A Lattice Based Public Key Cryptosystem Using Polynomial Representations

Seong-Hun Paeng[1] *, Bae Eun Jung[2], and Kil-Chan Ha[3] **

[1] Department of Mathematics, Konkuk University, Seoul, 143-701 Korea
shpaeng@konkuk.ac.kr
[2] ETRI, 161 Kajong-dong, Yusong-gu, Taejon, 305-350, Korea
bejung@etri.re.kr
[3] Department of Applied Mathematics, Sejong University, Seoul, 143-747 Korea
kcha@sejong.ac.kr

Abstract. In Crypto 97, a public key cryptosystem based on the closest vector problem was suggested by Goldreich, Goldwasser and Halevi [4]. In this paper, we propose a public key cryptosystem applying representations of polynomials to the GGH encryption scheme. Its key size is much smaller than the GGH system so that it is a quite practical and efficient lattice based cryptosystem.

Keywords: GGH cryptosystem, lattice based public key cryptosystem, polynomial representation

1 Introduction

In Crypto 97, Goldreich, Goldwasser and Halevi proposed a cryptosystem (GGH) using the closest vector problem (CVP) [4]. It is one of the most notable cryptosystem based on the complexity of lattices. The authors of the GGH published 5 numerical challenges for the security parameter $n = 200, 250, 300, 350, 400$, of which the public key sizes range from 330KBytes to 2MBytes. Nguyen solved all the GGH challenge except $n = 400$ [9]. For $n = 400$, the GGH is not practical since the key size is too large. It uses $n \times n$-matrices as a public key and a private key. Thus its key sizes are very large, so it is considered not to be practical. Almost every lattice based public key cryptosystem except for NTRU has an impractical key size. Micciancio suggested to express the public matrix as Hermitian normal form (HNF), whose key sizes are much smaller than those of the GGH system [8].

However, the GGH system has some advantages. For example, it seems to be asymptotically more efficient than RSA and ElGamal encryption schemes using modular exponentiations. Furthermore, it has a natural signature scheme. Currently, NTRU cryptosystem is the most efficient cryptosystem among lattice based PKC's. But in view of security, the GGH encryption scheme has an advantage. Attackers can find out only the message by known lattice attacks, i.e.

* Supported by the Faculty Research Fund of Konkuk University in 2002 and NSRI.
** Supported by NSRI.

Y.G. Desmedt (Ed.): PKC 2003, LNCS 2567, pp. 292–308, 2003.
© Springer-Verlag Berlin Heidelberg 2003

the secret key of the GGH cannot be obtained by solving the shortest vector problem (or CVP) [9]. But in NTRU, the secret key of NTRU can be obtained by finding the shortest vector of NTRU-lattice.

In this paper, we propose a public key cryptosystem applying polynomial representations to the GGH scheme whose key size is practical. In section 2, we shortly review the GGH system and explain the security related to the choice of a secret parameter T. In section 3, we study various representations of polynomials by $n \times n$-matrices and their direct applications to the GGH system. In section 4, we suggest a public key cryptosystem using the representations in section 3. Also we study its parameter selection, security analysis and key sizes. Its key sizes are much smaller than HNF expression and comparable with NTRU. In Appendix A, we introduce a scheme whose key size is smaller than that of the scheme proposed in section 4.

2 Description of the GGH System

2.1 The GGH System

In this section, we describe the GGH cryptosystem briefly. First, recall the definitions related to lattice:

Definition 1. *Let B be a real non-singular $n \times n$-matrix. The orthogonality defect of B is defined as*

$$\textit{orth-defect}(B) := \frac{\prod_i ||b_i||}{|det(B)|},$$

where $||b_i||$ is the Euclidean norm of the i-th column in B.

Then orth-defect$(B) = 1$ if and only if B is an orthogonal matrix.

Definition 2. *Let B be a real non-singular $n \times n$-matrix. The dual orthogonality defect of B is defined as*

$$\textit{orth-defect}^*(B) := \frac{\prod_i ||b_i^*||}{|det(B^{-1})|} = |det(B)| \prod_i ||b_i^*||,$$

where b_i^* is the i-th row in B^{-1}.

The GGH uses the closest vector problem (CVP). It is well known that CVP is an NP-hard problem. The GGH system is as follows:

Private key The private key is an $n \times n$-matrix R with a low dual-orthogonality-defect. It can be generated by $R' + kI$, where $R' = (R'_{ij})$ satisfies that $|R'_{ij}| \leq l$ and $k \approx \sqrt{n}l$ for some constant l.

Public key The public key is an $n \times n$-matrix B such that B generates the same lattice as R with a high dual-orthogonality-defect. Then $B = RT^{-1}$ for some $T \in \mathrm{GL}(n, \mathbb{Z})$.

Encryption The message v is an element of $\mathbb{Z}^n$. The ciphertext is obtained as follows:

$$c = Bv + e$$

for an error vector $e = (\delta_1\sigma, \cdots, \delta_n\sigma)$, where $\delta_i = -1$ or 1 and σ is a small constant, e.g. 4.

Decryption The deciphertext is obtained as follows:

$$v' = T\lceil R^{-1}c \rfloor,$$

where $\lceil v \rfloor$ denotes the vector in $\mathbb{Z}^n$ which is obtained by rounding each entry in v to the nearest integer.

Since T^{-1} is an integer matrix, we have

$$\begin{aligned} T\lceil R^{-1}c \rfloor &= T\lceil R^{-1}(RT^{-1}v + e) \rfloor \\ &= T\lceil T^{-1}v + R^{-1}e \rfloor \\ &= v + T\lceil R^{-1}e \rfloor. \end{aligned} \tag{2.1}$$

If $\lceil R^{-1}e \rfloor = 0$, then decryption works. We denote the maximum of L_∞-norm of the rows in R^{-1} by $\gamma/\sqrt{n}$. If $\sigma = [(\gamma\sqrt{8\ln(2n/\epsilon)})^{-1}]$ for some small real number $\epsilon > 0$, then the probability of decryption error is bounded by ϵ, where $[a] = \max\{\ x \mid x \text{ is an integer},\ x \leq a\}$.

