DBTRU, a new NTRU-like cryptosystem based on dual binary truncated polynomial rings

Cao Minh Thang
Posts and Telecommunications Institute of Technology
Hanoi, Vietnam
Email: thangcm@ptit.edu.vn

Nguyen Binh
Posts and Telecommunications Institute of Technology
Hanoi, Vietnam
Email: nguyenbinh@ptit.edu.vn

***Abstract*—NTRU is a probabilistic public key cryptosystem having security related to some hard problems in lattices. The original NTRU operates on the on a $N-th$ degree truncated polynomial ring $R = Z[x]/(x^n+1)$. In this paper, DBTRU, a new variant of NTRU based on two binary truncated polynomial rings $GF(2)[x]/(x^n+1)|n \in Z^+$ is proposed along with some security and performance advantages in comparison with standard NTRU.**

I. Introduction

NTRU [1], proposed by J.Hoffstein, J.Pipher and J.H.Silverman in 1996, is a probabilistic public key cryptosystem runs on a $N-th$ degree truncated polynomial ring $R = Z[x]/(x^n+1)$. NTRU is well-known as one of the fastest public-key encryption schemes and was standardized in 2008 by IEEE in standard P.1363.1.

The hard problem underlying this cryptosystem is finding short polynomials in R. This problem is not equivalent but related to Shortest Vector Problem (SVP) in lattice theory.

Since NTRU was proposed, it has been cryptanalyzed heavily by the cryptographic community, and some interesting results can be found in [3], [5] and [6].

In addition, some interesting variants of NTRU encryption schemes have been proposed. The first proposal is the generalized NTRU scheme [6], in which, the proposed scheme uses two invertible polynomials and respectively two public keys in stead of only one in classical NTRU. Another variant of NTRU proposed in 2002, CTRU [7], used polynomial rings over finite field instead of Z to avoid attacks based on LLL algorithm or Chinese Remainder Theorem on NTRU. MaTRU [8], a NTRU-like cryptosystem that runs on rings of $k-by-k$ matrics of polynomials in ring $R = Z[x]/(x^n-1)$, was introduced in 2005 with respectable speed improvements of $O(k)$ over NTRU. NNRU, introduced in 2009, is a break-through variant using non-commutative algebraic structure to avoid lattice-based attacks on NTRU. Quaterinions and octonions algebra were exploited to construct two NTRU-like cryptosystems, QTRU (in 2009) and OTRU (2010), respectively. As NTRU is a lattice-based cryptosystem, in 2012, by reducing lattice-based public-key cryptosystems to find some certain kinds of easy closest vector problems (CVPs), [10] proposed a general NTRU-like framework for constructing a new lattice-based cryptosystem. Recently, in [11], the Z in NTRU was replaced by the ring of Eisenstein integers to construct a new variant named ETRU which is faster and has smaller keys for the same and or better level of security than NTRU.

In this paper, we propose a new variant of NTRU, DBTRU, that runs on dual binary truncated polynomial rings along with some comparison in theoretical performance and security with standard NTRU.

In section II, we points out that there exists two special truncated polynomial rings having a large number of invertible elements and defines the conditions and an algorithm for identifying them and their inverses. The results in this section are used for finding private keys and selecting parameters of DBTRU. In section III, the cryptosystem DBTRU is proposed in detail with some theoretical performance and security comparison with standard NTRU. The conclusion and other future work are mentioned in section IV.

II. Invertible elements in binary truncated polynomial rings

In this section, we focus on polynomials invertible truncated polynomial rings. For convenience, we denote $GF(2)[x]/(x^n+1)|n \in Z^+$ as $R_n[x]$.

A. Definition and notations

Definition 2.1: The Hamming weight of arbitrary polynomial $f \in R_n[x]$ is denoted as $w(f)$.

Definition 2.2: A polynomial $f \in R_n[x]$ is invertible if there exists $g \in R_n[x]$ such that $f * g = 1 \bmod (x^n+1)$

Definition 2.3: The set of polynomials having odd Hamming weight in $R_n[x]$ is denoted as $I_n[x]$.

It is clear that $|I_n[x]| = 2^{n-1}$.

Definition 2.4: The ratio of the number of invertible elements over the total number of polynomials in $R_n[x]$ is denoted as K_n.

Since invertible polynomials always have odd Hamming weight, by Lemma 2.1, we can see that $max(K_n) = |I_n[x]|/|R_n[x]| = 1/2$.

