{aligned} m'_{\underline{k}} &\equiv ld(\tilde{m})^{e'} \equiv m_{\underline{k}}^{ee'} \equiv m_{\underline{k}} \pmod{d}, \\ \tilde{m}'_{\underline{k}} &\equiv \tilde{m}_{\underline{k}} \pmod{Nd}. \end{aligned}$$

Because $ld(m) < d$, we have

$$\begin{aligned} m_{\underline{k}} &= m'_{\underline{k}}, \\ \tilde{m}_{\underline{k}} &= \tilde{m}'_{\underline{k}}. \end{aligned}$$

Thus, $\tilde{\mu}' = \tilde{m}_{\underline{k}'}\underline{a}^{\underline{k}'} + \sum_{\underline{i}\in\Lambda'_X\smallsetminus\{\underline{k}'\}} \tilde{m}_{\underline{i}}\underline{a}^{\underline{i}}d^{\sum\underline{k}'-\sum\underline{i}}$. Because $\tilde{m}$ is of degree increasing type, we have $\sum\underline{k}' - \sum\underline{i} \geq 1$. It implies that we can get $m_{\underline{k}'}$ as above. Similarly, we can get $m_{\underline{i}}$ for $\underline{i} \in \Lambda_X \smallsetminus \{\underline{k}, \underline{k}'\}$. □

Remark 4.6.3. We give some remarks on our cryptosystem.

1. Suppose we want to construct a polynomial X by the method described in Remark 4.6.1. Let $\underline{s}$ be as in Remark 4.6.1. Then it is necessary to make a little change in the decryption process and assume $\gcd(d, \prod_i s_i) = 1$, because we must compute the inverse of s_i in $(\mathbb{Z}/d\mathbb{Z})^\times$ in the decryption process.

2. If $d = p$ is a prime number, we may choose $e = p$ and $e' = 1$.

4.6.5 Improvement in Recovering Algorithm

In step 2 of the decryption process we can write $g = f(\underline{a}/d)d^{w_X}t$ $(t \in \mathbb{Z})$. If $|t| > 1$, then, in step 3, g may not divide $s_1(\underline{a}/d)f(\underline{a}/d)d^{2w_X}$. If so, both $\tilde{\mu}$ and $\tilde{\mu} - g$ are not equal to $\tilde{m}(\underline{a}/d)d^{w_X}$. Then RA will return "false" with high probability because d is large, $\sharp\Lambda_X \leq w_X$ and hence $w_X - w'_X$ becomes ≥ 2 in the middle of the process of RA. In this case we must take the following steps:

1. If RA returned "false", then we choose a positive integer M and construct the set $F(g, M) := \{x \in \mathbb{Z} \mid 2 \leq x \leq M, x|g\} \subset \mathbb{Z}$.

2. If $F(g, M) \neq \emptyset$, then we choose an element $x \in F(g, M)$ and remove x from $F(g, M)$. Otherwise, go back to step 1 and choose an integer which is larger than M.

3. Compute $g' := \frac{g}{x}$, $H' := h_1 d^{2w_X} \pmod{g'}$ and $\tilde{\mu}' := H' d^{-w_X} \pmod{g'}$ and recover $m(\underline{x})$ from $\tilde{\mu}'$.

4. If RA returned "false" again, then go back to step 2.

We describe the reason why RA return "false" with high probability if we do not get $\tilde{m}(\underline{a}/d)d^{w_X}$. Because $\sharp\Lambda_X = w_X + 1$ implies $w_X - w'_X = 1$, $d^{w_X - w'_X} \mid (M - \tilde{m}'_{\underline{k}}\underline{a}^{-\underline{k}})$ is always satisfied. Thus in this case RA does not return "false" even if we do not get $\tilde{m}(\underline{a}/d)d^{w_X}$. On the other hand if $\sharp\Lambda_X \leq w_X$, then $w_X - w'_X \geq 2$ is satisfied in the middle of the process of RA and then RA returns "false" with high probability, if we do not get $\tilde{m}(\underline{a}/d)d^{w_X}$.

Remark 4.6.4. 1. In step 3 of the decryption process, we require that $|g| > |\tilde{m}(\underline{a}/d)d^{w_X}|$ to get $\tilde{m}(\underline{a}/d)d^{w_X}$. To satisfy this condition we impose the condition of step 3 in the encryption process on $ld(f)$. Note that the fact that X is of degree increasing type also helps to satisfy $|g| > |\tilde{m}(\underline{a}/d)d^{w_X}|$, because $O(f) = O(\underline{x}^{\underline{k}}) = O(\tilde{m})$ as $x_1, \ldots, x_n \to \infty$ ($\sum \underline{k} = w_X$), if X is of degree increasing type. Thus, if $f_{\underline{k}} > \tilde{m}_{\underline{k}}$ and $|a_1|, \ldots, |a_n| \gg d$, then $|f(\underline{a}/d)d^{w_X}| > |\tilde{m}(\underline{a}/d)d^{w_X}|$ is satisfied with high probability because $|\frac{a_1}{d}|, \ldots, |\frac{a_n}{d}| \gg 1$. We also note that we can estimate whether $\tilde{m}(\underline{a}/d)d^{w_X} > 0$ or not by the same reason with high probability.

2. If $|a_1|, \ldots, |a_n| \approx d$ or $|a_1|, \ldots, |a_n| \ll d$, then the argument in Remark 4.6.4.1 is not correct because $|\frac{a_1}{d}|, \ldots, |\frac{a_n}{d}| \approx 1$ or $|\frac{a_1}{d}|, \ldots, |\frac{a_n}{d}| \ll 1$. So in this case $\underline{a}$ and f should be chosen so that $a_1, \ldots, a_n > 0$ and, for each $\underline{i} \in \Lambda_f$, the absolute value of the $\underline{i}$-th coefficient of f is larger than that of the monomial $\underline{x}^{\underline{i}}$ of $\tilde{m}$ to satisfy $|f(\underline{a}/d)d^{w_X}| > |\tilde{m}(\underline{a}/d)d^{w_X}|$.