2.2 Why Is $|\det(T)| = 1$ Needed?

Let L_R and L_B be the lattices generated by columns of R and $B = RT^{-1}$, respectively. In the GGH, L_R and L_B are the same lattices so that T is unimodular. Even if L_B is a sublattice of L_R (i.e. T^{-1} is an integer matrix and $|\det(T^{-1})| \geq 1$), the decryption works. But in this case, its security can be weakened. In this section, we discuss the reason why we should use R and B such that $L_R = L_B$ in view of security.

Assume that L_B is a sublattice of L_R, i.e. $|\det(T)| < 1$. For the embedding attack ([4], [9]), L_B is embedded in $\bar{L}_B$ as (4.7). (see Section 4.) Note that $\det(\bar{L}_B) = \det(L_B) = \det(R)\det(T^{-1})$. Then CVP for L_B is changed to the shortest vector problem (SVP) for $\bar{L}_B$ [4],[9].

Recall the definition of the gap of the lattice.

Definition 3. *The gap of a lattice L, G_L is the ratio between the second successive minimum (the smallest real number r such that there are two linearly independent lattice points of length at most r) and the length of a shortest non zero vector in L.*

The larger the lattice gap is, the easier it becomes to find the shortest vector [9]. For $\bar{L}_B$, $(e, 1)$ will be the shortest vector with high probability and the second successive minima will be similar to the norm of the column vector of R if $|\det(T^{-1})| = 1$. Hence, in the case of the GGH, the gap of $\bar{L}_B$ could be estimated.

In the case that $|\det(T^{-1})| > 1$, since L_B is a sublattice of L_R, such an estimate is invalid. Instead, we can consider the security analysis used in NTRU. Gaussian heuristics says that the expected size of the smallest vector in a random lattice of dimension $n+1$ lies between

$$s_1 = \det(\bar{L}_B)^{1/n+1}\sqrt{\frac{n+1}{2\pi e}} = \det(T^{-1})^{1/n+1}\det(L_R)^{1/n+1}\sqrt{\frac{n+1}{2\pi e}}$$

and

$$s_2 = \det(\bar{L}_B)^{1/n+1}\sqrt{\frac{n+1}{\pi e}} = \det(T^{-1})^{1/n+1}\det(L_R)^{1/n+1}\sqrt{\frac{n+1}{\pi e}}.$$

Let $\lambda_1(\bar{L}_B)$ be the length of the shortest vector in $\bar{L}_B$. Since the second successive minima is expected to be larger than s_1, if we can find a vector b_1 such that $||b_1|| \leq s_1 = \frac{s_1}{\lambda_1(\bar{L}_B)}\lambda_1(\bar{L}_B)$, it will be the shortest vector. Hence the larger $\frac{s_1}{\lambda_1(\bar{L}_B)}$ is, the easier it is to find the shortest vector.

By LLL-algorithm, we can find a vector b_1 such that $||b_1|| \leq 2^{n/2}\lambda_1(\bar{L}_B)$. BKZ algorithm with block size β finds a vector of length at most $O(\beta^{n+1/\beta}\lambda_1(\bar{L}_B))$ [10]. Hence the larger β is, the higher the probability to find the shortest vector is. On the other hand, the run time of BKZ algorithm is exponential in the block size.

The authors of NTRU guessed the following conjecture based on experiments [5]:

Conjecture 1. For a given n-dimensional lattice L, let s_1 be $\det(L)^{1/n}\sqrt{\frac{n}{2\pi e}}$. The required time to find the shortest vector is $\exp(O(\frac{\lambda_1(L)}{s_1}n))$.

Based on this conjecture, the larger $|\det(T^{-1})|$ is, the easier it is to find the shortest vector in the embedded lattice $\bar{L}_B$. Hence it is an essential condition to use T^{-1} such that $|\det(T^{-1})|$ is small (especially 1). Assume that $n = 500$ and let t be the run time to find the shortest vector for the case that $|\det(T^{-1})| = 1$. If we choose T^{-1} such that $\det(T^{-1}) \approx 1.1^{500} = 5 \times 10^{20}$, then the run time to find the shortest vector will be $t^{0.91}$. But if we choose T^{-1} randomly in $M(n, \mathbb{Z})$, the probability to choose T^{-1} such that $|\det(T^{-1})| \leq 5 \times 10^{20}$ is almost 0.

3 Lattice Generated by Representations of Polynomial Rings

3.1 Representation of a Polynomial Ring

We introduce a representation of a polynomial ring as follows: We identify $c_{n-1}x^{n-1} + \cdots + c_0 \in \mathbb{Z}[x]/\langle r(x)\rangle$ with a vector $(c_0, \cdots c_{n-1}) \in \mathbb{Z}^n$, where $r(x)$ is a polynomial of degree n. Then we have the following representation of $\mathbb{Z}[x]/\langle r(x)\rangle$ into the set of $n \times n$ matrices with integer entries:

$$\begin{aligned} \Phi : \mathbb{Z}[x]/\langle r(x)\rangle &\to M(n, \mathbb{Z}) \\ h &\mapsto \Phi(h),\ \Phi(h)(f) = h(x)f(x). \end{aligned} \tag{3.2}$$

$$\begin{array}{ccc} (c_0, \cdots, c_{n-1}) \in \mathbb{Z}^n & \xrightarrow{\Phi(h)} & \Phi(h)(f) = (d_0, \cdots, d_{n-1}) \in \mathbb{Z}^n \\ \downarrow & & \uparrow \\ f(x) = \sum_{i=0}^{n-1} c_i x^i \in \mathbb{Z}[x]/\langle r(x)\rangle & \longrightarrow & h(x)f(x) = \sum_{i=0}^{n-1} d_i x^i \in \mathbb{Z}[x]/\langle r(x)\rangle \end{array}$$

Let $\{1, x, x^2, \cdots, x^{n-1}\}$ be a basis of $\mathbb{Z}^n = \mathbb{Z}[x]/\langle r(x)\rangle$. Depending on the choice of $r(x)$, we can find various representations.

