NTRU and RSA Cryptosystems for Data Security in IoT Environment

Ankitha Nandanavanam
Department of Computer Science and Engineering.
NMAM Institute of Technology, Nitte
Karnataka, INDIA
ankitha.5dec@gmail.com

Ipsita Upasana
Department of Information and Communication Technology
Manipal Institute of Technology,
Manipal Academy of Higher Education
Manipal, INDIA
ipsita.upasana@manipal.edu

Nanditha Nandanavanam
School of Engineering and Applied Sciences
SUNY, University of Buffalo
Buffalo, USA
nnandana@buffalo.edu

***Abstract*— The massive technological revolution that has taken place and that is going on at a massive rate, has led to zillions of digital information traveling in a communication network. Information traveling in the network needs security and privacy. There are various cryptographic algorithms available today that can secure messages and data being communicated. Symmetric cryptography, though simple, does not provide authorization efficiently. Asymmetric cryptography proves to be more secure as it provides all the components of information security.**

With the breakthrough development on the Internet-of-Things (IOT), any digital device can become an originator of data. The main concern which arises is how to protect these data. With an accelerated usage of IoT environment, the security of data collected and stored by these devices has become more crucial.

In this paper, we discuss two asymmetric algorithms NTRU (Nth degree truncated polynomial ring) and RSA (Rivest-Shamir-Adleman) in context of IoT. RSA is widely used but is heavy on key size and computation complexity. On the other hand, NTRU can give the same level of bit security with comparative key size and very less computational complexity. The captivity of NTRU algorithm lies in the fact that it works on simple polynomial operations. It is a NP-hard problem based on shortest vector problem (SVP) and is secure even for future quantum attacks. In a resource constrained environment like IoT environment, the performance of RSA and NTRU has been analyzed on the various security level parameters, viz. key generation, encryption and decryption time. Furthermore, performance of the various parameters has also been analyzed on various bits size plain text.

Keywords— Cryptography, Data Security, IoT, NTRU, Performance Analysis, RSA

I. INTRODUCTION

Today data security and privacy is an area which concerns every single user using any device connected to the internet. Information of single person or information of large firm travel in a communication channel which needs to be secured.

Cryptography is an art of giving a different face to any information. It helps to maintain data privacy and trust among users. Cryptography schemes have its own history stretching back to the pre-digital era, but in the digital age, various algorithms have been developed to secure data. These algorithms are broadly divided into two types: symmetric and asymmetric. Symmetric, as the name suggests, has symmetry. This symmetry is of keys being used at sender as well as receiver end. A single key is used and is shared by appropriate key exchange mechanisms. Asymmetric algorithms use two different keys for encryption and decryption. The sender encrypts data with receiver's public key. This encrypted information can be decrypted only by the receiver using his private key. Asymmetric cryptography or Public Key Cryptosystem (PKI) comes with the digital signature that helps in authentication and non-repudiation of messages besides giving confidentiality and integrity to the messages.

It is very hard to compare two different schemes as each one of them has its own pros and cons. In a situation where bulk messages are to be communicated between small parties, symmetric cryptography will help. However, in IoT environment where small messages are shared between billions of heterogeneous devices, asymmetric encryption would help. Unlike symmetric, which must store the shared key for each user, PKI must store only one public key and one private key. This may help in solving scalability problem in IoT [1]. A requirement of one key pair even on the large network in PKI makes the total number of keys required much smaller than that used in symmetric encryption. IoT devices, when deployed, has negligible chances of an update. In such case also PKI is helpful, as a symmetric key may be comprised of a long interval session and it must be regenerated and securely exchanged [2]. This is a burden on a device communicating in IoT environment.

There are various available schemes to give data security. The various algorithm provides confidentiality, integrity, and authentication using encryption, hashing, and signatures respectively. In this paper, we are discussing a very famous algorithm Rivest- Shamir-Adleman (RSA) and the not so popular Nth degree Truncated polynomial Ring (NTRU). We show through experiments how RSA could be easily replaced with NTRU that can not only give present but also post-quantum security. NTRU based on an NP-hard problem is hard to be broken when quantum computers will become reality.

