ISSN 2278-3091
Volume 9, No.5, September - October 2020
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse109952020.pdf
https://doi.org/10.30534/ijatcse/2020/109952020

NTRUEncrypt – A Quantum Proof Replacement to RSA Cryptosystem

Kunal Meher[1], DivyaMidhunchakkaravarthy[2]
[1] Research Scholar at Lincoln University College, Malaysia and Assistant Professor at Xavier Institute of Engineering, India. kunalmeher@gmail.com
[2] Associate Professor in Lincoln University College, Malaysia. divya@lincoln.edu.my

ABSTRACT

The purpose of encryption is to provide a secure environment for communication and to keep information safe from unauthorized. There are two types of cryptosystem symmetric cryptosystem and asymmetric cryptosystem. Asymmetric cryptosystem is also called as Public Key Encryption (PKE). In this paper, we analyze two asymmetric cryptosystem – RSA and NTRU. NTRU have reasonably short and easily created keys, high speed and low memory requirements. As, NTRU is relatively new and cryptosystem of future, we talk more about it in the paper.

Key words :PQC, RSA, NTRUEncrypt, Security Levels, Key Size, PKCS.

1. INTRODUCTION

Post-quantum cryptography (PQC) deals with cryptographic primitives those are secure against an attack by a quantum computer. Widely used asymmetric cryptographic algorithms such as RSA, ECC and Diffie-Hellman are not secure against quantum computer. The quantum computer is no more theoretical but is actually in practice. Post-quantum cryptography (PQC) is used for quantum-resistant. Lattice-based cryptography is one of the types of PQC. Learning with Errors (LWE), Ring Learning with Errors (Ring-LWE), and Module Learning with Errors (Module-LWE) are the mathematical hard problems in lattice-based cryptography [1].

In this paper, performance analysis of RSA and NTRU is compared. Public key encryption schemes are rarely used to actually encrypt messages; they are usually used to encrypt a symmetric key for future bulk encryption. Most public-key encryption schemes (like ECC) follow the KEM / DEM hybrid encryption paradigm. Of the schemes available only RSA-OKCS#1 v1.5 and RSA-OAEP can be considered as traditional public key encryption algorithms. But non-KEM based applications should only be used for encrypting small amounts of data [5]. NTRU is an open source public-key cryptosystem that uses lattice-based cryptography to encrypt and decrypt data.

2. RSA

Integer Factorization is a commonly used mathematical problem often used to secure public-key encryption system. RSA is an asymmetric cryptosystem which is based on Integer Factorization problem. Shor’s algorithm was introduced in 1994 and shows how to factor large products of prime numbers efficiently on a quantum computer, which undermines the security of RSA.

3. NTRU

The first version of the system, which was simply called NTRU was developed by Joseph H. Silverman, Jeffrey Hoffstein, Jill Pipher and Daniel Lieman in 1996. Now, it consists of two algorithms: NTRUEncrypt, which is used for encryption, and NTRUSign, which is used for digital signatures. The NTRUEncrypt public key cryptosystem was first presented by NTRU Cryptosystems Inc and is now included in the IEEE P1363 standard. The name NTRU is abbreviation for N-th degree truncated polynomial degree ring unit. The emergence of Quantum Computer (QC) posed a serious challenge on many current PKC that builds upon factorization (RSA) and elliptic curve (ECC) hard problems. NTRU is an emerging PKC that is resistant to QC attack and widely known as one of the strong candidate for post quantum cryptography. NTRU is built upon the shortest vector problem in a lattice, which is not known to be susceptible to QC attack. The main operation in NTRU encryption and decryption is polynomial multiplication which is the most complex operation and which is known to be faster than RSA. NTRU can be computed even faster if coefficients of the polynomial used are of small values (binary or ternary) and sparse [2] [3].

3.1NTRU Parameters

N - The polynomials in the truncated polynomial R have degree N-1.

q - Large modulo: The coefficients of the truncated polynomials will be reduced mod q.

p - Small modulo: The coefficients of the message are reduced to mod p.
f - A polynomial that is the private key.
g - A polynomial that is used to generate the public key h from f.
h - A polynomial that is the public key.
r - The random "blinding polynomial".
K - A security parameter which controls resistance to certain types of attacks, including plaintext awareness.
df - The polynomial f has df coefficients equal to 1, (df-1) coefficients equal to -1, and the rest equal to 0.
dg - The polynomial g has dg coefficients equal to 1, dg coefficients equal to -1, and the rest equal to 0.
dr - The polynomial r has dr coefficients equal to 1, dr coefficients equal to -1, and the rest equal to 0.
dm - Plaintext space.