3. We need to have $\gcd(f(\underline{a}/d)d^{w_X}, d) = 1$ to compute the inverse element of $d \pmod{g}$. We show that this condition is satisfied. Let $\underline{k}$ be the maximal element of Λ_f. It follows from the expression

$$f(\underline{a}/d)d^{w_X} = f_{\underline{k}}\underline{a}^{\underline{k}} + \sum_{\underline{i} \in \Lambda_f \setminus \{\underline{k}\}} f_{\underline{i}}\underline{a}^{\underline{i}} d^{w_X - \sum \underline{i}},$$

that if $\gcd(f(\underline{a}/d)d^{w_X}, d) = d' > 1$, then $f_{\underline{k}}$ is divisible by d' because $\gcd(\underline{a}^{\underline{k}}, d) = 1$ is satisfied, and $\sum_{\underline{i} \in \Lambda_f \setminus \{\underline{k}\}} f_{\underline{i}}\underline{a}^{\underline{i}} d^{w_X - \sum \underline{i}}$ is divisible by d. This contradicts our assumption because we assume $\gcd(f_{\underline{k}}, d) = 1$ in step 3 of the encryption process.

4. We also need to have $\tilde{m}(\underline{a}/d)d^{w_X} \neq 0$ to recover m. We show that this condition is satisfied. Let $\underline{k}$ be as above. It follows from the expression

$$\tilde{m}(\underline{a}/d)d^{w_X} = \tilde{m}_{\underline{k}}\underline{a}^{\underline{k}} + \sum_{\underline{i} \in \Lambda_{\tilde{m}} \setminus \{\underline{k}\}} \tilde{m}_{\underline{i}}\underline{a}^{\underline{i}} d^{w_X - \sum \underline{i}},$$

that if $\tilde{m}(\underline{a}/d)d^{w_X} = 0$, then $\tilde{m}_{\underline{k}}$ is divisible by d. This is a contradiction because $\gcd(m_{\underline{k}}, d) = 1$ implies $\gcd(\tilde{m}_{\underline{k}}, d) = 1$.

5. Experimentally, we can expect that t is small. So we can get $\tilde{m}(\underline{a}/d)d^{w_X}$ in practical time.

4.7 Security analysis

In this section although we have not been able to give a security proof, we analyze the effectiveness of some possible attacks for the one-wayness of our cryptosystem. We also discuss the sizes of d, e and N to achieve 128-bit security. First, we note that the attacks against ASC described in §1.3.7 are applicable also to our cryptosystem.

4.7.1 Reduction to solving a multivariate equation system I

Let

$$\begin{aligned} f'(\underline{x}) &= \sum_{\underline{i}\in\Lambda_f} f'_{\underline{i}}\underline{x}^{\underline{i}}, \\ s'(\underline{x}) &= \sum_{\underline{i}\in\Lambda_{s_1}} s'_{\underline{i}}\underline{x}^{\underline{i}}, \\ r'(\underline{x}) &= \sum_{\underline{i}\in\Lambda_{r_1}} r'_{\underline{i}}\underline{x}^{\underline{i}}, \end{aligned}$$

where $f'_{\underline{i}}$, $s'_{\underline{i}}$ and $r'_{\underline{i}}$ are variables. One may be able to get f by solving the following quadratic equation system

$$F_1 - F_2 = (s_1 - s_2)f + (r_1 - r_2)X = s'f' + r'X. \tag{4.7}$$

The number of variables of the system is smaller than that of the system in §1.3.7, but experimentally a Gröbner basis of the ideal generated by the coefficients of $F_1 - F_2 - (s'f' + r'X)$ consists of quadratic polynomials and there is no known general algorithm to solve a multivariate quadratic equation system over $\mathbb{Z}$ or $\mathbb{Q}$ in polynomial time. So solving the system would not be easy. Moreover, if $\Lambda_{s_1} = \Lambda_f = \Lambda_{r_1} = \Lambda_X$, then the equalities

$$\begin{aligned} s'f' + r'X &= s'(f' + tX) + (r' - ts')X \\ &= (s' + sX)f' + (r' - sf')X, \end{aligned}$$

where s and t are any integers, show that there are many solutions of the system (4.7). So we may avoid this attack.

4.7.2 Reduction to solving a multivariate equation system II

Let $f'(\underline{x}) = \sum_{\underline{i}\in\Lambda_f} f'_{\underline{i}}\underline{x}^{\underline{i}}$, $s'(\underline{x}) = \sum_{\underline{i}\in\Lambda_{s_1}} s'_{\underline{i}}\underline{x}^{\underline{i}}$ and $r'(\underline{x}) = \sum_{\underline{i}\in\Lambda_{r_1}} r'_{\underline{i}}\underline{x}^{\underline{i}}$ be as in §4.7.1. Let

$$\begin{aligned} \tilde{m}'(\underline{x}) &:= \sum_{\underline{i}\in\Lambda_{\tilde{m}}} \tilde{m}'_{\underline{i}}\underline{x}^{\underline{i}}, \\ F' &:= \tilde{m}' + s'f' + r'X, \end{aligned}$$

where $\tilde{m}'_{\underline{i}}$ are variables. Let $\underline{a}_1 = (a_{11}, \ldots, a_{1n}), \cdots, \underline{a}_\ell = (a_{\ell 1}, \ldots, a_{\ell n}) \in \mathbb{Z}^n$ be n-tuples of integers. Then we have the following multivariate equation system in $f'_{\underline{i}}$, $s'_{\underline{i}}$, $r'_{\underline{i}}$ and $\tilde{m}'_{\underline{i}}$:

$$\begin{cases} G_1(m'_{\mathrm{c}\underline{0}}, \cdots, r'_{\underline{k}}) := F'(\underline{a}_1) - F_1(\underline{a}_1) = 0 \\ \vdots \\ G_\ell(m'_{\mathrm{c}\underline{0}}, \cdots, r'_{\underline{k}}) := F'(\underline{a}_\ell) - F_1(\underline{a}_\ell) = 0. \end{cases} \tag{4.8}$$

One of the methods of solving (4.8) is to use the Gröbner basis technique. However, if $\{g_1, \cdots, g_h\}$ is a Gröbner basis of the ideal $(G_1, \cdots, G_\ell)$, experimentally, g_i is a cubic or a quadratic polynomial with rational coefficients having large denominators and numerators. Thus, as mentioned in §4.7.1, it would not be easy to solve (4.8). Moreover, for any integers s and t we have

$$\begin{aligned} F' &= (\tilde{m}' + ts' + sX) + s'(f' - t) + (r' - s)X \\ &= \tilde{m}' + s'(f' + tX) + (r' - ts')X. \end{aligned}$$

Noting that $\Lambda_X = \Lambda_{\tilde{m}} = \Lambda_f = \Lambda_{s_1}$, $\Gamma_{s_i} = \Gamma_X$ and $\Gamma_{r_i} = \Gamma_f$ for $1 \le i \le 3$, we see that there are many possible solutions of (4.8). Hence, we would conclude that this attack is not efficient if Nd is sufficiently large, say $Nd > 2^{128}H(X)$ (then the number of possible solutions is larger than 2^{128}). Note that it is also possible to compare $F'(\underline{a}_i) - \tilde{m}'(\underline{a}_i)$ and $F_1(\underline{a}_i) - F_2(\underline{a}_i)$ to get f, but it would be hard because of the same reason.

