Indeterminate Equation Public-key Cryptosystem (Giophantus$^{\text{TM}}$) 2017.11.30

Principal submitter

- Koichiro Akiyama, Toshiba Corporation

E-mail address (preferred): **koichiro.akiyama@toshiba.co.jp**
Telephone (if absolutely necessary): +81-44-549-2156
Postal Address (if absolutely necessary):1 Komukai-Toshiba-cho, Saiwai-ku, Kawasaki 212-8582, Japan

Auxiliary submitters:

- Yasuhiro Goto, Hokkaido University of Education at Hakodate
- Shinya Okumura, Osaka University
- Tsuyoshi Takagi, Kyushu University
- Koji Nuida, National Institute of Advanced Industrial Science and Technology
- Goichiro Hanaoka, National Institute of Advanced Industrial Science and Technology
- Hideo Shimizu, Toshiba Corporation
- Yasuhiko Ikematsu, Kyushu University

Inventors/developers: The inventors/developers of this submission are the same as the principal submitter and auxiliary submitters.

Owner: Same as principal submitter.

Signature: See also printed version of "Statement by Each Submitter".

Contents

1 Introduction

This document proposes a post-quantum public-key encryption scheme whose security is based on finding small solutions of indeterminate equations, to which approximation attacking algorithms (e.g., LLL [24] and BKZ [33]) cannot be applied directly when the equations are non-linear. This scheme has been disclosed in [3].

Our scheme was developed from algebraic surface cryptosystems (ASC), which are designed such that the security depends on the intractability of solving some non-linear indeterminate equation [2]. Although ASC can be broken by the ideal decomposition attack proposed in PKC2010 [15], we revise the scheme to be secure against known attacks (including ideal decomposition) by adding a noise term to the cipher polynomial. Our scheme is provably secure in the sense of IND-CPA (see Section 8) under the indeterminate equation of the learning with errors (IE-LWE) assumption, which is a new computational assumption coming from analogy to the LWE assumption. An IND-CCA2 secure scheme is provided by using a well-known conversion technique, such as Fujisaki–Okamoto conversion [14].

We refer to the public key encryption scheme as the **Giophantus$^{\text{TM}}$ encryption scheme**, which comes from the Diophantine equations used as the general term for the indeterminate equations[1]. In addition, the Giophantus encryption scheme has the ring homomorphic property described in Section 13.

Table 1 shows the difference between Giophantus and other post-quantum cryptography (PQC) candidates. In the table 1, "Linearity" indicates the linearity of the underlying problem. Giophantus provides public-key cryptosystem whose security depends on the computational hardness of solving indeterminate equations. Solving non-linear indeterminate equations is a well-known hard problem in general. In particular, it is known that there is no general solution for equations over $\mathbb{Z}$ or $F_q[t]$ and no general algorithm for solving them. Although this encryption scheme employs indeterminate equations over $F_q[t]/(t^n - 1)$, the scheme itself is potentially secure since we are able to apply non-linear equations to the scheme.

Table 1: Comparison with other PQC candidates

Cryptosystem	Underlying problem	Linearity	Provably secure
Code Based	Decoding Problem	Linear+noise	Yes
Lattice Based	Shortest/Closest Vector Problem	Linear+noise	Yes
Multivariate	Solving Multivariate Equations	Non-linear	No
Giophantus (Present)	Solving Indeterminate Equation	Linear/Non-linear +noise	Yes

The organization of this document is as follows.

[1] In an earlier document, we referred to this as the **IEC** (Indeterminate Equation Cryptosystem) **encryption scheme**, but "IEC" may be confused with the standard abbreviation for the International Electrotechnical Commission, and so we adopt "Giophantus" instead.

- Section 1 is an introduction.
- Section 2 defines some notation used in this document.
- Section 3 provides mathematical preliminaries and defines some basic mathematical primitives used in this document.
- Section 4 gives an overview of the proposed cryptographic primitives.
- Section 5 defines the data conversion primitives used for the cryptographic primitives and for the schemes described in Sections 6 and 11.
- Section 6 defines the proposed cryptographic encryption primitives, which satisfy IND-CPA security.
- Section 7 shows the background and the concept of these primitives.
- Section 8 defines the security assumption and describes a proof that the cryptographic primitive is IND-CPA secure under the given assumption.
- Section 9 provides cryptanalysis for known attacks and gives recommended parameters.
- Section 10 shows temporal parameters that satisfy the security level suggested by NIST.
- Section 11 defines a cryptographic scheme that satisfies IND-CCA2 security.
- Section 12 shows the performance of our scheme on our platform.
- Section 13 describes advantages and limitations of the proposed scheme.

2 Notation

The notation in this document includes the following.

M	Plaintext in the set $\{0,1\}^k$, where k is bit length of the plaintexts. The bit length is defined in Section 6.
ℓ	A small integer which is larger than 1
$(m_1 m_2 \cdots m_k)_\ell$	ℓ-ary representation of plaintext M, particularly the case $\ell = 2$, which is binary representation.
q	A prime number much larger than ℓ
F_q	The prime field identified with the set $\{0, \cdots, q-1\}$
x, y, t	Variables used for the cryptographic primitives and scheme
$F_q[t]$	Univariate polynomial ring over F_q
R_q	Quotient ring $F_q[t]/(t^n - 1)$, which is $F_q[t]$ modulo $t^n - 1$, where n is an integer larger than 1
R_ℓ	Subset of the quotient ring R_q, which consists of all univariate polynomials of t up to degree $n-1$ whose coefficients are within the range $\{0, \cdots, \ell - 1\}$
$\mathbb{Z}[t]/(t^n - 1)$	Quotient ring, which is $\mathbb{Z}[t]$ modulo $t^n - 1$ where n is an integer larger than 1
n	Degree of the modulus $t^n - 1$ of the quotient ring R_q
$max(S)$	Maximum value of ordered set S. If $S = \{3, 8, -3, 4, 9\}$, then $max(S) = 9$.
$X(x, y)$	Irreducible bivariate polynomial of x and y over the ring R_q, with $X(x, y)$ an element of $R_q[x, y]$
$X(x, y) = 0$	Indeterminate equation over the ring R_q
$r(x, y)$	Random bivariate polynomial of x and y over the ring R_q, with $r(x, y)$ an element of $R_q[x, y]$
$e(x, y)$	Noise bivariate polynomial of x and y over the ring R_q, with $e(x, y)$ an element of $R_\ell[x, y]$
$m(t)$	Plaintext polynomial that embeds a plaintext M into R_ℓ
$c(x, y)$	Ciphertext polynomial over the ring R_q such that $c(x, y)$ is an element of $R_q[x, y]$
$(u_x(t), u_y(t))$	Small solution of the indeterminate equation $X(x, y) = 0$ over the ring R_q, where $u_x(t)$ and $u_y(t)$ are polynomials of t in R_ℓ and satisfy the relation $X(u_x(t), u_y(t)) = 0$
$a_{ij}(t)$	Coefficient of the monomial $x^i y^j$ belonging to the irreducible bivariate polynomial $X(x, y)$ over the ring R_q, such that $a_{ij}(t)$ is an element of R_q
$r_{ij}(t)$	Coefficient of the monomial $x^i y^j$ belonging to the random bivariate polynomial $r(x, y)$ over the ring R_q, such that $r_{ij}(t)$ is an element of R_q
$e_{ij}(t)$	Coefficient of the monomial $x^i y^j$ belonging to the noise bivariate polynomial $e(x, y)$ over the set R_ℓ, such that $e_{ij}(t)$ is an element of R_ℓ

Γ_X	Support set of the irreducible polynomial $X(x,y)$. Each element is a pair (i,j) of the exponents of $x^i y^j$, which is a non-zero monomial of $X(x,y)$ such that $\Gamma_X = \{(i,j) \in (\mathbb{N} \cup \{0\})^2 \mid a_{ij}(t) \neq 0\}$.
$\#\Gamma_X$	Cardinality of the support set Γ_X
$\mathfrak{F}_{\Gamma_X}/R_q$	Set of bivariate polynomials whose support set is Γ_X over the ring R_q
Γ_r	Support set of the random polynomial $r(x,y)$. Each element is a pair (i,j) of the exponents of a non-trivial monomial $x^i y^j$.
$\#\Gamma_r$	Cardinality of the support set Γ_r
$\mathfrak{F}_{\Gamma_r}/R_q$	Set of bivariate polynomials whose support set is Γ_r over the ring R_q
Γ_e	Support set of the random polynomial $e(x,y)$. Each element is a pair (i,j) of the exponents of a non-trivial monomial $x^i y^j$.
$\#\Gamma_e$	Cardinality of the support set Γ_e
$\mathfrak{F}_{\Gamma_e}/R_\ell$	Set of bivariate polynomials whose support set is Γ_e over the ring R_ℓ
$\mathfrak{X}(\Gamma_X, \ell)/R_q$	Subset of $\mathfrak{F}_{\Gamma_X}/R_q$, consisting of all bivariate polynomials with a small zero point $(u_x(t), u_y(t))$ in R_ℓ
dX	Total degree of irreducible bivariate polynomial $X(x,y)$ such that $dX = max(\{\ i+j \mid X(x,y) = \sum_{(i,j)\in\Gamma_X} a_{ij}(t) x^i y^j\ \})$
dr	Total degree of random bivariate polynomial $r(x,y)$ such that $dr = max(\{\ i+j \mid r(x,y) = \sum_{(i,j)\in\Gamma_r} r_{ij}(t) x^i y^j\ \})$
$\|.\|$	Bit length of an integer, such as $\|5\| = 3$
$a\|\|b$	String concatenation of a and b.

3 Mathematical primitives

In this section, we introduce some basic mathematical definitions and operations needed in this document.

3.1 Finite fields and polynomial Rings

A field is defined as a set with operations such as addition, subtraction, multiplication and division that satisfy certain rules. Typical examples of fields are the real number field $\mathbb{R}$, the rational number field $\mathbb{Q}$ and finite fields F_q. Finite fields F_q are fields with q elements, where q is a positive integer, called the order. It is well known that the order is a prime p or a prime power p^k. A prime field is defined as a finite field whose order is prime. In this document, we focus on the case of prime fields written as sets:

$$F_q = \{0, 1, \cdots, q-1\}.$$

Its operations are described using the modulus of q as follows:

$$\begin{aligned} a+b &= a+b \mod q \\ a-b &= a-b \mod q \\ a \cdot b &= a \cdot b \mod q \\ a/b &= a \cdot b^{-1} \mod q \ , \end{aligned} \tag{1}$$

where b^{-1} satisfies the condition $b \cdot b^{-1} = 1 \mod q$.

Example. The prime field $F_5 = \{0, 1, 2, 3, 4\}$ can be equipped with operations modulo 5, such as

$$1+2=3, \quad 2+4=1, \quad 3-1=2, \quad 2-3=4,$$

$$2 \cdot 2 = 4, \quad 2 \cdot 3 = 1, \quad 2/3 = 2 \cdot 3^{-1} = 2 \cdot 2 = 4.$$

Let R be a ring. A univariate polynomial ring is a set defined as

$$R[t] = \{c_0 + c_1 t + \cdots + c_n t^n \mid c_i \in R \ (0 \le i \le n) \ n \in \mathbb{N}\}, \tag{2}$$

where t is a variable and c_i is the coefficient of the monomial $c_i t^i$. Univariate polynomials $f(t)$ and $g(t)$ can be described as

$$\begin{aligned} f(t) &= a_0 + a_1 t + \cdots + a_n t^n \\ g(t) &= b_0 + b_1 t + \cdots + b_n t^n \ , \end{aligned} \tag{3}$$

where a_i and b_i are elements of R. We note that neither $a_n \neq 0$ nor $b_n \neq 0$ is assumed in the expression of (3) above.

$R[t]$ is a ring since addition and multiplication are defined as follows:

$$\begin{aligned} f+g &= a_0 + b_0 + (a_1 + b_1)t + \cdots + (a_n + b_n)t^n \\ f \cdot g &= a_0 \cdot b_0 + (a_1 \cdot b_0 + a_0 \cdot b_1)t + \cdots + (a_n \cdot b_n)t^{2n} \ . \end{aligned} \tag{4}$$

Though an inverse of addition can be defined as

$$-f = -a_0 - a_1 t + \cdots - a_n t^n \ ,$$

an inverse of multiplication can be defined if and only if $f(t)$ is a non-zero constant, such as $f(t) = a_0$.

Example. Let us consider a univariate polynomial in $F_5[t]$ and set $f(t) = 2{+}3t{+}4t^2$ and $g(t) = 4{+}t{+}3t^2$. Then, $f(t) + g(t) = 1 + 4t + 2t^2$, $f(t) \cdot g(t) = 2t^4 + 3t^3 + 4t + 3$, and $-f(t) = 3 + 2t + t^2$.

$$F_5[t] = \{c_0 + c_1 t + \cdots + a_n t^n \mid a_i \in F_5 \ (0 \le i \le n) \ n \in \mathbb{N}\} \ . \tag{5}$$

If a polynomial is written in $f(t) = \sum_{i=0}^{n} c_i t^i$ such that the coefficient $c_n \neq 0$ then we define n to be the degree of f. Thus, the degree of f is the maximum integer n such that $a_n \neq 0$. We denote this by $\deg f = n$. In the example of $f(t)$ and $g(t)$ above,

$$\deg(f) = \deg(g) = 2, \ \deg(f(t) \cdot g(t)) = 4 \, .$$

A bivariate polynomial ring is a set defined as

$$R[x,y] = \{c_{n0}x^n + c_{(n-1)1}x^{n-1}y + \cdots + c_{0n}y^n + \cdots c_{10}x + c_{01}y + c_{00} \mid c_{ij} \in R \ (0 \le i,j \le n) \ n \in \mathbb{N}\}, \quad (6)$$

where x and y are variables and c_{ij} are coefficients of the monomial $c_{ij}x^iy^j$.

Set $f(x,y)$ and $g(x,y)$ as follows:

$$\begin{array}{rcl} f(x,y) & = & \sum_{i=j=1}^{n} a_{ij}x^iy^j = a_{n0}x^n + a_{(n-1)1}x^{n-1}y + \cdots + a_{0n}y^n + \cdots a_{10}x + a_{01}y + a_{00} \\ g(x,y) & = & \sum_{i=j=1}^{n} b_{ij}x^iy^j = b_{n0}x^n + b_{(n-1)1}x^{n-1}y + \cdots + b_{0n}y^n + \cdots b_{10}x + b_{01}y + b_{00} \, , \end{array} \quad (7)$$

where a_{ij} and b_{ij} are elements of R. Then we define addition and multiplication as follows:

$$\begin{array}{ll} f+g & = \sum_{i=j=0}^{n}(a_{ij}+b_{ij})x^iy^j \\ & = (a_{n0}+b_{n0})x^n + (a_{(n-1)1}+b_{(n-1)1})x^{2n-1}y + \cdots + (a_{10}+b_{10})x + (a_{01}+b_{01})y + a_{00} + b_{00} \\ f \cdot g & = \sum_{i_1+j_1=i_2+j_2=0}^{n}(a_{i_1j_1}b_{i_2j_2})x^iy^j \\ & = (a_{n0}b_{n0})x^{2n} + (a_{n0}b_{(n-1)1} + a_{(n-1)1}b_{n0})x^{2n-1}y + \cdots + (a_{01}b_{00} + a_{00}b_{01})y + a_{00}b_{00} \\ ,. & \end{array} \quad (8)$$

An inverse of addition can be defined as

$$-f = -a_{n0}x^n - a_{(n-1)1}x^{n-1}y - \cdots - a_{0n}y^n - \cdots a_{10}x - a_{01}y - a_{00} \, .$$

However, an inverse of multiplication does not exist in general.

Example. In the case of $F_5[x,y]$, set

$$\begin{array}{c} f(x,y) = 2x^2 + 3xy + y^2 + 3x + 3y + 4 \\ g(x,y) = x^2 + 2xy + 3y^2 + x + 3y + 3 \, , \end{array} \quad (9)$$

and then $f(x,y) + g(x,y) = 3x^2 + 4y^2 + 4x + y + 2$,

$$f(x,y) \cdot g(x,y) = 2\,x^4 + 2\,x^3y + 3\,x^2y^2 + xy^3 + 3\,y^4 + 3\,x^2y + 2\,y^3 + 3\,x^2 + 4\,xy + 4\,y^2 + 3\,x + y + 2$$

and $-f(x,y) = 3x^2 + 2xy + 4y^2 + 2x + 2y + 1$.

Setting the bivariate polynomial $f(x,y) = \sum_{i=j=0}^{n} c_{ij}x^iy^j$, the total degree of f, denoted $\deg f$, can be defined as

$$\deg f := max\{i+j | c_{ij} \neq 0\} \, .$$

We can determine the degrees for $f(x,y)$ and $g(x,y)$, described above, as follows.

$$\deg(f(x,y)) = \deg(g(x,y)) = 2, \;\; \deg(f(x,y) \cdot g(x,y)) = 4 \, .$$

3.2 The quotient ring R_q

The ring R_q is defined as the quotient ring of $F_q[t]$ modulo $t^n - 1$. Elements of R_q are polynomials over F_q with degree at most $n-1$ (since t^n is equivalent to 1).

We can represent $a \in R_q$ as a vector $(a_0, a_1, \cdots, a_{n-2}, a_{n-1})$ representing

$$a = a_0 + a_1 t + \cdots + a_{n-2} t^{n-2} + a_{n-1} t^{n-1}$$

on F_q. When elements $b, c \in R_q$ are represented in the same manner as a, we can express $ab + c$ as

$$\begin{pmatrix} b_0 & b_1 & \cdots & b_{n-2} & b_{n-1} \end{pmatrix} \begin{pmatrix} a_0 & a_1 & \cdots & a_{n-2} & a_{n-1} \\ a_{n-1} & a_0 & \cdots & a_{n-3} & a_{n-2} \\ a_{n-2} & a_{n-1} & \cdots & a_{n-4} & a_{n-3} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_1 & a_{n-1} & \cdots & a_{n-1} & a_0 \end{pmatrix} + \begin{pmatrix} c_0 & c_1 & \cdots & c_{n-2} & c_{n-1} \end{pmatrix} \tag{10}$$

on F_q.

Using this expression, the relation $ab + c = d$ can be described as

$$\vec{b}A + \vec{c} = \vec{d},$$

where vectors $\vec{b}$ and $\vec{c}$ correspond to b and c, respectively, and A is a matrix corresponding to a. The vector $\vec{d}$ corresponds to the result of $ab + c$.

3.3 monomial order

To describe a detailed specification of the proposal, we need to introduce the monomial order of polynomials, which defines the order of calculation. First, we define an exponent vector $\alpha = (i, j) \in \mathbb{Z}^2_{\geq 0}$ of monomialx $x^i y^j$, and then we denote a monomial $x^i y^j$ as x^α.

Example. The exponent vectors of monomials $3x^2y^3$ and $4x^3$ in $F_q[x, y]$ are $(2, 3)$ and $(3, 0)$ respectively.

We define the monomial ordering $x^\alpha > x^\beta$ as follows.

