# A Comparative Analysis & Enhancement of NTRU Algorithm for Network Security and Performance Improvement

Rashmi Jha
Ph.D. Research Scholar (Computer Applications)
Teerthanker Mahaveer University
Moradabad, UP, India
e-mail: rashmijha1909@gmail.com

Anil Kumar saini
Professor (IT), USMS
Guru Govind Singh Indraprastha University
New Delhi, India
e-mail: aksaini1960@gmail.com

*Abstract*: *Security is a vexing, costly and complicated business. A single lapse can be expensive in lost funds, records and reputation. The NTRU algorithm can be used in a variety of applications which involve security across a network. In this research paper, we introduce the general structure of NTRU cryptosystem, its performance analysis and some enhancements of the algorithm. We have also studied the previous algorithms, found out their exact limitations, and developed the NTRU algorithm which is providing a solution to overcome the limitations of these algorithms. In this way we are also increasing the usefulness of NTRU algorithm and its robustness.*

Keywords- NTRU Algorithm, Spam, Digital Signature, modulo, Convolution, PKCS, Cryptosystem.

## I. INTRODUCTION

A network like the Internet is not a secure channel unless we use a cryptosystem. When using symmetric key cryptosystems, two communicating parties have to agree on a secret key. Thus symmetric key cryptosystems require a secure way to make this agreement and the most common way is to use a public key cryptosystem. Also a secure channel can be established by using only a public key cryptosystem instead of using both a symmetric and a public key system. The NTRU cryptosystem, which is first introduced in 1996, is a relatively new cryptosystem. It is claimed to be much faster than the current and more widely used public key cryptosystems such as RSA and McEliece. The NTRU algorithm uses a mixing system based on polynomial algebra and reduction modulo two numbers p and q. Its validity depends on elementary probability theory [1][2]. Since NTRU is a ring based public key cryptosystem and is therefore quite different from the group based cryptosystems whose security relies on the integer factorization problem or the discrete logarithm problem. This extra structure can be exploited to obtain a very fast cryptosystem: to encrypt/decrypt a message block of length N, NTRU only requires $O(N^2)$ time, whereas the group based schemes like RSA etc. require $O(N^3)$ time. Furthermore, NTRU also has a very short key size of O(N) and very low memory requirements, which makes it ideal for constrained devices such as smart cards[6][7][13].

In this paper, we first reviewed several encryption algorithms and then studied the NTRU public key algorithm combined with Spam countering techniques. Analysis of our project showed that it can not only overcome the security flaws existing in other algorithms, but also counter Spam effectively. Though the NTRU algorithm is still in development and requires further research to perfect it, it is a good option for securing future wireless communications because of its greater security, speed and lower computational complexity.

## II. RING BASED PUBLIC KEY CRYPTOSYSTEM

The Ring based public key Cryptosystem features reasonable short, easily created keys, high speed and low memory requirements. The encryption procedure uses a mixing system based on polynomial algebra and reduction modulo two numbers p and q, while the decryption procedures uses an unmixing system whose validity depends on elementary probability theory. The security of the NTRU public key cryptosystem comes from the interaction of the polynomial mixing system with the independence of reduction modulo p and q. Security also relies on the fact that for most lattices, it is very difficult to find extremely short vectors. Public key cryptography is more appropriate than symmetric key cryptosystems for security purposes. Public key methods are very powerful and give us much more flexibility. However this flexibility comes at a computational cost. The amount of computation needed in public key systems is several orders of magnitude more than the amount of computation needed in symmetric key systems such as DES. For this reason, public key cryptography is used in applications where only small amount of data must be processed. There has been significant effort for the creation of computationally inexpensive public key cryptosystems since Diffie and Hellman [3][4].

The signatures produced by the NTRU algorithm are appended to the email as shown in figure 1. This signature is sent with the email to the recipient. On receiving the mail, the signatures are matched with the existing signatures in the database. If a pre-recorded spam is received, it is reported as spam. If a mail not as spam is received, it is sent to the mail box. If a mail is received, whose entry is not

978-0-7695-4437-3/11 $26.00 © 2011 IEEE
DOI 10.1109/CSNT.2011.23

present in the database, then the user is given an option to report it as a spam.