4.7.3 Reduction to solving a multivariate equation system III

The following attack was suggested by Professor Attila Pethő. Let f', s', r' and $\tilde{m}'$ be as in §4.7.2. Let $S := \sum_{\underline{i}\in\Lambda_{f's'}} S_{\underline{i}}\underline{x}^{\underline{i}}$ and define

$$F'' := \tilde{m}' + S + r'X,$$

where $S_{\underline{i}}$'s are variables. Then one can apply the similar attack in §4.7.2 to F''. However, we would also conclude that this attack is not efficient if Nd is sufficiently large, say $Nd > 2^{128}H(X)$. To see this, let $r \in \mathbb{Z}[\underline{x}]$ be a random polynomial with $\Lambda_r = \Lambda_X$. Then we have

$$\begin{aligned} F'' &= \tilde{m}' + (S + rX) + (r' - r)X \\ &= (\tilde{m}' - r) + (S + r) + r'X. \end{aligned}$$

It implies that the number of possible solutions is larger than 2^{128}. Note that $S + rX$ has the same form as S, and $r' - r$, $\tilde{m}' - r$ and $S + r$ have the same form as r', $\tilde{m}'$ and S, respectively.

4.7.4 Reduction by X

Since X is made public, one can try to divide $F_1 - F_2$ by X to find f in the remainder. But f does not appear in the remainder if $\Lambda_f = \Lambda_X$ and the absolute values of coefficients of f are larger than those of X. So this attack would not be effective.

4.7.5 Rational point attack (solving $X = 0$)

This attack is equivalent to solving the diophantine equation $X(\underline{x}) = 0$. Although it is hard in general as mentioned in introduction, one may wonder if the diophantine equation $X(\underline{x}) = 0$ may be solvable for X of degree increasing type. However, we think that in general, using polynomials of degree increasing type does not affect the security of our cryptosystem. For instance, in [36], it was proved that the problem for determining whether there are positive integer solutions for

$$ax_1^2 + bx_2 - c = 0,$$

where a, b and c are positive integers, is NP-complete. Moreover, we can prove the following theorem.

Theorem 4.7.1. *There is no general method to solve an arbitrary diophantine equation of degree increasing type in $\mathbb{Z}$.*

Proof. Let $T \in \mathbb{Z}[\underline{x}]$ be an arbitrary polynomial. We claim that by making a change of variables $x_i \mapsto x_i^{q_i}$ with suitable q_i's, we can make $T(x_1^{q_1}, \ldots, x_n^{q_n})$ of degree increasing type. We prove this claim by induction on $n \geq 2$. First, we assume $n = 2$. Let q_1 and q_2 be positive integers which are relatively prime to each other. We assume that $q_1 > \max\{|i_2 - j_2| \mid (i_1, i_2), (j_1, j_2) \in \Lambda_T\}$ and

$q_2 > \max\{|i_1 - j_1| \mid (i_1, i_2), (j_1, j_2) \in \Lambda_T\}$. For (i_1, j_1), $(i_2, j_2) \in \Lambda_T$, if $i_1q_1 + i_2q_2 = j_1q_1 + j_2q_2$, then $(i_1, j_1) = (i_2, j_2)$ because of the above assumptions on q_1 and q_2. Thus $T(x_1^{q_1}, x_2^{q_2})$ is of degree increasing type. Next, we assume that our claim is true for $n - 1 \geq 1$. By our inductive hypothesis, there are positive integers $q_1, \ldots, q_{n-1}$ such that $T(x_1^{q_1}, \ldots, x_{n-1}^{q_{n-1}}, 1)$ is of degree increasing type. Let q_n be a positive integer. We assume that

$$q_n > \max\{|(i_1 - j_1)q_1 + \cdots + (i_{n-1} - j_{n-1})q_{n-1}| \mid (i_1, \ldots, i_n), (j_1, \ldots, j_n) \in \Lambda_T\}.$$

For $(i_1, \ldots, i_n)$, $(j_1, \ldots, j_n) \in \Lambda_T$, we assume that the following equality holds:

$$i_1q_1 + \cdots + i_nq_n = j_1q_1 + \cdots + j_nq_n.$$

It implies that $(i_1 - j_1)q_1 + \cdots + (i_{n-1} - j_{n-1})q_{n-1} = (j_n - i_n)q_n$. Since $T(x_1^{q_1}, \ldots, x_{n-1}^{q_{n-1}}, 1)$ is of degree increasing type, if $i_n = j_n$ then $(i_1, \ldots, i_n) = (j_1, \ldots, j_n)$. On the other hand, if $i_n \neq j_n$ then this is a contradiction to the assumption on q_n. Thus $T(x_1^{q_1}, \ldots, x_n^{q_n})$ is of degree increasing type. So we have proved the claim.