Example 1. Let $h(x)$ be $h_{n-1}x^{n-1} + \cdots + h_0$.
(1) If $r(x) = x^n - 1$, then we have a circulant matrix

$$\Phi(h) = \begin{pmatrix} h_0 & h_{n-1} & \cdots & h_2 & h_1 \\ h_1 & h_0 & \cdots & h_3 & h_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ h_{n-2} & h_{n-3} & \cdots & h_0 & h_{n-1} \\ h_{n-1} & h_{n-2} & \cdots & h_1 & h_0 \end{pmatrix}. \tag{3.3}$$

(2) If $r(x) = x^n - x - 1$, then we have

$$\Phi(h) = \begin{pmatrix} h_0 & h_{n-1} & \cdots & h_2 & h_1 \\ h_1 & h_0 + h_{n-1} & \cdots & h_3 + h_2 & h_2 + h_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ h_{n-2} & h_{n-3} & \cdots & h_0 + h_{n-1} & h_{n-1} + h_{n-2} \\ h_{n-1} & h_{n-2} & \cdots & h_1 & h_0 + h_{n-1} \end{pmatrix}. \tag{3.4}$$

3.2 Direct Applications of Polynomial Representations

From representations of polynomials, we can obtain various lattices as we see in the above example. We can apply these representations to the GGH scheme directly as follows: Let $r(x)$ be $x^n - 1$. Then we obtain a circulant matrix as Example 1 (1). If $f(x) = a_{n-1}x^{n-1} + \cdots + a_0 \in \mathbb{Z}[x]/\langle r(x)\rangle$ satisfies that $|a_0| \approx \sqrt{n}l$ and other coefficients are contained in $[-l, l]$, then the dual-orthogonal-defect of $R = \Phi(f)$ would be low. In order to apply $R = \Phi(f)$ to the GGH system, it is necessary to find g such that $T^{-1} = \Phi(g)$ and $|\det(T^{-1})|$ is small (especially 1). But it is difficult to find a sufficiently large class of g such that $|\det(T^{-1})| = 1$ (i.e. $\Phi(g)$ is invertible in $M(n, \mathbb{Z})$).

4 Cryptosystem : Scheme I

In this section, we propose cryptosystems using a representation of polynomials.

4.1 Key Generation

We will take the private and public key in polynomial rings. Let n be a prime number and p be a positive integer. Experimentally, we can verify that sufficiently many elements of $\mathbb{Z}_p[x]/\langle x^n - 1\rangle$ have their inverses. Intuitively, if p is a prime number, then $|\mathbb{Z}_p^*| = \phi(p) = p - 1$, so almost every element of $\mathbb{Z}_p[x]/\langle x^n - 1\rangle$ has its inverse, where ϕ is Euler phi function. Even if p is not a prime number, $\mathbb{Z}_p$ has sufficiently many invertible elements, so sufficiently many elements of $\mathbb{Z}_p[x]/\langle x^n - 1\rangle$ have their inverses.

First, we generate 4 polynomials

$$f_1, f_2, h_1, h_2 \in \mathbb{Z}[x]/\langle x^n - 1\rangle$$

for the private key, which have the following properties:

- $f_1(x) = \alpha_{n-1}x^{n-1} + \cdots + \alpha_0$ and $f_2(x) = \beta_{n-1}x^{n-1} + \cdots + \beta_0$, where $|\alpha_{i_0}|, |\beta_{j_0}| \approx \sqrt{2nl}$ for some i_0, j_0 and the other coefficients are contained in $[-l, l]$ (l will be set to be 1).
- The coefficients of h_1 and h_2 are contained in $[-l, l]$.

We make the private matrix R as follows:

$$R = \begin{pmatrix} \Phi(f_1) & \Phi(h_1) \\ \Phi(h_2) & \Phi(f_2) \end{pmatrix}.$$

Since the diagonal entries of $\Phi(f_1), \Phi(f_2)$ are about $\sqrt{2nl}$ and other entries are contained in $[-l, l]$, the dual-orthogonality-defect of R would be low by the same reason as the GGH.

In order to generate the public key, we choose $g \in \mathbb{Z}[x]/\langle x^n - 1\rangle$ such that the coefficients of g are contained in $(-p/2, p/2]$. Then g can be considered as an element of a ring $F = \mathbb{Z}_p[x]/\langle x^n - 1\rangle$. We take g which is invertible in F. Then there exist g_p and Q in $\mathbb{Z}[x]/\langle x^n - 1\rangle$ such that $gg_p - 1 = pQ \in \mathbb{Z}[x]/\langle x^n - 1\rangle$. We generate 4-polynomials $P_1, P_2, P_3, P_4 \in \mathbb{Z}[x]/\langle x^n - 1\rangle$ as follows:

$$\begin{aligned} P_1 &= f_1 g + h_1 Q, \\ P_2 &= p f_1 + h_1 g_p, \\ P_3 &= h_2 g + f_2 Q, \\ P_4 &= p h_2 + f_2 g_p, \end{aligned} \tag{4.5}$$

which are expressed as

$$B = \begin{pmatrix} \Phi(P_1) & \Phi(P_2) \\ \Phi(P_3) & \Phi(P_4) \end{pmatrix}.$$

Then we have the following private key and public key:

- Private key : f_1, f_2, h_1, h_2 (i.e. R)
- Public key : P_1, P_2, P_3, P_4 (i.e. B)

4.2 Encryption and Decryption

Encryption A message is $M = (m_1, m_2) \in (\mathbb{Z}[x]/\langle x^n - 1\rangle)^2$. Then the ciphertext is

$$c = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = B \begin{pmatrix} m_1 \\ m_2 \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \end{pmatrix} = \begin{pmatrix} P_1 m_1 + P_2 m_2 + e_1 \\ P_3 m_1 + P_4 m_2 + e_2 \end{pmatrix} \in (\mathbb{Q}[x]/\langle x^n - 1\rangle)^2$$

for an error vector $e = (e_1, e_2)$, where $e_i \in \{-\sigma, \sigma\}^n$ (σ will be set to be $1/2$).