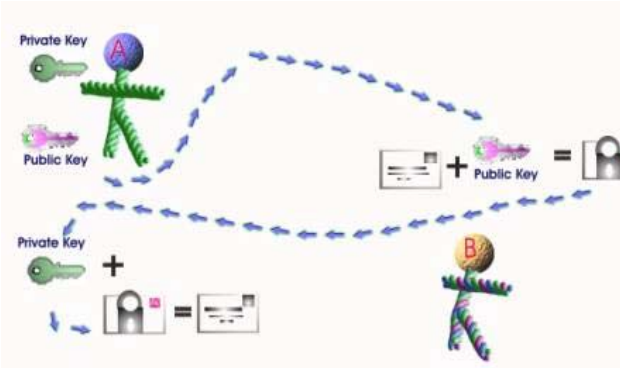


**Figure 1.** Ring Based Public Key Cryptosystem

## III. EXISTING ALGORITHMS

Classical methods of creating digital signatures rely on the fact that it is difficult to deduce the private key, which is used to create a signature, from the public key, which is used to verify it. Some of these are as follows [8][11]:

**a) PKCS:** Public Key Cryptography is based on the creation of mathematical puzzles that are difficult to solve without certain knowledge about how they were created. The creator keeps that knowledge secret (the private key) and publishes the puzzle (the public key). The puzzle can then be used to scramble a message in a way that only the creator can unscramble.

**b) RSA:** The RSA algorithm, used products of two large prime numbers as the puzzle: a user picks two large random primes as her private key and publishes their product as her public key. The difficulty of factoring ensures that no one else can derive the private key (i.e., the two prime factors) from the public one. However, due to recent progress in factoring, RSA public keys must now be thousands of bits long to provide adequate security.

**Drawbacks:**

The RSA encryption scheme can also be used to perform signatures, but only if the enemy is unable to factor large numbers. A forger with a quantum computer, for instance, could successfully create false-signed messages. The same problem arises when RSA is used for encoding secret information, and can be partially circumvented by quantum key distribution.

**c) Quantum Digital Signatures:** In fact, instead of having the public key be a string of classical bits, we let the public key be a number of quantum bits. Given n classical bits, there are only $2^n$ possible values, and by looking at the string, we can tell exactly which one we have. There are many more possible states of n qubits. This means that we can let the public key be one of these many possible quantum states, chosen at random, while the private key says which of the states it is. Only the person who picked the state knows the private key, which enables them to sign messages without fear of having them forged.

**Drawbacks:**

The quantum processing and storage required for this scheme are just beyond the edge of current technology. Also, an enemy who gets too many copies of the public key will be able to figure out its identity, so to prevent this; the sender has to limit the number of copies she distributes. It is possible to make that number very large, however, so this is not too severe a restriction.

**d) Elliptic Curve:** An elliptic curve is a plane curve defined by an equation of the form

$$y^2 = x^3 + a\,x + b$$

The set of points on such a curve (i.e., all solutions of the equation together with a point at infinity) can be shown to form an abelian group (with the point at infinity as identity element). If the coordinates $x$ and $y$ are chosen from a large finite field, the solutions form a finite abelian group. The discrete logarithm problem on such elliptic curve groups is believed to be more difficult than the corresponding problem in (the multiplicative group of nonzero elements of) the underlying finite field. Thus keys in elliptic curve cryptography can be chosen to be much shorter for a comparable level of security.

Finally, all these schemes are very inefficient: a large public key is needed to sign even a single-bit message, and the key can only be used once.

## IV. NTRU ALGORITHM (THE DESIGN)

In this section, we present the algorithm that is designed as having 6 classes:

(i) **KeyGenerator**: responsible for generating public and private keys.
(ii) **Encoder**: responsible for encoding.
(iii) **Decoder**: responsible for decoding.
(iv) **PolynomialOperations**: responsible for polynomial operations such as multiplication and inversion.
(v) **RandPolyGenerator**: responsible for construction of random polynomial.
(vi) **Analyzer**: contains test routines.

## V. STEPS IN THE ALGORITHM (With Explanation)

It depends on 3 integer parameters (N,p,q), where N is a prime integer, p and q are relatively prime integers and q is larger than p [5].