Now we prove this theorem. Let $q_1, \ldots, q_n$ be positive integers such that $T^{\text{inc}} := T(x_1^{q_1}, \ldots, x_n^{q_n})$ is of degree increasing type. We assume that there is a general method to solve $T^{\text{inc}} = 0$ in $\mathbb{Z}$. Let $(b_1, \ldots, b_n)$ be its solution. Then $(b_1^{q_1}, \ldots, b_n^{q_n})$ is an integral solution to $T = 0$. Thus if there is a method to solve an arbitrary diophantine equation of degree increasing type, then it can solve an arbitrary diophantine equation, which contradicts Matijasevič's result ([15]). So we have proved this theorem. □

Next, we discuss more general diophantine problems. If one can find a vector $\underline{a}$ such that $X(\underline{a}/d) = 0$, then one can get m by the same process of decryption. The solution $\underline{a}/d$ is not an integral solution but a rational solution. (Using rational solutions is suggested by Professor Noriko Hirata-Kohno.) However, finding such rational solutions is equivalent to finding integral solutions of $G(\underline{x}) := X(\underline{x}/d)d^{w_X} = 0$. (If we do not know the denominator d, finding rational solutions of $G(\underline{x}) = 0$ is reduced to finding integer solutions of the equation $G(\frac{x_1}{z}, \ldots, \frac{x_n}{z})z^{w_X} = 0$ in $n + 1$ variables.) If $n = 2$ and $G(\underline{x}) = 0$ defines a curve of genus 0, 1 or a hyperelliptic curve, then there are explicit algorithms to find all integral solutions ([48], [42], [11]). Otherwise, in special cases there are some algorithms to find all integral points ([8], [9]). Moreover, it is believed that in many cases, diophantine equations with two variables are solvable. Theoretically, using Baker's method and its improvements, explicit upper bounds of the size of solutions to special equations with two variables are known. (see [27] and the references given there). Note that if solutions of a diophantine equation

are sufficiently large, then Baker's method is not practical in general, but we want to use a solution which is as small as possible. However, no efficient methods are known to find integral solutions of diophantine equations of n variables with $n \geq 3$. So we should use a diophatine equations with at least 3 variables as a public key of our cryptosystem. Note that in case of 3 variables, our experience in arithmetic geometry suggests to use X of degree at least 5, because then the hypersurface in the projective 3-space defined by (the homogenized form of) X is of general type if it is non-singular (cf. [29], Example F.5.1.7 and §F.5.2).

4.7.6 Solving $X(\underline{x}/d)d^{w_X} \equiv 0 \pmod{d^{w_X+1}}$

If we use a single cipher polynomial $F := \tilde{m} + rX$, where r is an integer or a polynomial in $\mathbb{Z}[\underline{x}]$ such that rX is of degree increasing type, and $\Lambda_{\tilde{m}} = \Lambda_{rX}$, then it can be broken by finding a solution to the congruence equation

$$X(\underline{x}/d)d^{w_X} \equiv 0 \pmod{d^{w_{rX}+1}}, \tag{4.9}$$

which can be computable in probabilistic polynomial time. Let $\underline{b}$ be a solution of (4.9) and $\underline{k}$ the maximal element of Λ_{rX}. Then the same method as RA is applicable as follows:

$$\begin{aligned} M &:= F(\underline{b}/d)d^{w_{rX}} := \tilde{m}(\underline{b}/d)d^{w_{rX}} + r(\underline{b}/d)X(\underline{b}/d)d^{w_{rX}} \\ &= \tilde{m}(\underline{b}/d)d^{w_{rX}} + r(\underline{b}/d)d^{w_r}X(\underline{b}/d)d^{w_X}, \\ m_{\underline{k}} &= (M\underline{b}^{-\underline{k}})^{e'} \pmod{d}, \\ \tilde{m}_{\underline{k}} &= m_{\underline{k}}^{e} \pmod{Nd}. \end{aligned}$$

Similarly, we can compute the other coefficients of m. However, using cipher polynomials of the form

$$F_i := \tilde{m} + s_i f + r_i X \ (i = 1, 2, 3),$$

we may avoid this weakness because $s_i f$ obstructs to get $\tilde{m}(\underline{b}/d)d^{w_X} \pmod{d}$.

4.7.7 Ideal decomposition attack

By using the resultant as in §1.3.7, it is also possible in our case to reconstruct the ideals $I := (\tilde{m}, f, X) \subset \mathbb{Z}[\underline{x}]$, $J := (\tilde{m} + z, f, X) \subset \mathbb{Q}[\underline{x}, z]$ or $\overline{J}^{(\ell)} := (\overline{\tilde{m}}^{(\ell)} + z, \overline{f}^{(\ell)}, \overline{X}^{(\ell)}) \subset (\mathbb{Z}/\ell\mathbb{Z})[\underline{x}, z]$ from the data (F_1, F_2, X), where z is a new variable and ℓ is a prime number. If one can get $\tilde{m}$, then one can get

m. A simple method to avoid this attack is to let $\Lambda_{\tilde{m}} = \Lambda_f = \Lambda_X$ and the coefficients of $\tilde{m}$ be larger than $H(X)$. Then $\tilde{m}$ cannot be determined uniquely because $\tilde{m}' + z \in J$ implies $\tilde{m}' + z + sX + tf \in J$ for any s, $t \in \mathbb{Z}$ (note that $\Lambda_{\tilde{m}} = \Lambda_f = \Lambda_X$). However, in general, we cannot determine $\tilde{m}$ from $\tilde{m}(\underline{a}/d)d^{w_X}$ uniquely even if we know the secret key $\underline{a}$. This reason is as follows: for any $t \in \mathbb{Z}$, $\tilde{m}(\underline{x})$ and $\tilde{m}(\underline{x}) + tX(\underline{x})$ have the same value at $\underline{a}/d$. So, we use modular exponentiation to transform m into $\tilde{m}$ and use Euler's theorem as in the RSA cryptosystem to recover m from $\tilde{m}(\underline{a}/d)d^{w_X}$ in RA. This is the main idea to avoid this attack.

Now, we analyze the effectiveness of the ideal decomposition attack in detail. Note that as mentioned in §4.2.3, this attack only works well when $n = 2$. However, experimentally, this attack sometimes succeeds in constructing the proper ideal containing $\tilde{m}$ for $n \geq 3$. So, we need to discuss the effectiveness of this attack. We analyze only the Level 2 and the Level 3 attacks because, experimentally, the Level 1 attack is not efficient. First, we analyze the effectiveness of the ideal decomposition attack of Level 2 (see [23], §3.2), which uses the ideal decomposition

$$\begin{aligned}(F_1 - F_2, X) &= ((s_1 - s_2)f, X) = I_1 \cap I_2 \subset \mathbb{Q}[\underline{x}], \\ &\quad (f, X) \subset I_1,\end{aligned}$$