II. LITERATURE REVIEW

With the rapid advent of the IOT environment, the ease with which the data generated and stored in the endpoints could be accessed and possibly exploited has increased. In [3], the major vulnerabilities associated with IoT were studied. Communication protocols that do not employ

978-1-7281-7213-2/20/$31.00 ©2020 IEEE

security mechanisms is one of the primary reasons. Absence of proper device authentication and weak security policy stood out as the other reasons.

Traditional cryptosystems will not prove to be effective for the distributed nature of IoT endpoints. The taxonomy of the various IoT security protocols has been studied and evaluated in [4]. Key management is one of the security issues that has been evaluated based on several parameters like storage, computation and communication.

The sensitive nature of the data existing in the IoT environment necessitate an even more better security mechanism. However, the resource-constrained nature of the IoT environment poses as a major roadblock to the implementation of better security mechanisms. Constraints that result from the vulnerabilities of hardware, software and the network connectivity of the devices were studied in [5]. The analysis also presented some scenarios where the absence of security mechanisms could prove detrimental. The constraints on energy, computation power, and memory necessitates the deployment of a cryptosystem that provides a robust protection against the various threats while keeping the restrictions in consideration.

Various surveys, analysis, and experiments have been performed to find out the best algorithm that can encrypt and secure data transmitted between IoT devices. In [6], a survey on the various application of IoT in healthcare sector was performed. An analysis of security challenges was done which included performance analysis of two symmetric algorithm Advanced Encryption Standard (AES) and Data Encryption Standard (DES); and an asymmetric algorithm, RSA. According to the survey, RSA proves to provide optimal security against various attacks and is more secure than AES and DES. They proposed modified RSA which included anonymity in communication and storage that can prevent data from malicious users.

While looking for a suitable approach to secure the endpoints in an IoT environment, we need not only to answer the threats posed by the current scenario but also what may become of the future. With the existence of quantum computers, the cryptosystems based on integer factorization or discrete logarithm problem will be deemed as inadequate. Chaudhary et al. proposed a lattice-based public key cryptosystem as a post-quantum solution. [7] The authors have also evaluated the performances of some lattice-based cryptosystems based on message size and key sizes.

In [8], an ARM architecture was used as IoT endpoint to analyze the performance of various NTRU parameters in terms of runtime and memory utilization. It gives a detailed explanation of various operations performed in NTRU and some variation of NTRU for optimal performance. The authors suggest NTRU on top of various public key cryptosystem that can even prevent future quantum attacks and is viable for resource constrained IoT devices.

Communication of keys across IoT endpoints using NTRU has been shown to be more secure in [9]. The authors showcased an enhanced Datagram transport layer security with post quantum capabilities using NTRU.

III. CRYPTOGRAPHY SCHEMES

As in this paper, we are analyzing RSA and NTRU, below is the detailed explanation of operations performed in both the schemes.

A. RSA

RSA is an asymmetric cryptographic algorithm, introduced by Ron Rivest, Adi Shamir, and Leonard Adleman in 1977, as one of the first public key cryptosystems based on the factorization of large integers. The large size of keys contributes to its strength as well as weakness. The large size of keys makes it more secure but at the same time very slow on computation. RSA comprises key generation, encryption, and decryption for communication between sender A and receiver B explained below. The parameters used are modulus n, encryption exponent e, and decryption exponent d.

1) Key Generation

Original Algorithm Comprises of following steps:

i. Generate two large prime numbers, p and q and compute $n = p*q$ and $\phi = (p-1)(q-1)$.
ii. Choose an integer e ($1 < e < \phi$), such that $gcd(e, \phi) = 1$.
iii. Calculate secret exponent, d ($1 < d < \phi$), such that $e\,d \equiv 1 \pmod{\phi}$
iv. The public key is (n, e) and the private key is (d, p, and q).

2) Encryption

Encryption is done by sender A as:

i. Get receiver B's public key (n, e).
ii. Compute cipher text *c* from plain text m using (1).

$$c = m^e \bmod n \quad (1)$$

iii. Send c to B.