NTRU uses polynomial addition and multiplication in the ring R = Z[x] / ($x^N$ – 1). Any polynomial f in R is written as a vector

$$f = \sum_{i=0}^{N-1} F_i x^i = [F_0, F_1, \ldots, F_{N-1}]$$

The addition used by NTRU is regular polynomial or vector addition. However the multiplication is not regular polynomial multiplication. It is given as a cyclic convolution product and denoted as $\otimes$.

$$F \otimes G = H \quad \text{with} \quad H_k = \sum_{i=0}^{k} F_i G_{k-i} + \sum_{i=k+1}^{N-1} F_i G_{N+k-i}$$

NTRU uses 4 sets $L_f$, $L_g$, $L_r$, $L_m$ of polynomials of degree N-1 with integer coefficients. All these sets contain small polynomials. "Small" polynomial is a polynomial with coefficients close to zero.

### A. Key Creation

To create public and private keys, 2 small polynomials f and g are randomly chosen from the sets $L_f$ and $L_g$ respectively. The polynomial f must have inverses modulo p and modulo q, denoted as $F_p$ and $F_q$.

$F_p \otimes f \equiv 1 \pmod p$ and $F_q \otimes f \equiv 1 \pmod q$

Then the polynomials f and $F_p$ are the private keys. And the polynomial h, given by

$h \equiv p \otimes F_q \otimes g \pmod q$

is the public key.

### B. Encryption

The message m must be a polynomial from the set $L_m$. To encrypt m, a random polynomial r is chosen from the set $L_r$. Then the encrypted message e is the polynomial computed by

$e \equiv r \otimes h + m \pmod q$

where h is the public key.

### C. Decryption

To decrypt e, the polynomial a is computed first

$a \equiv f \otimes e \pmod q$

The coefficients of a must be chosen from the interval ( -q/2, q/2]. Then the original message can be computed by

$m \equiv F_p \otimes a \pmod q$

### D. How Decryption Works

The polynomial a computed during decryption is

$$\begin{aligned} a &\equiv f \otimes e \\ &\equiv f \otimes (r \otimes h + m) \\ &\equiv f \otimes r \otimes h + f \otimes m \\ &\equiv f \otimes r \otimes p \otimes F_q \otimes g + f \otimes m \\ &\equiv r \otimes p \otimes g + f \otimes m \pmod q \end{aligned}$$

For appropriate parameter choices, the coefficients of the polynomial $r \otimes p \otimes g + f \otimes m$ lie between $-q/2$ and $q/2$. So reducing modulo p doesn't affect the coefficients.

$$r \otimes p \otimes g + f \otimes m \equiv r \otimes p \otimes g + f \otimes m \pmod q$$

Then reducing a modulo p gives the polynomial

$r \otimes p \otimes g + f \otimes m \equiv f \otimes m \pmod p$

And multiplying by $F_p^{-1}$ gives m (mod p). If the coefficients of m is in $Z_p$, then we can recover m by decrypting e.

## VI. Optimizations of NTRU Algorithm

Some enhancements of the NTRU algorithm are that we describe involve optimization techniques for speed.

**Enhancement 1: Invertibility of f Modulo p**

f can be chosen to have form

$f = 1 + p \otimes f_1$

where $f_1$ is a random polynomial. Notice that it is not necessary to compute the inverse of f modulo p, because $F_p$ is equal to 1 regardless of $f_1$. So we don't need to compute Fp in key creation and the second multiplication in decryption. But it is important to obtain the desired level of security while using f of this form.

The random polynomial $f_1$ has to be chosen appropriately. However there is not any parameter described for $f_1$ in [2]. Also if we take f in the form $1 + p \otimes f_1$, then the coefficients of polynomial a that lie outside (-q/2 , q/2] will each affect a single coefficient of the decrypted message. It is possible to correct such errors by using error correction techniques. But it is not recommended for high security applications. We realized a problem which affects the efficiency gain of this enhancement. When p is 3, which is the suggested value, then f doesn't have an inverse modulo 2. The solution we used is taking $f = 2 + p \otimes f_1$. But the second multiplication doesn't disappear by this solution. Rather it takes less time than before.

**Enhancement 2: Taking p to be a polynomial**

p can be a polynomial instead of an integer. But the ideals generated by p and q must be relatively prime in the ring. Also p should has small coefficients, because the polynomial a has to be 'narrow'.

The most convenient polynomial is

$$p = X + 2$$

It is natural to use binary polynomials instead of trinary polynomials, when p is chosen in that form. This makes encoding messages much simpler.