to reconstruct an ideal $J \subset \mathbb{Q}[\underline{x}, z]$ which coincides with $(\tilde{m} + z, f, X)$ from the data (F_1, F_2, X). To get $\tilde{m}$, we use the fact that if a Gröbner basis of J is computed, then $\tilde{m}' + z \in J$ if and only if $NF_J(\tilde{m}' + z) = 0$ (see §1.3.7 for more detail). But, if $\tilde{m}' + z \in J$, then for any integers s and t, $\tilde{m}' + z + sX + tf \in J$ is also satisfied. If the number of choices of the pairs $(s, t) \in \mathbb{Z}^2$ is larger than 2^{128}, we may avoid this attack. All coefficients of $\tilde{m}$ and f are smaller than Nd, but in many cases they are as large as Nd, if $m_{\underline{i}}^e > Nd$. So the possible choices of t may be only 0 , 1 or 2. But, if $Nd > 2^{128}H(X)$, the number of the possible choices of s may be larger than 2^{128}. So N should be chosen so that $Nd > 2^{128}H(X)$ and e should be so large that $m_{\underline{i}}^e \geq 2^e > Nd$ for $\underline{i} \in \Lambda_m$. In this case, this attack is not assumed to be effective. Note that because the absolute value of coefficients of f are as large as those of $\tilde{m}$, the above argument implies that choosing N satisfying $Nd > 2^{128}H(X)$ may complicate finding f from the ideal J or I_1.

Next, we analyze the effectiveness of the ideal decomposition attack of Level 3 (see [23], §3.3). We assume that d is a prime number. We note that if one got $\overline{\tilde{m}}^{(d)}$, then one can get m. So one does not need to get $\tilde{m}$. It is possible to reconstruct an ideal $\overline{J}^{(d)} \subset (\mathbb{Z}/d\mathbb{Z})[\underline{x}, z]$ which coincides with $(\overline{\tilde{m}}^{(d)} + z, \overline{f}^{(d)}, \overline{X}^{(d)})$ from tha data (F_1, F_2, X) (see the algorithm in 1.3.7). Let $\tilde{m}'(\underline{x}) := \sum_{\underline{i} \in \Lambda_{\tilde{m}}} \tilde{m}'_{\underline{i}} \underline{x}^{\underline{i}}$, where $\tilde{m}'_{\underline{i}}$ are variables for $\underline{i} \in \Lambda_{\tilde{m}}$. Assume that a

Gröbner basis of $\overline{J}^{(d)}$ is computed. Let J be the ideal of $(\mathbb{Z}/d\mathbb{Z})[m'_{\underline{c}0}, \cdots, \tilde{m}'_{\underline{k}}]$ generated by the coefficients of $NF_{\overline{J}^{(d)}}(\tilde{m}'+z)$. Let $\{g_1, \cdots, g_h\}$ be a Gröbner basis of J. Then g_i is linear with respect to its variables for each $1 \leq i \leq h$. So we can use linear algebra techniques to solve $NF_{\overline{J}^{(d)}}(\tilde{m}' + z) = 0$. Let A be the coefficient matrix of the equation system $g_1 = \cdots = g_h = 0$. Let D be the dimension of the kernel of the linear map $\mathbb{F}_d^{\#\Lambda_{\tilde{m}}} \to \mathbb{F}_d^h$ defined by A. Then the number of polynomials in $\overline{J}^{(d)}$ having the same form as $\overline{\tilde{m}}^{(d)} + w$ is d^D. So if $d^D > 2^{128}$, the Level 3 attack is not effective. Experimentally, D is at least 2. Thus, this attack is not assumed to be effective if $d^2 \geq 2^{128}$ ($d \geq 2^{64}$).

Next, we assume that $d = \prod_{1 \leq i \leq k} p_i$ ($k \geq 2$ and p_i are distinct prime numbers for $1 \leq i \leq k$). If one got $\overline{\tilde{m}}^{(p_i)}$ for $1 \leq i \leq k$, then one can get $\overline{\tilde{m}}^{(d)}$ and m by the Chinese Remainder Theorem. However, because of the above argument we may also avoid this attack, if d is sufficiently large, for example $d^2 > 2^{128}$. Note that if $d = \prod_{1 \leq i \leq k} p_i^{e_i}$ and $e_i \geq 2$ for some i, this attack may not be directly applicable, because $\mathbb{Z}/p^{e_i}\mathbb{Z}$ is not a domain if $e_i \geq 2$. But, it is possible to lift a polynomial $\overline{\tilde{m}}^{(p_i)} \in (\mathbb{Z}/p_i\mathbb{Z})[\underline{x}]$ to a polynomial $\overline{\tilde{m}}^{(p_i^{e_i})} \in (\mathbb{Z}/p_i^{e_i}\mathbb{Z})[\underline{x}]$ for $1 \leq i \leq n$. There are $p_i^{e_i-1}$ ways of such a lifting. So we may also avoid this attack, if d is sufficiently large, for example $d \geq 2^{64}$.

4.8 Sizes of keys and cipherpolynomials

In this section we estimate the sizes of keys and cipherpolynomials so that our cryptosystem can be expected to have 128-bit security. First, we estimate the size of a secret key and a public key. A typical brute force attack is as follows: One chooses a random vector $(b_1, \ldots, b_{n-1})$ and factorize the polynomial $X(\frac{b_1}{d}, \ldots, \frac{b_{n-1}}{d}, x_n)$ in x_n. If $X(\frac{b_1}{d}, \ldots, \frac{b_{n-1}}{d}, x_n)$ has a factor of the form $(x_n - \frac{b_n}{d})$ for some integer b_n, then $(\frac{b_1}{d}, \ldots, \frac{b_n}{d})$ is a solution to $X = 0$. If $\gcd(\prod_i b_i, d) = 1$, then using the solution $(\frac{b_1}{d}, \ldots, \frac{b_n}{d})$, one can get m by taking the same steps as the decryption process. So we should choose a secret key $\underline{a} = (a_1, \ldots, a_n)$ such that $|a_i|$ is sufficiently large for $i = 1, \ldots, n$ to avoid the brute force attack. Since the probability that a random integer b is prime to d is $\frac{\varphi(d)}{d}$ ($\varphi(\cdot)$ is the Euler's function), the number of choices of the vector $(b_1, \ldots, b_{n-1})$ which satisfies $\frac{2^{\lceil \frac{128}{n-1} \rceil} d}{\varphi(d)} \leq |b_i| < \frac{2^{\lceil \frac{128}{n-1} \rceil + 1} d}{\varphi(d)}$ and $\gcd(\prod_i b_i, d) = 1$ is at least $2^{\lceil \frac{128}{n-1} \rceil (n-1)} \geq 2^{128}$. Thus we should choose a secret key so that

