OPTIMIZATION STUDY ON NTRU POLYNOMIAL MULTIPLICATION

Yunpeng Zhang\textsuperscript{1,2*}, Peng Sun\textsuperscript{3}, Haozhi Deng\textsuperscript{1}, Peiyong Hou\textsuperscript{1}, Xi Liu\textsuperscript{1}

1. School of Software and Microelectronics, Northwestern Polytechnical University  
710072 Xi’an, China, poweryp@163.com,  
2. Imperial College London, SW3 6NH, London, U.K.  
3. School of Computer, Northwestern Polytechnical University, 710072, Xi’an, China

ABSTRACT
This paper expatiated that the most of PKC had a problem on speed. It had a research on NTRU and found out the influence of the NTRU polynomial multiplication optimization to the algorithm security. After optimization NTRU in process of encryption algorithm using the multinomial decompose, in process of decryption using the recursive method of bisection. At last, it analysed the efficiency and security of this algorithm and proved the efficiency of this algorithm is increased highly and the security is not decreased at the same time.

KEY WORDS
Encrypt Public Key, NTRU, and Polynomial multiplication.

1. Introduction

At present, most PKC(Public-Key Cryptography) algorithms have a low working efficiency.\textsuperscript{[1]} Based on its overwhelming advantages, the RSA(a famous algorithm for public-key encryption) Encrypt Public Key cryptosystem has been the mainstream of Encrypt Public Key cryptosystem. However, the RSA is quite slow on speed. In order to meet the requirement of security, at the same time to satisfy the increasing need for information, a new cryptosystem with a high speed is seriously wanted. Therefore, to solve the bottleneck of PKC speed, Joffery Hoffstein with other mathematicians developed the NTRU(Number Theory Research Unit) \textsuperscript{[2]} Encrypt Public Key cryptosystem, whose security was based on the "shortest vector" problem.

The NTRU is a relatively young but move efficient cryptosystem, and its security is based on a mathematical problem of the "shortest vector" problem.\textsuperscript{[3, 4]} It has a preferable security, so it would be extremely difficult for modern cryptogram analysing technology to break it. Compared with other Public Key cryptosystem, though the speed of the NTRU Polynomial multiplication is immeasurable, O(n2)'s time complexity is obviously not suitable if we want to develop further for a higher speed and lower requirement.

Thus, the research of how to optimize the efficiency of the NTRU Polynomial Multiplication but not debase

the security at the same time becomes a new problem. For those problems above, this paper mainly research on the time-consuming part of the NTRU.\textsuperscript{[5-7]} In the end, this paper concluded that the efficiency of this algorithm is highly increased and the security is not apparently decreased at the same time.

2. Algorithm Description

2.1 NTRU Public Key Algorithm Analysis

Working space:
NTRU cryptosystem depends on there parameters (N, P, q) and N-1 polynomial with integer coefficients of four polynomial set \( L_f, L_g, L_r, L_m \). In this part: p, q need not to be prime numbers, but \( \gcd(p, q) = 1 \), and the general assumption is \( q > p \); Then we make this:

\[ L = L(d_1, d_2) = \{ f | f \in R \} \]

Where F's \( d_i \) coefficient is 1, the coefficient of \( d_2 \) is -1, the remains are 0; so we have

\[ L_f = L(d_f, d_f - 1), L_g = L(d_g, d_g), \]
\[ L_r = L(d_r, d_r - 1) \]

(2) Definition \textsuperscript{[8]}: For \( \forall f, g \in R, f = (f_0, f_1, \cdots, f_{N-1}), g = (g_0, g_1, \cdots, g_{N-1}) \),

\[ h = f \ast g = (\sum_{j=0}^{N-1} f_j x^j) (\sum_{i=0}^{N-1} g_i x^i) \mod x^N - 1 = \sum_{i+j \equiv k \mod N} f_i g_j x^k \]

Can be called an R loop on the sum-product algorithm.
(3) For \( \forall f \in \mathbb{R} \), \( f = (f_0, f_1, \ldots, f_{N-1}) \) lets define:

\[
    f \mod q = (f_0 \mod q, f_1 \mod q, \ldots, f_{N-1} \mod q)
\]

and this is called a polynomial modulo.