**Enhancement 3: Low Hamming Weight Products**

We can fasten encryption and decryption by using small hamming weight products. The most time consuming operations of encryption and decryption are computations of products $r \otimes h$ and $e \otimes f$ respectively. Since r and f have small coefficients, the computations required for these products can be reduced by taking

$r = r_1 \otimes r_2$

$f = 1 + p \otimes (f_1 \otimes f_2 + f_3)$

and using low hamming weight product. Suppose $r_1$ and $r_2$ are binary polynomials with $d_1$ and $d_2$ 1's respectively, then r will be a binary polynomial with nearly $d_1d_2$ 1's. To make $f_1 \otimes f_2 + f_3$ purely binary, the polynomials $f_1$, $f_2$, and $f_3$ have to be chosen very carefully. Taking f of the form $1 + p \otimes (f_1 \otimes f_2 + f_3)$ may increase the decryption failures. The low hamming weight products can be performed using only additions and subtractions and take O(N) time.

## VII. NTRU VARIATIONS FOR SECURITY AND PERFORMANCE IMPROVEMENTS

There is no guarantee that the coefficients of NTRU lie between –q/2 and q/2. For appropriate parameter choices, the coefficients lie between these values almost every time. We mentioned before that NTRU uses 4 sets of polynomials, namely $L_f$, $L_g$, $L_r$, and $L_m$ . $L_m$ is the set of messages and includes all polynomials with coefficients in {-1,0,1}.The parameters of NTRU algorithm are N, p, q, and the other 3 sets. The sets are denoted as L(x,y) and depends on two numbers x and y[6][7]. L(x,y) is the set of polynomials with exactly x many coefficients equal to 1 and y many coefficients equal to –1 and other coefficients equal to 0 [7][9].

Based on the parameters, three different NTRU implementation with 3 different level of security are described here. Table-1 shows these three implementations with the parameter values.

| | N | Q | p | df | dg | dr |
|---|---|---|---|---|---|---|
| Low Security | 107 | 64 | 3 | 15 | 12 | 5 |
| Moderate | 167 | 128 | 3 | 61 | 20 | 18 |
| High Security | 263 | 128 | 3 | 50 | 24 | 16 |
| Highest Security | 503 | 256 | 3 | 216 | 72 | 55 |

**Table 1.** NTRU parameters for various levels of security

Using the latest suggested parameters, the NTRUEncrypt public key cryptosystem is secure to most attacks. By encrypting and decrypting a message according to the NTRUEncrypt the attacker can check whether the function f is the correct secret key or not. There continues however to be a struggle between performance and security. It is hard to improve the security without slowing down the speed and vice versa.

## VIII. COMPARATIVE ANALYSIS FOR HIGH SPEED OF NTRU PKCS

Besides the reason for the high speed of NTRU Public-Key Cryptosystem are its basic features of super fast key generation as depicted in Table 2 below:

- NTRU key generation 30-1000+ times faster than RSA, ECC
- All NTRU keys are fully Independent.
- NTRU sign/verify up to 100 times faster than RSA or ECC
- Key generation is 1-3 orders of magnitude faster than RSA, ECC
- NTRU encrypt/decrypt runs up to 475 times faster than RSA/ECC on servers.
- Encrypts 40-50 times faster than hand-optimized (for speed) RSA, ECC.
- Decrypts 333 times faster than hand-optimized (for speed) RSA.
- Up to 2000 times faster than RSA on DSPs.

| System | Basic Operation | Number of Operations Required | |
|---|---|---|---|
| | | Encrypt | Decrypt |
| NTRU | Convolution Product | 1 | 2 |
| RSA | Modular Multiplication | 17 | ~1000 |
| ECC | Elliptic Curve Addition | ~160 | ~160 |
| NTRU and ECC basic operations take approximately the same time (basic operations for RSA are a little faster). | | | |

**Table 2.** Reasons for the High Speed of NTRU PKCS

Advantages of the NTRU PKCS is not only limited to NTRU's fast key creation that enables new public key paradigms like Message Integrity ( Encrypting audio and video with a different key for every few seconds of content), It also uses independent keys for every transaction. E-mail related Master Key/Disposable Key Protocol feature saves storage and increases security. NTRU is also easy to program, easy to build into hardware and ideal for Digital Signal Processors (DSPs). NTRU requires less memory (RAM) in software, less storage in software and fewer gates in hardware. NTRU easily fits into low power smart cards, handheld devices, Cellular telephones etc [13] [14].