$$\frac{2^{\lceil \frac{128}{n-1} \rceil} d}{\varphi(d)} \leq |a_i| < \frac{2^{\lceil \frac{128}{n-1} \rceil + 1} d}{\varphi(d)} \tag{4.10}$$

for $i = 1, \ldots, n$. We assume (4.10). Let $\underline{k}$ be an element of Λ_X such that $\underline{k} = w_X$ and Λ'_X be as in §4.6.1. We assume that X is constructed by the method described in §4.6.1. There are infinitely many solutions of (4.5). We claim that we can choose a solution $(c_{\underline{0}}, c_{\underline{k}})$ such that $|c_{\underline{0}}| \leq |\underline{a}^{\underline{k}}|$ and $|c_{\underline{k}}| \leq d^{w_X}$, if the following inequality is satisfied:

$$|\underline{a}^{\underline{k}} d^{w_X}| > \left| \sum_{\underline{i} \in \Lambda'_X} c_{\underline{i}} \underline{a}^{\underline{i}} d^{w_X - \sum \underline{i}} \right|. \tag{4.11}$$

To see this, let $A := \left| \sum_{\underline{i} \in \Lambda'_X} c_{\underline{i}} \underline{a}^{\underline{i}} d^{w_X - \sum \underline{i}} \right|$. If (x_0, y_0) is a solution to

$$|\underline{a}^{\underline{k}}| x + d^{w_X} y = A,$$

then all solutions are given by $(x_0 + k d^{w_X}, y_0 - k \underline{a}^{\underline{k}})$ for $k \in \mathbb{Z}$. Looking at the first lattice point (x, y) on the line $|\underline{a}^{\underline{k}}| x + d^{w_X} y = A$ with $x > 0$, we find a solution (x, y) such that $x \leq d^{w_X}$ and $y \leq \underline{a}^{\underline{k}}$. Thus, we have proved the above claim.

In many cases the minimum size of the solutions of (4.5) satisfies $c_{\underline{0}} \approx \underline{a}^{\underline{k}}$ and $c_{\underline{k}} \approx d^{w_X}$. If the $|c_{\underline{i}}|$'s are so small that (4.11) is satisfied, then we may assume that

$$H(X) = \begin{cases} c_{\underline{0}} \approx \underline{a}^{\underline{k}} < \left(\frac{2^{(\lceil \frac{128}{n-1} \rceil + 1)} d}{\varphi(d)} \right)^{w_X} & \text{if } \underline{a}^{\underline{k}} \gg d^{w_X}, \\ c_{\underline{k}} \approx d^{w_X} & \text{if } \underline{a}^{\underline{k}} \ll d^{w_X}. \end{cases}$$

On the other hand, as mentioned in §4.7.7, N, d and e should be chosen so that $Nd > 2^{128} H(X)$, $d \leq 2^{64}$ and $2^e > Nd$, respectively. We must determine an upper bound of Nd and d to estimate the size of e and $c_{\underline{k}}$, respectively. We assume that $H(X) = c_{\underline{k}}$, $2^{64} \leq d < 2^{65}$ and $2^{128} H(X) \leq 2^{128} d^{w_X} < 2^{128+65 w_X} \leq Nd$. Then $c_{\underline{k}} \leq 2^{65 w_X}$ and $N \geq 2^{128+65(w_X - 1)}$. If we assume that $2^{128+65(w_X-1)} \leq N < 2^{128+65(w_X-1)+1} = 2^{129+65(w_X-1)}$, then we should choose e so that $e \geq 129 + 65 w_X$ because $Nd < 2^{129+65 w_X}$. It remains to estimate the size of $|c_{\underline{i}}|$ for $\underline{i} \in \Lambda'_X$. We think that the size of these coefficients may be small enough to keep the size of the public key reasonable even though we cannot prove it. For example, if $|c_{\underline{i}}| < 2^{10}$, then the size of X, that is $\sum_{\underline{i} \in \Lambda_X}$(bit length of $c_{\underline{i}}$), is at most $(\lceil \frac{128}{n-1} \rceil + 1 + \lceil \log_2 d - \log_2 \varphi(d) \rceil) w_X + 65 w_X + 10(\#\Lambda_X - 2) = (\lceil \frac{128}{n-1} \rceil + 66 + \lceil \log_2 d - \log_2 \varphi(d) \rceil) w_X + 10 \#\Lambda'_X$ bits under the above assumptions. If $w_X \approx \#\Lambda_X = \Lambda'_X + 2$, then the size of X $\approx (\lceil \frac{128}{n-1} \rceil + 76 + \lceil \log_2 d - \log_2 \varphi(d) \rceil) w_X$ bits. Then the size of the secret key and the public key is at most $(\lceil \frac{128}{n-1} \rceil + 1) n + \lceil \log_2 d - \log_2 \varphi(d) \rceil$ bits and $(\lceil \frac{128}{n-1} \rceil + 76 + \lceil \log_2 d - \log_2 \varphi(d) \rceil) w_X + 65 + \lceil \log_2 e \rceil$ bits, respectively.

Next, we estimate the size of F_i for $i = 1, 2, 3$. We may assume that the size of F_i is about the same as that of $2s_i f$ because $\Gamma_f = \Gamma_{r_i}$ and $\Gamma_{s_i} = \Gamma_X$. Since $\Lambda_X = \Lambda_f = \Lambda_{s_i}$, $\#\Lambda_X \leq w_X$, $H(f) < Nd < 2^{129+65w_X}$ and $H(s_i) \approx H(X) < 2^{65w_X}$, we have

$$H(s_i f) \leq \#\Lambda_X H(f) H(s_i) < 2^{129+130w_X} w_X.$$

It implies that the size of $2s_i f$ is at most $(130 + 130w_X + \lceil \log_2 w_X \rceil)\#\Lambda_{s_i f}$ bits. So, it is important to estimate $\#\Lambda_{s_i f}$, explicitly. We assume $\Lambda_f = \Lambda_{s_i} = \{\underline{k}_1, \ldots, \underline{k}_{\#\Lambda_f}\}$. Then we can write

$$\begin{aligned} s_i f &= \left(\sum_{\underline{j} \in \Lambda_{s_i}} \underline{s}_{\underline{j}}^{(i)} \underline{x}^{\underline{j}} \right) \left(\sum_{\underline{j} \in \Lambda_f} \underline{f}_{\underline{j}} \underline{x}^{\underline{j}} \right) \\ &= \sum_j \underline{s}_{\underline{k}_j}^{(i)} \underline{f}_{\underline{k}_j} \underline{x}^{2\underline{k}_j} + \sum_{j \neq h} \left(\underline{s}_{\underline{k}_j}^{(i)} \underline{f}_{\underline{k}_h} + \underline{s}_{\underline{k}_h}^{(i)} \underline{f}_{\underline{k}_j} \right) \underline{x}^{\underline{k}_j + \underline{k}_h}. \end{aligned}$$