In the two following lemmas, we show that there are two special classes of binary truncated polynomial rings having large K_n.

Definition 2.5: The set of integers n such that $x^n+1 = (x+1)*T$ where $T = \sum_{i=0}^{n-1} x^i$ is irreducible in $GF(2)$ is denoted as N_{2C}. In this case, $R_n[x]$ is called a polynomial ring with only two cyclotomic cosets.

978-1-4673-6640-3/15/$31.00 ©2015 IEEE

Definition 2.6: The set of integers $n = 2^k$ where k is an arbitrary positive integer is denoted as N_{2^k}.

B. Hamming weight of invertible polynomials

Lemma 2.1: In $R_n[x]$, if $w(f) = 2k, w(g) = 2l|k, l \in Z^+$ then $w(f + g)$ is even.

Proof: By presenting polynomials as $f = \sum_{i=0}^{N-1} f_i x^i$ and $g = \sum_{i=0}^{N-1} g_i x^i$, respectively, we have

$$h = f + g = \sum_{i=0}^{N-1} h_i.x^i$$

where $h_i = (f_i + g_i) \bmod 2$.

Since $f_i, g_i \in GF(2)$ then $h_i = 0$ if and only if $f_i = g_i$. Let S denote the set containing the values i such that $f_i = g_i = 1$. It easy to see that

$$w(h) = w(f) - |S| + w(g) - |S| = 2(k + l - |S|).$$

■

In more general, if h is the summation of polynomials having even Hamming weight then $w(h)$ is even.

Lemma 2.2: In $R_n[x]$, if $w(f)$ is even then $\forall g \in R_n[x]$, $w(g * f)$ is even.

Proof: Suppose that the presentation of g is $g = \sum_{i=0}^{N-1} g_i x^i$ we have

$$h = g * f \bmod (x^N + 1) = \sum_{i=0}^{N-1} g_i.x^i * f.$$

Since $w(g_i.x^i * f) = g_i.w(f)$ and is thus always even, by lemma 2.1, $w(h)$ is even. ■

Lemma 2.3: In $R_n[x]$, all polynomials having even Hamming weight are not invertible.

Proof: Suppose that $f \in R_n[x]$ is a polynomial having even Hamming weight. By lemma 2.2, $\forall g \in R_n[x]$, $w(f * g)$ is always even. Since $w(1) = 1$ then there does not exist any polynomial h such that $w(f * h) = 1$. As a result, we can say, f is not invertible. ■

C. In $R_n[x]|n \in N_{2C}$

Lemma 2.4: In $R_n[x]|n \in N_{2C}$, all polynomials in $I_n[x] \setminus T$ are invertible. Consequently, the number of invertible elements in those rings is $2^{n-1} - 1$.

Proof: According to [18], if $R_n[x]$ is a ring having only two cyclotomic cosets then n is odd prime thereby $w(T)$ is odd. Since $\deg T = n-1$ is the $\max \deg f|f \in R_n[x]$ and T is irreducible over $GF(2)$ then, by definition 2.5, $gcd(f, T) = 1$ $\forall f \in I_n[x] \setminus T$.

In addition, if $f \in I_{2^k}[x] \setminus T$ then $\gcd(f, x^n + 1) = \gcd(f, (x + 1) * T) = 1$. By Euclidean theorem, there always exists two polynomials $u, v \in R_n[x]$ such that $u * f + v * (x^n + 1) = 1 \bmod (x^n + 1)$ or $u * f = 1 \bmod (x^n + 1)$, thus f is invertible with some inverse u. ■

According to [19], $m = 2^{n-1} - 1$ is the maximum order of all polynomials in $I_n[x] \setminus T$ i.e, $f^m = 1 \bmod (x^n + 1)$ and $u = f^{m-1} \bmod (x^n + 1)$ is inverse of f in $R_n[x]|n \in N_{2C}$.

In order to compute the inverse of an invertible polynomial in $R_n[x]$ we can use algorithm 2.226 in [20] which is based on extended Euclidean algorithm for polynomials.

D. In $R_n[x]|n \in N_{2^k}$

Lemma 2.5: In $R_n[x]|n \in N_{2^k}$, all polynomials in $f \in I_n[x]$ are invertible. Consequently, the number of invertible elements in those rings is 2^{n-1} and K_n gets maximum.