3) Decryption

B uses private key (n, d) to get plain text m using (2).

$$m = c^d \bmod n \quad (2)$$

When we talk about key length in RSA, it's the size of n. Today, RSA key size of 1024 and more is secure. But in near future when quantum computers will become a reality, even best algorithm today will be broken. Many algorithms have been developed to secure data from quantum attacks. Some of them are NTRU, McEliece, Ring-LWE, etc. Though these are not widely used, but NTRU is more famous due to its security, speed, and low complexity. According to the data available on official website of NTRU [10], NTRU is 370 times faster in key generation, 60% faster in encryption, and 92 times faster in decryption when compared with RSA. Due to its small size, large scale deployments, minimal battery and CPU usage, and security against quantum attacks, NTRU can be said to be of one the strong contender to provide secure communication in IoT environment. NTRU has been explained in detail below.

B. NTRU

This section explains NTRU in details based on EESS Version 3.1 [11]. We discuss basic definitions, NTRU parameters and algorithm used in key generation, encryption, and decryption.

$$R_{N,q} = \frac{Z_q[X]}{X^{N-1}} \quad (3)$$

1) Basic Definitions

NTRU is ring based public key cryptosystem defined in lattice. All operations are based on truncated polynomial convolution ring, R, defined in (3). All polynomial in the ring has integer coefficient and maximum degree of N-1.

$Z_q[X]$ is a polynomial with integer coefficient reduced by modulo q. The polynomial has degree of atmost N-1. If a and b are two polynomials in ring R, they can be defined as in (4) and (5):

$$a = a_0 + a_1 x + a_2 x^2 + \cdots + a_{N-2} x^{N-2} + a_{N-1} x^{N-1} = \sum_{i=0}^{N-1} a^i x^i \quad (4)$$

Vector coefficients are represented as:

$$\text{a} = (a_0, a_1, a_2, \ldots\ldots\ldots\ldots, a_{N-2}, a_{N-1})$$

$$b = b_0 + b_1 x + b_2 x^2 + \cdots + b_{N-2} x^{N-2} + b_{N-1} x^{N-1} = \sum_{i=0}^{N-1} b^i x^i \quad (5)$$

Vector coefficients are represented as:

$$\text{b} = (b_0, b_1, b_2, \ldots\ldots\ldots\ldots, b_{N-2}, b_{N-1})$$

The basic operations used in convolution ring polynomial are addition, subtraction and convolution multiplication as defined in (6), (7) and (8). The addition is same as done in polynomial but the difference between ordinary and convolution polynomial is that degree extending N-1 is replaced by mod N. Like, x^N is by 1, x^{N+1} is replaced by x, x^{N+2} is replaced by x^2 and so on.

$$a + b = \left(\sum_{i=0}^{N-1} a_i x^i\right) + \left(\sum_{i=0}^{N-1} b_i x^i\right) = \sum_{i=0}^{N-1} (a_i + b_i) x^i \quad (6)$$

$$a - b = \left(\sum_{i=0}^{N-1} a_i x^i\right) - \left(\sum_{i=0}^{N-1} b_i x^i\right) = \sum_{i=0}^{N-1} (a_i - b_i) x^i \quad (7)$$

$$a * b = \left(\sum_{i=0}^{N-1} a_i x^i\right) * \left(\sum_{i=0}^{N-1} b_i x^i\right) = \sum_{k=0}^{N-1} \left(\sum_{i+j=k \bmod N} a_i b_j\right) x^k \quad (8)$$

2) NTRU Parameter

NTRU polynomial can be of binary, ternary or product form. Binary polynomial has coefficients in the range [0, 1]. Ternary polynomials have coefficient in the range [-1, 0, 1], while product form polynomials consist of three polynomial *f1, f2, f3* satisfying $f = f1 + f2 * f3$. Binary polynomials are said to be unbalanced and has chances to leak the message, so it is avoided for better security [12].

The various parameters used in entire cryptographic system are given in Table I.

TABLE I.
VARIOUS PARAMETERS OF NTRU ALGORITHM

Parameter	Description
N	Degree of polynomial at most N-1
p	Small modulus
q	Big modulus
f	Private Key
h	Public Key
m	Plain text
F	Polynomial used to calculate private key f
F_p	Multiplicative inverse of $f \bmod p$
F_q	Multiplicative inverse of $f \bmod q$
g	Temporary polynomial used in key generation
r	Blinding polynomial for hiding message
d_f	Number of ones in the polynomials that comprise the private key value f in simple polynomial (and as $df1, df2, and\ df3$ in case of product form polynomial)
d_g	Number of ones in the polynomials that comprise the temporary polynomial g
d_r	Number of ones in the blinding polynomial r. (and as dr1, dr2, and dr3 in product form polynomial.)