Definition 1. A monomial ordering on bivariate polynomial ring $F_q[x, y]$ is any relation $>$ on the set of monomials in $F_q[x, y]$ or $\mathbb{Z}^2_{\geq 0}$ satisfying:

1. $>$ is a total ordering such that any pair of monomials α and β satisfies exactly one of the relations $\alpha < \beta$, $\alpha = \beta$, and $\alpha > \beta$.

2. $>$ is compatible with multiplication in $F_q[x, y]$. If $\alpha > \beta$ and there is some $\gamma \in \mathbb{Z}^2_{\geq 0}$ then $\alpha + \gamma > \beta + \gamma$ since the relation $x^\alpha x^\gamma > x^\beta x^\gamma$ is satisfied.

3. $>$ induces a well ordering, such that there is a minimum element in any non-empty subset of $\mathbb{Z}^2_{\geq 0}$ or monomial set.

Lexicographic ordering is an example of monomial ordering satisfying these rules. It is defined as follows.

Definition 2. For any $\alpha = (\alpha_1, \alpha_2) \in \mathbb{Z}^2_{\geq 0}$ and $\beta = (\beta_1, \beta_2) \in \mathbb{Z}^2_{\geq 0}$, the relation $\alpha >_{lex} \beta$ (resp., $\alpha <_{lex} \beta$) holds when the leftmost non-zero entry of the difference of the exponent vectors $\alpha - \beta$ is positive (resp., negative). We write $x^\alpha >_{lex} >_{lex} x^\beta$ if $\alpha >_{lex} \beta$ and analogously for $<_{lex}$.

For example, $(2,1) >_{lex} (1,2)$ since the difference of the exponent vectors $\alpha - \beta = (1,-1)$. Similarly, $(2,1) <_{lex} (2,2)$ since $\alpha - \beta = (0,-1)$, and the leftmost non-zero entry is negative.

In this document, we employ the graded lexicographic order, which is defined as follows.

Definition 3. Let x^α and x^β be monomials in $F_q[x,y]$. We define $x^\alpha <_{grlex} x^\beta$ if $\alpha_1 + \alpha_2 > \beta_1 + \beta_2$, or if $\alpha_1 + \alpha_2 = \beta_1 + \beta_2$ and in the difference vector $\alpha - \beta$, the leftmost non-zero entry is positive.

For example, we have $(0,2) >_{grlex} (1,0)$ since $\alpha_1 + \alpha_2 = 2 > 1 = \beta_1 + \beta_2$. In the case of $(1,1) >_{grlex} (0,2)$, we have $\alpha_1 + \alpha_2 = 2 = \beta_1 + \beta_2$ and $\alpha - \beta = (1,-1)$, the leftmost non-zero entry is positive.

3.4 Mathematical primitives

This section provides some mathematical primitives that are used in the cryptographic primitives described in Section 6.

3.4.1 RandRingElement

Input:	n	Dimensionality of quotient ring R_q
	q	Modulus of the quotient ring r_q
	ℓ	An integer
Output:	f	Random element of the ring R_q with coefficients restricted to the range 0 to $\ell - 1$
(inclusive)	n, q, ℓ (ℓ is an integer larger than 1)	
Assumption:		
	q is greater than or equal to ℓ.	
Step:		

1. An element $f = c_0 + c_1 t + \cdots + c_{n-1} t^{n-1}$ whose coefficients c_i are restricted to the range 0 to $\ell - 1$ is generated as follows.
   (a) Set $f = 0$
   (b) For $i = 0$ to $n - 1$
      i. Set c_i=RandInteger$(0, \ell)$
      ii. Set $f = f + c_i t^i$

3.4.2 GenSupportList

Input:	d	Total degree of bivariate polynomial
Output:	Γ	Support list of bivariate polynomial whose degree is d in the graded lexicographic order.
Assumption:	d is a positive integer	
Step:		

1. An ordered list consisting of the exponentials pairs (i, j) that correspond to possible terms $x^i y^j$ in the bivariate polynomial is generated as follows. We call such a list a "support list."

```
(a) Set $\Gamma = [\ ]$ (an empty list)
(b) For $k = d$ to 0 by $-1$
    i. For $i = d$ to 0 by $-1$
        A. Set $\Gamma = \Gamma \cup [(i, d - i)]$
```

4 Overview of proposed post-quantum cryptosystem

This section provides an overview of the proposed PQC.

4.1 Domain parameters

We introduce parameters for the proposed scheme to be input to the key generation algorithm. Appropriate parameter settings are discussed in Section 10.

1. ℓ: A small integer
2. q: A prime which is cardinality of prime field F_q and is much larger than ℓ.
3. n: Degree of the modulus polynomial of the quotient ring $R_q(= F_q[t]/(t^n - 1))$. The n should be prime for the security reason.
4. dX: Total degree of the irreducible bivariate polynomial $X(x, y)$
5. dr: Total degree of the random bivariate polynomial $r(x, y)$
6. $mlen$ Length of the message M

The relation between ℓ and q is a critical condition for decryption. We require the condition

$$q > \sum_{k=0}^{dX+dr} (k + 1)n^k (\ell - 1)^{k+1} \tag{11}$$

to decrypt any ciphertext encrypted by the proposed encryption primitive.

The support set of the irreducible polynomial $X(x,y)$ with total degree dX is defined such that

$$\Gamma_X = \{(i,j) \in (\mathbb{N} \cup \{0\})^2 \ \mid \ 0 \le i, j, i+j \le dX\}$$

with graded lexicographic order. If dX is equal to 2, then

$$\Gamma_X = \{(2,0),(1,1),(0,2),(1,0),(0,1),(0,0)\},$$

whose elements correspond to the monomials x^2,xy,y^2,x,y, and 1, in that order, and the monomial order is called the graded lexicographic order.

The support set of the random polynomial $r(x,y)$ with total degree dr is also defined such that

$$\Gamma_r = \{(i,j) \in (\mathbb{N} \cup \{0\})^2 \ \mid \ 0 \le i, j, i+j \le dr\}$$

with graded lexicographic order. Since the total degree of the noise polynomial $e(x,y)$ is defined to be $dX + dr$, the Support set of the noise polynomial $e(x,y)$ is

$$\Gamma_e = \{(i,j) \in (\mathbb{N} \cup \{0\})^2 \ \mid \ 0 \le i, j, i+j \le dX + dr\}$$

with graded lexicographic order. If $dX = dr = 2$, then

$$\Gamma_e = \{(4,0),(3,1),(2,2),(1,3),(0,4),(3,0),(2,1),(1,2),(0,3),(2,0),(1,1),(0,2),(1,0),(0,1),(0,0)\},$$

whose elements correspond to the monomials x^4,x^3y,x^2y^2,xy^3,y^4,x^2y,xy^2,y^3,x^2,xy,y^2,x,y, and 1, in that order.

4.2 Key Generation

The secret key is a small (not necessarily smallest) solution of the indeterminate equation $X(x,y) = 0$:

$$(x,y) = (u_x(t), u_y(t)), \quad u_x(t), u_y(t) \in R_\ell, \tag{12}$$

where $\deg u_x(t) = \deg u_y(t) = n-1$. Note that ℓ is much smaller than q, and thus we call $(u_x(t), u_y(t))$ a small solution. The public key is the indeterminate equation $X(x,y) = 0$, which is irreducible and has the small solution $(u_x(t), u_y(t))$:

$$X(x,y) = \sum_{(i,j)\in\Gamma_X} a_{ij}(t) x^i y^j = \ , \tag{13}$$

where $a_{ij}(t) \in R_q$.

The key generation algorithm takes the parameters ℓ,q,n,dX, and dr as parameters, and is defined in Section 4.1. The secret key is generated as degree $n-1$ random polynomials $u_x(t), u_y(t) (\in R_\ell)$. The indeterminate equation $X(x,y) = 0$ is constructed according to the following procedure.

1. Generate a degree dX support set Γ_X with graded lexicographic order, applying the function GenSupportSet described in Section 3.

2. Choose a coefficient of each monomial (except the constant term) as follows.
   (a) Set $X(x, y) = 0$
   (b) For each element (i, j) in $\Gamma_X - \{(0, 0)\}$
      i. Choose a coefficient $a_{ij}(t)$ whose degree is $n - 1$, uniformly at random from the set R_q
      ii. Set $X(x, y) = X(x, y) + a_{ij}(t)x^i y^j$
3. Calculate the constant term $a_{00}(t)$ as
   $a_{00}(t) = -\sum_{(i,j)\in\Gamma_X-\{(0,0)\}} a_{ij}(t)u_x(t)^i u_y(t)^j (\in R_q)$
4. Confirm the polynomial $X(x, y)$ is irreducible; if not, return to step 2a.

4.3 Encryption

1. Embed a plaintext M into the coefficients of the plaintext polynomial $m(t)(\in R_\ell)$ whose degree is $n - 1$. As an example, in the case of $\ell = 4, n = 3$, a plaintext $M = (312)_4$ can be embedded such as $m(t) = 3t^2 + t + 2$.
2. Generate a support set Γ_r of degree dr with graded lexicographic order
3. Create a random polynomial $r(x, y)$ as follows:
   (a) Set $r = 0$
   (b) For each (i, j) in Γ_r
      i. Choose a coefficient $r_{ij}(t)$ uniformly at random from the set R_q
      ii. Set $r(x, y) = r(x, y) + r_{ij}(t)x^i y^j$
4. Generate a support set Γ_e of degree $dX + dr$ with graded lexicographic order
5. Create a noise polynomial $e(x, y)$ as follows:
   (a) Set $e(x, y) = 0$
   (b) For each (i, j) in Γ_e
      i. Choose a coefficient $e_{ij}(t)$ uniformly at random from the set R_ℓ
      ii. Set $e(x, y) = e(x, y) + e_{ij}(t)x^i y^j$
6. Construct the cipher polynomial $c(x, y)$ as

$$c(x, y) = m(t) + X(x, y)r(x, y) + \ell \cdot e(x, y) \tag{14}$$

4.4 Decryption

1. Substitute the secret key that is a small solution $(u_x(t), u_y(t))$ over R_q of $X(x, y) = 0$ into $c(x, y)$:

$$c(u_x(t), u_y(t)) = m(t) + \ell \cdot e(u_x(t), u_y(t)) \tag{15}$$

   When the parameters ℓ and q satisfy the relation described above (11), each coefficient of $m(t) + \ell \cdot e(u_x(t), u_y(t)) \in \mathbb{Z}/(t^n - 1)$ is within the range 0 to $q - 1$. Theorem 4 gives a proof of this fact.

2. Extract $m(t)$ from $c(u_x(t), u_y(t))$ as

$$c(u_x(t), u_y(t)) \quad (mod \quad \ell) = m(t)$$

   where we consider $c(u_x(t), u_y(t))$ as an element of $\mathbb{Z}[t]$

3. Recover the plaintext M from the coefficients of $m(t)$

Theorem 4. Let a ciphertext polynomial $c(x, y)(\in R_q[x, y])$ encrypt a plaintext polynomial $m(t)(\in R_\ell)$ with a public key $X(x, y)$ and public parameters (n, ℓ, q, dX, dr), applying the encryption process IECENC. The plaintext polynomial $m(t)$ can be recovered from the ciphertext $c(x, y)$ with a corresponding secret key $(u_x(t), u_y(t))$ and public parameters (n, ℓ, q, dX) by applying the decryption process IECDEC.

Proof. Since a secret key $(u_x(t), u_y(t))$ is a solution of the equation $X(x, y) = 0$, we obtain

$$c(u_x(t), u_y(t)) = m(t) + \ell \cdot e(u_x(t), u_y(t)) \ (mod\ \ell)\,,$$

where the calculation is in the ring $R_q[x, y]$.

Take $m(t)+\ell \cdot e(u_x(t), u_y(t))$ of R_q as a univariate polynomial over the integers $\mathbb{Z}$, where the coefficients are integers within the range 0 to $q - 1$. Now we denote by $MC(f(t))$ the maximum coefficient of a univariate polynomial $f(t)$ over the integer $\mathbb{Z}$. If the condition

$$MC(m(t) + \ell \cdot e(u_x(t), u_y(t)) < q \tag{16}$$

is satisfied in the univariate polynomial ring $\mathbb{Z}[t]$ for any possible $m(t), e(x, y), (u_x(t), u_y(t)), \ell$, then the conclusion

$$m(t) + \ell \cdot e(u_x(t), u_y(t))(mod\ell) = m(t)$$

follows. Here, $m(t)$ is an element of R_ℓ whose coefficients are restricted to the range 0 to $\ell - 1$.

To see the relation 16, we assume the coefficients of the polynomials $u_x(t), u_y(t)$ are maximum, such as

$$u_x(t) = u_y(t) = (\ell - 1)(t^{n-1} + t^{n-2} + \cdots + t + 1)\,.$$

We can see $(t^{n-1} + t^{n-2} + \cdots + t + 1)^k = n^{k-1}(t^{n-1} + t^{n-2} + \cdots + t + 1)$ for any positive integer k since the multiples have to be reduced by $t^n - 1$. Then

$$u_x(t)^k = u_y(t)^k = (\ell - 1)^k \cdot n^{k-1}(t^{n-1} + t^{n-2} + \cdots + t + 1)\,,$$

The support set Γ_e is

$$\Gamma_e = \{(i,j) \in (\mathbb{N} \cup \{0\})^2 | 0 \leq i, j, i+j \leq dX + dr\} .$$

Since there are ${}_2H_k$ degree-k elements in Γ_e, the value of $MC(e(u_x(t), u_(t))$ is as follows:

$$\begin{aligned}
MC(e(u_x(t), u_(t)) &= MC(\textstyle\sum_{(i,j)\in\Gamma_e} e_{ij}(t)u_x(t)^i u_y(t)^j) \\
&\leq MC(\textstyle\sum_{(i,j)\in\Gamma_e} (\ell-1)(t^{n-1} + t^{n-2} + \cdots + t + 1)u_x(t)^i u_y(t)^j) \\
&= (\ell-1)\textstyle\sum_{k=0}^{dX+dr} {}_2H_k n^k \dot{(}\ell-1)^k \\
&= \textstyle\sum_{k=0}^{dX+dr} {}_{k+1}C_k n^k \dot{(}\ell-1)^{k+1} \\
&= \textstyle\sum_{k=0}^{dX+dr} (k+1) n^k \dot{(}\ell-1)^{k+1}.
\end{aligned}$$

So, we obtain the relation

$$MC(m(t) + \ell \cdot e(u_x(t), u_(t)) \leq \ell - 1 + \ell \sum_{k=0}^{dX+dr} (k+1) n^k \dot{(}\ell-1)^{k+1}.$$

The condition (16) is always satisfied since $q > \ell - 1 + \ell \sum_{k=0}^{dX+dr} (k+1) n^k \dot{(}\ell-1)^{k+1}$.

$\square$

5 Data conversion primitives

This section defines some data conversion primitives, such as conversion from octet strings to integers or polynomials, and their inverses. Here, dp denotes a set of the domain parameters defined in Section 4.1. We can refer to elements of the set individually, such as

$dp.\ell$	A small integer	
$dp.q$	The cardinality of a prime field F_q, much larger thanℓ	
$dp.n$	Degree of the modulus polynomial of the quotient ring $R_q(= F_q[t]/(t^n - 1))$	(17)
$dp.dX$	Total degree of the irreducible bivariate polynomial $X(x, y)$	
$dp.dr$	Total degree of the random bivariate polynomial $r(x, y)$	
$dp.mlen$	Length (in bits) of the message M.	

In this document, these domain parameters satisfy the conditions as follows.

$dp.\ell$	A divisor of 16 except 1 (i.e., 2, 4, 8, or 16)	
$dp.n$	An integer larger than 1.	(18)
$dp.dX, dp.dr$	Positive integers.	
$dp.q$	Satisfies the condition (11)	

In addition, we define $qlen$ as the byte length of $dp.q$ to simplify the notation. $qlen$ is described as follows:

$$qlen := \lceil |dp.q|/8 \rceil \tag{19}$$

5.1 Finite field elements

This section defines functions that transfer an element of the finite field F_q to an octet string and vice versa. Figure 1 describes how these functions work.

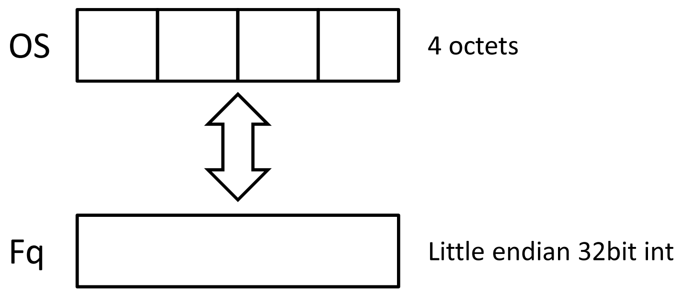

Fig. 1: OS2Fq/Fq2OS

5.1.1 OS2Fq

Input:	os	Octet string
	dp	Set of the domain parameters
Output:	c	Element of F_q
Assumption:	os is a $qlen$-byte array of octet characters	
Step:		

```
1. Set $c = 0$
2. For $i = 0$ to $qlen - 1$
   (a) Set $c_i = os[i]$
3. Set $c = \sum_{i=0}^{qlen-1} c_i \cdot 2^{8i}$
```

5.1.2 Fq2OS

Input:	c	Element of F_q
	dp	Set of domain parameters
Output:	os	Array of octet strings
Assumption:	os is an octet string with byte length $qlen$	
Step:		

```
1. For $i = 0$ to $qlen - 1$
   (a) Set $os[i] = c \bmod 2^8$
   (b) Set $c = c \gg 8$
```

5.2 Quotient ring R_q

This section defines functions that transfer an element of the quotient ring R_q or a subset R_ℓ to an octet string and vice versa. Figure 2 describes how functions to transfer between R_q and an octet string work. Figure 3 describes how functions to transfer between R_ℓ and an octet string work.

5.2.1 OS2Rq

OS2Rq is a conversion function that transfers an octet string to a univariate polynomial over F_q whose coefficients is restricted to the range 0 to $q - 1$.