Proof: With $n = 2^k$, f can be presented as

$$f = \sum_{i=0}^{2^k-1} f_i x^i | f_i \in GF(2).$$

Since $f_i \in GF(2)$ then

$$\begin{aligned} f^{2^k} &= \sum_{i=0}^{2^k-1} (f_i * x)^{i.2^k} \bmod (x^{2^k} + 1) \\ &= \sum_{i=0}^{2^k-1} {f_i}^{2^k} * x^{i.2^k \bmod 2^k} = \sum_{i=0}^{2^k-1} f_i \\ &= w(f) \bmod 2. \end{aligned}$$

If $f \in I_n[x]$ then $w(f)$ is odd thus $w(f) \bmod 2 = 1$. As a result, $f^{2^k} = 1 \bmod (x^{2^k} + 1)$. Let $g = f^{2^k-1} \bmod (x^{2^k} + 1)$ we have $g * f = 1 \bmod (x^{2^k} + 1)$ and g is the inverse of f. ■

In [19], $n = 2^k$ is pointed out the maximum order of all polynomials $f \in R_{2^k}[x]$ and $ord(f)$ divides 2^k. Hence, we can find the inverse of more efficiently by the following algorithm instead of computing $g = f^{2^k-1} \bmod (x^{2^k} + 1)$.

Algorithm 2.1: Algorithm for finding the inverse of invertible element in $R_{2^k}[x]$
INPUT: A polynomial $f \in I_{2^k}[x]$.
OUTPUT: A polynomial $g \in I_{2^k}[x]$ such that $g * f = 1 \bmod x^{2^k} + 1$.
ALGORITHM:

```
1) Set $f \leftarrow g$.
2) Set $a \leftarrow f^2 \bmod (x^{2^k} + 1)$.
3) For $i$ from 1 to $k - 1$ do
    a) If $f * g = 1 \bmod (x^{2^k} + 1)$ return $g$.
    b) Set $g \leftarrow g * a \bmod (x^{2^k} + 1)$.
    c) Set $a \leftarrow a^2 \bmod (x^{2^k} + 1)$.
```

III. Proposed public-key cryptosystem

In this section, the modular multiplication and reduction in two truncated polynomial rings $R_s[x]$ and $R_l[x]$ are exploited to construct DBTRU cryptosystem whose security comes from the hard problem of finding polynomials invertible in both those rings.

A. Notations

For easy comparison between NTRU and DBTRU we reuse some notations in NTRU including parameters f, g, ϕ, m, h, e, a and truncated ring $R = Z[x]/(x^N - 1)$.

In DBTRU, we use four values d_f, d_g, d_ϕ and d_m to denote the maximum degree and Hamming weight of four polynomials f, g, ϕ and m, respectively.

Besides, we replace the definition $\mathcal{L}(d_1, d_2)$ by

$$\mathcal{B}(d) = \{b \in R_l x]| \deg b \leq d\}.$$

In addition, because DBTRU runs on dual binary truncated polynomial rings, $R_l[x]$ is larger than $R_s[x]$, we denote two inverses of private key f in those rings as F_l and F_s, respectively. At last, for convenience, we set $S = (x^s + 1)$ and $L = (x^l + 1)$.

B. Key generation

Bob chooses two arbitrary positive integers $s < l$ and set $d_f = s - 1$. In addition, Bob chooses an small positive integer N_f and arbitrary N_f polynomials $f_i \in \mathcal{B}_f | i \in [1, N_f]$ which are invertible in both $R_s[x]$ and $R_l[x]$. For each f_i, Bob computes $F_{i,s} \in R_s[x]$ and $F_{i,l} \in R_l[x]$ where $F_{i,s} * f_i = 1 \bmod S$ and $F_{i,l} * f = 1 \bmod L$. After that, Bob computes

$$f = \prod_{i=1}^{N_f} f_i \tag{1}$$

with its two inverses

$$F_s = \prod_{i=1}^{N_f} F_{i,s}$$

and

$$F_l = \prod_{i=1}^{N_f} F_{i,l}.$$

Notice that $\deg f \leq N_f.d_f$.

To obtain $L - bit$ public key, Bob chooses a non-zero $g \in \mathcal{B}_g$ and computes

$$h = g * F_l * S \bmod L \tag{2}$$

and then Bob keeps f, f_i and F_s as private keys (F_l can be discarded) and sends s, l and h to Alice.