NTRU uses public parameter $N, p, q, d_f, d_g, d_r, d_m, h$ and private parameters $f\ and\ g$. Parameters p and q are used to modulo reduce the coefficient of polynomial. p is very small as compared to q and gcd(p, q) should be 1. N is taken to be a prime number, p is taken 3 and q is taken power of 2.

The multiplicative inverse of a polynomial a, satisfying $a * a^{-1} \equiv 1\ mod\ q$ can be computed using Almost Inverse Algorithm in the ring polynomial explained in [13] [14].

3) Key Generation

NTRU is a public key cryptosystem, thus necessitating the generation of a private key and a public key. NTRU has keys in polynomial form: private key, f and public key, h. Two random polynomial F and g are generated first with the coefficients in the range {-1, 0, 1} using Pseudo Random Generators. For F, there should be d_f number of ones, d_f number of negative ones and rest coefficients as zero. Similarly, for g, there should be d_g ones and negative ones and rest coefficients as zero. Private Key is computed using (9). Multiplicative inverse of F mod p, (Fp), multiplicative inverse F mod q, (Fq) and inverse of g mod q is calculated as in (10) and (11) using Almost Inverse Algorithm [13] [14]. If inverse do not exist, a polynomial is regenerated until a polynomial having inverse is found. Public Key polynomial is computed using (12).

$$f = 1 + p\,F \quad (9)$$

$$Fp = f^{-1} \, mod \, \mathrm{p} \tag{10}$$

$$Fq = f^{-1} \, mod \, q \tag{11}$$

$$h \equiv p \, Fq * g \, (mod \, q) \tag{12}$$

4) Encryption

For encrypting plain message m to encrypted message e, first message m is padded with random bits, b, length of message, len, and remaining bits are filled with zeros (p0), to get padded message, M (b || len || m || p0). Three bits of M is then taken and converted to ternary polynomial using look up table given in Table II. A random polynomial, r (OID || m || b || hTrunc) is generated using Blinding Polynomial Generation (BPGM) method [11] with input as message m, unique parameter ID, OID, random bits, b and truncated public key, hTrunc. A mask is generated using Mask Generation Function (MGF) [11]. This mask is XORed with M to get m'. The encrypted text polynomial e can be computed using (13).

$$e = r * h + m' \, mod \, q \tag{13}$$

5) Decryption

Receiver decrypt the encrypted message e to get plaintext m using private key f. First message m' is retrieved using (14) and (15). Coefficients of a is constrained in the range $[-q/2, q/2]$. $cR = r * h$ is retrieved using (16). Finally, plain text is retrieved using (17).

TABLE II
CONVERSION FROM BINARY TO TERNARY

Input	Output
{0, 0, 0}	{0, 0}
{0, 0, 1}	{0, 1}
{0, 1, 0}	{0, –1}
{0, 1, 1}	{1, 0}
{1, 0, 0}	{1, 1}
{1, 0, 1}	{1, –1}
{1, 1, 0}	{–1, 0}
{1, 1, 1}	{–1, 1}

TABLE III
CONVERSION FROM TERNARY TO BINARY

Input	Output
{0, 0}	{0, 0,0}
{0, 1}	{0, 0, 1}
{0, -1}	{0, 1, 0}
{1, 0}	{0, 1, 1}
{1, 1}	{1, 0, 0}
{1, –1}	{1, 0, 1}
{–1, 0}	{1, 1, 0}
{–1, 1}	{1, 1, 1}

$$a = e * f \, mod \, q \tag{14}$$

$$m' = a \, mod \, p \tag{15}$$

$$cR = r * h = e - m' \, mod \, q \tag{16}$$

$$cM = m' - MGF(cR) \, mod \, p \tag{17}$$

Plain text in the form of ternary polynomial is converted to binary using Table III. Original message cm is retrieved by parsing through padded message, cM to the given length. Plain text, cm, is used to generate a blinding polynomial cR' using BPGM (OID || m || b || hTrunc), which should satisfy (18).

$$cR = cR' * h \, mod \, q \tag{18}$$

IV. PERFORMANCE EVALUATION

For analyzing the performance of NTRU and RSA, we implemented it on 2 GHz Intel(R) Core(TM) i3-5005U Processor operating under windows 8.1 64-bit operating system using bouncy castle java library. We analyzed NTRU key generation, encryption and decryption speed for various bit length for various standard encryption parameter available in [15]-[18] at various security level for a 20-byte plain message. Time in milliseconds and operations per second for various operations are noted in Table IV. We calculated the time required at various hashing algorithm; SHA-256, SHA-512, SHA3-256, and SHA3-512. There was no significant difference in time while using different hash algorithms. Hashing algorithms can be used as per user's requirement.