Decryption Let T be a matrix defined as follows:

$$T = \begin{pmatrix} \Phi(g) & pI \\ \Phi(Q) & \Phi(g_p) \end{pmatrix}^{-1}.$$

Then we decrypt as follows:

$$M = (m_1, m_2) = T\lceil R^{-1} c \rfloor.$$

Why decryption works? As we see in the above, $2n \times 2n$-matrix R has also a low-dual-orthogonality defect. Furthermore, we have the following lemma:

Lemma 1. $\det(T) = 1$.

Proof. Since

$$\begin{pmatrix} \Phi(g) & pI \\ \Phi(Q) & \Phi(g_p) \end{pmatrix} \begin{pmatrix} \Phi(g_p) & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} I & 0 \\ -\Phi(Q) & I \end{pmatrix} = \begin{pmatrix} I & pI \\ 0 & \Phi(g_p) \end{pmatrix},$$

we obtain that

$$\det(T^{-1})\det(\Phi(g_p)) = \det(\Phi(g_p)),$$

which implies that $\det(T^{-1}) = 1$. □

Also we can easily verify that

$$B = \begin{pmatrix} \Phi(f_1 g + h_1 Q) & \Phi(p f_1 + h_1 g_p) \\ \Phi(h_2 g + f_2 Q) & \Phi(p h_2 + f_2 g_p) \end{pmatrix} = R T^{-1}.$$

The decryption works by the same reason as the GGH scheme.

4.3 Security

Algebraic View In the GGH, we can have the equation $B = RT^{-1}$, where R and T are unknown. Then we have n^2 linear equations with $2n^2$ unknown variables. (In fact, we have an additional non linear equation $|\det(T)| = 1$.)

Assume that p is not a secret parameter. ¿From the equation (4.5), we have $4n$ equations with $5n$ unknown variables. For any subsets of equations of (4.5), the number of unknown variables $\geq$ the number of equations $+ n$. Hence if n is sufficiently large, we cannot obtain secret keys by solving equations algebraically.

Also note for each equation in (4.5), the lattice attack in NTRU is not applicable.

Gap of an Embedded Lattice and Selections of σ and l Nguyen attacked the GGH by the embedding attack [9]. To our knowledge, the embedding attack seems to be the most efficient attack to the GGH. So we select the parameter σ and l under the consideration of the embedding attack.

By the attack to the GGH system used in [9], the security of the system is not so closely related to the size of σ. Precisely, the linear equation $c = Bm + e$ can be reduced to

$$\bar{c} = \frac{c - Bm_{2\sigma}}{2\sigma} = Bm' + \frac{e}{2\sigma}, \tag{4.6}$$

where $m_{2\sigma}$ is the solution of

$$c + (\sigma, \cdots, \sigma) = Bm \pmod{2\sigma}.$$

So the error vector $\bar{e} = e/2\sigma$ is an element of $\{\pm 1/2\}^n$. Hence the choice of a large σ is not so essential condition for the security of the GGH scheme if $\sigma \geq 1/2$. Hence we take σ to be $1/2$.

The embedding technique builds the lattice $\bar{L}_B$ such that

$$\bar{L}_B = \begin{pmatrix} \mathbf{b_1} & \mathbf{b_2} & \cdots & \mathbf{b_n} & c \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix}, \tag{4.7}$$

where $\mathbf{b_i}$ are the column vector of B and c is the ciphertext. If v is the closest vector to c, then one can hope that $c - v$ is the shortest vector in $\bar{L}_B$. Recall that the gap of lattice (Definition 4). By experiments, the smaller the lattice gap is, the larger block size for BKZ algorithm we need in finding the shortest vector (Table 7). For a lattice whose gap size is about 10, Nguyen found the shortest vector by BKZ algorithm with block size 20 in 300-dimensional lattice reduced by (4.6) [9]. In $\bar{L}_B$, the second successive minimum is smaller than the minimal norm of column vectors of R, which is smaller than $2\sqrt{n}l$. If $\sigma = 1/2$, then we have

$$G_{\bar{L}_B} \leq \frac{||2\sqrt{n}l||}{||\bar{e}||} \leq 2.83l.$$

So the smaller l is, the harder it is to find the shortest vector in $\bar{L}_B$. Hence, we take l to be 1. Experimentally, if $l = 1$, then the probability of decryption error is sufficiently small if $n \geq 30$ and $G_{\bar{L}_B} \leq 2.4$ which is much smaller than the gap of the reduced lattice of the GGH system. Since the gap of our lattice is smaller than 2.4, BKZ algorithm with block size 20 cannot find the shortest vector in $158 = 79 \times 2$-dimensional lattice in many cases (Table 5,Table 7). Note that the run time of BKZ algorithm is exponential in the block size.

Assuming that $k \approx O(n/G_{\bar{L}})$-block size is needed for BKZ algorithm in finding the shortest vector (Table 7), we have the following natural conjecture:

Conjecture 2. The run time to solve the shortest vector in $\bar{L}$ by a lattice reduction algorithm is about $\exp(O(n/G_{\bar{L}}))$.

Also note that the reduction of our lattice is not easier than non reduced GGH lattice as we see experimental results for low dimensions (Table 6). (We used the implementation of the GGH in `http://theory.lcs.mit.edu/~cis/lattice/lattice.html`.)