It implies that

$$\#\Lambda_{s_i f} \leq \frac{\#\Lambda_f^2 - \#\Lambda_f}{2} + \#\Lambda_f \leq \frac{w_X^2 - w_X}{2} + w_X.$$

Thus, the size of $2s_i f$ is at most

$$\begin{aligned} &(\frac{w_X^2 - w_X}{2} + w_X)(130 + 130w_X + \lceil \log_2 w_X \rceil) \\ &= \frac{1}{2}(w_X^2 + w_X)(129 + 130w_X + \lceil \log_2 w_X \rceil) \end{aligned}$$

bits. Since $2^{128+65(w_X-1)} \leq N < 2^{129+65(w_X-1)}$, we conclude that the size of ciphertext is at most

$$\frac{3}{2}(w_X^2 + w_X)(129 + 130w_X + \lceil \log_2 w_X \rceil) + 129 + 65(w_X - 1)$$

bits.

4.9 Examples

In Table 1 and Table 2 we give examples of the size of keys and ciphertexts. In Table 3 we also give examples of the time which it took to encrypt and decrypt. We use a computer Windows 8.1 Pro 64 bit with Intel(R) Core(TM) i7-3840QM CPU 2.80 GHz, with 8 GB of RAM. We implemented in Magma V2.19-7 ([10]) and the source code of our cryptosystem (file name: `crypto-okumura.txt`) is available at
`http://www2.math.kyushu-u.ac.jp/ ~ s-okumura/`.

Table 4.9.1: Size of keys of our cryptosystem.

No.	n	w_X	$\#\Lambda_X$	secret key (bit)	public key (bit)
1	3	5	4	198	739
2	3	5	5	198	747
3	3	7	4	198	1000
4	3	7	7	198	1031
5	3	10	4	198	1393
6	3	10	7	198	1420
7	3	10	10	198	1450

Table 4.9.2: Size of ciphertext of our cryptosystem.

No.	n	w_X	$\#\Lambda_X$	F_1 (bit)	F_2 (bit)	F_3 (bit)	N (bit)
1	3	5	4	7442	7443	7440	387
2	3	5	5	10755	10748	10752	390
3	3	7	4	9946	9942	9947	521
4	3	7	7	23907	23915	23917	515
5	3	10	4	13685	13684	13688	717
6	3	10	7	33658	33659	33667	717
7	3	10	10	57740	57749	57767	719

Table 4.9.3: Encryption time and decryption time.

No.	n	w_X	$\#\Lambda_X$	enc. time (ms)	dec. time (ms)
1	3	5	4	39	34
2	3	5	5	38	33
3	3	7	4	38	34
4	3	7	7	38	34
5	3	10	4	39	34
6	3	10	7	39	36
7	3	10	7	40	40

4.10 Conclusion

In this chapter we proposed a new public key cryptosystem based on diophantine equations and analyzed its security. It is a number field analogue of the ASC, incorporating a key idea, to avoid some attacks, of "twisting" the plaintext by using some modular arithmetic and Euler's theorem as in the RSA cryptosystem. Another key idea is to use a polynomial, as the public key, of degree increasing type to recover the plaintext.

Bibliography

[1] K. Akiyama, Y. Goto and H. Miyake, *An Algebraic Surface Cryptosystem*, In Proc. of PKC'09, Vol. 5443 (2009), 425–442.

[2] A. Baker, *Transcendental Number Theory*, Cambridge Univ. Press, 1975.

[3] M. Bardet, *Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie*, PhD thesis, Université Paris VI, Décembre 2004.

[4] M. Bardet, J.-C. Faugére, B. Salvy, and B. Y. Yang, *Asymptotic behaviour of the degree of regularity of semi-regular polynomial systems*, Proce. of MEGA 2005, 2005.

[5] D. Bayer and M. Stillman, *Computation of Hilbert Functions*, J. Symbolic Comp. **14**, (1992), no. 1, 31–50.

[6] T. Becker and V. Weispfenning, *Gröbner Bases: A Computational Approach to Commutative Algebra*, Graduate Texts in Mathematics, Vol. 141, Springer New York, 1993.

[7] E.R. Berlekamp, *Factoring Polynomials over Large Finite Fields*, Math. of Computation **24** (1970), 713–735.

[8] F. Beukers and S. Tengely, *An implementation of Runge's method for Diophantine equations*, available at arXiv:math/0512418.

[9] Y. Bilu, *Effective analysis of integral points on algebraic curves*, Israel J. Math. **90** (1995), 235–252.

[10] W. Bosma, J. Cannon, and C. Playoust, *The Magma algebra system. I. The user language*, J. Symbolic Comput. **24** (1997), 235–265.

[11] Y. Bugeaud, M. Mignotte, S. Siksek, M. Stoll and S. Tengely, *Integral points on hyperelliptic curves*, Algebra Number Theory **2** (2008), 859–885.

[12] D.G. Cantor and H. Zassenhaus, *On Algorithms for Factoring Polynomials over Finite Fields*, Math. of Computation **36** (1981), 587–592.

[13] D. Cox, J. Little and D. O'Shea *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*, 3rd., Undergraduate Texts in Mathematics, Springer Verlag 2007.

[14] T. W. Cusick, *Cryptoanalysis of a public key system based on diophantine equations*, Inform. Processing Letters, **56** (1995), 73–75.

[15] M. Davis, Y. Matijasevič and J. Robinson, *Hilbert's tenth problem, Diophantine equations: positive aspects of a negative solution*, in: Mathematical Developments Arising from Hilbert Problems, Ed.: F.E. Browder, Symp. in Pure Math., (1974), AMS, Providence, RI., (1976), pp. 323–378.