C. Encryption

To encrypt $S - bit$ plain-text message m, Alice selects a non-zero arbitrary $\phi_0 \in \mathcal{B}_\phi$, a small positive integer N_ϕ and arbitrary N_ϕ polynomials $\phi_i \in \mathcal{B}_\phi | i \in [1, N_\phi]$ to compute

$$e = (\phi_0 * h + S * \sum_{i=1}^{N_\phi} \phi_i + m) \bmod L \tag{3}$$

and sends $L - bit$ cipher-text e to Bob.

D. Decryption

When receiving e, Bob computes

$$a = f * e \bmod L \tag{4}$$

and then recovers

$$m = F_s * a \bmod S. \tag{5}$$

E. Proof of decryption

By replacing (4) into (5) we have

$$\begin{aligned} a &= f * e \bmod L \\ &= f * (\phi * h + S * \textstyle\sum_{i=1}^{N_\phi} \phi_i + m) \bmod L \\ &= (f * \phi * g * F_l * S + f * S * \textstyle\sum_{i=1}^{N_\phi} \phi_i + f * m) \bmod L \\ &= ((\phi * g + f * \textstyle\sum_{i=1}^{N_\phi} \phi_i) * S + f * m) \bmod L. \end{aligned}$$

Hence, $F_s * a \bmod S = F_s * f * m \bmod S$ thereby

$$m = F_s * a \bmod S.$$

F. Decryption criteria

Since

$$\deg(\phi * g * S) \leq d_\phi + d_g + s,$$

$$\deg(f * S * \textstyle\sum_{i=1}^{N_\phi} \phi_i) \leq N_f.d_f + s + d_\phi$$

and

$$\deg(f * m) \leq d_f + d_m$$

then, if we choose N_f such that

$$d_g < N_f.d_f \tag{6}$$

then, by (4), we have

$$\deg a \leq N_f.d_f + s + d_\phi$$

thereby, for successful decryption, we must ensure

$$l > \max(\deg a) = N_f.d_f + d_\phi + s. \tag{7}$$

G. Security analysis

In this sub-section, some attacks in [1] are applied on DBTRU to analyze the security of this cryptosystem in comparison with NTRU. For convenience, we denote $key - security$ and $message - security$ of DBTRU as S_k and S_m, respectively.

1) Brute-force attacks: Since $f * h = g * S \bmod L$, to recover private key f attackers can try all $f_i \in \mathcal{B}_f$ and check whether $\deg g * S = \deg(f * h) \bmod l < 2s$. Therefore,

$$S_k = C_{|\mathcal{B}_f|}^{N_f}$$

Similarly, attackers can guess m by trying all $\phi_0, \phi_i \in \mathcal{B}_\phi$ and testing if $\deg m = \deg(e + \phi_0 * h + \sum_{i=1}^{N_\phi} \phi_i) \bmod l < s$. Hence,

$$S_m = C_{|\mathcal{B}_\phi|}^{N_\phi + 1}$$

2) Meet-in-the-middle attacks: According to [17], the meet-in-the-middle attacks can be applied to DBTRU and cut the brute-force search time for f and m by a factor of square root. Hence, the key-security and message-security of DBTRU are

$$S_k = \sqrt{C^{N_f}_{|\mathcal{B}_f|}} \tag{8}$$

and

$$S_m = \sqrt{C^{N_\phi}_{|\mathcal{B}_\phi|}}. \tag{9}$$

3) Multiple transmission attacks: In DBTRU, since e depends not only one ϕ as in NTRU but $N_\phi + 1$ blinding polynomials so that the multi-transmission in [1] is not a threat.

However, in two following subsections, we show that the algebraic attack on CTRU in [21] can be applied on f and m in DBTRU and we must choose parameters l carefully to avoid this kind of attack.

4) Attack on f by using algebraic linear equations: By (2) we can write

$$f * h = g * S + u * L. \tag{10}$$

Let d_u is the maximum degree of u we have

$$\begin{aligned}\deg u &= \deg(u * L) - \deg L\\ &= \deg(f * h + g * S) - l = \deg(f * h) - l\\ &\leq N_f.d_f + (l-1) - l\\ &= N_f.d_f - 1.\end{aligned}$$