Selection of p In order to estimate the public key size, the bit size of p should be determined. If $p \geq 2^{80}$, then it can be regarded as a private parameter. But if p takes 10 bits, then p cannot be considered as a private parameter. If p is larger than 2^{80} and it is kept secret, then we have the following advantages in the security. First, even if an attacker obtains g, he cannot obtain g_p and Q. Second, the reduction time for 80-bit p is longer than that for 10-bit p. The run time of lattice reduction algorithm for 10-bit p is shorter than 1/6 of that for 80-bit p (Table 5). It is a natural result since the run time of BKZ algorithm is proportional to $\log B$ where B is the maximal norm of input basis [10].

However, the bit size of p does not seem to be a critical point for the security. Instead, if we use small p, then the efficiency increase significantly. If we use a 10-bit p, then the key size is comparable to NTRU and its efficiency can be significantly increased.

By our limited and non-optimized experiments, the run time to find the shortest vector in $\bar{L}_B$ with 10-bit number p is longer than $e^{0.1n}$-seconds with Pentium III 866 MHz. (see Table 5.) Based on these experiments, we estimate the security for 10-bit number p as Table 1.

Remark 1. If we use a 10-bit integer p, then p cannot be considered as a secret key. Even if p is not a secret key, it would be better to keep p secret for increasing the security.

Key Sizes Let p be about 10-bit number. The coefficients of P_i will take about 18 bits for 514-dimensional lattice ($n = 257$). Then public key takes 2.3 KBytes.

Let p be about an 80-bit number. The coefficients of P_i will take 88 bits for 514-dimensional lattice ($n = 257$), the public key takes 11.3 KBytes, which is much smaller than the key sizes of both 200-dimensional GGH and 200-dimensional GGH using HNF expression [8].

4.4 Other Representations

Let $r(x)$ be $x^n - x - 1$. Then $r(x)$ is irreducible polynomial in $\mathbb{Z}_p[x]/\langle r(x)\rangle$. Hence every non zero element has its inverse [7]. Let $f_1, f_2, h_1, h_2, g \in \mathbb{Z}[x]/\langle x^n - x - 1\rangle$ be defined by the same method except that $|\alpha_0|, |\beta_0| \approx \sqrt{8n}$ instead of $|\alpha_{i_0}|, |\beta_{j_0}| \approx \sqrt{2n}$ for some i_0, j_0. Then $\Phi(f_i)$ has a low dual-orthogonality-defect.

Table 1. Expected run time to find the shortest vector for Scheme I

n	expected run time
211	1.46×10^9-seconds$\approx$ 46-years
257	1.45×10^{11}-seconds$\approx 4.6 \times 10^3$-years
373	1.58×10^{16}-seconds$\approx 5 \times 10^8$-years
503	5.18×10^{21}-seconds$\approx 1.6 \times 10^{14}$-years

Table 2. Comparison of key sizes (KB) of Scheme I with the GGH

rank of B	10-bit p	80-bit p	GGH	GGH(HNF)
200	0.85	4.4	330	32
300	1.4	6.6	990	75
400	1.8	8.8	2370	140
500	2.3	11		
750	3.6	16.7		
1000	4.8	22.3		

The gap of the embedded lattice is smaller than 6. Our experiments say that the gap is about 4, which is larger than the gap for $r(x) = x^n - 1$. The larger the gap size is, the larger the dimension we need for the security is. As we see in Table 8, if we use $x^n - x - 1$ as $r(x)$, the shortest vector for $n = 79$ is found by BKZ algorithm with block size 10. When we use $x^n - 1$ as $r(x)$, we cannot find the shortest vector for $n = 79$ with block size 20. For the similar complexity of lattice generated by $x^{211} - 1$, we need $n \approx 400$ based on Conjecture 2, the public key size is about 18KBytes, which is also much smaller than the key sizes of both 200-dimensional GGH and 200-dimensional GGH using HNF expression but it is two times larger than the scheme with $r(x) = x^{211} - 1$.

When we use this representation, we have the following advantages: First, $\Phi(f_i)$ is more complicated. Second, if p is a prime number, then every non zero g is invertible in $\mathbb{Z}_p[x]/\langle r(x)\rangle$. But since it seems that there are no special lattice reduction algorithm for $r(x) = x^n - 1$, the scheme with $r(x) = x^n - 1$ is more efficient than that with $r(x) = x^n - x - 1$.

5 Conclusion

We proposed a lattice based public key cryptosystem using polynomial representations. The proposed cryptosystem is an improvement of the GGH system. Our scheme has the advantages of the GGH system written in the introduction. Furthermore, our scheme is practical in key sizes compared with the GGH.

It has not been proved that the security of our scheme is equivalent to that of the GGH scheme since our schemes use specific lattices generated by polynomial representations. Although the further research on the security of the proposed schemes is required, any serious weakness has not been found yet.

As we see in Section 3, 4 and Appendix A, we can make various lattices with representations of polynomials. By studying various representations and size of coefficients of polynomials, the key size might be decreased and the efficiency could be increased. Furthermore, the security of the cryptosystem is closely related to the choice of representations. (See Section 4.4.)