## VIII. CONCLUSION AND DISCUSSIONS

The Internet has evolved far beyond a collection of research and government technology labs and communications centers for which it was founded. During the past years, the increasing use of Internet for commercial transactions and the growing potential of electronic commerce created new demands for security of exchanging private information over a non-secure channel. We should not wait for disaster to strike, but stay proactive, employing a security expert where necessary. This paper implemented the NTRU algorithm including some enhancements suggested by

NTRU Cryptosystems. Although these enhancements decreased the time required for the algorithm, some of them affected the accuracy of the cryptosystem significantly [9][10]. NTRU Cryptosystems claims that it is possible to recover these effects by using error correction techniques, which are not suggested for high secure systems. Thus it is not convenient to make a judgment unless analyzing the security aspects of these enhancements.

The tests showed that the most time consuming part of NTRU algorithm is polynomial multiplication. There are some suggested algorithms for multiplication in some documents of NTRU Cryptosystems. We didn't consider any of them, because these are out of the subject for this paper. But these multiplication algorithms can be considered for future study. The parameters we used are obtained by trial and denial method. The analysis of the effects of different parameters on the accuracy and efficiency of NTRU algorithm will be a good subject for a future study.

## IX. FUTURE SCOPE

There is quite a lot of improvement that can be made to our project to make it a complete cryptography package to give the user complete peace of mind [6][12]. Listed below are a few major improvements that can be made in the future:

- The key size of NTRU encryption can be increased for further security.
- Mail server can be maintained to make it dynamically feasible.
- A better and faster implementation of the algorithm is possible which would efficiently and quickly encrypt and decrypt large files.
- More efficient algorithms can be implemented for polynomial generation.

## REFERENCES


[1] J. Hoffstein, J. Pipper, and J. H. Silverman, "NTRU: A Ring Based Public Key Cryptosystem", in Algorithmic Number Theory: Third International Symposium (ANTS 3) (J. P. Buhler, ed.), vol. LNCS 1423, pp. 267-288, Springer-Verlag, June 21-25 1998.

[2] J. Hoffstein and J. H. Silverman, "Optimizations for NTRU", in Proceedings of Public Key Cryptography and Computational Number Theory, de Gruyter, Warsaw, September 2000.

[3] Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joe Silverman,William Whyte, "NTRUsign Introduced" 2002, Cryptolab, NTRU, MA 01720, USA.

[4] N. Gama and N. Howgrave-Graham and P.Q. Nguyen, "Symplectic Lattice Reduction and NTRU", In: Proc. of EuroCrypr2006, 2006.

[5] NTRU Cryptosystems, The NTRU Encrypt Public Key Cryptosystem (BasicTutorial):http://www.ntru.com/cryptolab/tutorial_pkcs.htm

[6] NTRU Cryptosystems, The NTRU Encrypt Public Key Cryptosystem: EnhancementsI:http://www.ntru.com/cryptolab/tutorial_advanced.htm

[7] NTRU Cryptosystems, The NTRU Encrypt Public Key Cryptosystem: EnhancementsII:http://www.ntru.com/cryptolab/tutorial_hamming.htm

[8] J. Hoffstein and J. H. Silverman, "Random Small Hamming Weight Products With Applications to Cryptography", NTRU Cryptosystems,http://www.ntru.com/cryptolab/articles.htm#003

[9] J. H. Silverman, "Invertibility in Truncated Polynomial Rings", NTRU Cryptosystems (Technical Note #009):http://www.ntru.com/cryptolab/pdf/NTRUTech009.pdf

[10] J. H. Silverman, "Almost Inverses and Fast NTRU Key Creation", NTRU Cryptosystems, (Technical Note#014):http://www.ntru.com/cryptolab/pdf/NTRUTech014.pdf

[11] http: // en.wikipedia.org/ wiki/ NTRU_Cryptosystems,_Inc.

[12] http://en.wikipedia.org/wiki/NTRUEncryt

[13] Asper Scholten and Frederik Vercauteren, "An Introduction to Elliptic and Hyperelliptic Curve Cryptography and the NTRU Cryptosystem", http://www.math.unibonn.de/~saxena/courses/WS2010-ref4.pdf

[14] Arunesh Ramalingam, "Cryptography Terminology & NTRU Public Key Cryptosystem", web.mst.edu/~install/Student_Corner/