and, consequently, $d_u = N_f.d_f - 1$ and

$$\deg(f * h) \leq N_f.d_f + l - 1.$$

Notice that the number of linear equations built on (10) is

$$E_f = \deg(f * h) + 1 = N_f.d_f + l$$

while the hidden polynomials include $f_i | i \in [1, N_f]$, g, u with totally

$$\begin{aligned}V_f &= (N_f.d_f + 1) + (d_g + 1) + (d_u + 1)\\ &= 2N_f.d_f + d_g + 2\end{aligned}$$

unknown coefficients. As a result, to avoid recovering f, g and u we must ensure $E_f < V_f$ that means

$$l < N_f.d_f + d_g + 2 \tag{11}$$

5) Attack on m by using algebraic linear equations: By (3) we can write

$$e = \phi_0 * h + S * \sum_{i=1}^{N_\phi} \phi_i + m + v * L \tag{12}$$

Let d_v is the maximum degree of v we have

$$\begin{aligned}\deg v &= \deg(v * L) - \deg L\\ &= \deg(e + \phi_0 * h + S * \sum_{i=1}^{N_\phi} \phi_i + m) - l\\ &= \deg(\phi_0 * h) - l\\ &\leq d_\phi + l - 1 - l\\ &= d_\phi - 1\end{aligned}$$

and, consequently, $d_v = d_\phi - 1$ and $\deg e \leq d_\phi + l - 1$.

Notice that the number of linear equations built on (12) is

$$\begin{aligned}E_m &= \max(\deg e) + 1\\ &= (d_\phi + l - 1) + 1\\ &= d_\phi + l.\end{aligned}$$

On the other hand, the hidden polynomials include ϕ, $\phi_i | i \in [1, N_\phi]$, m, v with totally

$$\begin{aligned}V_m &= (d_\phi + 1) + (N_\phi.d_\phi + 1) + (d_m + 1) + (d_v + 1)\\ &= (N_\phi + 2).d_\phi + d_m + 3\end{aligned}$$

unknown coefficients. As a result, to avoid recovering ϕ, ϕ_i, m and v we must ensure $E_m < V_m$ that means

$$l < (N_\phi + 1).d_\phi + d_m + 3 \tag{13}$$

6) Lattice-based attacks: In DBTRU, the associated lattice is generated by the rows of the following matrix

$$M_{DBTRU} = \left(\begin{array}{c|c} \alpha I_{l\times l} & H_{l\times l}\\ \hline 0_{l\times l} & 2.I_{l\times l}\end{array}\right)$$

Similar to NTRU, in theory, attackers can use some lattice-reduction technique as LLL [2] to find short vectors close to $(\alpha.f, g)$. However, this kind of attack is out of scope of this paper and we will consider it thoroughly in the future works.

H. Theoretical performance analysis

The important advantage of DBTRU is computation speed, both algorithms for encryption and decryption of DBTRU are only one simple modular polynomial multiplication and cost $O((\log_2 l)^2)$ bit operations. However, like NTRU, the cipher-text in DBTRU is expanded and larger than the plain-text by the factor of l/s.

I. Parameter selection

It is note that if s divides l then attackers can easily recover m from e by compute $m = e \bmod S$ without private keys f and F_s. Hence, in order to prevent trivial reduction attack, we must choose s and l such that

$$\gcd(s, l) = 1. \tag{14}$$

In order to get the maximum key-security we should choose s and l such that K_s gets maximum. By lemma 2.4 and lemma 2.5, we can choose

$$s, l \in N_{2C} \tag{15}$$

or

$$s \in N_{2C}, l \in N_{2^k} \tag{16}$$

or

$$s \in N_{2^k}, l \in N_{2C}. \tag{17}$$

It is very interesting that those values, naturally, not only satisfy condition (14) but also ensure that nearly every invertible $f \in R_s[x]$ is also invertible in $R_l[x]$ where $l > s$.

For simple, normally, we choose $d_f = d_m = s - 1$. Since $d_\phi = d_f$ then $|\mathcal{B}_\phi| = 2.\max(|\mathcal{B}_f|)$, we should choose $d_\phi < d_f$ to balance S_m and S_k.

J. Comparison with NTRU

The comparison underlying algebraic structures of DBTRU in comparison with NTRU are described in table I.