[16] W. Diffie and M. Hellman, *New direction in cryptography*,Trans. on Information Theory, **22** (1976), 644–654.

[17] J. Ding, J. E. Gower and D. Schmidt, *Zhuang-Zi: A new algorithm for solving multivariate polynomial equations over a finite field*, In: PQCrypto 2006: International Workshop on Post-Quantum Cryptography, May 23-26. Katholieke Universiteit Leuven, Belgium (2006).

[18] G. Faltings, *Endlichkeitssätze für abelsche Varietäten über Zahlkörpern*, Invent. Math. **73** (1983), 349–366.

[19] G. Faltings, *Diophantine approximation on abelian varieties*, Annals of Math. **133** (1991), 549–576.

[20] G. Faltings, *The general case of Lang's conjecture*, In Symposium in Algebraic geometry, Barsotti, eds., Acad. Press, 1994, 175–182.

[21] J.-C. Faugére, *A new efficient algorithm for computing Gröbner basis (F_4)*, Journal of Pure and Applied Algebra 139, 1-3 (1999), 61–88.

[22] J.-C. Faugére, *A new efficient algorithm for computing Gröbner basis without reduction to zero (F_5)*, In T. Mora, editor, Proceedings of ISSAC, 75–83. ACM Press, July 2002.

[23] J.-C. Faugére and P.-J. Spaenlehauer, *Algebraic Cryptanalysis of the PKC'2009 Algebraic Surface Cryptosystem*, Proc. of PKC'10, Vol. 6056 (2010), 35–52

[24] R. Fröberg, *An introduction to Gröbner bases*, Pure and Applied Mathematics. John Wiley and Sons Ltd., Chichester, 1997.

[25] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman, New York (1979).

[26] M. Giusti, *Some effectivity problems in polynomial ideal theory*, In Proc. Int. Symp. on Symbolic and Algebraic Computation EUROSAM 84, Cambridge (England), Vol. 174 of LNCS, 159–171, (1984) Springer.

[27] K. Győry, *Solving Diophantine equations by Baker's theory*, In A panorama of number theory of the view from Baker's garden (Zürich, 1999), 38–72, Cambridge Univ. Press, 2002.

[28] R. Hartshorne, *Algebraic Geometry*, Graduate Texts in Mathematics Vol. 52 (1977), Springer New York.

[29] M. Hindry and J. H. Silverman *Diophantine Geometry: An Introduction*, Graduate Texts in Mathematics, Vol. 201, Springer New York 2000

[30] N. Hirata-Kohno and A. Pethő, *On a key exchange protocol based on Diophantine equations*, Infocommunications Journal, **5** (2013), 17–21.

[31] N. Koblitz, *Elliptic curve cryptosystems*, Math. of Computation **48** (1987), 203-209.

[32] S. Lang, *Algebra*, 3rd ed., Graduate Texts in Mathematics, Vol. 211, Springer, New York, 2002.

[33] D. Lazard, *Gaussian Elimination and Resolution of Systems of Algebraic Equations*, In Proc. EUROCAL 83, Vol. 162 of LNCS, (1983) 146–157.

[34] A. K. Lenstra and H. W. Lenstra, Jr. (eds.), *The Development of the Number Field Sieve*, Lecture Notes in Mathematics, **1554**, Springer-Verlag, Berlin, 1993.

[35] C. H. Lin, C. C. Chang and R. C. T. Lee, *A New Public-Key Cipher System Based Upon the Diophantine Equations*, IEEE Trans. Comp. **44** (1995), 13–19.

[36] K. Manders and L. Adleman, *NP-complete decision problems for binary quadratics*, J. Comput. System Sci. **16** (1978), no. 2, 168–184.

[37] R. C. Mason, *Diophantine Equations over Function Fields*, London Mathematical Society Lecture Note Series, **96**, Cambridge, England: Cambridge University Press.

[38] V.S. Miller, *Use of elliptic curves in cryptography*, Abstracts for Crypto. '85. Lecture Notes in Computer Science, **218** (1986), 417–426.

[39] S. Mochizuki *Inter-universal Teichmüller Theory I: Construction of Hodge Theaters, II: Hodge-Arakelov-theoretic Evaluation, II I: Canonical Splittings of the Log-theta-lattice, IV: Log-volume Computations and Set-theoretic Foundations*, available in `http://www.kurims.kyoto-u.ac.jp/~motizuki/papers-english.html`.

[40] N. Ogura, *On Multivariate Public-key cryptosystems*, PhD thesis, Tokyo Metropolitan University.

[41] T. Pheidas, *Hilbert's tenth problem for fields of rational functions over finite fields*, Invent. Math. **103** (1991), no. 1, 1–8.

[42] D. Poulakis and E. Voskos, *On the practical solution of genus zero Diophantine equations*, J. Symbolic Comput. **30** (2000), 573-582.

[43] R. L. Rivest, A. Shamir and L. Adleman, *A method for obtaining digital signatures and public key cryptosystems*, Commununications of the ACM, **21** (1987), 120–126.

[44] A. Shamir, J. Patarin, N. Courtois and A. Klimov, *Efficient Algorithms for solving Overdefined Systems of Multivariate Polynomial Equations*, Eurocrypt ' 2000, LNCS 1807, Springer, 392-407.

[45] P. Shor, *Algorithms for Quantum Computation: Discrete Logarithm and Factoring*, Proc. 35th Annual Symposium on Foundations of Computer Science (1994), 124–134 and SIAM J. Comput. **26** (1997), 1484–1509.

[46] J. H. Silverman, *The arithmetic of elliptic curves*, 2nd ed., Graduate Texts in Mathematics, Vol. 106, Springer 2009.

[47] W. W. Stothers, *Polynomial identities and hauptmoduln*, Quart. J. Math. Oxford Ser. (2) **32** (1981), no. 127, 349–370.

[48] R. J. Stroeker and N. Tzanakis, *Computing all integer solutions of a genus 1 equation*, Math. Comp. **72** (2003), 1917–1933.

[49] A. Weil, *Sur les courbes algébriques et les variétés qui s'en déduisent*, Actualités Sci. Ind., no. 1041 = Publ. Inst. Math. Univ. Strasbourg **7** (1945). Hermann et Cie., Paris, 1948. iv+85 pp.

[50] H. Yosh, *The key exchange cryptosystem used with higher order Diophantine equations*, International Journal of Network Security & Its Applications **3** (2011), 43–50.