References

[1] D. Coppersmith, A. Shamir *Lattice Attacks on NTRU*, Advances in Cryptology-Eurocrypt '97, LNCS 1233 (1997), 52–61

[2] E. Fujisaki, T. Okamoto *Secure Integration of Asymmetric and Symmetric Encryption Schemes*, Advances in Cryptology-Crypto '99, LNCS 1666 (1999), 537–554 306

[3] C. Gentry *Key Recovery and Message Attacks on NTRU-Composite*, Advances in Cryptology-Eurocrypt '01, LNCS 2045 (2001), 182–194

[4] O. Goldreich, S. Goldwasser, S. Halevi *Public Key Cryptosystems from Lattice Reduction Problems*, Advances in Cryptology-Crypto '97, LNCS 1294 (1997), 112–131 292, 294

[5] J. Hoffstein, J. Pipher , J. Silverman *NTRU : a Ring Based Public Key Cryptosystem*, ANTS III, LNCS 1423 (1998), 267–288 295

[6] E. Jaumels, A. Joux *A Chosen-Ciphertext Attack against NTRU*, Advances in Cryptology-Crypto 2000, LNCS 1880 (2000), 20–35

[7] R. Lidl, H. Niederreiter *Introduction to Finite Fields and Their Applications*, Cambridge University Press, (1986) 300

[8] D. Micciancio *Improving Lattice Based Cryptosystems Using the Hermite Normal Form*, CaLC 2001, LNCS 2146 (2001), 126–145 292, 300, 305

[9] P. Nguyen *Cryptanalysis of the Goldreich-Goldwasser-Halevi Cryptosystem from Crypto '97*, Advances in Cryptology-Crypto '99, LNCS 1666 (1999), 288–304 292, 293, 294, 299

[10] C. P. Schnorr *A Hierarchy of Polynomial Time Lattice Basis Reduction Algorithms*, Theoretical Computer Science 53 (1987), 201–224 295, 300

[11] L. C. Washington *Introduction to Cyclotomic Fields*, Springer-Verlag, GTM 83 (1996)

Appendix A : Scheme II

In this section, we introduce a scheme whose key size is smaller than that of Scheme I.

Key Generation Let n be a prime number and p be a positive integer as Scheme I.

Private key First, we generate 9 polynomials

$$f_i,\ h_k \in \mathbb{Z}[x]/\langle x^n - 1\rangle\ i = 1, 2, 3,\ k = 1, 2, \cdots 6$$

for the private key such that

- $f_1 = \alpha_{n-1}x^{n-1} + \cdots + \alpha_0$, $f_2 = \beta_{n-1}x^{n-1} + \cdots + \beta_0$ and $f_3 = \gamma_{n-1}x^{n-1} + \cdots + \gamma_0$ satisfy that $\alpha_{i_0} = \beta_{j_0} = \gamma_{k_0} \approx \sqrt{3n}$ for some i_0, j_0, k_0 and other coefficients are contained in $\{-1, 0, 1\}$.
- All coefficients of h_i's are contained in $\{-1, 0, 1\}$. Furthermore, $f_2 + h_4 = f_3 + h_6 = q \approx \sqrt{3n}$ for a positive integer q and $h_1 + h_2 = 0$.

The secret data are $\{f_1, f_2, f_3, h_1, h_3, h_5\}$. We make the private matrix R as follows:

$$R = \begin{pmatrix} \Phi(f_1)\ \Phi(h_1)\ \Phi(h_2) \\ \Phi(h_3)\ \Phi(f_2)\ \Phi(h_4) \\ \Phi(h_5)\ \Phi(h_6)\ \Phi(f_3) \end{pmatrix}.$$

In order to generate the public key, we choose $g \in \mathbb{Z}[x]/\langle x^n - 1\rangle$ such that the coefficients of g are contained in $(-p/2, p/2]$. Then g can be considered as an element of a ring $F = \mathbb{Z}_p[x]/\langle x^n - 1\rangle$. We take g which is invertible in F. Then there exist g_p and Q in $\mathbb{Z}[x]/\langle x^n - 1\rangle$ such that $gg_p - 1 = pQ \in \mathbb{Z}[x]/\langle x^n - 1\rangle$. We obtain that

$$\begin{aligned} P_{13} &= h_1 + h_2 = 0 \\ P_{23} &= f_2 + h_4 = q \\ P_{33} &= f_3 + h_6 = q \\ P_{31} &= h_5 g - f_3 Q (\mathrm{mod}\ q) \\ P_{32} &= p h_5 + h_6 g_p (\mathrm{mod}\ q). \end{aligned} \tag{5.8}$$

Every coefficient of P_{31} and P_{32} is contained in $(-q/2, q/2]$. We define T_1, T_2 as follows:

$$\begin{aligned} T_1 &= q^{-1}(P_{31} - h_5 g + f_3 Q) \\ T_2 &= q^{-1}(P_{32} - p h_5 - h_6 g_p). \end{aligned} \tag{5.9}$$

Then we obtain

$$\begin{aligned} P_{11} &= f_1 g - h_2 Q + T_1 P_{13} = f_1 g - h_2 Q \\ P_{12} &= p f_1 + h_1 g_p + T_2 P_{13} = p f_1 + h_1 g_p \\ P_{21} &= h_3 g - h_4 Q + T_1 P_{23} = h_3 g - h_4 Q + q T_1 \\ P_{22} &= p h_3 + f_2 g_p + T_2 P_{23} = p h_3 + f_2 g_p + q T_2 \end{aligned} \tag{5.10}$$

Then we have the public matrix B as follows:

$$B = \begin{pmatrix} \Phi(P_{11})\ \Phi(P_{12})\ \Phi(P_{13}) \\ \Phi(P_{21})\ \Phi(P_{22})\ \Phi(P_{23}) \\ \Phi(P_{31})\ \Phi(P_{32})\ \Phi(P_{33}) \end{pmatrix} = \begin{pmatrix} \Phi(P_{11})\ \Phi(P_{12})\ 0 \\ \Phi(P_{21})\ \Phi(P_{22})\ q \\ \Phi(P_{31})\ \Phi(P_{32})\ q \end{pmatrix}.$$

Consequently, we have the following private key and public key:

- Private key : $f_1, f_2, f_3, h_1, h_3, h_5$ (i.e. R)
- Public key : $P_{11}, P_{12}, P_{21}, P_{22}, P_{31}, P_{32}$ (i.e. B)

Encryption and Decryption

Encryption A message is $M = (m_1, m_2, m_3) \in (\mathbb{Z}[x]/\langle x^n - 1\rangle)^3$. The ciphertext is

$$c = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} P_{11}m_1 + P_{12}m_2 + P_{13}m_3 + e_1 \\ P_{21}m_1 + P_{22}m_2 + P_{23}m_3 + e_2 \\ P_{31}m_1 + P_{32}m_2 + P_{33}m_3 + e_3 \end{pmatrix} = BM + e. \tag{5.11}$$

for an error vector $e = (e_1, e_2, e_3)$, where $e_i \in \{-1/2, 1/2\}^n$ for $i = 1, 2, 3$.