TABLE I. UNDERLYING ALGEBRAIC STRUCTURES OF EQUIVALENT PARAMETERS OF NTRU AND DBTRU

NTRU	DBTRU
$N, p, q \in Z^+$, $\gcd(p, q) = 1$	$s, l \in Z^+$, $\gcd(s, l) = 1$
$\mathcal{L}_f = \mathcal{L}(d_f, d_f - 1)$	$\mathcal{B}_f = \mathcal{B}(d_f)$
$\mathcal{L}_g = \mathcal{L}(d_g, d_g)$	$\mathcal{B}_g = \mathcal{B}(d_g)$
$\mathcal{L}_\phi = \mathcal{L}(d, d)$	$\mathcal{B}_\phi = \mathcal{B}(d_\phi)$
$\mathcal{L}_m = \{m \in R, m_i \in [-(p-1)/2, (p-1)/2]\}$	$\mathcal{B}_m = B(d_m)$

Table II shows the theoretical performance of DBTRU in comparison with NTRU. It is clear that DBTRU is as fast as NTRU.

TABLE II. THEORETICAL PERFORMANCE AND SECURITY OF DBTRU IN COMPARISON WITH NTRU

	DBTRU	NTRU
Public Key (bits)	l	$M = N.log_2 q$
Private Key (bits)	$2.N_f.d_f$	$2N.log_2 p$
Cipher-text	l	$N.log_2 q$
Plain-text	s	$N.log_2 p$
Encryption	$O((log_2 l)^2)$	$O((log_2 M)^2)$
Decryption	$O((log_2 l)^2)$	$O((log_2 M)^2)$
Message-expansion	l/s	$log_p q$
Decryption criteria	$l > N_f.d_f + d_\phi + s$	$\lvert f * m + p\phi * g \rvert_\infty < q$
S_m	$C_{\lvert \mathcal{B}_f \rvert}^{N_f}$	$\frac{1}{d!}\sqrt{\frac{N!}{(N-2d)!}}$
S_k	$C_{\lvert \mathcal{B}_\phi \rvert}^{N_\phi + 1}$	$\frac{1}{d_g!}\sqrt{\frac{N!}{(N-2d_g)!}}$

The comparison in three proposed security cases of NTRU and DBTRU in [1] is given in table III, table IV and table V. The theoretical results show that, at nearly the same security levels, DBTRU always uses much smaller keys. However, the massage-expansion factors in DBTRU are a little higher than those in NTRU.

TABLE III. COMPARISON IN MODERATE SECURITY MODE OF NTRU

Moderate security	NTRU	DBTRU
Basic parameters	$(N, p, q, d_f, d_g, d) = (107, 3, 64, 15, 12, 5)$	$(S, L, d_\phi, d_g, N_f, N_\phi) = (37, 197, 27, 105, 3, 4)$
S_m	$2^{26.5}$	$2^{51.21}$
S_k	2^{50}	$2^{51.71}$
Public key (bits)	642	197
Private key (bits)	340	222
Message-expansion	3.78	5.32

IV. CONCLUSION

By exploiting two special kinds of binary truncated polynomial rings $R_n[x]$, DBTRU is a new variant of NTRU having some important advantages in both security and performance comparison with standard version. However, since NTRU has been attacked heavily by cryptographic community [3], [5], [6], in the future, we will consider DBTRU thoroughly under other various attacks to evaluate the security of this cryptosystem.

TABLE IV. COMPARISON IN HIGH SECURITY MODE OF NTRU

High security	NTRU	DBTRU
Basic parameters	$(N, p, q, d_f, d_g, d) = (167, 3, 128, 61, 20, 18)$	$(s, l, d_\phi, d_g, N_f, N_\phi) = (59, 293, 44, 120, 3, 4)$
S_m	$2^{77.5.5}$	$2^{85.71}$
S_k	$2^{82.9}$	$2^{85.71}$
Public key (bits)	1169	293
Private key (bits)	530	354
Message-expansion	4.23	4.97

TABLE V. COMPARISON IN HIGHEST SECURITY MODE OF NTRU

Highest security	NTRU	DBTRU
Basic parameters	$(N, p, q, d_f, d_g, d) = (503, 3, 256, 216, 72, 55)$	$(s, l, d_\phi, d_g, N_f, N_\phi) = (197, 1019, 147, 500, 3, 4)$
S_m	2^{170}	$2^{292.70}$
S_k	2^{285}	$2^{291.71}$
Public key (bits)	4024	1019
Private key (bits)	1595	1182
Message-expansion	5.05	5.17

REFERENCES

[1] Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman. NTRU: Alice ring-based public key cryptosystem. Lecture Notes in Computer Science Volume 1423, pp 267-288, Springer Verlag 1998.