We could see from the result that key generation time is large compared to encryption and decryption time. As the security level increases the key generation time increases. We could also see a visible difference between the time that is taken by simple ternary polynomial and time that is taken by product form polynomial. Product form ternary polynomial gives better performance as compared to a simple ternary polynomial. At the same value of N, the operations performed per second for product form polynomial is more compared to the simple polynomial. This leads to less execution time and more efficiency. We did not get any standard parameter for 160-bit security level NTRU, hence it is not mentioned in the result.

We compared the result with RSA asymmetric cryptography for a plain text of 256 bit. RSA is not suitable for IoT environment because of its large key size and it takes more time and operations to give the security level that can be provided by other algorithms. A comparison of the algorithm's key size at various security level is given in Table V. Comparison of time taken in millisecond for key generation, encryption, and decryption, as well as operation per second, is analyzed in detail in Table VI. It is clearly visible that key generation time is very large for RSA. When it comes to encryption RSA and NTRU is comparable till 160-bit security level, but NTRU outperforms RSA beyond 160-bit security level. Same is the case in decryption, NTRU is faster than RSA. Due to less complex operations, the time taken to perform various operations per second in NTRU is very high.

TABLE V.
KEY SIZE COMPARISON

Security Level (bits)	N	NTRU Key Sizes (bits)	RSA Key Sizes (bits)
80	251	2761	1024
112	401	4411	2048
128	443	4873	3072
160	491	5401	4096
192	593	6523	7680
256	743	8173	15360

For comparing RSA and NTRU, we took different bit size plain text and compared both on 128-bit security level with SHA-256 hashing. For 128-bit security level RSA-3072, and for NTRU simple and product form (NTRU_FAST) N=439 is considered. Key generation, encryption, and decryption time are shown in Fig. 1, Fig. 2, and Fig. 3 respectively.

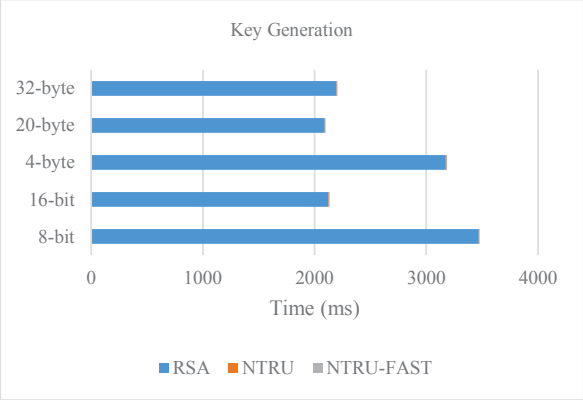

Fig. 1. Key Generation Time

It can be observed from Fig. 1 that key generation time for RSA is very high as compared to NTRU. Fig. 2 and Fig. 3 shows that encryption and decryption of text takes very less time with NTRU compared with RSA.