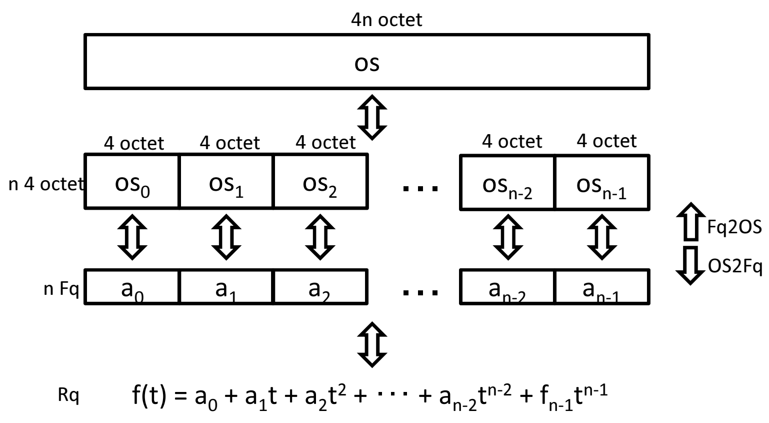

Fig. 2: OS2Rq/Rq2OS

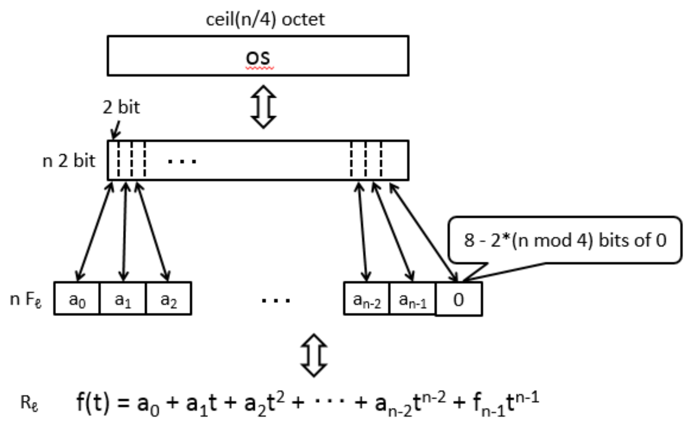
ceil(n/4) octet
OS
2 bit
n 2 bit
...
8 - 2*(n mod 4) bits of 0
$n\ F_\ell$
a_0
a_1
a_2
...
a_{n-2}
a_{n-1}
0
R_ℓ
$f(t) = a_0 + a_1t + a_2t^2 + \cdots + a_{n-2}t^{n-2} + f_{n-1}t^{n-1}$

Fig. 3: OS2Rℓ/Rℓ2OS

Input:	*os*	Octet string
	dp	Set of domain parameters
Output:	*elm*	Degree-n element of the quotient ring F_q

Assumption: *os* is an array of octet characters that has byte length $dp.n \cdot qlen$.

Step:

1. Decompose *os* into elements os_i each of byte length *qlen*, such that

$$os = os_0 || os_1 || \cdots || os_{qlen}$$

2. For $i = 0$ to $dp.n - 1$

   (a) Set $a_i = OS2Fq(os_i, dp)$

3. Set $elm = a_0 + a_1 t + \cdots + a_{dp.n-1} t^{dp.n-1}$

5.2.2 OS2Rℓ

OS2Rℓ is a conversion function that transfers an octet string to the quotient ring R_q with coefficients restricted to the range 0 to $\ell - 1$.

Input:	*os*	An Octet string
	dp	Set of domain parameters
Output:	*elm*	Degree-n element of the quotient ring F_q

Assumption: *os* is an array of octet characters, with byte length $\lceil dp.n \cdot |dp.\ell/8| \rceil$.

Step:

1. Decompose *os* into elements bs_i each of bit length $|dp.\ell|$

2. Set $elm = bs_0 + bs_1 \cdot t + \cdots + bs_{dp.n-1} t^{dp.n-1}$

5.2.3 Rq2OS

Rq2OS is a conversion function that accepts a univariate polynomial over F_q with coefficients restricted to the range 0 to $q - 1$ and produces an octet string.

Input:	elm	Degree-n element of the quotient ring F_q
	dp	Set of domain parameters
Output:	os	An octet string
Assumption:	os is an array of octet characters, with length $dp.n|dp.q|$ bits.	
Step:		

1. Extract the coefficients of elm to elements a_i $(i = 0, \cdots dp.n - 1)$, each with byte length $qlen$
2. For $i = 0$ to $dp.n - 1$
   (a) Set $os_i = Fq2OS(a_i)$
3. Set $os = os_0||os_1||\cdots||os_{dp.n-1}$

5.2.4 Rℓ2OS

Rℓ2OS is a conversion function that accepts a univariate polynomial over F_q with coefficients restricted to the range 0 to $\ell - 1$ and produces an octet string.

Input:	elm	Degree-n element of the quotient ring F_q
	dp	Set of domain parameters
Output:	os	An octet string
Assumption:	os is an array of octet characters, with byte length $\lceil dp.n \cdot |dp.\ell|/8\rceil$.	
Step:		

1. Extract the coefficients of elm to bit strings bs_i $(i = 0, \cdots, dp.n - 1)$ each with bit length $|dp.\ell|$.
2. Set $os = bs_0||bs_1||\cdots||bs_{dp.n-1}||0\cdots0$, where right-hand side $||0\cdots0$ indicates concatenation $(8 - |dp.\ell| \cdot (n \bmod (8/|dp.\ell|)))$ bits of 0.)

5.3 Bivariate polynomial ring $R_q[x, y]$

This section defines functions that transfer an element of the bivariate polynomial ring $R_q[x, y]$ to an octet string and vice versa. Figure 4 describes how functions to transfer between $R_q[x, y]$ and an octet string work when the total degree of polynomial $f(x, y)$ is 1. Figure 5 describes how functions to transfer between $R_q[x, y]$ and an octet string work when the total degree of polynomial $f(x, y)$ is 2.

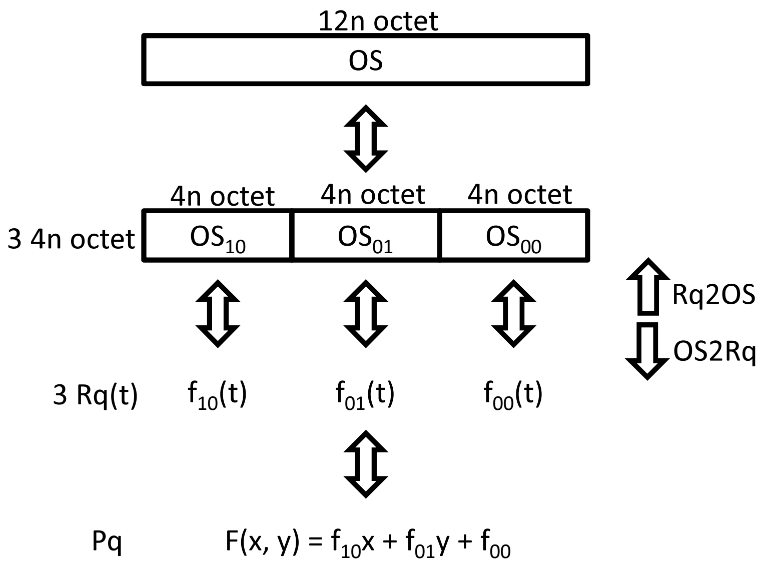

Fig. 4: OS2Pq/Pq2OS (total degree = 1)

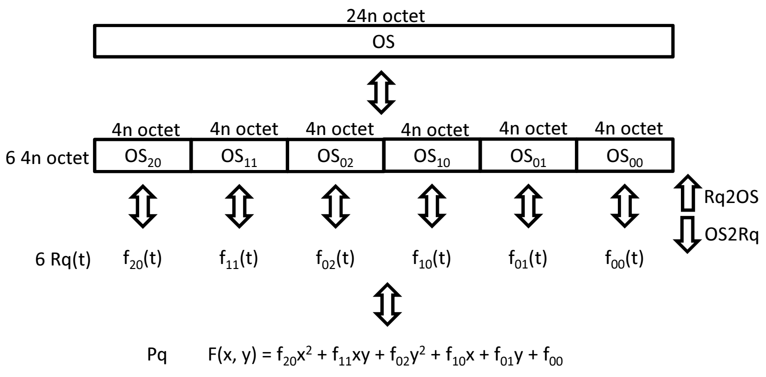

Fig. 5: OS2Pq/Pq2OS (total degree = 2)

5.3.1 OS2Pq

OS2Pq is a conversion function that transfers an octet string to a bivariate polynomial over R_q.

Input:	*os*	An octet string
	d	Total degree of the output bivariate polynomial
	dp	Set of domain parameters
Output:	*ply*	Degree-n bivariate polynomial over R_q
Assumption:	*os* is an array of octet characters, with length $d\dot{|}q|$ bits.	
	d is a positive integer	

Step:

```
1. Set $\Gamma = GetSupportList(d)$
2. Decompose $os$ into elements $os_k$ each of byte length $qlen$
3. For $k = 0$ to $\#\Gamma - 1$
    (a) Set $elm_k = OS2Rq(os_k, dp)$
4. Set $ply = 0$
5. For $k = 0$ to $\#\Gamma - 1$
    (a) Set $(i, j) = \Gamma[k]$
    (b) Set $ply = ply + elm_k \cdot x^i y^j$
```

5.3.2 Pq2OS

Pq2OS is a conversion function trat transfers a bivariate polynomial over F_q to an octet string.

Input:	*ply*	Bivariate polynomial of t over F_q, with total degree d
	d	Total degree of the output bivariate polynomial
	dp	Set of domain parameters
Output:	*os*	An octet string
Assumption:	*os* is an array of octet characters, with length $d\|q\|$ bits	
	d is a positive integer	
Step:		

1. Set $\Gamma = GetSupportList(d)$
2. Decompose the bivariate polynomial *ply* into
$$ply(x,y) = \sum_{(i,j)\in\Gamma} c_{ij}x^i y^j$$
3. For $k = 0$ to $\#\Gamma - 1$
   (a) Set $(i,j) = \Gamma[k]$
   (b) Set $os_k = Rq2OS(c_{ij}, dp)$
4. Set $os = os_0||os_1||\cdots||os_{\#\Gamma-1}$

6 Cryptographic primitive

An encryption primitive uses a public key to create a ciphertext from a plaintext, and a decryption primitive recovers the plaintext from the ciphertext given the secret key that corresponds to the public key.

A key generation primitive creates a pair of a public key and a secret key, with control parameters, here ℓ,n,q,dX, and dr, described in Section 4. The parameter values depend on the needed security.

The prime q is defined to satisfy the condition (11), such as

$$q > \ell - 1 + \ell \sum_{k=1}^{dX+dr+1} kn^{k-1}(\ell-1)^k\,.$$

6.1 Key generation

6.1.1 Secret key

The secret key is a small solution of the indeterminate equation $X(x,y) = 0$ in R_q such that

$$(x,y) = (u_x(t), u_y(t)),$$

where $u_x(t)$ and $u_y(t)$ are elements of R_ℓ. Namely,

$$u_x(t) = \sum_{i=0}^{n-1} \alpha_i t^i, \quad u_y(t) = \sum_{i=0}^{n-1} \beta_i t^I,$$

where α_is and β_is are restricted to the range 0 to $\ell - 1$.

6.1.2 Public key

The public key is an irreducible bivariate polynomial $X(x, y)$ described by

$$X(x, y) = \sum_{i=0}^{dX} \sum_{j=0}^{dX} a_{ij}(t) x^i y^j,$$

where $a_{ij}(t)$ is an element of R_q. Here, the indeterminate equation $X(x, y) = 0$ has a small solution $(x, y) = (u_x(t), u_y(t))$ that is the secret key described above.

6.1.3 Key Generation

This section gives details of the key generation algorithm shown in Section 4.2.

6.1.4 IECKG

Input:	dp	Set of domain parameters
Output:	$(u_x(t), u_y(t))$	Small solution in R_ℓ^2 with coefficients restricted to the range 0 to $\ell - 1$
	$X(x, y)$	Irreducible bivariate polynomial that has small solution $(u_x(t), u_y(t))$
Assumption:	dp satisfies the condition (64)	
Step:		

1. The secret key (the smallest solution) is generated as follows.
   (a) $u_x(t)$ =RandRingElement($dp.n$,$dp.q$,$dp.\ell$)
   (b) $u_y(t)$ =RandRingElement($dp.n$,$dp.q$,$dp.\ell$)
2. Generate the support set Γ_X of a bivariate polynomial whose degree is $dp.dX$ by using the function GenSupportSet
$$\Gamma_X = \text{GenSupportSet}(dp.dX)$$
3. Generate a coefficient $a_{ij}(t)$ of the degree $dp.n - 1$ function $X(x, y)$ uniformly at random from the ring R_q by using the function RandRingElement($dp.n$, $dp.q$, $dp.q$).
   (a) Set $X_0(x, y) = 0$
   (b) For $k = 0$ to $\#\Gamma_X - 1$
      i. Set $(i, j) = \Gamma[k]$
      ii. Set $a_{ij}(t)$ =RandRingElement($dp.n$, $dp.q$, $dp.q$)
      iii. Set $X_0(x, y) = X_0(x, y) + a_{ij}(t)x^i y^j$
   (c) Calculate the constant term $a_{00}(t)$ as
$$a_{00}(t) = -\sum_{(i,j)\in\Gamma_X} a_{ij}(t)u_x(t)^i u_y(t)^j$$
   (d) Set $X_0(x, y) = X_0(x, y) + a_{00}(t)$
   (e) Confirm that the polynomial $X(x, y)$ is irreducible; if it is not, return to step 3a

6.2 Encryption

This section gives details of the encryption algorithm of Section 4.3.

The encryption process consists of two sub-processes: plaintext embedding and encryption. The plaintext embedding process produces a plaintext polynomial $m(t)$ from a plaintext M up to $n|\ell|$ bits long. The encryption process produces a ciphertext polynomial $c(x, y)$ from a plaintext polynomial $m(t)$ with a public key X, a set of domain parameters (n, ℓ, q, dX, dr), a random bivariate polynomial $r(x, y)$, and a noise bivariate polynomial $e(x, y)$. These are defined in this section or Section 2.

6.2.1 IECENC

Input:	$m(t)$	Plaintext polynomial up to $dp.n|dp.\ell|$ bits long
	$X(x, y)$	Public key (an irreducible bivariate polynomial)
	dp	Set of domain parameters
Output:	$c(x, y)$	Ciphertext polynomial
Assumption:	$X(x, y)$ is a valid public key	

Step:

1. Create a random bivariate polynomial $r(x.y)$ over R_q

   (a) Generate a support set Γ_r of random bivariate polynomials whose degree is dr, applying the function GenSupportSet

   $$\Gamma_r = \text{GenSupportSet}(dr)$$

   (b) Generate a random polynomial $r(x, y)$ as follows

      i. Set $r(x, y) = 0$

      ii. For $i = 0$ to $\#\Gamma_r$

         A. Set $r_{ij}(t) = \mathit{RandRingElement}(dp.n, dp.q, dp.q)$

         B. Set $r(x, y) = r(x, y) + r_{ij}(t)x^i y^j$

2. Create a random bivariate polynomial $e(x.y)$ over R_ℓ

   (a) Generate the ordered support set Γ_e of a noise bivariate polynomial whose degree is $dX + dr$ by using the function GenSupportSet

   $$\Gamma_e = \text{GenSupportSet}(dp.dX + dp.dr)$$

   (b) Generate a noise polynomial $e(x, y)$ as follows

      i. Set $e(x, y) = 0$

      ii. For $i = 0$ to $\#\Gamma_e$

         A. Set $e_{ij}(t) = \mathit{RandRingElement}(dp.n, dp.q, dp.\ell)$

         B. Set $e(x, y) = e(x, y) + e_{ij}(t)x^i y^j$

3. Create the ciphertext $c(x, y)$ by letting

   $$c(x, y) = m(t) + X(x, y)r(x, y) + dp.\ell \cdot e(x, y)$$

   These calculations are carried out consistently in the ring $R_q[x, y]$

6.3 Decryption

This section gives details of the decryption algorithm of Section 4.4.

The decryption process recovers a message M from a cipher polynomial $c(x, y)$ when provided with the secret key $(u_x(t), u_y(t))$ corresponding to the secret key and the parameters used during the encryption process.

6.3.1 IECDEC

Input:	$c(x,y)$	Ciphertext polynomial
	$(u_x(t),u_y(t))$	Secret key (the smallest solution of $X(x,y)=0$)
	dp	Set of domain parameters
Output:	$m(t)$	Plaintext polynomial in R_ℓ
Assumption:	The secret key and the ciphertext polynomial are valid	
Step:		

1. Substitute the secret key $(u_x(t),u_y(t))$ into the ciphertext polynomial $c(x,y)$

$$w(t)=c(u_x(t),u_y(t))$$

2. Extract the plaintext polynomial $m(t)$ from $c(u_x(t),u_y(t))$ as follows:
   (a) Decompose $w(t)$ into $w(t)=w_0+w_1t+\cdots+w_{n-1}$
   (b) For $i=0$ to $dp.n-1$
      i. Set $m_i=Fq2Int(w_i) \mod \ell$
   (c) Set $m(t)=m_0+m_1t+\cdots+m_{dp.n-1}t^{dp.n-1}$

7 Design concept

7.1 Algebraic Surface Cryptosystem

The ASC was first announced in 2006 by K. Akiyama and Y. Goto [1]. The algebraic surfaces are defined as a solution space of a three-variable polynomial equation $X(x,y,t)=0$ over a field K. The security of ASC depends on the section-finding problem, defined as follows.

Definition 5. *(Section-finding problem) If $X(x,y,t)=0$ is an algebraic surface over a field K, then the problem of finding a parameterized curve*
$(x,y,t)=(u_x(t),u_y(t),t)$ on X is called the section-finding problem on X.

A section can be considered as a solution of $X(x,y)=0$, which is an indeterminate equation over the ring $K[t]$.

The problem of solving indeterminate equations over an arbitrary ring or field is known to be hard. For example, the class of indeterminate equations over the integer ring $\mathbb{Z}$, called Diophantine equations, is undecidable (Hilbert's 10th problem). Being "undecidable" means that there is no general algorithm to solve such indeterminate equations. The section-finding problem has been proven to be undecidable [12].

We recall the method of algebraic surface encryption to see the conceptual design for the scheme described in this paper. First, the simplest ASC can be described as

$$c(x,y) = m(x,y) + X(x,y)r(x,y),$$

where $X(x,y)$ is the public key, which defines an algebraic surface with a section. The polynomials $c(x,y)$ and $r(x,y)$ are a ciphertext polynomial and a random polynomial, respectively. The polynomial $m(x,y)$ is a plaintext polynomial in which a plaintext message is embedded. In the decryption phase, we substitute the secret key (a section of $X(x,y)$) into $c(x,y)$. By the relation $X(u_x(t),u_y(t)) = 0$, we obtain

$$c(u_x(t),u_y(t)) = m(u_x(t),u_y(t)).$$

From the polynomial $m(u_x(t),u_y(t))$, we can recover the plaintext message as follows. We can describe $m(x,y)$ as

$$m(x,y) = \sum_{(i,j,k)\in\Gamma_m} m_{ijk}x^iy^jt^k,$$

where each m_{ijk} is a variable, and substitute the section into $m(x,y)$. We thus obtain

$$m(u_x(t),u_y(t)) = \sum_{(i,j,k)\in\Gamma_m} m_{ijk}u_x(t)^iu_y(t)^jt^k.$$

Comparing the coefficient of t, the simultaneous linear equations containing m_{ijk} are constructed. When the number of variables is less than or equal to the number of equations, we can detect the correct plaintext message by solving the equations.

However, there exists an attack to break the scheme. We can expand the cipher polynomial $c(x,y)$ to

$$c(x,y) = \sum_{(i,j,k)\in\Gamma_m} m_{ijk}x^iy^jt^k + \left(\sum_{(i,j,k)\in\Gamma_X} a_{ijk}x^iy^jt^k\right)\left(\sum_{(i,j,k)\in\Gamma_r} r_{ijk}x^iy^jt^k\right), \tag{20}$$

where Γ_m,Γ_X, and Γ_r are given as parameters, and the values a_{ijk} are the given coefficients of the public key X. Each m_{ijk} and r_{ijk} is a variable. Comparing the coefficients of the monomials, we obtain the simultaneous linear equations having the variables m_{ijk} and r_{ijk}. For decryption, the relation

$$\#\Gamma_m + \#\Gamma_r < \#\Gamma_{Xr}$$

is required. However, in this case, the equations have unique solution with high probability. We refer to this type of attack as the **Linear Algebra attack**.