In this part:

(1) During the encryption process, it will use three mode operations--- (mod \( q \)) , (mod \( p \)) and (mod \( x^N - 1 \)). The result of (mod \( q \)) will be limited to \((-q/2, q/2)\), instead of \((0, q)\). (mod \( p \))’s results will be limited to \((-p/2, p/2)\), instead of \((0, p)\). And \( GF(p) \) elements are also restricted in \((-p/2, p/2)\).

(2) In general, calculation of \( f \cdot g \) need \( N^2 \) times' calculation, however, for NTRU, the coefficient for \( f \) and \( g \) is very small, which is -1 or 0 or 1, thus making the calculation of \( f \cdot g \) move quick.

2.2 NTRU Public Key Algorithm

Working space:

NTRU depends on three parameter of \((N, p, q)\), and \( N - 1 \) polynomial with inter coefficient of 4 polynomial set \( L_f, L_g, L_r, L_m \), \( p \) And \( q \) need not to be prime, but \( \gcd(p, q) = 1 \), and the general assumption is \( q \gg p \);

We let \( p = 2 + x \)

Then: \( L = L(d) = \{ f \in \mathbb{R} \} \) the coefficient of \( F \) is 1, while others are 0; so we get \( L_f = L(d_f) \), \( L_g = L(d_g) \), \( L_r = L(d_r) \).

Key Generation:

Randomly select two polynomials of degree \( N - 1 \)

\( F \in L_f \), \( g_1, g_2, g_3 \in L_g \), \( p \) and \( q \) all have inverse elements, that is to say, there exist \( F_p \) and \( F_q : f = 1 + p \cdot F \) \( F_p = 1 \mod p \), \( F_q \cdot f = 1 \mod q \)

then calculate

\[
    h = F_q \cdot g \mod q = ((g_1 \cdot F_q) \cdot g_2 + g_3 \cdot F_q) \mod q
\]

so public key generated is \( h \), and private key is \( f \) (\( F_p \) here has no need for storage.)

The Encryption Process:

If the sending plaintext is \( m \in L_m \), then randomly select \( r_1, r_2, r_3 \in L_r \), and use the public key \( h \) to calculate

\[
    c = (r \cdot h + m) \mod q = ((r_1 \cdot h) \cdot r_2 + r_3 \cdot h) + m) \mod q
\]

So we can get the plaintext \( c \).

The Decryption Process:

After receiving the encrypted \( c \), use your own private key \( f \) to calculate \( a = c \cdot f \mod q \); during this process please use the score of recursive algorithm. Then calculate \( I = e(I) - r(1)h(1) \mod q \) and put \( I \) in \((N - q/2, N + q/2)\) then

\[
    A = (p(1) - r(1)g(1) + If(1) \mod q)
\]

adjust the coefficient of \( a \) in \((A - q/2, A + q/2)\), followed by

---

Figure 1. Optimization NTRU algorithm flow
calculation of \( m = a \mod p \), at last we recover \( m \) clearly.

2.3 Algorithm Flow
Optimization NTRU algorithm flow chart is in Figure 1:

3. The Experimental Results and Performance Analysis

3.1 Time Complexity of the Algorithm
First of all, the encryption process needs \((d_r + d_s + d_t)N\) times of addition operation; here it transfers all multiplications into addition operations, thus greatly improving the efficiency. The original algorithm's time complexity is \( O(N^2) \) operation, but this algorithm involves only one multiplication and one addition.

Secondly, as the encryption process uses the module operation of once linear operations, so the complexity can be ignored, then the final complexity is \( O(3/4N^2) \), while the original complexity is \( 2O(N^2) \). From this we can easily sense the effect of optimization.