Decryption The deciphertext is

$$M = T\lceil R^{-1}c \rfloor$$

for

$$T = \left\{ \begin{pmatrix} \Phi(g) & pI & 0 \\ 0 & \Phi(g_p) & I \\ -\Phi(Q) & 0 & I \end{pmatrix} \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ T_1 & T_2 & I \end{pmatrix} \right\}^{-1}. \tag{5.12}$$

We can easily check that $B = RT^{-1}$. By the same reason as the GGH and Scheme I, the decryption works.

We can prove that $|\det(T)| = 1$.

Lemma 2. $|\det(T)| = 1$.

Proof. From the equation

$$\begin{pmatrix} \Phi(g_p) & -pI & pI \\ -\Phi(Q) & \Phi(g) & -\Phi(g) \\ \Phi(g_pQ) & -\Phi(pQ) & \Phi(gg_p) \end{pmatrix} \begin{pmatrix} \Phi(g) & pI & 0 \\ 0 & \Phi(g_p) & I \\ -\Phi(Q) & 0 & I \end{pmatrix} = I,$$

we obtain that T^{-1} is invertible, so $|\det(T)| = 1$. □

Remark 2. In (5.12), if $g_1g_2g_3 - 1 = pQ$ generally, we obtain an invertible matrix

$$T^{-1} = \begin{pmatrix} \Phi(g_1) & pI & 0 \\ 0 & \Phi(g_2) & I \\ -\Phi(Q) & 0 & \Phi(g_3) \end{pmatrix}.$$

Note that

$$\begin{pmatrix} \Phi(g_2g_3) & -p\Phi(g_3) & pI \\ -\Phi(Q) & \Phi(g_1g_3) & -\Phi(g_1) \\ \Phi(g_2Q) & -\Phi(pQ) & \Phi(g_1g_2) \end{pmatrix} \begin{pmatrix} \Phi(g_1) & pI & 0 \\ 0 & \Phi(g_2) & I \\ -\Phi(Q) & 0 & \Phi(g_3) \end{pmatrix} = I.$$

In order to reduce the public key size, we replace g_3 by 1 and make modular reduction.

Security and Key Size In algebraic view, we can use the similar arguments on the security as Scheme I.

By our experiments, if we use an 80-bit number as p, the run time to find the shortest vector in $\bar{L}_B$ in Scheme II is about half of the run time for Scheme I. We guess that such results are obtained since the entries of B are smaller than that of Scheme I. If p is a 10-bit number, then the run time is shorter than $1/7$ similarly as Scheme I. Our limited experiments say that the run time of BKZ algorithm for the embedded lattice $\bar{L}$ is longer than $\exp(0.14n)$-seconds with Pentium III 866 MHz.

Based on our experiments (Table 5) , we obtain the security for 10-bit number p as Table 3.

Table 3. Expected run time to find the shortest vector for Scheme II

n	expected run time
137	2.1×10^8-seconds$\approx$ 6.8-years
167	1.4×10^{10}-seconds$\approx$ 451-years
251	1.8×10^{15}-seconds$\approx 5.8 \times 10^7$-years
331	1.3×10^{20} seconds$\approx 4.2 \times 10^{12}$ years

Table 4. Comparison of key sizes (KB) of Scheme II

rank of B	10-bit p	80-bit p	Scheme I with 10-bit p
400	1.4	6.1	1.8
500	1.7	7.6	2.3
750	2.8	11.6	3.6
1000	3.8	15.4	4.8

By modular operations, coefficients of $P_{13}, P_{23}, P_{31}, P_{32}, P_{33}$ are smaller than q, which are relatively small numbers. If $n = 167$, the dimension of the lattice is 501, and the public key size is about 1.7 KBytes for 10-bit p and about 7.6KBytes for 80-bit p.

Remark 3. (1) Our lattice reduction programs used for experiments in Appendix D are not optimized. If the programs are optimized, then the expected run time to solve SVP in Table 1 and Table 3 will be decreased.
(2) Even if the key size of Scheme II is slightly smaller than that of Scheme I, Scheme I seems to be more secure than Scheme II when we use 10-bit p.

Appendix B : IND-CCA2

The GGH system encrypts as follows:

$$c = BM + e,$$

where M is a message and e is an error vector. But this encryption does not satisfy the indistinguishability and is insecure against adaptive chosen ciphertext attack.

Indistinguishability If one encrypts one of two messages M_1 and M_2 and obtain a ciphertext c, then an adversary can distinguish a plaintext as follows: if $||BM_i - c|| < ||BM_j - c||$, then M_i is a plaintext.

In [8], the ciphertext for a message M is as follows:

$$c = B\phi + M,$$

where ϕ is a random vector in $\mathbb{Z}^n$ and $M \in \{-\sigma, \sigma\}^n$. In this case, an adversary distinguishes which of M_i is a message by checking which of $c - M_i$ is contained in $\mathrm{Im}(B)$.

Adaptive Chosen Ciphertext Attack Given a ciphertext c of a message M, i.e. $c = BM + e$, if an adversary inputs the $c + BM'$ to the decryption oracle for some M', then the decryption oracle outputs $\bar{M}$. Then the adversary can find out the original message M by calculating $M = \bar{M} - M'$.