To avoid the attack, K. Akiyama, Y. Goto and H. Miyake constructed the latest ASC scheme in 2009 [2]. From the cryptographic point of view, the ciphertext is equivalent to

$$c(x,y) = m(x,y)s(x,y) + X(x,y)r(x,y). \tag{21}$$

Here, $s(x,y)$ is employed as another random polynomial and the term product $m(x,y)s(x,y)$ equals $X(x,y)r(x,y)$ (with $\Gamma_{ms} = \Gamma_{Xr}$). To decrypt the ciphertext, we have to divide $m(u_x(t),u_y(t))s(u_x(t),u_y(t))$ into $m(u_x(t),u_y(t))$ and $s(u_x(t),u_y(t))$ by factoring. Since polynomial factoring is computationally easy via the Berlekamp method, we can obtain $m(u_x(t), u_y(t))$ as a factor. The plaintext is then recovered from $m(u_x(t), u_y(t))$ in the same way as in the previous scheme.

Applying the Linear Algebra attack to this scheme, we need to consider $m(x, y)s(x, y)$ as a single polynomial $g(x, y)$, since quadratic equations are derived from the variables m_{ijk} and s_{ijk}. Therefore, the number of variables, $\#\Gamma_r + \#\Gamma_{Xr}$, is greater than the number of equations, $\#\Gamma_{Xr}$, and so the Linear Algebra attack does not work.

Unfortunately, this scheme was also broken by the **ideal decomposition attack**, which was described by Faugere et al. [15]. They found that the ideal $(c(x, y), X(x, y))$ can be decomposed into $(m(x, y), X(x, y))$ and $(s(x, y), X(x, y))$ by calculating the resultant $Res_x(c(x, y), X(x, y))$ and $Res_y(c(x, y), X(x, y))$. The plaintext message $m(x, y)$ is then recovered by solving the linear equations.

The proposed primitive avoids both attacks. Our idea is to apply for ℓ the polynomial structure employed in NTRU encryption. The ciphertext is

$$c(x, y) = m(x, y) + X(x, y)r(x, y) + \ell \cdot e(x, y),$$

where $e(x, y)$ is a random polynomial whose coefficients are small. The polynomial $e(x, y)$ works as a noise factor in the cipher, and we claim the condition

$$\#\Gamma_e = \#\Gamma_{Xr}$$

for resistance against the Linear Algebra attack. Needing the smallest solution of $X(x, y)$ to decrypt the message ensures this.

8 Security assumption and proof for primitives (IND-CPA)

In this section, we introduce a computational assumption and discuss some possible attacks under this assumption, based on the attacks for ASCs.

8.1 The smallest-solution problem

Let us express the solution $u = (u_x(t), u_y(t))$ $(\in (\mathbb{Z}_q[t]/(t^n - 1))^2)$ of an indeterminate equation as

$$u_x(t) = \sum_{i=0}^{n-1} \alpha_i t^i, \quad u_y(t) = \sum_{i=0}^{n-1} \beta_i t^i.$$

The norm of the solution is defined as follows.

$$Norm(u) = \max\{\alpha_i, \beta_i \in \mathbb{Z}_q^+ \mid 0 \le i \le n-1\}$$

The security of our system depends on the smallest-solution problem, defined as follows.

Definition 6. *(Smallest-solution Problem) If $X(x, y) = 0$ is an indeterminate equation over the ring $\mathbb{Z}_q[t]/(t^n - 1)$, then the problem of finding the solution $(x, y) = (u_x(t), u_y(t))$ on $\mathbb{Z}_q[t]/(t^n - 1)$ with the smallest norm is called the smallest-solution problem on X.*

Approximate lattice reduction algorithms cannot be directly applied to solving the problem because the solution space is non-linear.

8.2 Security assumption

Polynomials over $\mathbb{Z}_q$ whose coefficients are in the range 0 to $p - 1$ are called size-ℓ polynomials. If a polynomial is size ℓ, this means that its coefficients are much smaller than those of an ordinary polynomial, since ℓ is much smaller than q. We define the set of polynomials that have zero points in size ℓ as follows:

$$\mathfrak{X}(\Gamma_X, p)/R_q = \{X \in \mathfrak{F}_{\Gamma_X}/R_q \ \mid \ \exists u_x(t), u_y(t) \in R_\ell \ \ X(u_x(t), u_y(t)) = 0\}.$$

Given sets of polynomials, such as $\mathfrak{X}(\Gamma_X, \ell)/R_q$, $\mathfrak{F}_{\Gamma_r}/R_q$, and $\mathfrak{F}_{\Gamma_{Xr}}/R_\ell$, that satisfy the condition

$$(0, 0) \in \Gamma_X, (0, 0) \in \Gamma_r$$

are given, we define the decision problem as follows.

Definition 7. (IE-LWE problem) Writing the sets U_X and T_X as

$$U_X = \mathfrak{X}(\Gamma_X, p)/R_q \times \mathfrak{F}_{\Gamma_{Xr}}/R_q, \tag{22}$$

$$T_X = \{(X, Xr + e) | X \in \mathfrak{X}(\Gamma_X, \ell)/R_q, r \in \mathfrak{F}_{\Gamma_r}/R_q, e \in \mathfrak{F}_{\Gamma_{Xr}}/R_\ell\}, \tag{23}$$

the IE-LWE problem is to distinguish the multivariate polynomials chosen from a "noisy" set T_X of polynomials or from a set $U_X - T_X$, where T_X is a subset of U_X.

We define the IE-LWE assumption.

Definition 8. (IE-LWE assumption) The IE-LWE assumption is the assumption that the advantage

$$Adv_{\mathfrak{B}}^{\text{IE-LWE}}(k) :=$$

$$\left| \begin{array}{l} Pr\left[\mathfrak{B}(\ell,q,n,\Gamma_r,\Gamma_X,X,Y) \to 1 \;\middle|\; \begin{array}{l} (\ell,q,n,\Gamma_X,\Gamma_r,X) \overset{R}{\leftarrow} GenG(1^k); \\ r \overset{U}{\leftarrow} \mathfrak{F}_{\Gamma_r}/R_q; e \overset{U}{\leftarrow} \mathfrak{F}_{\Gamma_{Xr}}/R_\ell; \\ Y := Xr + e \end{array} \right] \\ -Pr\left[\mathfrak{B}(\ell,q,n,\Gamma_r,\Gamma_X,X,Y) \to 1 \;\middle|\; \begin{array}{l} (\ell,q,n,\Gamma_X,\Gamma_r,X) \overset{R}{\leftarrow} GenG(1^k); \\ \\ Y \overset{U}{\leftarrow} \mathfrak{F}_{\Gamma_{Xr}}/R_q \end{array} \right] \end{array} \right| \tag{24}$$

is negligible, where the function $GenG(1^k)$ outputs the domain parameters (i.e., ℓ,q,n,Γ_X, and Γ_r) from the security parameter k and creates X from these domain parameters by the function IECKG. In other words,

$$Adv_{\mathfrak{B}}^{\text{IE-LWE}}(k) < \epsilon(k),$$

where $\epsilon(k)$ is a negligible function in the security parameter k.

IE-LWE is an extended variation of R-LWE$^{\times}_{\text{HNF}}$, which is one of the variants of R-LWE defined by the polynomial ring R_q. This is claimed by a provably secure NTRU modification [35] and can be reduced to the shortest-vector problem of the lattice derived from R_q. In this paper, we extend R-LWE$^{\times}_{\text{HNF}}$ to the multivariate polynomial ring $R_q[x,y]$ so that the dimensionality of the lattice is larger than that of the lattice derived from R_q.

Theorem 9. Under the IE-LWE assumption, the Giophantus encryption scheme $\Sigma = (Gen, Enc, Dec)$ is secure in the sense of IND-CPA. Specifically, if there is an adversary that runs in polynomial time and breaks the Giophantus encryption scheme Σ in the sense of IND-CPA, then there exists an algorithm $\mathfrak{B}$ that solves the IE-LWE problem in probabilistic polynomial time. Moreover, the following relation holds:

$$Adv_{\Sigma,\mathfrak{A}}^{\text{IND-CPA}}(k) = 2 \cdot Adv_{\mathfrak{B}}^{\text{IE-LWE}}(k).$$

Proof. Assume that Σ is not secure in the sense of IND-CPA. Then, there exists an adversary $\mathfrak{A}$ who breaks Σ in polynomial time with non-negligible advantage

$$Adv_{\Sigma,\mathfrak{A}}^{\text{IND-CPA}}(k) \geq \epsilon(k),$$

where k is a security parameter. By using $\mathfrak{A}$, we construct an algorithm $\mathfrak{B}$ solving the IE-LWE problem in probabilistic polynomial time as follows. Without loss of generality, we assume $\mathfrak{B}$ outputs 1 when it decides that the input is sampled from T_X, and otherwise outputs 0.

Assume an oracle $\mathcal{O}$ that picks set $S \hookleftarrow U(\{T_X, U_X - T_X\})$ and samples from the set of S uniformly at random. Algorithm $\mathfrak{B}$ first calls $\mathcal{O}$ to get a sample $(X^{'}(x,y), C^{'}(x,y))$ from S. Then, the algorithm runs $\mathfrak{A}$ with the public key $X(x,y)(= \ell X^{'}(x,y) \in \mathfrak{X}(\Gamma_X,\ell)/R_q)$. Here, $X(x,y)$ is chosen uniformly at random from $\mathfrak{X}(\Gamma_X,\ell)/R_q$ since the map $X'(x,y) \to \ell X^{'}(x,y)$ is invertible due to the invertibility of ℓ modulo q.

When $\mathfrak{A}$ outputs challenge messages $m_0(t), m_1(t) \in R_\ell$, the algorithm $\mathfrak{B}$ picks b either 0 or 1 uniformly at random, computes the challenge ciphertext $c(x,y) = \ell \cdot C^{'}(x,y) + m_b(t) \in \mathfrak{F}_{\Gamma_e}/R_q$, and returns $c(x,y)$ to $\mathfrak{A}$. Finally, when $\mathfrak{A}$ outputs its guess $b^{'}$ for b, the algorithm $\mathfrak{B}$ outputs 1 if $b^{'} = b$ and 0 otherwise. Here, $c(x,y)$ is calculated as follows.

$$c(x,y) = \ell \cdot C^{'}(x,y) + m_b(t) = m_b(t) + X(x,y)r(x,y) + \ell \cdot e(x,y)$$

If the sample $(X^{'}(x,y), C^{'}(x,y))$ is from T_X, then it is impossible to distinguish $c(x,y)$ from an element chosen from the ciphertext space uniformly at randombecause $r(x,y)$, and $e(x,y)$ are chosen from $\mathfrak{F}_{\Gamma_r}/R_q$ and $\mathfrak{F}_{\Gamma_e}/R_\ell$, respectively, uniformly randomly. If the algorithm $\mathfrak{A}$ outputs $b^{'} = b$ with non-negligible advantage $Adv^{\text{IND-CPA}}_{\Sigma,\mathfrak{A}}(k)$, then we can calculate $Adv^{\text{IND-CPA}}_{\Sigma,\mathfrak{A}}(k)$ as follows.

$$\begin{array}{ll} & Adv^{\text{IND-CPA}}_{\Sigma,\mathfrak{A}}(k) \\ = & |Pr[b = b^{'} | (X^{'}(x,y), C^{'}(x,y)) \overset{U}{\leftarrow} T_X] - Pr[b \neq b^{'} | (X^{'}(x,y), C^{'}(x,y)) \overset{U}{\leftarrow} T_X]| \\ = & |Pr[\mathfrak{B}(X^{'}(x,y), C^{'}(x,y)) \to 1 | (X^{'}(x,y), C^{'}(x,y)) \overset{U}{\leftarrow} T_X] \\ & -Pr[\mathfrak{B}(X^{'}(x,y), C^{'}(x,y)) \to 0 | (X^{'}, C^{'}) \overset{U}{\leftarrow} T_X]| \\ = & |Pr[\mathfrak{B}(X^{'}(x,y), C^{'}(x,y)) \to 1 | (X^{'}(x,y), C^{'}(x,y)) \overset{U}{\leftarrow} T_X] \\ & -(1 - Pr[\mathfrak{B}(X^{'}(x,y), C^{'}(x,y)) \to 1 | (X^{'}(x,y), C^{'}(x,y)) \overset{U}{\leftarrow} T_X])| \\ = & |2Pr[\mathfrak{B}(X^{'}(x,y), C^{'}(x,y)) \to 1 | (X^{'}(x,y), C^{'}(x,y)) \overset{U}{\leftarrow} T_X] - 1| \\ = & 2|Pr[\mathfrak{B}(X^{'}(x,y), C^{'}(x,y)) \to 1 | (X^{'}(x,y), C^{'}(x,y)) \overset{U}{\leftarrow} T_X] - 1/2|. \end{array} \tag{25}$$

If the sample is picked from the set $U_X - T_X$, then the map

$$C^{'}(x,y) \mapsto m_b(t) + \ell \cdot C^{'}(x,y) (= c(x,y)) (\in \mathfrak{F}_{\Gamma_e}/R_q)$$

is invertible, since

$$c(x,y) \mapsto \ell^{-1}(c(x,y) - m_b(t)) (\in \mathfrak{F}_{\Gamma_e}/R_q).$$

Then, $c(x,y)$ is uniformly randomly in $\mathfrak{F}_{\Gamma_e}/R_q$, and independent of b. It follows that $\mathfrak{B}$ outputs 1 with probability 1/2.

We are able to compute $Adv^{\text{IE-LWE}}_{\mathfrak{B}}(k)$ as follows.

$$\begin{array}{rcl} Adv^{\text{IE-LWE}}_{\mathfrak{B}}(k) & = & |Pr[\mathfrak{B}(X^{'}(x,y), C^{'}(x,y)) \to 1 | (X^{'}(x,y), C^{'}(x,y)) \overset{U}{\leftarrow} T_X] \\ & & -Pr[\mathfrak{B}(X^{'}(x,y), C^{'}(x,y)) \to 1 | (X^{'}(x,y), C^{'}(x,y)) \overset{U}{\leftarrow} U_X - T_X]| \\ & = & |Pr[\mathfrak{B}(X^{'}(x,y), C^{'}(x,y)) \to 1 | (X^{'}(x,y), C^{'}(x,y)) \overset{U}{\leftarrow} T_X] - 1/2| \end{array}$$

Comparing the equation (25), we have

$$Adv^{\text{IND-CPA}}_{\Sigma,\mathfrak{A}}(k) = 2 \cdot Adv^{\text{IE-LWE}}_{\mathfrak{B}}(k).$$

This is a contradiction to the assumption, since a polynomial time algorithm $\mathfrak{B}$ satisfying $Adv^{\text{IE-LWE}}_{\mathfrak{B}}(k) \geq \epsilon(k)/2$ can be constructed. We conclude the desired claim. $\square$

In addition, one can make the Giophantus encryption scheme IND-CCA2 secure by using well-known conversions, such as those in [14]. However, the converted scheme is no longer homomorphic.

9 Security analysis

In this section, we introduce two possible attacks for the IE-LWE assumption. Other attacks against ASC, which this scheme was developed from, cannot be applied to this problem. For example, the ideal decomposition attack described in section 7.1 does not work on our scheme because our scheme does not have a product structure such as $m(x,y)s(x,y)$ in (21).

9.1 The Linear Algebra attack

For a given pair of polynomials $(X(x,y),Y(x,y))$, we can determine that $(X(x,y),Y(x,y))$ is sampled from T_X if we find $r \in \mathfrak{F}_{\Gamma_r}/R_q$ and $e \in \mathfrak{F}_{\Gamma_{Xr}}/R_\ell$ such that $Y(x,y) = X(x,y)r(x,y) + e(x,y)$.

The IE-LWE searching problem, which finds polynomials $r(x,y)$ and $e(x,y)$ of this type, can be solved by using the Linear Algebra attack (see Section 7.1) as follows. We construct a system of linear equations by comparing the coefficients of x^iy^j in the relation

$$\sum_{(i,j)\in\Gamma_e} d_{ij}(t)x^iy^j = \left(\sum_{(i,j)\in\Gamma_X} a_{ij}(t)x^iy^j\right)\left(\sum_{(i,j)\in\Gamma_r} r_{ij}(t)x^iy^j\right) + \left(\sum_{(i,j)\in\Gamma_e} e_{ij}(t)x^iy^j\right), \tag{26}$$

where $r_{ij}(t)$ and $e_{ij}(t)$ are elements of R_q and R_ℓ, respectively.

In the case $\deg X = \deg r = 1$, we can express X,r,e, and Y in the following manner.

$$\begin{aligned} X(x,y) = &\quad a_{10}(t)x + a_{01}(t)y + a_{00}(t) \\ r(x,y) = &\quad r_{10}(t)x + r_{01}(t)y + r_{00}(t) \\ e(x,y) = &\quad e_{20}(t)x^2 + e_{11}(t)xy + e_{02}(t)y^2 + e_{10}(t)x + e_{01}(t)y + e_{00}(t) \\ Y(x,y) = &\quad d_{20}(t)x^2 + d_{11}(t)xy + d_{02}(t)y^2 + d_{10}(t)x + d_{01}(t)y + e_{00}(t) \end{aligned} \tag{27}$$

In this section, we employ a small example (28),

$$\begin{aligned} X(x,y) = &\quad (818+1072t)x + (301+264t)y + (371+916t) \\ (u_x,u_y) = &\quad (1+3t, 3+2t) \\ r(x,y) = &\quad (1234+83t)*x + (188+675t)*y + (853+1285t) \\ e(x,y) = &\quad 3x^2 + (2+t)xy + 3ty^2 + (1+2t)*x + 2y + (2+t), \end{aligned} \tag{28}$$

to clarify the attack procedure. Here, $n = 2, \ell = 4, q = 1459$, and a small solution (u_x,u_y) satisfies $X(u_x(t),u_y(t)) = 0$. Then, $Y(x,y)(= X(x,y)r(x,y) + e(x,y))$ is

$$Y(x,y) = (1223+315t)*x^2+(1402+1442t)xy+(1348+403t)y^2+(425+48t)x+(123+179t)y+(968+426t).$$

When this sample (X,Y) is given by the IE-LWE oracle, we can establish simultaneous linear equations (29) by comparing coefficients from both sides of the equation $Y(x,y) = X(x,y)r(x,y) + e(x,y)$, where

$r(x, y)$ and $e(x, y)$ are unknown.

$$\begin{aligned}
&a_{10}(t)r_{10}(t) + e_{20}(t) = d_{20}(t) \\
&a_{10}(t)r_{01}(t) + a_{01}(t)r_{10}(t) + e_{11}(t) = d_{11}(t) \\
&a_{01}(t)r_{01}(t) + e_{02}(t) = d_{02}(t) \\
&a_{10}(t)r_{00}(t) + a_{00}(t)r_{10}(t) + e_{10}(t) = d_{10}(t) \\
&a_{01}(t)r_{00}(t) + a_{00}(t)r_{01}(t) + e_{01}(t) = d_{01}(t) \\
&a_{00}(t)r_{00}(t) + e_{00}(t) = d_{00}(t)
\end{aligned} \tag{29}$$

In the case of example (28), we can write $r_{ij}(t) = r_{ij0} + r_{ij1}t$, where r_{ij0} and r_{ij1} are variables valued at $\{0, \cdots q - 1\}$ in F_q, and also write $e_{ij}(t) = e_{ij0} + e_{ij1}t$, where e_{ij0} and e_{ij1} are variables valued at $\{0, \cdots \ell - 1\}$ in F_q.