NTRUEncrypt requires the message to be in a polynomial form, therefore the need of dm to define the form of the message to be encrypted. The more relevant properties of NTRU PKC are the following:

1. The parameters (N, p, q) are public and p and q must satisfy gcd(p, q) = 1.
2. Coefficients of polynomials are bounded modulo p and modulo q.
3. The inverse of a(X) mod q is the polynomial A(X):

a(X) *A(X) = 1 mod q.

3.2 Key Generation

The key generation consists in the generation of the private key (f, fp) and the public key h. Choose random polynomials f and g from R with "small" coefficients. Meaning "small" much smaller than q, typically f {-1, 0, 1} for p = 3. Then compute fp, i.e. the inverse of f (mod p) defined by f * fp = 1 (mod p). Compute fq, the inverse of f (mod q) that analogously satisfies the requirement: f * fq = 1 (mod q). Compute the polynomial h = g * p fq. The public key is h and the private key is the set (f, fp).

3.3Encryption

The plaintext m is a polynomial with coefficients taken mod p. Note that converting the message m to a polynomial form is not part of NTRU public-key algorithm. Choose a blinding message r randomly from R with small coefficients. The cipher text is e = r * h + m (mod q).

3.4Decryption

The decryption returns the message m from the encrypted message e using the private key (f, fp). Compute a = e * f (mod q); choosing the coefficients of a to satisfy $-q/2 < a_i < q/2$. Reduce a modulo p: b = a (mod p): Compute c = b * fp (mod p). Then c mod p is equal to the plaintext m [3].

4. COMPARISION OF RSA AND NTRU

Modular exponentiation used in RSA is very time-consuming when the key-size is large (above 1024 bits) and today even RSA-1024 is considered insecure. Recommended key size for RSA is 2048 bits. The speed performance of RSA comes to concerns if large data are needed to handle by the server at a time, through a massive number of connected mobile and wireless devices.

In contrast, NTRU has become popular these days because of its ability to resist attacks from quantum computer. NTRU can offer classical security levels of 192 bits and 256 bits using relatively shorter length keys than RSA. So, NTRU requires less space for key storage and less time for key transmission. NTRU is one of the best choices for securing devices with constrained resources such as mobile devices, PDAs etc.

4.1Mathematical Problem

RSA is based on integer factorization problem. It is very hard to find two prime factors of a very large number. NTRU uses short vector problem (geometrical problem). It is based on hardness of lattice problems.

4.2Plaintext Size

Theoretically, maximum size of plaintext block in RSA is equal to the key size (N). For example, if N = 2048 bits, then plaintext block should be less than or equal to 2048 bits. But, RSA without padding is considered to be insecure. The technique of encoding a message and then encrypting it with RSA is provably secure in the random oracle model. There are two popular encoding schemes used with RSA – PKCS#1-v1.5 and PKCS#1 OAEP. Padding used in PKCS#1 OAEP is 42 bytes whereas in PKCS#1-v1.5 it is 11 bytes. From the two schemes, only RSA-OAEP is secure. Table 1 shows maximum block size of message can be encrypted in RSA using two different padding schemes for different key size.

In NTRU, plaintext block size = N * $\log_2(p)$. NTRU is also vulnerable to CPA attack and CCA attack if a proper padding scheme is not used.

Table 1: Maximum block size of plaintext in RSA

	PKCS#1 OAEP	PKCS#1-v1.5
RSA – N	(N / 8 – 42) bytes	(N / 8 – 11) bytes
RSA – 1024	86 bytes	117 bytes
RSA – 2048	214 bytes	245 bytes
RSA - 3072	342 bytes	373 bytes
RSA - 4096	470 bytes	501 bytes
RSA - 7680	918 bytes	949 bytes
RSA - 15360	1878 bytes	1909 bytes

4.3Ciphertext Size

In RSA, size of generated ciphertext is equal to N bits whereas in case of NTRU ciphertext block size is $N*\log_2(q)$.

4.4Key Size

In RSA, key size is equal to N bits. In NTRU, public key consists of one ring of N element. Each ring element is a value between 0 and q – 1 ($\log_2(q)$ bits for each element). So key size is $N*\log_2(q)$.

4.5Execution (Running) Time

Execution time in RSA is sub-exponential whereas in case of NTRU it is exponential. In average case, RSA's time complexity is O (n^3) and NTRU's time complexity is O (n log (n)). In worst case, NTRU's time complexity is O (n^2). That is, NTRU delivers encryption and decryption multiple times faster than RSA. As key size increases by a factor of n, RSA's operations/second decrease by about n^3 whereas NTRU's decrease at n^2.