From the aspect of time complexity, the process of encryption is efficient than the process of decryption. Because the decryption process doesn't use a polynomial, so it cannot use calculation of encryption to calculate.

3.2 Safety Analysis
Because NTRU optimization algorithm is polynomial decomposition from the perspective of identity transformation, so compared with the original algorithm, its security is just like the original one. But special attention should be paid to the usage of the binary small polynomial, which causes aloes of key space. So in the choice of parameters, a compensation of two times non zero should be used to ensure the safe performance of the algorithm.

A. Security of the cryptographic key
The security of the cryptographic key reflects on the key random and the key length. For multinomial \( f \), because of the addition to the coefficient of \( F \in L_f \), after polynomial transform, the random of \( f \) weakens. But this is enough for itself as a private key. For public key \( h \), after a modular multiplication, it becomes a pseudo random polynomial. At this point, security of both this arithmetic and the original one are the same. The main quotation's key length on two parameters, \( p \) and \( q \). Here we transform \( p \) into a polynomial, and the minimum of the polynomial is 3. So the new arithmetic has a longer key length, making it more secure than the original one. Compared with the same level of RSA, NTRU has more key length than RSA, so the NTRU is superior.

B. Exhaustive attack
Here we have \( h = F_q \ast g(\mod q) \) and \( f \ast h = g(\mod q) \), because \( f \) and \( g \) both have very small coefficient 1, so the attacker can try all possible \( f \in L_f \) and inspect whether \( f \ast h(\mod q) \) has small part. He can also try all possible \( g \in L_g \) and inspect whether \( g \ast h^{-1}(\mod q) \) small part has, or try all possible \( r \in L_r \) and inspect whether \( (e - r \ast h) \mod q \) small part has. In fact, \( L_g \) is smaller than \( L_f \), so the security of the cryptographic key depends on \( \#L_g \) (\#\( L_g \) represents the element number of \( L_g \)), while the security of the plain text depends on \( \#L_r \). However, this type of attack aims at the loophole of key length, that is, if the key length is not enough, it may be attacked. But from the above analysis, we can see that, this new arithmetic compensates some 1 to make up for the loss. Besides, \( p \) has particular random. So in the aspect of security, the new arithmetic doesn't have many differences from the original one.

C. Multiple transmission attack method
If we use the same public key to send the plaintext \( m \), for many times, then the attacker can recover most parts of plaintext. If the private text we send is \( e_i = (r_i \ast h + m) \mod q \), here we have \( i = 1, 2, \cdots, N \), then the attacker can work out \( (e_i - e_i) \ast h^{-1}(\mod q) \) and he can recover \( r_i \). Because the coefficient of \( r_i \) is very small, so the attacker recover \( r_i - r_i(\mod q) \) in fact, then he can recover most parts of the plaintext. In our daily use, we had better from a good habit to change the cryptographic key after a period. The asymmetric cryptography itself has solved the problem of key distribution to a large extent, so changing the cryptographic key for symmetric cryptosystem is an easy task.

D. The attack method based on lattice
The target of lattice attack is one or more short vectors chosen in the given lattices. Theoretically, we can use the exhaustive count method to find the shortest vector. But in fact, the dimension of the lattices is very big, so it is impossible to calculate it. At this point, the new arithmetic doesn't change the dimension of the lattice, so compared to the original arithmetic, its security remains unchanged.

4. Conclusion
Based on NTRU Arithmetic, this text uses cryptography and authentication theory to optimize the key process of
NTRU Arithmetic. Ring product algorithm. This text forms an optimized code system and makes system researches on its security. It exhibits some common attack methods, and makes some discussion: The result proves that the efficiency of the polynomial multiplication is increased highly and the security is not debasing at the same time.

Acknowledgments

This work is supported by Science and Technology Development Project of Shaanxi Province Project (2010K06-22g), Industrial application technology research and development project of Xi’an (CXY1118 (1)), Basic research fund of Northwestern Polytechnical University (GAKY100101), R Fund of College of Software and Microelectronics of Northwestern Polytechnical University (2010R001).

References