By using the example (28) and considering (X, Y), we can specify the equation (29) as follows.

$$\begin{aligned}
&(818 + 1072t)(r_{100} + r_{101}t) + e_{200} + e_{201}t = 1223 + 315t \\
&(818 + 1072t)(r_{010} + r_{011}t) + (301 + 264t)(r_{100} + r_{101}t) + e_{110} + e_{111}t = 1402 + 1442t \\
&(301 + 264t)(r_{010} + r_{011}t) + e_{020} + e_{021}t = 1348 + 403t \\
&(818 + 1072t)(r_{000} + r_{001}t) + (371 + 916t)(r_{100} + r_{101}t) + e_{100} + e_{101}t = 425 + 48t \\
&(301 + 264t)(r_{000} + r_{001}t) + (371 + 916t)(r_{010} + r_{011}t) + e_{010} + e_{011}t = 123 + 179t \\
&(371 + 916t)(r_{000} + r_{001}t) + e_{000} + e_{001}t = 968 + 426t
\end{aligned} \tag{30}$$

The system has a solution space with dimensionality at least 6 since there are 18 variables and 12 equations. In the general case, a linear system obtained by this attack has a solution space with dimensionality at least $3n$ since the system has $9n$ variables and $6n$ equations.

If we can find a solution such that the values $e_{ij}(t)$ are in R_ℓ, then we conclude that $(X(x, y), Y(x, y))$ is in T_X. We can find them exactly by an exhaustive search for the polynomial $e(x, y)$, but this attack can be avoided by increasing $\#\Gamma_e = 6n$ to

$$((\ell - 1)\ell^{n-1})^{6n} > 2^k,$$

where k is a security parameter.

We employ a lattice-reduction attack to find a suitable small e_{ij}. Let us represent $a \in R_q$ as a vector $(a_0, a_1, \cdots, a_{n-2}, a_{n-1})$ for

$$a = a_0 + a_1t + \cdots + a_{n-2}t^{n-2} + a_{n-1}t^{n-1}$$

on F_q. When the elements $b, c \in R_q$ are represented in the same manner as a, we can express $ab + c$ as

$$\begin{pmatrix} b_0 & b_1 & \cdots & b_{n-2} & b_{n-1} \end{pmatrix} \begin{pmatrix} a_0 & a_1 & \cdots & a_{n-2} & a_{n-1} \\ a_{n-1} & a_0 & \cdots & a_{n-3} & a_{n-2} \\ a_{n-2} & a_{n-1} & \cdots & a_{n-4} & a_{n-3} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_1 & a_{n-1} & \cdots & a_{n-1} & a_0 \end{pmatrix} + \begin{pmatrix} c_0 & c_1 & \cdots & c_{n-2} & c_{n-1} \end{pmatrix} \tag{31}$$

on F_q.

Using this expression, the first equation of (29) is described as

$$\vec{r_{10}}A_{10} + \vec{e_{20}} = \vec{d_{20}}, \tag{32}$$

where A_{10} is expressed as

$$A_{10} = \begin{pmatrix} a_0 & a_1 & \cdots & a_{n-2} & a_{n-1} \\ a_{n-1} & a_0 & \cdots & a_{n-3} & a_{n-2} \\ a_{n-2} & a_{n-1} & \cdots & a_{n-4} & a_{n-3} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_1 & a_2 & \cdots & a_{n-1} & a_0, \end{pmatrix}$$

and r_{10},e_{20}, and d_{20} are denoted by

$$\begin{aligned} \vec{r_{10}} &= \begin{pmatrix} r_{100} & r_{101} & \cdots & r_{10n-2} & r_{10n-1} \end{pmatrix}, \\ \vec{e_{20}} &= \begin{pmatrix} e_{200} & e_{201} & \cdots & e_{20n-2} & e_{20n-1} \end{pmatrix}, \\ \vec{d_{20}} &= \begin{pmatrix} d_{200} & d_{201} & \cdots & d_{20n-2} & d_{20n-1} \end{pmatrix}, \end{aligned}$$

respectively. By using our sample, this relation can be described as

$$\begin{pmatrix} r_{100} & r_{101} \end{pmatrix} \begin{pmatrix} 818 & 1072 \\ 1072 & 818 \end{pmatrix} + \begin{pmatrix} e_{200} & e_{201} \end{pmatrix} = \begin{pmatrix} 1223 & 315 \end{pmatrix},$$

where each element is in F_q.

To apply lattice reduction to (32), we add the integer vector

$$\vec{u_{20}} = (u_{200}, \cdots, u_{20n-1})$$

to (32), such as

$$\vec{r_{10}}A_{10} + q\vec{u_{20}} + \vec{e_{20}} = \vec{d_{20}}. \tag{33}$$

This equation is defined over the integer ring $\mathbb{Z}$. Then we can consider an integer lattice

$$\mathcal{L}_{LAA_1} = \begin{pmatrix} A_{10} \\ qI_n \end{pmatrix},$$

where I_n denotes the $n \times n$ unit matrix. By using the example (28),

$$\begin{pmatrix} r_{100} & r_{101} & u_{100} & u_{101} \end{pmatrix} \begin{pmatrix} 818 & 1072 \\ 1072 & 818 \\ 1459 & 0 \\ 0 & 1459 \end{pmatrix} + \begin{pmatrix} e_{200} & e_{201} \end{pmatrix} = \begin{pmatrix} 1223 & 315 \end{pmatrix}.$$

If we find a point v closest to $\vec{d_{20}}$ in the lattice $\mathcal{L}$, then we can conclude that $\vec{v} = \pm\vec{e_{20}}$ with high probability since

$$\vec{r_{10}}A_{10} + q\vec{u_{20}} = \vec{d_{20}} - \vec{e_{20}}\,.$$

Therefore, we need to find the vector closest to $\vec{d_{20}}$ in the lattice $\mathcal{L}_1$ to find $\vec{e_{20}}$, since the vectors $\vec{r_{20}}$ and $\vec{u_{20}}$ corresponding to $\vec{e_{20}}$ are found at the same time.

In the same way, $\pm\vec{e_{11}}$,$\vec{r_{10}}$, and $\vec{r_{01}}$ can be detected from a point w closest to the $\vec{d_{11}}$ in the lattice

$$\begin{pmatrix} A_{10} \\ A_{01} \\ qI_n \end{pmatrix}.$$

By using our sample,

$$\mathcal{L}_{LAA_2} = \begin{pmatrix} 301 & 264 \\ 264 & 301 \\ 818 & 1072 \\ 1072 & 818 \\ 1459 & 0 \\ 0 & 1459 \end{pmatrix}.$$

Therefore, we need to consider all equations in (29) simultaneously. Doing so, we see that the linear algebra attack can be reduced to the closest-vector problem (CVP) on the lattice

$$\mathcal{L}_{LAA} = \begin{pmatrix} A_{10} & A_{01} & & A_{00} & & \\ & A_{10} & A_{01} & & A_{00} & \\ & & & A_{10} & A_{01} & A_{00} \\ qI_n & & & & & \\ & qI_n & & & & \\ & & qI_n & & & \\ & & & qI_n & & \\ & & & & qI_n & \\ & & & & & qI_n \end{pmatrix} \tag{34}$$

and the vector $\vec{d} = \begin{pmatrix} \vec{d_{20}} & \vec{d_{11}} & \vec{d_{02}} & \vec{d_{10}} & \vec{d_{01}} & \vec{d_{00}} \end{pmatrix}$, where the blank spaces in (29) indicate zero matrices.

To specify the example, the lattice (34) is described as follows.

$$\mathcal{L}_{LAA} = \begin{pmatrix} 818 & 1072 & 301 & 264 & 0 & 0 & 371 & 916 & 0 & 0 & 0 & 0 \\ 1072 & 818 & 264 & 301 & 0 & 0 & 916 & 371 & 0 & 0 & 0 & 0 \\ 0 & 0 & 818 & 1072 & 301 & 264 & 0 & 0 & 371 & 916 & 0 & 0 \\ 0 & 0 & 1072 & 818 & 264 & 301 & 0 & 0 & 916 & 371 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 818 & 1072 & 301 & 264 & 371 & 916 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1072 & 818 & 264 & 301 & 916 & 371 \\ 1459 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1459 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1459 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1459 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 \end{pmatrix}$$

The Hermite normal form is calculated as

$$B = \begin{pmatrix} 1 & 0 & 0 & 0 & 116 & 982 & 0 & 0 & 447 & 93 & 1220 & 712 \\ 0 & 1 & 0 & 0 & 982 & 116 & 0 & 0 & 93 & 447 & 712 & 1220 \\ 0 & 0 & 1 & 0 & 1257 & 183 & 0 & 0 & 239 & 1311 & 0 & 0 \\ 0 & 0 & 0 & 1 & 183 & 1257 & 0 & 0 & 1311 & 239 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1257 & 183 & 239 & 1311 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 183 & 1257 & 1311 & 239 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 \end{pmatrix}.$$

This is a special case of a q-array lattice, such as

$$\begin{pmatrix} I & A \\ O & qI \end{pmatrix}. \tag{35}$$

Here, A consists of sparse cyclic matrices. The difference will be discussed in Section 9.3.

While the CVP on lattices is NP-hard, we need to apply known approximation algorithms for solving CVP to evaluate appropriate parameters. This document introduces the embedding technique, which is an efficient method to solve CVP. To simplify, we start by describing the embedding technique in the case of $\deg X(x,y) = \deg r(x,y) = 1$. The relation $\vec{r}A + q\vec{u} + \vec{e} = \vec{d}$ is satisfied, where

$$\begin{aligned}
\vec{r} &= \begin{pmatrix} r_{100} & r_{101} & r_{010} & r_{011} & r_{000} & r_{001} \end{pmatrix} \\
\vec{u} &= \begin{pmatrix} u_{200} & u_{201} & u_{110} & u_{111} & u_{020} & u_{021} & u_{100} & u_{101} & u_{010} & u_{011} & u_{000} & u_{001} \end{pmatrix} \\
\vec{e} &= \begin{pmatrix} e_{200} & e_{201} & e_{110} & e_{111} & e_{020} & e_{021} & e_{100} & e_{101} & e_{010} & e_{011} & e_{000} & e_{001} \end{pmatrix}.
\end{aligned}$$

Since the vector $\vec{e}$ is short, we may find $\vec{e}$ by calculating the vector in the lattice $(A|qI_n)$ closest to the vector $\vec{d}$. If vector $\vec{c}$ is the closest vector, then there is a possibility that the vector $\vec{e}$ is equal to the vector $\pm(\vec{d} - \vec{c})$. In our sample, the correct vector of $\vec{e}$ is

$$\vec{e} = \begin{pmatrix} 2 & 3 & 1 & 1 & 2 & 3 & 3 & 1 & 3 & 2 & 3 & 2 \end{pmatrix}. \tag{36}$$

This document shows computational experiments intended to find the closest vector by the embedding technique.

The embedding technique finds the closest vector from the lattice found by adding the target vector to the original lattice, such as

$$\mathcal{L}_d = \begin{pmatrix} B & \vec{0} \\ \vec{d} & \lambda \end{pmatrix},$$

where $\vec{d}$ is a target vector and λ is a small integer, such as 1 or 2. When we reduce the lattice $\mathcal{L}_d$ by applying the LLL or BKZ method, we can find the vector $\vec{e}$ as a row vector whose last element equals λ or $-\lambda$ in the reduced lattice.

For the example (28), the embedded lattice is

$$\begin{pmatrix}
818 & 1072 & 301 & 264 & 0 & 0 & 371 & 916 & 0 & 0 & 0 & 0 & 0 \\
1072 & 818 & 264 & 301 & 0 & 0 & 916 & 371 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 818 & 1072 & 301 & 264 & 0 & 0 & 371 & 916 & 0 & 0 & 0 \\
0 & 0 & 1072 & 818 & 264 & 301 & 0 & 0 & 916 & 371 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 818 & 1072 & 301 & 264 & 371 & 916 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1072 & 818 & 264 & 301 & 916 & 371 & 0 \\
1459 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1459 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1459 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1459 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1459 & 0 \\
1223 & 315 & 1402 & 1442 & 1348 & 403 & 425 & 48 & 123 & 179 & 968 & 426 & 2
\end{pmatrix},$$

since the vector $\vec{d}$ is $\begin{pmatrix} 1223 & 315 & 1402 & 1442 & 1348 & 403 & 425 & 48 & 123 & 179 & 968 & 426 \end{pmatrix}$. Applying LLL to the lattice, we can detect a shortest vector

$$\begin{pmatrix} 3 & 0 & 2 & 1 & 0 & 3 & 1 & 2 & 2 & 0 & 2 & 1 \end{pmatrix}$$

as the first row from the reduced lattice

$$\begin{pmatrix}
3 & 0 & 2 & 1 & 0 & 3 & 1 & 2 & 2 & 0 & 2 & 1 & 2 \\
14 & 7 & -4 & -11 & -2 & 1 & -13 & 0 & -6 & 15 & -10 & 6 & 0 \\
-11 & 14 & -12 & 15 & 6 & -3 & -7 & 10 & 3 & -1 & 7 & -4 & 2 \\
12 & 15 & -1 & -8 & 16 & -11 & -8 & 1 & 1 & 12 & 9 & -7 & 4 \\
7 & 14 & -11 & -4 & 1 & -2 & 0 & -13 & 15 & -6 & 6 & -10 & 0 \\
2 & 4 & -6 & 12 & 19 & -13 & 11 & -5 & 7 & -3 & -4 & 10 & 4 \\
1 & -15 & 20 & -1 & 3 & -3 & -12 & -9 & 6 & 23 & -7 & -2 & 2 \\
2 & -3 & -4 & -12 & 21 & 12 & -1 & 11 & -16 & -2 & -11 & -12 & 4 \\
3 & 11 & -2 & 10 & -1 & -1 & 1 & 11 & 10 & 5 & -17 & -24 & 4 \\
8 & 11 & -12 & 19 & -4 & -9 & 19 & -3 & -6 & 11 & 9 & -9 & 2 \\
7 & 16 & 17 & 2 & 9 & 7 & 5 & -3 & -22 & 0 & -3 & 3 & -18 \\
8 & -10 & -3 & 13 & -9 & -17 & -4 & 10 & -12 & -20 & -14 & 2 & 22 \\
-61 & 61 & 36 & -23 & -21 & 23 & 28 & -30 & -39 & 18 & -23 & 13 & 100
\end{pmatrix}.$$

The vector equals the correct vector $\vec{e}$ in (36).

9.1.1 Subring restriction technique

The linear algebra attack also works in a subring of $R_q[x, y]$. The ring $R_q[x, y]$ has three types of subrings. These are $\tilde{R_q}[x, y]$, $\tilde{R_q}[x, f(t)]$, and $\tilde{R_q}[f(t), y]$, where $\tilde{R_q}$ is a sub-ring of $R_q[x, y]$ and $f(t)$ is an element of $\tilde{R_q}$. If n is a composite number written as $n = ab$ where a and b are integers, then the quotient polynomial $t^n - 1$ can be factored into $(t^a - 1)$ and $(t^{a(b-1)} + t^{a(b-2)} + \cdots + t^a + 1)$. The ring $F_q[t]/(t^a - 1)$ is a subring of R_q. The paper [17] suggests that n be chosen as prime since our scheme employs the same algebra R_q as NTRU. From now on, we assume n is prime and consider the effect of the attack in the subrings $R_q[x, f(t)]$ and $R_q[f(t), y]$. Since these subrings have the same structure, it is sufficient to consider $R_q[x, f(t)]$.

This section describes how this technique works on the ring $\tilde{R_q}[x, f(t)]$ with an example (28).

Let $(X(x, y), Y(x, y))$ be a sample from a distribution of T_X or $U_X - T_X$. Then we can detect whether the pair belongs to T_X or not by solving the equation $Y(x, y) = X(x, y)r(x, y) + e(x, y)$ for $r \in \mathfrak{F}_{\Gamma_r}/R_q$ and $e \in \mathfrak{F}_{\Gamma_{X_r}}/R_\ell$. The lattice-reduction algorithms (which have complexity exponential with respect to the dimensionality of the lattice in general) can be applied as described above. By using the subring technique, we can expect to make the reduction easier than the original problem, since that allows reducing the dimensionality of the lattice.

For simplicity, let $f(t)$ be zero. Then the polynomials (27) can be described as follows.

$$\begin{aligned} X(x,0) =& \quad a_{10}(t)x + a_{00}(t) \\ r(x,0) =& \quad r_{10}(t)x + r_{00}(t) \\ e(x,0) =& \quad e_{20}(t)x^2 + e_{10}(t)x + e_{00}(t) \\ Y(x,0) =& \quad d_{20}(t)x^2 + d_{10}(t)x + e_{00}(t) \end{aligned}$$

Hence, the example (28) becomes the following.

$$\begin{aligned} X(x,y) =& \quad (818+1072t)x + (301+264t)y + (371+916t) \\ Y(x,y) =& \quad (1223+315t)*x^2 + (1402+1442t)xy + (1348+403t)y^2 + (425+48t)x + (123+179t)y \\ & \quad +(968+426t) \\ r(x,y) =& \quad (1234+83t)*x + (188+675t)*y + (853+1285t) \\ e(x,y) =& \quad 3x^2 + (2+t)xy + 3ty^2 + (1+2t)*x + 2y + (2+t) \end{aligned} \tag{37}$$

Then, we can find some partial solution of $r(x,y)$ and $e(x,y)$, such as $e_{20}(t)$, $e_{10}(t)$, and $e_{00}(t)$, by solving the following linear equations.

$$\begin{aligned} & a_{10}(t)r_{10}(t) + e_{20}(t) = d_{20}(t) \\ & a_{10}(t)r_{00}(t) + a_{00}(t)r_{10}(t) + e_{10}(t) = d_{10}(t) \\ & a_{00}(t)r_{00}(t) + e_{00}(t) = d_{00}(t) \end{aligned} \tag{38}$$

By using the example (37) and considering (X,Y), we can specify the equation (29) as follows.

$$\begin{aligned} & (818+1072t)(r_{100}+r_{101}t) + e_{200} + e_{201}t = 1223+315t \\ & (818+1072t)(r_{000}+r_{001}t) + (371+916t)(r_{100}+r_{101}t) + e_{100} + e_{101}t = 425+48t \\ & (371+916t)(r_{000}+r_{001}t) + e_{000} + e_{001}t = 968+426t \end{aligned} \tag{39}$$

This system has a solution space whose dimensionality is 4 since there are 10 variables and 6 equations. In the general case, a linear system obtained in this attack has a solution space with at least $3n$ dimensions since the system has $5n$ variables and $2n$ equations.