4.6 Security Levels

For RSA, key sizes are well known and used as standard key sizes. RSA with given keys is not secure against quantum computers. In contrast, NTRU's security is reduced only slightly by quantum computers. Compared to RSA, NTRU keys are not well standardized. The coefficients in polynomial R are reduced to modulus q. q can be set to 128,256, 512, 1024 or 2048. But if q is less than 2048, decryption failure increases.
Table 2 and 3 show the key requirement for corresponding security level in case of RSA and NTRU respectively.

Table 2:RSA key size for differentsecuritylevels

Classical Security Level (K bits)	RSA Key Size (N bits)
80	1024
112	2048
128	3072
160	4096
192	7680
256	15360

Table 3: NTRU key size for different security levels (Considering q = 2048)

Classical Security Level (K bits)	Quantum Security Level (bits)	NTRU Ring Size (N bits)	NTRU Key Size ($N*\log_2(q)$ bits)
112	112	401	4411
128	128	439	4829
192	128	539	5929
256	128	743	8173

5. METHODOLOGY AND RESULT

In this paper, a python library for NTRU "pyNTRUEncrypt" is used for testing performance of the NTRU encryption and decryption. For RSA, python "Crypto" library is used along with OAEP encoding. Pypy3 implementation of python is used for running the python files. The testing is done on Ubuntu 18.04, Intel(R) Core(TM) i3-3110M CPU @ 2.40GHz, 4GB RAM.

The NTRU parameters [4] used for different security levels and key generation time required is listed in table 4. The RSA key generation times for different key sizes are listed in table 5. Specially for RSA key generation time can vary in wider range. Encryption and decryption times required for security level of 192 bits and 256 bits for RSA Cryptosystem and NTRUEncrypt using different file sizes are listed in table 6 and table 7.

Table 4: NTRU Key Generation Time (Considering p = 3)

Classical Security Level (K bits)	Ring Size (N)	df	dg	dr	Key Generation Time (sec)
112	401	133	133	133	0.346
128	439	146	146	146	0.394
192	593	197	197	197	0.622
256	743	247	247	247	0.88

Table 5: RSA Key Generation Time

Classical Security Level (K bits)	RSA Key Size (N)	Key Generation Time (sec)
80	1024	0.337
112	2048	1.354
128	3072	4.8
160	4096	11.2
192	7680	32.58
256	15360	512

Table 6: Encryption Time (Sec)

File Size	RSA - 7680	NTRU-539	RSA-15360	NTRU-743
8 KB	0.100	0.911	0.1319	1.148
10 KB	0.133	1.139	0.158	1.418
50 KB	0.476	5.7	0.588	6.98
100 KB	0.709	11.34	0.952	14.087

Table 7: Decryption Time (Sec)

File Size	RSA - 7680	NTRU-539	RSA-15360	NTRU-743
8 KB	2.444	1.73	8.634	2.193
10 KB	3.221	2.158	10.41	2.728
50 KB	15.02	10.69	47.96	13.44
100 KB	29.68	21.62	93.78	27.048

6. CONCLUSION

The key generation time is much faster in case of NTRUEncrypt compared to RSA cryptosystem. Also, the private key operations are much faster in NTRUEncrypt than RSA cryptosystem. As the plaintext blocks size increases, efficiency of the algorithm increases. In the experiment RSA plaintext block size is much larger than NTRU. If we reduce the RSA plaintext block size, we can see more decrease in efficiency of RSA in comparison to NTRU.

REFERENCES

1. Kunal Meher and Divya Midhunchakkaravarthy, "Hybrid Solution (ECDHE + NewHope) for PQ Transition", International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249 – 8958, Volume-9 Issue-2, page no.3893-3894, Dec-2019.
2. Xian-Fu Wong, Bok-Min Goi, Wai-Kong Lee, and Raphael C.-W. Phan, “Performance Evaluation of RSA and NTRU over GPU with Maxwell and Pascal Architecture”, Journal of Software Engineering, page no. 201-220, Nov-2017.
3. H T Loriya, A. Kulshreshta, D.R. Keraliya, “Security Analysis of Various Public Key Cryptosystems for Authentication and Key Agreement in Wireless Communication Network”, International Journal of Advanced Research in Computer and Communication Engineering, ISSN: 2278-1021, Vol. 6, Issue 2, page no. 267-274, February 2017
4. Jeff Hoffstein, Jill Pipher, John M. Schanck, Joseph H. Silverman, William Whyte, and Zhenfei Zhang, “Choosing Parameters for NTRUEncrypt”, ReseachGate, page no. 1-22, February 2017.
5. “Algorithms, key size and parameters report – 2014”, by European Union Agency for Network and Information Security, Nov – 2014.