Performance and analysis of NTRU and Advanced NTRU cryptosystems

Kala Swamy, Rakesh Nayak,
B.Tech student, Assoc. Professor,
Dept. of Information Technology, Sri Vasavi Engineering College,
Tadepalligudem, A.P,INDIA
nayakrakesh8@gmail.com

Abstract— Authentication and Confidentiality is the main goal of the public key cryptosystems deal when communication takes place between both the parties. The mostly used cryptographic methods are broadly classified as symmetric and asymmetric. The DES comes under the category of symmetric key cryptosystem whereas RSA, NTRU[1] belongs to the category of asymmetric key cryptosystem. NTRU (Nth degree truncated polynomial ring units) is again extended to Advanced NTRU. This paper describes and represents a comparative study on both the cryptosystems by using variable sized text files as input and the results were analyzed in an effective way. This has given an overall idea about the performance of the cryptosystem.

Keywords— Encryption, Decryption, Polynomial, NTRU

I. INTRODUCTION

Communication is the bane of life in the present day world. Therefore it is necessary to find methods of sending information through a secure channel, secure from third party attacks. Several methods have been proposed which include public key-private key algorithms such as RSA. All the public key- private key algorithms depend on the hardness of finding the private key given the public key.

Security is one of the main issues in designing encryption algorithm. Many encryption algorithm are widely available that can be classified into Symmetric (private) and Asymmetric (public) keys encryption.

The Public Key Crypto System (PKCS) provides a methodology for transmitting documents and allows two or more people to communicate while maintaining confidentiality and assuring authentication.

In this paper, we presented the implementations of NTRU and Advanced NTRU systems for different text files and finally compared the computational running times to find the suitable method for the business applications.

II. MATHEMATICAL PRELIMINARIES

A. NTRU

NTRU is the latest in the line of Public Key Cryptographic Systems. It is relatively new and was conceived by Jeffrey Hoffstein, Jill Pipher and Joseph. H. Silverman. NTRU uses polynomial algebra combined with the clustering principle based on elementary mathematical theory. The security of NTRU comes from the interaction of polynomial mixing modulo two relatively prime numbers.

A.1 NTRU Key Generation

User B wants to create a public/private key pair for the NTRU[1] PKCS. B first randomly chooses two small polynomials f and g in the ring of truncated polynomials. A small polynomial is small relative to a random polynomial mod q. In a random polynomial the coefficients are much smaller than q. B must keep the values of the Polynomials f and g private, since anyone who knows the value of either of them will be able to decrypt messages sent to B. B’s next step is to compute the inverse of the f modulo q and the inverse of f modulo p. Thus B computes polynomial fp and fq with the property that f*fp = 1 (modulo q) and f*fq = 1 (modulo p). If by some chance if the inverse does not exist, B will need to go back and choose another f. For information about computing inverses in the ring of truncated polynomials,

User B chooses two polynomials f and g, where polynomial f is invertible polynomial. B keeps the polynomials f and g private and generates a public key has follows:

\[ h = pf_f g \pmod q \]

Here \( f_q \) is \( f' \pmod q \) or \( f\pmod q \). B’s private key is the pair of polynomials f and \( f_q \). B’s public key is the polynomial h.
A.2 NTRU Encryption

The NTRU Crypto-system [1] is based on three parameters p, q and N where p is a small prime number and q and p are relatively-prime and N is the degree of the polynomial in the ring of polynomials.

When User A wants to send a message to User B, A converts the message to the form of binary polynomial m (which is of the same order as f and q). A uses B’s public key and generates the cipher text e as follows:
\[ e = h*r + m \pmod{q}, \]
where r is any small polynomial to obscure the original message m.

A.3 NTRU Decryption

User B has received A’s encrypted message e and B wants to decrypt it. B begins by using the private polynomial f to compute the polynomial
\[ a = \text{f}^e \pmod{q} \]
Then calculate the centering value of a by choosing the coefficients of a to lie between -q/2 and q/2. In general B will choose the coefficients of a to lie in an interval of length q. The specific interval depends on the form of the small polynomials. It is very important that B does this before performing the next step. B next computes the polynomial
\[ b = a \pmod{p} = f^*m \]
That is, B reduces each of the coefficients of a modulo p. Finally B uses the other private polynomial fp to compute
\[ c = \text{f}^p*B \pmod{p} = \text{fp}^*f^*m \pmod{p} = m \]
The polynomial c will be A’s original message m.

B. Advanced NTRU

The Advanced NTRU (A-NTRU) also has three parameters p, q and N, where p=2+x, q is a number which is in the form 2^n and N is the degree of the polynomial in the ring of polynomials.

In Advanced NTRU, take \( f = 1+pF \) where F is a small binary polynomial. This choice means that f is equal to 1 mod p, which has the following advantages:

1. f is always invertible mod p (\( f^i = 1 \pmod{p} \)). This speeds up key generation, because we don’t have to explicitly calculate the inverse mod p.

2. Because \( f^i = 1 \pmod{p} \), we no longer have to carry out the second polynomial multiplication when decrypting. This speeds up decryption considerably. It also means that we don’t have to store \( fp = f^i = f \pmod{p} \) as part of the private key.