If we can find a solution such that elements $e_{i0}(t)$ are in R_ℓ, then we can conclude that $(X(x,y), Y(x,y))$ is in T_X with non-negligible probability $1 - 1/q^6$. We need an n that satisfies

$$((\ell-1)\ell^{n-1})^{3n} > 2^k$$

to avoid an exhaustive attack on the polynomial $e(x,y)$, where k is a security parameter.

We can establish the lattice $Lat_{LAA}^{R_q[x,0]}$ instead of Lat_{LAA} to find the solution by solving the CVP.

$$\mathcal{L}_{LAA}^{R_q[x,0]} = \begin{pmatrix} A_{10} & A_{00} & \\ & A_{10} & A_{00} \\ qI_n & & \\ & qI_n & \\ & & qI_n \end{pmatrix} \tag{40}$$

and the vector $\vec{d} = \begin{pmatrix} \vec{d_{20}} & \vec{d_{10}} & \vec{d_{00}} \end{pmatrix}$, where the blank spaces in (40) indicate zero matrices.

To specify the example, the lattice (40) is described as follows.

$$\mathcal{L}_{LAA} = \begin{pmatrix} 818 & 1072 & 371 & 916 & 0 & 0 \\ 1072 & 818 & 916 & 371 & 0 & 0 \\ 0 & 0 & 818 & 1072 & 371 & 916 \\ 0 & 0 & 1072 & 818 & 916 & 371 \\ 1459 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1459 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1459 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1459 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1459 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1459 \end{pmatrix},$$

The Hermite normal form is calculated as B.

$$B = \begin{pmatrix} 1 & 0 & 0 & 0 & 1220 & 712 \\ 0 & 1 & 0 & 0 & 712 & 1220 \\ 0 & 0 & 1 & 0 & 239 & 1311 \\ 0 & 0 & 0 & 1 & 1311 & 239 \\ 0 & 0 & 0 & 0 & 1459 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1459 \end{pmatrix}$$

This matrix can be described as

$$\begin{pmatrix} I & & A \\ & \mathrm{I} & B \\ & & qI \end{pmatrix}, \tag{41}$$

where A and B cyclic matrices. The difference is discussed in Section 9.3.

By using the embedding technique, we can find a shortest vector

$$\begin{pmatrix} 3 & 0 & 1 & 2 & 2 & 1 & 2 \end{pmatrix}$$

as the first row from the reduced lattice

$$\begin{pmatrix} 3 & 0 & 1 & 2 & 2 & 1 & 2 \\ 2 & 3 & -2 & 8 & -4 & -3 & 0 \\ -3 & -2 & 4 & 1 & 8 & 1 & -4 \\ -6 & 6 & 2 & 3 & -3 & -2 & 4 \\ -3 & -2 & -8 & 2 & 3 & 4 & 0 \\ -5 & -3 & 5 & 7 & -6 & 2 & 2 \\ 1 & -9 & 4 & -3 & -1 & -5 & 6 \end{pmatrix}.$$

The vector equals the correct vector $\vec{e}$ in (36).

Let us assume that $f(t)$ is not zero. Since the polynomial $e(x, y)$ can be described by

$$e(x, f(t)) = e_{20}(t)x^2 + (e_{11}(t)f(t) + e_{10}(t))x + (e_{02}(t)f(t)^2 + e_{01}(t)f(t) + e_{00}(t)),$$

the coefficients and the degree of $f(t)$ should be small enough to detect the polynomial $e(x, f(t))$ with small coefficients. Since $MC(e)$ can be estimated from $MC(e) \leq n^2 \cdot \ell MC(f(t))^2$, we can see the case

$f(t) = 0$ is most effective for detecting the polynomial $e(x, f(t))$. So for the discussion of the subring technique in Section 9.3 we consider only the ring $R_q[x, f(t)]$.

9.2 Key recovery attack

If a solution $(\tilde{u}_x(t), \tilde{u}_y(t)) \in R_q^2$ to $X(x, y) = 0$ (not necessarily the secret key), in which all coefficients are less than ℓ is found, then the IE-LWE problem can be solved with high probability, as follows. For an IE-LWE instance $(X(x, y), Y(x, y))$, if all coefficients of $\ell \cdot Y(\tilde{u}_x(t), \tilde{u}_y(t))$ are multiples of ℓ, then it can be concluded that (X, Y) is sampled from T_X. In fact, sampling (X, Y) from T_X implies that

$$\ell \cdot Y(\tilde{u}_x(t), \tilde{u}_y(t)) = \ell(X(\tilde{u}_x(t), \tilde{u}_y(t))r(\tilde{u}_x(t), \tilde{u}_y(t)) + e(\tilde{u}_x(t), \tilde{u}_y(t)) = \ell \cdot e(\tilde{u}_x(t), \tilde{u}_y(t)),$$

and $MC(e(\tilde{u}_x(t), \tilde{u}_y(t))) < q$ implies that all coefficients of $\ell \cdot e(\tilde{u}_x(t), \tilde{u}_y(t))$ are multiples of ℓ. On the other hand, if $(X(x, y), Y(x, y))$ is sampled from U_X, then the probability that all coefficients of $\ell \cdot Y(\tilde{u}_x(t), \tilde{u}_y(t))$ are multiples of ℓ is about $1/\ell^n$. Therefore if a small solution, such as $(\tilde{u}_x(t), \tilde{u}_y(t))$, can be found, then the IE-LWE problem can be solved with a probability higher than $1 - 1/\ell^n$ by checking whether all coefficients of $\ell \cdot Y(\tilde{u}_x(t), \tilde{u}_y(t))$ are multiples of ℓ. Since $n, \ell \geq 2$, the probability $1 - 1/\ell^n$ is at least $3/4$, which is non-negligible.

In the following, we consider the key recovery attack on our encryption scheme (i.e., finding the small solution belonging to R_ℓ^2, to $X(x, y) = 0$ over R_q, by using lattice reduction techniques). First, we consider the case $\deg X = 1$. In this case, we need to find $u_x(t), u_y(t) \in R_\ell^2$ satisfying

$$a_{10}(t)u_x(t) + a_{01}(t)u_y(t) + a_{00}(t) = 0. \tag{42}$$

We write this equation with a matrix and vectors, in the same manner as the algebraic attack described above, as

$$\begin{pmatrix} \vec{u_x} & \vec{u_y} \end{pmatrix} \begin{pmatrix} A_{10} \\ A_{01} \end{pmatrix} = \begin{pmatrix} -\vec{a_{00}} \end{pmatrix}, \tag{43}$$

where the vectors $\vec{u_x}$ and $\vec{u_y}$ corresponding to $u_x(t)$, and $u_y(t)$, respectively, with elements restricted to $\{0, \cdots, p-1\}$ in F_q.

As the Linear Algebra attack, we apply lattice reduction to find a small solution $(u_x(t), u_y(t))$. Then we add the integer vector $\vec{u}$

$$\vec{u} = (u_0, \cdots, u_{n-1})$$

to (42), such as

$$\vec{u_x}A_{10} + \vec{u_y}A_{01} + q\vec{u} = -\vec{a_{00}}. \tag{44}$$

This equation is defined over the integer ring $\mathbb{Z}$. We consider an integer matrix

$$A = \begin{pmatrix} A_{10} \\ A_{01} \\ qI_n \end{pmatrix}$$

and

$$\begin{pmatrix} \vec{u_x} & \vec{u_y} & \vec{u} \end{pmatrix} \begin{pmatrix} A_{10} \\ A_{01} \\ qI_n \end{pmatrix} = \begin{pmatrix} -\vec{a_{00}} \end{pmatrix}. \tag{45}$$

Then, we consider the lattice

$$\mathcal{L}_{KRA} = \{ \vec{x} \mid \vec{x}A = \vec{0} \} \tag{46}$$

and let $\vec{v}$ be a solution to the system (42). Then, any solution of (42) can be written as $\vec{v}+\vec{w}$ ($\vec{w} \in \mathcal{L}_{KRA}$). Observe that our target solution $(\vec{u_x}, \vec{u_y}, \vec{u})$ of (42) is expected to be relatively short among the solutions of (42) because all of the coefficients of $u_x(t)$ and $u_y(t)$ are restricted to $\{0, \cdots, \ell-1\}$, where ℓ is much smaller than q. This observation leads us to an approach to the key-recovery attack as follows. First, we solve the system and find its solution space $\mathcal{L}_{KRA}$ and a solution $\vec{v}$. Second, we solve CVP to find the vector $\vec{w}$ closest to $\vec{v}$, and then $\vec{v} - \vec{w}$ is the smallest solution of (42) and is expected to be our target solution $(\vec{u_x}, \vec{u_y}, \vec{u})$.

We recall the example (28) to see the relation in a concrete manner.

$$\begin{aligned} X(x,y) = &\quad (968+302t)x + (861+442t)y + (1109+271t) \\ (u_x, u_y) = &\quad (2+2t, 1+3t) \end{aligned}$$

Here, $n=2$,$\ell=4$, and $q=1459$. Then we define a small solution (u_x, u_y) as

$$\begin{aligned} u_x = &\quad u_{x0} + u_{x1}t \\ u_y = &\quad u_{y0} + u_{y1}t \end{aligned},$$

where u_{x0},u_{x1},u_{y0}, and u_{y1} are variables valued at $\{0, \cdots, \ell-1\}$ in F_q. This gives

$$(968+302t)(u_{x0}+u_{x1}t) + (861+442t)(u_{y0}+u_{y1}t) + (1109+271t) = 0.$$

Moreover, we can transfer the above formula to the polynomial ring $\mathbb{Z}[t]$ by adding $q\dot{u}$, which is described by $u = u_0 + u_1 t$. Thus,

$$(968+302t)(u_{x0}+u_{x1}t) + (861+442t)(u_{y0}+u_{y1}t) + 1459(u_0+u_1t) + (1109+271t) = 0.$$

We can describe the above formula as

$$A = \begin{pmatrix} 968 & 302 \\ 302 & 968 \\ 861 & 442 \\ 442 & 861 \\ 1459 & 0 \\ 0 & 1459 \end{pmatrix}$$

and

$$\mathcal{L}_{KRA} = \begin{pmatrix} 1 & 0 & 1014 & 1033 & -912 & -917 \\ 0 & 1 & 1033 & 1014 & -917 & -912 \\ 0 & 0 & 1459 & 0 & -861 & -442 \\ 0 & 0 & 0 & 1459 & -442 & -861 \end{pmatrix}, \tag{47}$$

which is the same as the Hermite normal form. We can describe the lattice (47) as

$$\mathcal{L}_{KRA} = \begin{pmatrix} I & A & C \\ O & qI & D \end{pmatrix}, \tag{48}$$

where A,C, and D are cyclic matrices.

We also apply the embedding technique to find the lattice point of $\mathcal{L}^{+}_{KRA}$ that is closest to the solution $\vec{v}$. Let

$$\mathcal{L}^{+}_{KRA} = \begin{pmatrix} B & \vec{0}^T \\ \vec{v} & \lambda \end{pmatrix},$$

where λ is 2 and B is the lattice $\mathcal{L}_{KRA}$, and reduce the lattice $\mathcal{L}^{+}_{KRA}$ by applying LLL or BKZ. In a reduced lattice, we expect to find the vector $(u_x(t), u_y(t), u(t), \pm\lambda)$ as the row vector whose last element is equal to λ or $-\lambda$.

For the example, since we can take a solution to the system (42)

$$\vec{v} = \begin{pmatrix} 261060 & 0 & 0 & -458 & -173067 & -53767 \end{pmatrix},$$

we construct an embedding lattice such that

$$\mathcal{L}^{*}_{KRA} = \begin{pmatrix} 1 & 0 & 1014 & 1033 & -912 & -917 & 0 \\ 0 & 1 & 1033 & 1014 & -917 & -912 & 0 \\ 0 & 0 & 1459 & 0 & -861 & -442 & 0 \\ 0 & 0 & 0 & 1459 & -442 & -861 & 0 \\ 261060 & 0 & 0 & -458 & -173067 & -53767 & 2 \end{pmatrix},$$

and we obtained a reduced lattice

$$\begin{pmatrix} 2 & 2 & 1 & 3 & -4 & -4 & 2 \\ 7 & 7 & -16 & -4 & 0 & 0 & 12 \\ 1 & 3 & 20 & -16 & -9 & 1 & 2 \\ -13 & -12 & -1 & 6 & 0 & 5 & 26 \\ 35 & -42 & 6 & 4 & -17 & 17 & -6 \end{pmatrix}$$

by applying the LLL algorithm. So we find the shortest vector

$$v = \begin{pmatrix} 2 & 2 & 1 & 3 & -4 & -4 & 2 \end{pmatrix},$$

which corresponds to $(u_{x0}, u_{x1}, u_{y0}, u_{y1}, u_0, u_1, \lambda)$ from the first row. The vector $\begin{pmatrix} 2 & 2 & 1 & 3 \end{pmatrix}$ is equal to the correct vector $(u_{x0}, u_{x1}, u_{y0}, u_{y1}) = (2, 2, 1, 3)$.

Recent lattice attacks, such as the lattice-decoding attack and the subfield-lattice attack, do not apply to our scheme. See Subsection 9.4 for details.

9.2.1 Kernel technique

Moreover, we applied the same reduction to the lattice

$$\mathcal{L}'_{KRA} = \begin{pmatrix} I & A \\ O & qI \end{pmatrix}, \tag{49}$$

omitting the cyclic matrices C and D from the original $\mathcal{L}_{KRA}$ since these matrices are not related to $\vec{u_x}$ and $\vec{u_y}$ directly. If we can get a small solution from $\mathcal{L}'_{KRA}$, then we should consider that reduction.

The result in table 2 shows that the attack is also valid, that is, small solutions can be found by the reduction. The result tells us this attack is the most effective against the proposed cryptosystem since the attack did not fail until $n \leq 120$, which is larger than $\mathcal{L}_{KRA}$.

9.3 Dominant attack to the proposed system

By the discussion of the last two sections, dimension of the lattice to attack as follows.

Attack	Original LLA	Improved LLA	KRA
Rank	$6n$	$3n$	$2n$

This table shows the key recovery attack is dominant to evaluate security of the proposed system.

9.4 Further discussion on lattice attacks

In this section, we discuss and analyze the impact of other lattice attacks, such as a subfield lattice attack [22], on the proposed primitive.

9.4.1 Subfield lattice attack

Here, we discuss the subfield lattice attack in the context of our scheme. This attack can be applied to homomorphic variants of NTRU. The attack reduces the lattice problem on certain number fields to the problem on their appropriate subfields by using norm maps from the original number fields to the subfields.

NTRU variants (i.e., the NTRU on $\mathbb{Z}_q[x]/(x^{2^k}+1)$ and $\mathbb{Z}_q[x]/(x^p - x - 1)$ with prime numbers p and positive integers q) have been addressed in previous experiments by Kirchner et al. [22, Section 5]. There is no proper nontrivial subfield of the number field $\mathbb{Q}[x]/(x^p - x - 1)$, but the attack on $\mathbb{Z}_q[x]/(x^p - x - 1)$ succeeds for many parameters. We infer that the size of the parameter q is strongly related to the success or failure of the attack. As the size of q increases, the volume of the lattice becomes larger, and SVP on the lattice becomes easier. In fact, the subfield attacks on NTRU with relatively small q fail in some cases (see [22, Figures 1 and 2]). Moreover, the form $h = f/g$ of the public key for NTRU seems to have a positive effect on the attack, where f and g are secret polynomials with small coefficients and f is invertible in $\mathbb{Z}_q[x] = (x^{2^k} + 1)$ or $\mathbb{Z}_q[x] = (x^p - x - 1)$.

However, when comparing Table 5 in this document with [22, Figures 1 and 2], it is evident that the size of q in our scheme is much smaller than that in the NTRU variants. Moreover, there is a gap between the forms of the keys (public/secret keys) in our scheme and those in the above NTRU variants. The data show that the lattices derived from the two attacks on our scheme are very different from those derived from the subfield attacks on the above NTRU variants. Therefore, the subfield attack does not appear to be applicable to our scheme.

10 Appropriate parameter values

10.1 Embedding technique in $\mathcal{L}'_{KRA}$

This section intends to clear the mathematical structure of lattice $\mathcal{L}'_{KRA}$ which is associated with the key recovery attack, under the condition of $\deg X = \deg r = 1$ and $\ell = 4$.

Recall the discussion in the subsection 9.2.1, the lattice $\mathcal{L}'_{KRA}$ can be described as follows.

$$\mathcal{L}'_{KRA} = \begin{pmatrix} I & A \\ O & qI \end{pmatrix}, \tag{50}$$

where A is $n \times n$ cyclic lattice and I and qI are $n \times n$ identity matrix and q times identity matrix respectively. Then the lattice $\mathcal{L}'_{KRA}$ is a q-array lattice, where q is the minimum prime within the condition (11) such as

$$q > \sum_{k=0}^{2} (k+1) n^k \dot{(}\ell - 1)^{k+1} = 3 + 2 \cdot 3^2 n + 3 \cdot 3^3 n^2 = 81n^2 + 18n + 3 \tag{51}$$

then we choose q as the smallest prime larger than $81n^2 + 18n + 3$, so we conclude q is $O(n^2)$.

In this subsection, we investigate the structure of the lattice $\mathcal{L}^+_{KRA}$ which is described as

$$\mathcal{L}^+_{KRA} = \begin{pmatrix} I & A & 0 \\ O & qI & 0 \\ & \vec{v} & M \end{pmatrix}, \tag{52}$$

where $\vec{v}$ is a solution to the system (42) and M is the embedding factor. We choose $M = 2$. In our experiments, we used the embedding technique, which is a standard algorithm for solving CVP approximately and we use LLL/BKZ algorithm as a lattice-basis reduction algorithm.

It is clear that the rank of $\mathcal{L}^+_{KRA}$ is $2n$ and the determinant is $2q^n$. The lattice $\mathcal{L}^+_{KRA}$ has a shortest vector that is corresponding to the smallest solution $(u_x(t), u_y(t))$ whose coefficients are restricted within $\{0, 1, 2, 3\}$. Then the average norms of the shortest vectors v as follows.

$$||v|| \sim \sqrt{7n} \tag{53}$$

Our computing environment is as follows:

Table 2: Experimental results for the key-recovery attack by embedding technique($\ell = 4$)

n	q	Rank	Norm1	Norm2	Gap	Results	Time (s)
10	33149	21	8	186	22	Success	0.02
20	131059	41	12	619	50	Success	0.09
30	293791	61	15	1416	97	Success	0.26
40	521299	81	17	3236	191	Success	0.76
50	813623	101	19	6013	315	Success	1.77
60	1170751	121	21	11444	552	Success	3.52
70	1592659	141	22	20796	943	Success	6.45
80	2079401	161	24	37181	1563	Success	10.74
90	2630917	181	25	66292	2641	Success	57.79
100	3247243	201	27	106864	4026	Success	318.16
110	3928361	221	28	186219	6724	Success	788.46
120	4674289	241	29	307382	10474	Success	1361.19
130	5484979	261	373397	574752	2	Failure	2315.24

- CPU: AMD Opteron(TM) Processor 848
- Memory: 64 GB
- OS: Linux version 2.6.18-406.el5.centos.plus
- Software: fplll 4.0.0

We conducted the key-recovery attack to clear the characteristics of the lattice $\mathcal{L}'_{KRA}$ (see Section 9.2). We suppose that the key-recovery attack succeeds even if we find two polynomials with small coefficients $< \ell$ that differ from the correct secret key $(u_x(t), u_y(t))$.