IND-CCA2 For the security against IND-CCA2, we can apply the Fujisaki-Okamoto scheme.[2] We denote $2n$ (resp. $3n$) by N for Scheme I (resp. Scheme II). Let $\mathcal{E}_K, \mathcal{D}_K$ be a symmetric encryption and a decryption from $\mathbb{Z}[x]/\langle x^n - 1\rangle$ to $\mathbb{Z}[x]/\langle x^n - 1\rangle$ with a key K, respectively. Also M, e, B, T and R are the same notations which appeared in 4.1 and 4.2. Let H, G be random oracles. Then $M' = H(e, M)$ and the ciphertext is obtained as follows:

$$c = c_1 || c_2 = (BM' + e) || \mathcal{E}_{G(e)}(M).$$

For the decryption, first we obtain $\bar{M}' = T\lceil R^{-1} c_1 \rfloor$ and $\bar{e} = c_1 - B\bar{M}'$. Second, we obtain a deciphertext $\bar{M}$ with the symmetric key $G(e)$. Finally, if $B(H(\bar{e}, \bar{M})) + \bar{e} = c_1$, then decryption oracle outputs $\bar{M}$, otherwise the decryption fails. Then the security against IND-CCA2 depends on one-wayness of the function $f(m) = Bm + e$.

Remark 4. We can simplify the above scheme as follows:

$$M' = \mathcal{E}_{h(e)}(M) \text{ and } c = BM' + e,$$

for a hash function h. The decryption is as follows: Compute

$$M' = T\lceil R^{-1} c \rfloor \text{ and } e = c - BM'.$$

If

$$e \notin \{-1/2, 1/2\}^N,$$

then the decryption fails. Otherwise, the decryption oracle outputs

$$M = \mathcal{D}_{h(e)}(M').$$

The security of this scheme has not been proved yet but this scheme prevents message expansion in Fujisaki-Okamoto scheme, trivial distinguishability and chosen ciphertext attack described in the above.

Appendix C : Experimental Results

We have the following data for the run time to find the shortest vector in $\bar{L}$. Our program is simply using BKZ algorithm in NTL 5.2, so it is not optimized.

Table 5. Run time ($r(x) = x^n - 1$, Pentium III 866)

n	Scheme	p's bit size	block size	run time (sec)	succeed
31	I	10	4	33.72	succeed
41	I	10	4	147.86	succeed
47	I	10	4	280.78	fail
47	I	10	10	280.67	succeed
59	I	10	10(prune 12)	1003.49	succeed
67	I	10	10(prune 12)	1568.89	fail
79	I	10	20(prune 12)	5602.56	fail
79	I	10	20	7691.87	fail
29	II	10	4	109.3	succeed
31	II	10	4	144.67	succeed
47	II	10	4	1098.9	succeed
47	II	10	10	1169.85	succeed
53	II	10	10(prune 12)	2222.35	fail
53	II	10	20(prune 12)	2373.22	fail
53	II	10	20	2544.43	succeed
59	II	10	25(prune 12)	4704.26	fail
59	II	10	25	4758.63	succeed
41	I	80	4	1388.57	succeed
41	I	80	10	1352.54	fail
47	I	80	4	2681.9	succeed
47	I	80	10	2747.15	succeed
47	I	80	10(prune 12)	2797.96	succeed
59	I	80	4	7421.31	fail
59	I	80	10(prune 12)	8066.63	succeed
67	I	80	10(prune 12)	15117.1	succeed
79	I	80	20(prune 12)	34736.1	succeed
29	II	80	4	952.35	succeed
31	II	80	4	1312.45	succeed
41	II	80	10	5036	succeed
47	II	80	4	10760	fail
47	II	80	10	9597.35	succeed
53	II	80	20(prune 12)	16952.8	succeed
53	II	80	10(prune 12)	17130.1	succeed

Table 6. Comparison of the key sizes for the non reduced GGH and Scheme I with 80-bit p (Pentium III 866)

dimension	block size(GGH)	run time (sec)	succeed	block size(Scheme I)	run time(sec)	succeed
94	4	561.51	succeed	4	280.78	fail
118	4	1768.06	succeed	4	7421.31	fail
118	10(prune 12)	1958.26	succeed	10(prune 12)	8066.63	succeed
158	4	7553.46	succeed	10		fail
158	20(prune 12)	16164.6	succeed	20(prune 12)	34736.1	succeed

Table 7. Run time of Scheme I($r(x) = x^n - 1$), SUN BLADE 1000 750MHZ (Experimentally, for $l = 1$, $G_{\bar{L}} \leq 2.4$. For $l = 3$, $G_{\bar{L}} \leq 6.8$ and for $l = 5$, $G_{\bar{L}} \leq 12$.)

n	Scheme	p's bit size	block size	l	run time (sec)	suceeed
67	I	10	20	1	769	fail
79	I	10	20(prune 12)	1	2218.57	succeed
67	I	80	4	1	3172.6	fail
67	I	80	4	3	3216.02	fail
67	I	80	4	5	3257.28	succeed
67	I	80	10	1	3276.23	fail
67	I	80	10	3	3319.16	fail
67	I	80	10	5	3373.16	succeed
67	I	80	20	1	3771.76	succeed
67	I	80	20	3	3662.1	fail
67	I	80	20	5	3754.34	succeed
79	I	80	4	5	6935.06	succeed
79	I	80	10	1	7053.14	fail
79	I	80	10	3	7151.54	fail
79	I	80	10	5	7137.82	succeed
79	I	80	20	1	8672.55	fail
79	I	80	20	3	8901.04	fail
79	I	80	20	5	9598.09	fail
79	I	80	20(prune 12)	1	8278.63	fail

Table 8. Run time of Scheme I($r(x) = x^n - x - 1$, $G_{\bar{L}} \leq 4.1$, Pentium III 866)

n	block size	run time (sec)	succeed
23	4	81.12	succeed
31	4	336.2	succeed
41	4	1337.66	succeed
47	4	2776.43	succeed
59	4	7543.8	succeed
59	10	8250.79	succeed
67	4	13632.1	succeed
67	10	13773.2	succeed
79	10	29102.3	succeed
79	10	29118.3	fail
89	10	51793.1	fail
89	20	63542.6	succeed