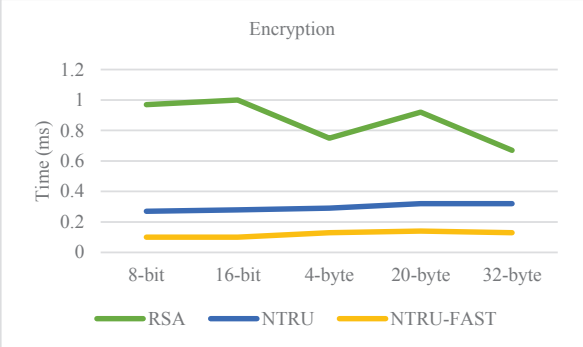

Fig. 2. Encryption Time

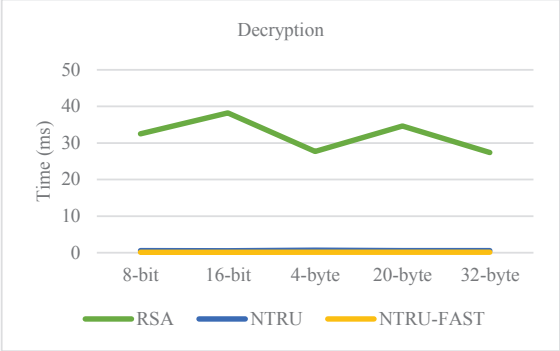

Fig. 3. Decryption Time

V. CONCLUSION

Although today everyone is more familiar with RSA and its operation. When it comes to post-quantum security and less complexity, NTRU can replace RSA. Simple polynomial operations in ring makes it much simpler and faster. Our results clearly show how NTRU can be used instead of RSA especially for IoT device which are constrained over size, memory, and power. Various parameter available for NTRU can be scaled for various size devices. For small devices, N could be small and for large devices, N could be taken large. This would solve the problem of scalability in IoT environment.

REFERENCES

[1] "Symmetric and Asymmetric Encryption", Infosec Resources, 2019. [Online]. Available: https://resources.infosecinstitute.com/symmetric-asymmetric-encryption/. [Accessed: 19- Dec- 2019].

[2] P. Thorsteinson and G. Ganesh, "Asymmetric Cryptography | Problems with Symmetric Algorithms | InformIT", Informit.com, 2019. [Online]. Available: https://www.informit.com/articles/article.aspx?p=102212. [Accessed: 19- Dec- 2019].

[3] Z. Ling, J. Luo, Y. Xu, C. Gao, K. Wu and X. Fu, "Security Vulnerabilities of Internet of Things: A Case Study of the Smart Plug System," in IEEE Internet of Things Journal, vol. 4, no. 6, pp. 1899-1909, Dec. 2017.

[4] A.K. Das,S. Zeadally and D. He, "Taxonomy and Analysis of Security Protocols for Internet of Things," in Future Generation Computer Systems, vol. 89, pp. 110-125, 2018.

[5] M. M. Hossain, M. Fotouhi and R. Hasan, "Towards an Analysis of Security Issues, Challenges, and Open Problems in the Internet of Things," 2015 IEEE World Congress on Services, New York, NY, 2015, pp. 21-28.

[6] A. B. Pawar and Shashikant Ghumbre, "A survey on IoT applications, security challenges and counter measures", International Conference on Computing, Analytics and Security Trends (CAST), pp. 294 - 299, 2016.

[7] R. Chaudhary, G. S. Aujla, N. Kumar and S. Zeadally, "Lattice-Based Public Key Cryptosystem for Internet of Things Environment: Challenges and Solutions," in IEEE Internet of Things Journal, vol. 6, no. 3, pp. 4897-4909, June 2019.

[8] O. M. Guillen, T. Pöppelmann, J. M. Bermudo Mera, E. F. Bongenaar, G. Sigl and J. Sepulveda, "Towards post-quantum security for IoT endpoints with NTRU," Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017, Lausanne, 2017, pp. 698-703.

[9] J. Sepúlveda, S. Liu and J. M. Bermudo Mera, "Post-Quantum Enabled Cyber Physical Systems," in IEEE Embedded Systems Letters, vol. 11, no. 4, pp. 106-110, Dec. 2019.

[10] "NTRU Post Quantum Cryptography | OnBoard Security", Onboardsecurity.com, 2019. [Online]. Available: https://www.onboardsecurity.com/products/ntru-crypto. [Accessed: 23- Dec- 2019].W. Whyte, "EESS 1: Implementation Aspects of NTRUEncrypt, Version 3.1," Consortium for Efficient Embedded Security, Tech., September 2015.

[11] J. Hermans, F. Vercauteren, and B. Preneel, “Speed records for NTRU”,.Springer, pp. 73–88, 2010.