The experimental results given in Table 2 show that the key-recovery attack for $\deg X = 1$ failed for $n \geq 130$, which is a much higher threshold than for the linear algebra attack. Here, the experiment took LLL algorithm to reduce lattices.

In the Table 2, Norm1 and Norm2 are the norms of the shortest vector v_1 and the second shortest vector v_2 in the reduced lattice respectively and Gap indicates the gap of Norm1 and Norm2 such that Gap = Norm2/Norm1. We conclude SVP of $\mathcal{L}^{+}_{KRA}$ is unique-SVP since the gap increases until the attack failed.

On the other hand, we may assume that

$$||\lambda_2(\mathcal{L}^{+}_{KRA})|| \approx GH(\mathcal{L}'_{KRA}) \tag{54}$$

in [6], where $||\lambda_2(\mathcal{L}^{+}_{KRA})||$ denote the norm of the smallest vector linearly independent from the shortest non-zero vector in the embedding lattice $\mathcal{L}^{+}_{KRA}$, and $GH(\mathcal{L}'_{KRA})$ is the Gaussian heuristic for the lattice $\mathcal{L}'_{KRA}$, namely

$$GH(\mathcal{L}'_{KRA}) = \sqrt{nq/\pi e}\ ,$$

Table 3: The behavior $log_2||b_i^*||$ and the comparison $||b_2^*||/||b_1^*||$ and $||b_2||/||b_1||$

Beta	slope	y-intercept	correlation coefficient	p-value	$\|\|b_2^*\|\|/\|\|b_1^*\|\|$	$\|\|b_2\|\|/\|\|b_1\|\|$
10	-0.08354	32.2740	> 0.999	< 0.001	4320402	4320505
20	-0.07493	31.2279	> 0.999	< 0.001	1783504	1783497

So we conclude the norm $||\lambda_2(\mathcal{L}_{KRA}^+)||$ increases as the n increases.

We investigate the behavior of $log||b_i^*|| \quad (i = 1, \cdots, 2n+1)$ at $n = 120$ which is the last n where all examples are succeed to attack in the Table 2. Let $(b_1, \cdots, b_{2n+1})$ be a sufficiently reduced basis of the lattice $\mathcal{L}_{KRA}^+$ and we write $(b_1^*, \cdots, b_{2n+1}^*)$ as a basis which is given by Gram-Schmidt orthonormalization from the basis $(b_1, \cdots, b_{2n+1})$.

The Fig. 6 shows the behavior, where the red line and the blue line indicate the behavior in the case of $\beta = 10$ and $\beta = 20$ respectively. The figure tells us the lattice $\mathcal{L}_{KRA}^{'}$ satisfies the geometric series assumption (GSA) and the slope of the line gentry decreases as the β increases. Therefore the norm of the second vector decreases as the β increases.

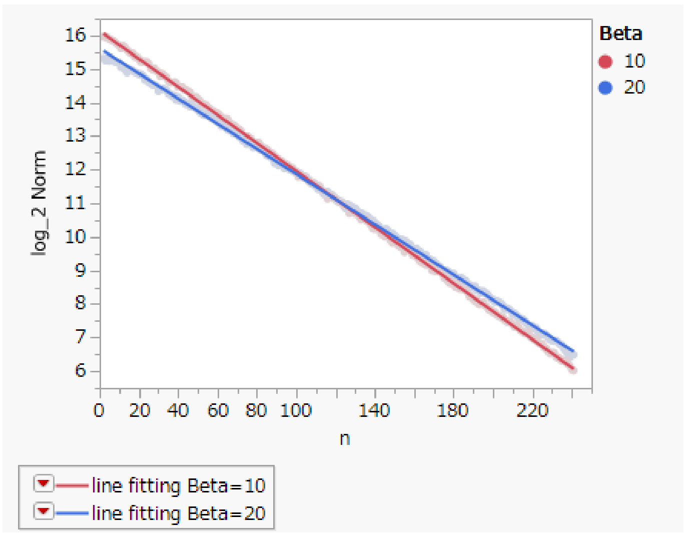

Fig. 6: Behavior of $log_2||b_i^*|| \quad (i = 2, \cdots, 2n+1)$

Table 3 shows the slope and the y-intercept of the fitting line described in Fig.6 and the correlation coefficient and the p-value indicate these lines well approximate the samples. The table 3 also shows comparison the values of $||b_2^*||/||b_1^*||$ and $||b_2||/||b_1||$ then we observe $||b_2^*||/||b_1^*||$ is almost same as $||b_2||/||b_1||$.

By the above observation, for the reduced basis b_i we can set as follows.

$$\begin{aligned} ||b_1^*|| = &\ \sqrt{7n} \\ ||b_2^*|| = &\ \delta^{2n}\sqrt{q} \\ ||b_i^*|| = &\ -slope \cdot ||b_{i-1}^*|| \ \ (i = 3, 4, .., 2n+1) \ \ \text{with} \ \ -1 < slope < 0 \ , \end{aligned} \tag{55}$$

where we assume that $||b_2^*||$ is equal to the non-zero shortest vector v obtained by the underlying the lattice reduction algorithm over the lattice $\mathcal{L}_{KRA}^{'}$ and δ denotes the root of Hermite factor which is defined

$$\delta = (||v||/(det\mathcal{L}_{KRA}^{'})^{1/2n})^{1/2n} \ .$$

In the Table 3 "slope" indicates the slope of the line of the GSA and thus we have the following relationship:

$$||b_1^*|| \cdot ||b_2^*|| \cdot ||b_3^*|| \cdots ||b_{2n+1}^*|| = det(\mathcal{L}_{KRA}^{+}) = 2q^n. \tag{56}$$

Then we can calculate

$$\begin{aligned} ||b_3^*|| = &\ \ |slope| \cdot ||b_2^*|| \\ ||b_4^*|| = &\ \ |slope|^2 \cdot ||b_2^*|| \\ &\ \ \vdots \\ ||b_{2n}^*|| = &\ \ |slope|^{2n-2} \cdot ||b_2^*|| \\ ||b_{2n+1}^*|| = &\ \ |slope|^{2n-1} \cdot ||b_2^*|| \ . \end{aligned} \tag{57}$$

So we have

$$\sqrt{7n} \cdot |slope|^{2n^2-n+1} \cdot (\delta^{2n} \cdot \sqrt{q})^{2n} = 2q^n \ , \tag{58}$$

and

$$log(|slope|) = (-log(7n))/2 - 4n^2 \cdot log(\delta) + log(2))/(2n^2 - n + 1) \ . \tag{59}$$

The formula specifies the relation of $log(|slope|)$ and δ with fixed n and q, where q is the next prime larger than $81n^2 + 18n + 3$ (See (51)).

10.2 Parameter estimation

In this section, we assume that the computational complexity for the key recovery attack is as same as LWE problem(n, m, q, σ) since the last subsection observes the same property of the LWE problem. The parameters n, m, q, σ are described as follows. Here $m = 2n$, and q is the smallest prime larger than $81n^2 + 18n + 3$ in our system. The standard deviation σ of the elements of the error vector associated with LWE-problem, which is known as unique-SVP, can be calculated as $\sqrt{m}\sigma$, when we use embedding technique with the embedding factor equals 2. In our system, the average norm for elements of the shortest vector v is $\sqrt{7n}$ by (53). So we have $\sqrt{2n}\sigma = \sqrt{7n}$, then $\sigma = \sqrt{7/2}$ follows.

To estimate secure parameter n we apply "2016 Estimate" in [4], which is applied to "New Hope" [5], to our system.

First, Y.Chen suggests

$$\delta_0 = (((\pi\beta)^{1/\beta}\beta)/(2\pi e))^{1/(2(\beta-1))} \tag{60}$$

Table 4: The parameters of LWE-problem and relation to our system

parameter	description	our system
n	dimension of the associated lattice	n
m	number of samples	$2n$
q	modulo	$> 81n^2 + 18n + 3$
σ	standard deviation	1.87

in [11], where δ_0 is the root of Hermite factor of the shortest vector in the reduced basis of $\mathcal{L}'_{KRA}$ obtain by BKZ with block size β.

In "2016 Estimate", Albrecht et.al. suggest the following inequality

$$\sqrt{\beta/(2n)}\lambda_1(\mathcal{L}'_{KRA}) \geq \delta_0^{2\beta-2n} det\mathcal{L}^{+}_{KRA} \ , \tag{61}$$

holds for the basis reduce by the BKZ algorithm with block size β, where $\lambda_1(\mathcal{L}'_{KRA})$ is the norm of the shortest vector in the lattice $\mathcal{L}'_{KRA}$, namely $\mathcal{L}'_{KRA} = \sqrt{7n}$. So we find the pair of n and β to satisfy the both condition of (60) and (61) and estimate the complexity of lattice reduction by the formula

$$8 \cdot 2n \cdot 2^{0.292\beta+12.31} \tag{62}$$

which is given by [7]. So, we design appropriate parameter values for our encryption scheme as shown in Table 5.

Table 5: Appropriate parameter values for our scheme

NIST Category	k	n	q	Secret key (bytes)	Public key (bytes)	Ciphertext (bytes)
I	143	1201	467424411	600.5	14412	28824
Ⅲ	207	1733	973190427	866.5	20796	41592
V	272	2267	1665292875	1133.5	27204	54408

Here, we denote the security parameter by k where $k = 143, 207, 272$ for AES128, AES192, AES256, respectively.

11 Cryptographic scheme

This section shows a cryptographic scheme that satisfies IND-CCA2 security. This scheme is constructed by applying Fujisaki–Okamoto conversion [14] to our cryptographic primitive (which satisfies IND-CPA security as described in Section 6).

11.1 Fujisaki–Okamoto conversion

Let $\Pi := (\mathfrak{K}, \mathfrak{E}, \mathfrak{D})$ be a public-key encryption scheme that satisfies IND-CPA security, where $\mathfrak{K}$ is the key-generation algorithm, $\mathfrak{E}$ is the encryption algorithm, and $\mathfrak{D}$ is the decryption algorithm. Fujisaki–Okamoto conversion tells us that the public-key encryption scheme $\bar{\Pi} := (\bar{\mathfrak{K}}, \bar{\mathfrak{E}}^H, \bar{\mathfrak{D}}^H)$ satisfies IND-CCA2 security, such that

$$\mathfrak{E}^{\bar{H}}_{pk} = \mathfrak{E}_{pk}((x||s), H(x||s)), \tag{63}$$

where

- s is a random string chosen from an appropriate domain,
- H is a hash function
$$H : \{0,1\}^* \to \{0,1\}^{\kappa_0}$$
- $\mathfrak{E}_{pk}(message, coins)$ indicates the encryption of the indicated message using the indicated coins as random bits.

More precisely, the basic scheme $\bar{\Pi} := (\bar{\mathfrak{K}}, \bar{\mathfrak{E}}^H, \bar{\mathfrak{D}}^H)$ can be described as follows.

- Let $\bar{\mathfrak{K}}(1^\kappa) := \mathfrak{K}(1^\kappa)$.
- $\bar{\mathfrak{E}}^H_{pk} : \{0,1\}^{\kappa-\kappa_0} \times \{0,1\}^{\kappa_0} \to C$ is defined by
$$\bar{\mathfrak{E}}^H_{pk}(x, s) := \mathfrak{E}_{pk}((x||s), H(x||s)),$$
where $x \in \{0,1\}^{\kappa-\kappa_0}$, $s \in \{0,1\}^{\kappa_0}$, and C is a cipher space whose elements are valid ciphertexts.
- $\bar{\mathfrak{D}}^H_{sk} : C \to \{0,1\}^{\kappa-\kappa_0} \cup \{\perp\}$ is defined by
$$\bar{\mathfrak{D}}^H_{sk}(y) := \begin{cases} [\mathfrak{D}_{sk}(y)]^{\kappa-\kappa_0} & \text{if the condition (64) holds} \\ \perp(null) & \text{otherwise} \end{cases},$$
where $[\mathfrak{D}_{sk}(y)]^{\kappa-\kappa_0}$ indicates the first $(\kappa - \kappa_0)$ bits of $\mathfrak{D}_{sk}(y)$. The condition for ciphertext verification is
$$y = \mathfrak{E}_{pk}(\mathfrak{D}_{sk}(y), H(\mathfrak{D}_{sk}(y))). \tag{64}$$

Specifically, the following theorem holds.

Theorem 10. Suppose Π is γ-uniform and $(t', 0, 0, \epsilon')$-secure in the sense of IND-CPA. Then, for any q_H and q_D, the scheme $\bar{\Pi}$ is (t, q_H, q_D, ϵ)-secure in the sense of IND-CCA2 in the random oracle model where

$$\begin{aligned} t = &\ t' - q_H \cdot (T_\epsilon(\kappa) + c \cdot \kappa) \\ \epsilon = &\ \epsilon' \cdot (1-\gamma)^{-q_D} + q_H \cdot 2^{-(\kappa_0 - 1)}. \end{aligned} \tag{65}$$

In this, $T_\epsilon(\cdot)$ denotes the computational running time of $\mathfrak{E}_{pk}(\cdot)$ and c is a constant.

Here, γ-uniformity is defined as follows.

Definition 11. Let $\Pi = (\mathfrak{K}, \mathfrak{E}, \mathfrak{D})$ be a public-key encryption scheme. Let the parameters $mlen$ and $clen$ denote the length of a plaintext message and a coins tuple s, respectively. For a given $x \in \{0,1\}^{mlen}$ and $y \in C$, define

$$\gamma(x,y) = Pr[s \leftarrow_R \{0,1\}^{clen} : y = \mathfrak{E}_{pk}(x,s)].$$

We say that Π is γ-uniform (for any $k \in \mathbb{N}$) if, for any $x \in \{0,1\}^{mlen}$ and any $y \in C$, $\gamma(x,y) \leq \gamma$.

Now, we estimate the sizes of the parameters q_D,qH, and γ of our encryption scheme. First, we set the parameter q_D to 2^{64} since the call for proposal indicates q_D is no more than 2^{64} (Section 4.A.2 "Security Definition for Encryption/Key-Establishment"). We assume the parameter q_H is 2^{2k} to consider an exhaustive search for a quantum computer with Grover's algorithm. To calculate the parameter γ, we need to estimate the probability $\gamma(x,y)$ for any $x \in \{0,1\}^{mlen}$ and $y \in C$. Let x_0 be the fixed plaintext in $\{0,1\}^{mlen}$ and suppose the probability

$$Pr\{\forall s_1, s_2 \in \{0,1\}^{clen} | \mathfrak{E}_{pk}(x_0, s_1) = \mathfrak{E}_{pk}(x_0, s_2)\}, s_1 \neq s_2.\} \tag{66}$$

If we write

$$\mathfrak{E}_{pk}(x_0, s_i) = m(t) + X(x,y)r_i(x,y) + \ell \cdot e_i(x,y) \quad (i = 1, 2),$$

the condition (66) can be described as

$$\ell^{-1} \cdot X(x,y)(r_1(x,y) - r_2(x,y)) = e_2(x,y) - e_1(x,y).$$

If $e_1(x,y)$ equals $e_2(x,y)$, then $r_1(x,y) = r_2(x,y)$ since neither $X(x,y)$ nor ℓ equals 0. So, we can assume $e_1(x,y) \neq e_2(x,y)$. By the definition of the encryption scheme, the coefficients of $e_2(x,y) - e_1(x,y)$ are in R_q with coefficients restricted to the range $\{0, \cdots, \ell-1, q-\ell+1, \cdots, q-1\}$. Further, the coefficients of $\ell^{-1} \cdot X(x,y)(r_1(x,y) - r_2(x,y))$ are in R_q with coefficients in the range $\{0, \cdots, q-1\}$ and the coin r is in $\{0,1\}^{\kappa_0}$. Then, we can estimate the probability (66) as

$$\max\{(2p/q)^{\#\Gamma_e \cdot n}, 2^{-\kappa_0}\}.$$

Since the probability does not depend on the fixed x_0, we can estimate

$$\gamma \leq \max\{(2\ell/q)^{\#\Gamma_e \cdot n}, 2^{-\kappa_0}\}. \tag{67}$$

By the condition (11), we can estimate q as follows.

$$\begin{array}{rcl} q & \geq & \ell \sum_{i=0}^{dX+dr} (i+1) n^i (\ell-1)^{i+1} \\ & > & \ell \sum_{i=dX+dr}^{dX+dr} (i+1) n^i (\ell-1)^{i+1} \\ & = & \ell(dX+dr+1) n^{dX+dr} (\ell-1)^{dX+dr+1} \\ & > & 2\ell \cdot n^2 (\ell-1)^3 \\ & \geq & 2\ell \cdot 2^2 (\ell-1)^3 \end{array}$$

The last inequality is satisfied since dX and dr are each larger than or equal to 1. Then

$$(2\ell/q)^{\#\Gamma_e \cdot n} < (1/2^2(\ell-1)^3)^{\#\Gamma_e \cdot n} < 1/2^{2n}.$$

If we set ℓ equals to 4, then $1/2^{2n} = 1/2^{\kappa} < 1/2^{\kappa_0}$ is satisfied since $2n = |\ell|n = \kappa$. We can conclude $\gamma < 1/2^{\kappa_0}$ in the case of $\ell = 4$.

Then, ϵ can be calculated as follows.

$$\begin{aligned}\epsilon &= \epsilon' \cdot (1 - 2^{-\kappa_0}))^{-q_D} + q_H \cdot 2^{-(\kappa_0-1)} \\ &\sim \epsilon' \cdot (1 + (q_D + q_H) \cdot 2^{-\kappa_0})\end{aligned}$$

Since k is larger than or equal to 128,

$$\begin{aligned}\epsilon &< \epsilon' \cdot (1 + 2q_H \cdot 2^{-\kappa_0}) \\ &= \epsilon' \cdot (1 + 2 \cdot 2^{2k} \cdot 2^{-\kappa_0}) \ . \\ &= \epsilon' \cdot (1 + 2^{2k+1-\kappa_0})\end{aligned}$$

According to the relation, we set $\kappa_0 \geq 2k + 1$ since ϵ is negligible (such as $\epsilon < 2\epsilon'$).