B.1 Advanced NTRU Key Generation

Take \( p=2+x \), a small binary polynomial F and finds \( f = 1+pF \) and chooses a polynomial g to generate a public key has follows:

\[ H = \text{pfq}^*g \pmod{q} \]
Here \( fq = f^i \pmod{q} \) or \( f^i \text{pq}^*g \pmod{q} = 1 \).
The private key is the pair of polynomials \( f \) and fp and the public key is the polynomial h.

B.2 Advanced NTRU Encryption

The Encryption process is same as the encryption process described in the earlier section.

B.3 Advanced NTRU Decryption

On receiving encrypted message e use the private key f to compute the polynomial \( a = f^e \pmod{q} \).
Then calculate the centering value of a by choosing the coefficients of a to lie between -q/2 and q/2.
As \( f \) is chosen so that it is inverse mod p equals to 1, now reduce \( a \pmod{p} \), where \( p=2+x \).
Now find a binary polynomial d which has the property that \( d(-2) = \text{a}(-2) \pmod{2^d + 1} \). Finally, convert the polynomial d into the binary message.

III. EXPERIMENTAL SET-UP DESIGN

For our experiment a Laptop with Dual-Core Processor having speed 2.30 GHz CPU, 2GB RAM and Windows 7 Ultimate (32-Bit) is used in which the performance data are collected. In this experiment, we encrypt the text file size that ranges from 1 Kb to 10 Kb. Their implementation is thoroughly tested and is optimized to give the maximum performance for the algorithm. Several performance metrics are collected by using NTRU and Advanced NTRU technique are:

1. Key Generation time
2. Encryption time
3. Decryption Time
4. Throughput during Encryption
5. Throughput during Decryption
6. Power consumption during Encryption (joule/Kb)
7. Power consumption during Decryption (joule/Kb)

The encryption/decryption time is considered the time that an encryption/decryption algorithm takes to produce a cipher text from a plaintext (or from cipher text to plain text). Encryption/Decryption time is used to calculate the throughput of an encryption/decryption scheme. It indicates the speed of encryption/decryption. The throughput of the encryption/decryption scheme is calculated as the total plaintext in Kilobytes encrypted divided by the encryption/decryption time [7]- [9].

www.ijrccct.org
IV. EXPERIMENTAL RESULTS

A. Key Generation

The keys of size 167 bits, 251 bits, 347 bits and 503 bits are generated. The following figure shows the time CPU time for generating key with NTRU and A-NTRU.

![Comparison during Key Generation](image1.png)

**Fig 1.** Time consumed during Key generation for various key sizes and various text files with NTRU and A-NTRU.

B. Encryption of different text size

Encryption time is used to calculate the throughput of encryption using a scheme. The input to encryption is various size text file, which is encrypted using various key sizes. The various key sizes are used. The key size used are 167 bits, 251 bits, 347 bits and 503 bits. The following figures shows the CPU time (Millisecond), throughput (Kilobytes/second).

![NTRU Encryption](image2.png)

**Fig 2.** Time consumed during encryption for various key sizes and various text files with NTRU.

![Throughput NTRU Encryption](image3.png)

**Fig 3.** Throughput of each NTRU encryption algorithm for different key size to different text data (Kb/MilliSeconds)

![A-NTRU Encryption](image4.png)

**Fig 4.** Time consumed during encryption for various key sizes and various text files with Advanced NTRU(A-NTRU).

![Throughput A-NTRU Encryption](image5.png)

**Fig 5.** Throughput of each NTRU encryption algorithm for different key size to different text data (Kb/MilliSeconds)

C. Decryption of different text size

Decryption time is used to calculate the throughput of decryption using different scheme. The input to decryption is various size cipher texts generated during decryption process. It is decrypted using various size...
D. Power Consumption during Encryption

We present a basic cost of encryption by the product of the total number of clock cycles taken by the encryption and the average current drawn by each CPU clock cycle. The basic encryption cost is in unit of ampere-cycle. To calculate the total power consumed we use the formula $(\text{Average Encryption Time per second} \times \text{CPU clock cycle per second}) / \text{processor operating voltage}$.

Then, we multiply the ampere-seconds with the processor’s operating voltage, and we obtain the energy cost in Joule.
Fig 11. Power Consumption for A-NTRU encryption algorithm for different key size to different text data (Ampere-Seconds)

E. Power Consumption during Decryption

The same method as described in the previous subsection is adopted for finding power consumption during decryption.

Fig 12. Power Consumption for NTRU Decryption algorithm for different key size to different text data (Ampere-Seconds)

Fig 13. Power Consumption for A-NTRU Decryption algorithm for different key size to different text data (Ampere-Seconds)

V. CONCLUSIONS

This paper presents the comparative study of both the cryptosystems for variable sized text files as input. The results were observed, analyzed and compared so as to identify which method is appropriate to the business needs. It is clear from the graphs plotted for NTRU and A-NTRU during key generation, encryption, decryption, that A-NTRU has performed better than NTRU. So we can conclude that A-NTRU can be used in real time so as to reduce the CPU time and energy consumption.

This paper can further be enhanced by comparing the security level of each technique.

ACKNOWLEDGMENT

We thank the principal and the Management of Sri Vasavi Engineering College, Tadepalligudem to provide the necessary support and encouragement to do this work.

REFERENCES