[12] C. M. O’Rourke, “Efficient NTRU implementations”, Master’s thesis, Worcester Polytechnic Institute, 2002

[13] J. H. Silverman, “Almost Inverses and Fast NTRU Key Generation”, 2019 [Online] Available : https://assets.onboardsecurity.com/static/downloads/NTRU/resources/NTRUTech014.pdf

[14] J. Hoffstein, J. Pipher, J. M. Schanck, J.H. Silverman, W. Whyte, and Z. Zhang, "Choosing parameters for NTRUEncrypt", Cryptology ePrint Archive, Report 2015/708 (2015). [Online] Available :http://eprint.iacr.org/2015/708

[15] P. Hirschhorn, J. Hoffstein, N. Howgrave-graham, and W.Whyte, "Choosing NTRUEncrypt parameters in light of combined lattice reduction and MITM approaches" in M. Abdalla, D. Pointcheval, P. A. Fouque, and D. Vergnaud (eds.) ACNS 2009. LNCS, vol. 5536, pp. 437–455. Springer, Heidelberg (2009)

[16] M. U. Sharif and K. Gaj, "High-Speed Hardware for NTRUEncrypt-SVES: Lessons Learned", 2017. [Online] Available : https://2017.pqcrypto.org/conference/slides/recent-results/Kris_Gaj_PQCrypto.pdf

[17] N. Howgrave-Graham, J.H. Silverman, and W. Whyte, "Choosing Parameter Sets for NTRUEncrypt with NAEP and SVES-3", in A.J. Menezes, (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 118–135. Springer, Heidelberg, 2005.

[18] Open Source NTRU Public Key Cryptography and Reference Code. [Online] Available: https://github.com/tbuktu/ntru.

TABLE IV
PERFORMANCE ANALYSIS OF NTRU ENCRYPTION PARAMETERS

N	Security Level (bits)	Key Size (bits)	Key Generation		Encryption		Decryption		Polynomial Form
			Time (ms)	Operation/sec	Time (ms)	Operation/sec	Time (ms)	Operation/sec	
251	80	2761	3.68	164.89	0.20	5110.24	0.28	3527.31	Simple
251	80	2761	1.81	552.96	0.09	10930.8	0.10	10314	Product
401	112	4411	5.64	177.15	0.27	3772.63	0.49	2049.02	Simple
401	112	4411	2.68	372.92	0.08	12855.9	0.10	9565.89	Product
439	128	4829	5.96	167.8	0.32	3170.56	0.58	1718.09	Simple
439	128	4829	3.26	306.62	0.14	7394.66	0.13	7640.99	Product
743	256	8173	14.46	69.16	0.34	2902.6	0.96	1040.91	Simple
743	256	8173	8.36	119.55	0.16	6067.31	0.20	5071.46	Product
1087	256	11957	29.55	33.85	0.45	2227	1.47	680.75	Simple
1087	256	11957	13.26	75.44	0.21	4847.08	0.25	4006.85	Product
1171	256	12881	32.59	30.68	0.43	2352.76	1.51	660.71	Simple
1171	256	12881	15.99	62.53	0.20	5065.26	0.25	4030.57	Product
1499	256	16489	51.05	19.59	0.43	2301.76	1.94	516.78	Simple
1499	256	16489	26.04	38.4	0.28	3602.47	0.37	2685.3	Product

TABLE VI
PERFORMANCE ANALYSIS OF NTRU AND RSA

Security Level (bits)	NTRU							RSA						
	Key Size (bits)	Key Generation		Encryption		Decryption		Key Size (bits)	Key Generation		Encryption		Decryption	
		Time (ms)	Operation/sec	Time (ms)	Operation/sec	Time (ms)	Operation/sec		Time (ms)	Operation/sec	Time (ms)	Operation/sec	Time (ms)	Operation/sec
112	4411	3.18	314.41	0.09	10859.12	0.1	9831.96	2048	629.96	1.59	0.38	2639.2	11.48	87.11
128	4873	3.52	284.03	0.08	12878.57	0.11	9058.95	3072	2195.55	0.46	0.67	1491.86	27.39	36.51
160	-	-	-	-	-	-	-	4096	11071.32	0.09	0.96	1043.19	61.47	16.27
192	6523	7.18	139.19	0.12	8121.83	0.14	7263.55	7680	146691.5	0.01	3.22	310.33	400.5	2.5
256	8173	8.55	116.97	0.16	6207.19	0.26	3882.40	15360	1129128	0	11.45	87.3	2827.29	0.35