11.2 Key generation

11.2.1 keygen

Input:	dp	Set of domain parameters
Output:	SK	Octet string for secret key $(u_x(t), u_y(t))$
	PK	Octet string for public key $X(x, y)$
Assumption:	n and ℓ are integers larger than 1.	
	dX is a positive integer.	
Step:		

1. Set $(u_x(t), u_y(t), X(x, y)) = IECKG(dp)$
2. Set $SK = R\ell 2OS(u_x(t), dp)||R\ell 2OS(u_y(t), dp)$
3. Set $PK = Rq2OS(X, dp)$

11.3 Encryption

11.3.1 encrypt

Input:	m	Message in octet string
	PK	Public key in octet string
	dp	Set of the domain parameters
Output:	cip	Ciphertext in octet string
	$ciplen$	Length of the ciphertext in bytes
Assumption:	$\|m\|$ is equal to $mlen$	
	dp satisfies the condition (64).	

Step:

1. Set the length of the payload $plen = \lceil dp.n \cdot |dp.\ell|/8 \rceil$
2. Create a plaintext M whose payload is $plen$ bytes in size
$$M = m||randombytes(plen - dp.mlen)$$
3. Set the lower $8 - |dp.l| \cdot (n \bmod (8/|dp.\ell|))$ bits of M to 0.
4. Initialize the seed expander with coins equal to $H(M)$.
5. Create a plaintext polynomial $m(t)$ from the octet string M.
$$m(t) = OS2R\ell(M, dp)$$
6. Create a public key X from the octet string PK
$$X(x, y) = OS2Pq(PK, dp.dX)$$
7. c(x,y)=IECENC(m(t),X(x,y),dp)
8. Create a ciphertext c
$$cip = Pq2OS(c(x, y), dp.dX + dp.dr, dp)$$
9. Calculate the length of the ciphertext
$$ciplen = \#\Gamma_e \cdot dp.n \cdot |dp.q|$$

11.4 Decryption

11.4.1 decrypt

Input:	cip	Ciphertext in an octet string
	$ciplen$	
Byte length of the input ciphertext	SK	Secret key in an octet string
	PK	Public key in an octet string
	dp	Set of domain parameters
Output:	m	Message
	$flag$	Validity of the ciphertext cip
Assumption:	The secret key and public key are valid	
Step:		

1. Extract the secret key $(u_x(t), u_y(t)$ from SK
   (a) Decompose SK into $SK1$ and $SK2$ $|dp.\ell| \cdot n$ bits each
   (b) Extract $u_x(t)$ from the ciphertext $SK1$, setting
   $$u_x(t) = OS2R\ell(SK1, dp)$$
   (c) Extract $u_y(t)$ from the ciphertext $SK2$, setting
   $$u_y(t) = OS2R\ell(SK2, dp)$$
2. Decrypt the cipher text cip, setting
$$m(t) = IECDEC(c(x, y), u_x(t), u_y(t), dp)$$
3. Recover the plaintext M, setting
$$M = R\ell 2OS(m(t), dp)$$
4. Initialize the seed expander with coins equal to $H(M)$.
5. Extract the public key X from the octet string PK, setting
$$X(x, y) = OS2Pq(dp.dX, PK, dp)$$
6. Encrypt the plaintext polynomial $m(t)$, setting
$$c2(x, y) = IECENC(m(t), X(x, y), dp)$$
7. If $c2$ equals c then $m = [M]^{dp.mlen}$ and $flag = valid$; otherwise, $m = null$ and $flag = invalid$

Here, $[x]^{len}$ denotes extraction of the most significant len bits of x.

12 Performance analysis

This section shows the results of the preliminary performance analysis, which is carried out by using a reference implementation and an optimized implementation (including this proposal). Table 6 and Table 7 show the cycles of each function described in Sections 11.2 to 11.4. We carried out this analysis on a platform with the following characteristics:

CPU	Xeon E5-1620 3.6GHz
OS	Windows 7, 64bit
memory	32 GB memory

Table 6: Performance of reference implementations

Name	Security (bits)	keygen (cycles)	encrypt (cycles)	decrypt (cycles)
IEC602	143	92909566	178456036	335353573
IEC868	207	160497017	378860493	716243384
IEC1134	272	239510004	626677271	1186128486

Table 7: Performance of optimized implementations

Name	Security (bits)	keygen (cycles)	enceypt (cycles)	decrypt (cycles)
IEC602	143	78272627	116773401	216049724
IEC868	207	131971731	248815749	466577361
IEC1134	272	191246205	420543208	792576864

Figure 7 shows the differences between these implementations.

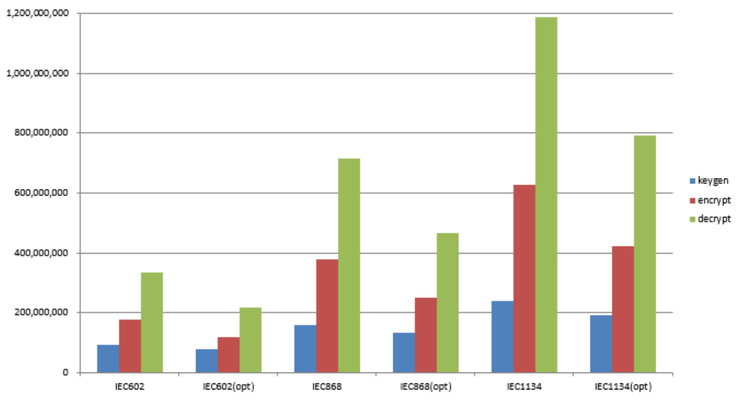

Fig. 7: Performance of the reference and optimized implementations

13 Advantages and limitations

One of the advantage of the proposed cryptographic primitives described in Section 6 is that the system has homomorphic properties. Homomorphic properties allow us to compute on encrypted data without decoding. They can calculate addition and/or multiplication of single- or multiple-bit integers in the encrypted state. These properties are attractive to industries such as the smart device community and cloud computing. These industries need to handle personal data that must be kept secret from others. In the smart device community, we need to send (for example) meter data to the electric power company. Although the smart mater encrypts its output, the amount of data is too large to send at short intervals. The data must be conslidated at intermediate nodes by calculating in an encrypted state.

Table 8 describes the classification of homomorphic encryption (HE) with respect to the computable number of times and sizes of public keys and encrypted data. From the table, we see that sizes of the public keys and the ciphertext increase with increasing the computable number of times. Somewhat homomorphic encryption (SHE) and fully homomorphic encryption (FHE) are realized by lattice-based encryption such as LWE, NTRU, or Nuida–Kurosawa's scheme. It is clear that lattice-based encryption is as important as homomorphic encryption.

Table 8: Classification of homomorphic encryption

Type	Number of operations		Message	Public key	Ciphertext	Example
	Addition	Multiplication	(bit)	size	size	
HE	Any	No	Multi	Small	Small	Paillier [31]
	No	Any	Multi	Small	Small	ElGamal [13]
SHE	Any	Once	Multi	Small	Small	Pairing [8]
	Any	Several	Single	Large	Medium	LWE [25]
	Any	Several	Multi	Large	Medium	Proposed, NTRU [35]
FHE	Any	Any	Single	Large	Large	Lattice-based [18]
	Any	Any	Multi	Large	Large	Nuida–Kurosawa [28]

13.1 Homomorphic property

The Giophantus cryptographic primitives have the ring homomorphic property. When two different cipher polynomials

$$\begin{aligned} c_1(x,y) &= m_1(t) + X(x,y)r_1(x,y) + \ell \cdot e_1(x,y) \\ c_2(x,y) &= m_2(t) + X(x,y)r_2(x,y) + \ell \cdot e_2(x,y) \end{aligned} \tag{68}$$

are given, we define the addition and multiplication as $c_1(x,y)+c_2(x,y)$ and $c_1(x,y)c_2(x,y)$, respectively.

13.1.1 Additive homomorphism

In the case of addition, we can decrypt as

$$(c_1 + c_2)(u_x(t), u_y(t)) = m_1(t) + m_2(t) + p \cdot (e_1 + e_2)(u_x(t), u_y(t)) \tag{69}$$

and extract the plaintext

$$(c_1 + c_2)(u_x(t), u_y(t))\ (mod\ \ell) = m_1(t) + m_2(t) \tag{70}$$

under the following conditions.

$$MC(m_1(t) + m_2(t)) < \ell \tag{71}$$

$$\ell \cdot MC((e_1 + e_2)(u_x(t), u_y(t))) < q \tag{72}$$

The condition (71) is to prevent the coefficients of the plaintext $m_1(t)+m_2(t)$ from overflowing beyond the range 0 to $q-1$. The condition (72) is to prevent the coefficients of the noise term $\ell \cdot (e_1+e_2)(u_x(t), u_y(t))$ from overflowing beyond the range of 0 $q-1$.

Let N_a be the number of times to add ciphertext. Then we obtain the condition of ℓ and q as follows:

$$N_a \cdot \lambda < p, \tag{73}$$

$$N_a \cdot \#\Gamma_e \cdot p(p-1) \cdot (n(\lambda - 1))^{d_X + d_r} < q, \tag{74}$$

where λ is a parameter giving the maximum size of the coefficients in m, $u_x(t)$ and $u_y(t)$ in the case of using the homomorphic operations. So, we can perform additional homomorphic operation between $n \log_2 \ell$ bit integers.

13.1.2 Multiplicative homomorphism

We can multiply

$$(c_1c_2)(u_x(t), u_y(t)) =$$
$$m_1(t)m_2(t) + p \cdot (m_1(t)e_2(u_x(t), u_y(t)) + m_2(t)e_1(u_x(t), u_y(t)) + \ell \cdot e_1(u_x(t), u_y(t))e_2(u_x(t), u_y(t)))$$

and extract the plaintext

$$(c_1c_2)(u_x(t), u_y(t))\ (mod\ \ell) = m_1(t)m_2(t) \tag{75}$$

under the following conditions.

$$MC(m_1(t)m_2(t)) < \ell \tag{76}$$

$$\deg m_1(t)m_2(t) < n \tag{77}$$

$$\ell^2 \cdot MC((e_1e_2)(u_x(t), u_y(t))) < q \tag{78}$$

The condition (76) is to prevent the coefficients of the plaintext $m_1(t)m_2(t)$ from overflowing beyond the range 0 to ℓ. Also, $m_1(t)m_2(t)$ requires keeping the degree under n since $m_1(t)m_2(t)$ must be included in R_ℓ, as required by the condition (77).

The condition (78) is to prevent the coefficients of the noise term from overflowing beyond the range 0 to $q-1$ after the multiplication.

Let N_m be the number of times to multiply ciphertext. Then, we obtain the conditions for ℓ,q, and n as follows.

$$\lambda^{N_m+1} < \ell \tag{79}$$

$$N_m \deg m(t) < n \tag{80}$$

$$(n\ell)^{N_m} \cdot \ell \cdot (\#\Gamma_{Xr} \cdot (\ell-1) \cdot (n(\lambda-1))^{d_X+d_r})^{N_m+1} < q \tag{81}$$

We can therefore perform N_m multiplicative homomorphic operations between
$n \log_2 \lambda / N_m$ bit integers.

References

[1] K. Akiyama, Y. Goto, *A Public-key Cryptosystem using Algebraic Surfaces*, In: Proc. of PQCrypto'06, pp. 119–138, (2006), available at `http://postquantum.cr.yp.to/`.

[2] K. Akiyama, Y. Goto, H. Miyake, *An Algebraic Surfaces Cryptosytem*, In: Proc. of PKC'09, **5443**, Lecture Notes in Computer Science, pp. 425–442, Springer Berlin Heidelberg, (2009).

[3] K. Akiyama, Y. Goto, S. Okumura, T. Takagi, K. Nuida, G. Hanaoka, *A Public-key Encryption Scheme Based on Non-linear Indeterminate Equations*, Pre-procceeding of SAC2017, available at http://sacworkshop.org/SAC17/prepro.pdf.

[4] M.Albrecht, F.Göpfert, F.Virdia, T.Wunderer, *Revisiting the Expected Cost of Solving uSVP and Application to LWE*, In: Proc. of ASIACRYPT'17, **10624**, Lecture Notes in Computer Science, pp. 297-322, Springer Berlin Heidelberg, (2017).

[5] E. Alkim, L. Ducas, T. Pöppelmann, P.Schwabe, *Post-quantum Key Exchange - A New Hope*, 25th USENIX Security Symposium (USENIX Security 16), pp.327–343,(2016)

[6] S. Bai, S. D. Galbraith, *Lattice Decoding Attacks on Binary LWE*, ACISP'14, **8544**, Lecture Notes in Computer Science, pp. 322–337, Springer International Publishing, (2014).

[7] A. Becker, L. Ducas, N. Gama, T. Laarhoven,"New directions in nearest neighbor searching with applications to lattice sieving", https://eprint.iacr.org/2015/1128.pdf

[8] D. Boneh, E.-J. Goh, K. Nissim, *Evaluating 2-DNF Formulas on Ciphertexts*, In: Proc. of TCC'05, **3378**, Lecture Notes in Computer Science, pp.325–341, Springer Berlin Heidelberg, (2005).

[9] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, D. Stehlé, *Classical hardness of learning with errors*, In: Proc. of STOC'13, ACM, pp. 575–584, ACM New York, NY, USA, (2013).

[10] Y. Chen, P. Nguyen, *BKZ 2.0: Better lattice security estimates*, In: Proc. of ASIACRYPT'11, **7073**, Lecture Notes in Computer Science, pp. 1–20, Springer Berlin Heidelberg, (2011).

[11] Y. Chen, *Reduction de reseau et securite concrete du chirement completement homomorphe* PhD thesis, Paris 7, (2013).

[12] J. Denef, *The Diophantine Problem for Polynomial Rings of Positive Characteristic*, In: Proc. of Logic Colloquium '78, Stud. Logic Foundations Math., **97**, pp. 131–145, North Holland, Amsterdam-New York, (1979).

[13] T. ElGamal, *A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms*, In: Proc. of CRYPTO'84, **196**, Lecture Notes in Computer Science, pp. 10–18, Springer Berlin Heidelberg, (1984).

[14] E. Fujisaki, T. Okamoto, "How to Enhance the Security of Public-Key Encryption at Minimum Cost,"In: Proc. of PKC'99, **1560**, Lecture Notes in Computer Science, pp.53–68, Springer Berlin Heidelberg, (1999). Springer-Verlag(1999)

[15] J. Faugère, P. Spaenlehauer, *Algebraic Cryptanalysis of the PKC'09 Algebraic Surface Cryptosystem*, In: Proc. of PKC'10, **6056**, Lecture Notes in Computer Science, pp. 35–52, Springer Berlin Heidelberg, (2010).

[16] N. Gama, P. Q. Nguyen, *Predicting lattice reduction*, In: Proc. of EUROCRYPT'08, **4963**, Lecture Notes in Computer Science, pp. 31–51, Springer Berlin Heidelberg, (2008).

[17] C. Gentry, *Key Recovery and Message Attacks on NTRU-Composite*, In: Proc. of EUROCRYPT'01, **2045**, Lecture Notes in Computer Science, pp. 182–194 Springer Berlin Heidelberg, (2001).

[18] C. Gentry, *Fully Homomorphic Encryption Using Ideal Lattices*, In: Proc. of STOC'09, ACM, pp.169–178, ACM New York, NY, USA, (2009).

[19] O. Goldreich, S. Goldwasser, S. Halevi, *Public-Key Cryptosystems from Lattice Reduction Problems*, In: Proc. of CRYPTO'97, **1294**, Lecture Notes in Computer Science, pp.112–131, Springer Berlin Heidelberg, (1997).

[20] J. Hoffstein, J. Pipher, J. H. Silverman, *NTRU: A Ring-Based Public Key Cryptosystem*, In: Proc. of ANTS'98, **1423**, Lecture Notes in Computer Science, Springer-Verlag, pp.267–288, Springer Berlin Heidelberg, (1998).

[21] R. Kannan, *Minkowski's convex body theorem and integer programming*, Math. of Oper. Res., **12**(3), pp. 415–440, INFORMS, Linthicum, Maryland, USA, (1987).

[22] P. Kirchner, P.-A. Fouque, *Comparison between Subfield and Straightforward Attacks on NTRU*, IACR Cryptology ePrint Archive: Report 2016/717, available at `http://eprint.iacr.org/2016/717`.

[23] K. Kobara, H. Imai, *Semantically secure McEliece public-key cryptosystems - conversion for McEliece PKC*, In: Proc. of PKC'01, **1992**, Lecture Notes in Computer Science, pp.19–35, Springer Berlin Heidelberg, (2001).

[24] A. K. Lenstra, H. W. Lenstra, L. Jr.Lovasz, *Factoring polynomials with rational coefficients*, Math. Ann. **261**(4), pp. 515–534, Springer-Verlag, (1982).

[25] R. Lindner, C. Peikert, *Better key sizes (and attacks) for LWE-based encryption*, In: Proc. of CT-RSA'11, **6558**, Lecture Notes in Computer Science, pp. 319–339, Springer Berlin Heidelberg, (2011).

[26] M. Liu, P. Q. Nguyen *Solving BDD by Enumeration: An Update*, In: Proc. of CT-RSA'13, **7779**, Lecture Notes in Computer Science, pp. 293–309, Springer Berlin Heidelberg, (2013).

[27] D. Micciancio, C. Peikert, *Hardness of SIS and LWE with Small Parameters*, In: Proc. of CRYPTO'13, **8042**, Lecture Notes in Computer Science, pp. 21–39, Springer Berlin Heidelberg, (2013).

[28] K. Nuida, K. Kurosawa, *(Batch) Fully Homomorphic Encryption over Integers for Non-Binary Message Spaces*, In: Proc. of EUROCRYPT'15, **9056**, Lecture Notes in Computer Science, pp.537–555, Springer Berlin Heidelberg, (2015).

[29] `http://www.etsi.org/news-events/events/949-etsi-iqc-3`, (2015).

[30] `https://www.nsa.gov/ia/programs/suiteb\_cryptography/`, (2015).

[31] P. Paillier, *Public-Key Cryptosystems Based on Composite Degree Residuosity Classes*, In: Proc. of EUROCRYPT'99, **1592**, Lecture Notes in Computer Science, pp. 223–238, Springer Berlin Heidelberg, (1999).

[32] O. Regev, *On Lattices, Learning with Errors, Random Linear Codes, and Cryptography*, J. of the ACM, **56**(6), pp. 1–40, ACM New York, NY, USA, (2009).

[33] C. R. Schnorr, M. Euchner, *Lattice basis reduction: improved algorithms and solving subset sum problems*, in Math. and Programming, **66**(1), pp. 181-189, Springer-Verlag, (1994).

[34] P. W. Shor, *Algorithms for Quantum Computation: Discrete Log and Factoring*, In: Proc. of SFCS'94, pp. 124–134, IEEE Computer Society Washington, (1994).

[35] D. Stehle, R. Steinfeld, *Making NTRU as secure as worst-case problems over ideal lattices*, In: Proc. of EUROCRYPT'11, **6632**, Lecture Notes in Computer Science, pp. 27–47, Springer Berlin Heidelberg, (2011).

[36] C. Tao, A. Diene, S. Tang, J. Ding, *Simple matrix scheme for encryption*, In: Proc. of PQCrypto'13, **7932**, Lecture Notes in Computer Science, pp.231–242, Springer Berlin Heidelberg, (2013).