[2] A.K. Lenstra, H.W. Lenstra, L. Lovsz, Factoring polynomials with polynomial coefficients, Math. Annalen 261 (1982), 515-534.

[3] E. J aulme s and A. Joux. A Chosen Ciphertext Attack on NTRU. In Proceeding ofCRYPTO 00, LNCS, vol. 1880, Springer-Verlag, pp. 20-35, 2000.

[4] C. Gentry. Key recovery and message attacks on NTRU-composite. In Proceeding of Eurocrypt 01, LNCS, vol. 2045, Springer-Verlag, pp.182-194, 2001.

[5] Daewan Han, Jin Hong, Jae Woo Han and Daesung Kwon. Key recovery attacks on NTRU without ciphertext validation routine. In Proceeding of ACISP 03, LNCS, vol. 2727, Springer-Verlag, pp.274-284, 2003.

[6] William D. Banks, Igor E. Shparlinski. A Variant of NTRU with Non-invertible Polynomials. Lecture Notes in Computer Science Volume 2551, 2002, pp 62-70.

[7] Gaborit, P., Ohler, J., Sole, P.: CTRU, a Polynomial Analogue of NTRU, INRIA. Rapport de recherche, N.4621 (November 2002), (ISSN 0249-6399).

[8] Michael Coglianese, Bok-Min Goi. MaTRU: A New NTRU-Based Cryptosystem. Lecture Notes in Computer Science Volume 3797, 2005, pp 232-243.

[9] Malekian, E. Zakerolhosseini. OTRU: A non-associative and high speed public key cryptosystem. A.Computer Architecture and Digital Systems (CADS), 2010 15th CSI International Symposium on, Tehran, pp 83 90, ISBN: 978-1-4244-6267-4.

[10] Yanbin Pan, Yingpu Deng. A General NTRU-Like Framework for Constructing Lattice-Based Public-Key Cryptosystems. Lecture Notes in Computer Science Volume 7115, 2012, pp 109-120.

[11] Katherine Jarvis, Monica Nevins. ETRU: NTRU over the Eisenstein integers. Springer Date: 13 Jul 2013.

[12] Yanbin Pan, Yingpu Deng, Yupeng Jiang, Ziran Tu. A New Lattice-Based Public-Key Cryptosystem Mixed with a Knapsack. Lecture Notes in Computer Science Volume 7092, 2011, pp 126-137.

[13] Jin -Yi Cai, Thomas W. Cusick. A Lattice- Based Public-Key Cryptosystem. Lecture Notes in Computer Science Volume 1556, 1999, pp 219-233.

[14] J. Hoffstein and J.H. Silverman. Optimizations for NTRU. In Public-key Cryptography and Computational Number Theory, DeGruyter, 2000.

[15] J. Hoffstein and J.H. Silverman. Random small hamming weight products with applications to cryptography. Discrete Applied Mathematics, vol. 130, Issue 1 - special issue on the 2000 com2MaC workshop on cryptography, pp. 37 - 49, 2003.

[16] P. Karu and J. Loikkanen. Practical comparison of fast public-key cryptosystems. Seminar on Network Security, Telecommunications Software and Multimedia Laboratory, Kelsinki University of Technology. Available at $http : //www.tml.tkk.fi/Opinnot/Tik - 110.501/2000/papers/loikkanen_karu.pdf$.

[17] N. Howgrave-Graham, J.H. Silverman, W. Whyte, NTRU Cryptosystems Technical Report 004, Version 2: A Meet-In-The-Middle Attack on an NTRU Private Key.

[18] Dang Hoai Bac, Nguyen Binh, Nguyen Xuan Quynh , Young Hoon Kim (2007). Polynomial rings with two cyclotomic cosets and their applications in Communication, MMU International Symposium on Information and Communications Technologies 2007, Malaysia, ISBN: 983-43160-0-3.

[19] Nguyen Binh, Le Dinh Thich (2002), The order of polynomials and algorithms for defining Oder of Polynomial over polynomial rings, VICA-5, Hanoi, Vietnam.

[20] Menezes A. J, Van Oorchot P. C. (1998), Handbook of Applied Cryptography, CRC Press.

[21] Nitin Vats. Algebraic Cryptanalysis of CTRU Cryptosystem. Computing and Combinatorics Lecture Notes in Computer Science Volume 5092, Springer-Verlag, 2008, pp 235-244.