BAREKENG: Journal of Mathematics and Its Applications
December 2023 Volume 17 Issue 4 Page 2187–2196
P-ISSN: 1978-7227 E-ISSN: 2615-3017
https://doi.org/10.30598/barekengvol17iss4pp2187-2196

ALGEBRAIC CRYPTANALYSIS ON NTRU-HPS AND NTRU-HRSS

Fadila Paradise[1*], **Kiki A. Sugeng**[2]

[1,2]*Department of Mathematics, Faculty of Mathematics and Natural Sciences, University of Indonesia Kampus UI Depok, Depok, 16424, Indonesia*

*Corresponding author's e-mail: * **fadila@sci.ui.ac.id***

ABSTRACT

Article History:

Received: 8th July 2023
Revised: 5th October 2023
Accepted: 4th November 2023

Keywords:

Algebraic Cryptanalysis;
NTRU-HPS;
NTRU-HRSS.

NTRU is a lattice-based public-key cryptosystem designed by Hoffstein, Pipher, and Silverman in 1996. NTRU published on Algorithmic Number Theory Symposium (ANTS) in 1998. The ANTS'98 NTRU became the IEEE standard for public key cryptographic techniques based on hard problems over lattices in 2008. NTRU was later redeveloped by NTRU Inc. in 2018 and became one of the finalists in round 3 of the PQC (Post-Quantum Cryptography) standardization process organized by NIST in 2020. There are two types of NTRU algorithms proposed by NTRU Inc., which are classified based on parameter determination, NTRU-HPS (Hoffstein, Pipher, Silverman) and NTRU-HRSS (Hulsing, Rijnveld, Schanck, Schwabe). Algebraic cryptanalysis on ANTS'98 NTRU had previously been carried out in 2009 and 2012. In this paper, algebraic cryptanalysis is performed on NTRU-HPS with q=2048, n=509 (ntruhps2048509) and NTRU-HRSS with n=701 (ntruhrss701). This research aims to evaluate the resistance of NTRU-HPS and NTRU-HRSS algorithms against algebraic cryptanalysis by reconstructing the private key value. As a result, NTRU-HPS and NTRU-HRSS resistance to algebraic cryptanalysis.

This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution-ShareAlike 4.0 International License.

How to cite this article:

F. Paradise and K. A. Sugeng, "ALGEBRAIC ATTACK ON NTRU-HPS AND NTRU-HRSS," *BAREKENG: J. Math. & App.*, vol. 17, iss. 4, pp. 2187-2196, December, 2023.

Copyright © 2023 Author(s)
Journal homepage: *https://ojs3.unpatti.ac.id/index.php/barekeng/*
Journal e-mail: *barekeng.math@yahoo.com; barekeng.journal@mail.unpatti.ac.id*
Research Article · Open Access

1. INTRODUCTION

The concept of quantum computing has changed many scientific fields, including cryptography. Quantum computers can run several code breaking methods faster than classical computers [1]. For example, the Shor algorithm created by Peter Shor in 1994 can solve the large integer factorization problem in RSA if run on a quantum computer [2].

Classical public key cryptosystems such as RSA are widely used in key exchange mechanisms and digital signatures [1]. Advances in computing and algorithm development increase the need for cryptosystem development to provide a replacement for classical cryptosystems that are vulnerable to quantum computer-based cryptanalysis. These replacement cryptosystems are referred to as post-quantum cryptography [3].

In the context of the Post Quantum Cryptography (PQC) standardization process, the National Institute of Standards and Technology (NIST) conducted a selection process for PQC-based public key algorithms starting in 2017 with a total of 69 candidates. In July 2020, NIST published the candidates that became finalists in round 3, one of which was NTRU [4]. NTRU is a lattice-based public key cryptosystem that provides encryption algorithm solutions [3]. NTRU was published at the Algorithmic Number Theory Symposium (ANTS) in 1998 [5] and became the standard public key cryptography technique based on hard problems on lattice in IEEE in 2008 [6]. NTRU ANTS'98 was then redeveloped by NTRU Inc. in 2018 and underwent several changes during the NIST standardization process. There are two types of NTRU algorithms proposed by NTRU Inc. in round 3 of the PQC standardization process, namely NTRU-HPS (Hoffstein, Pipher, Silverman) and NTRU-HRSS (Hulsing, Rijnveld, Schanck, Schwabe) [7].

Currently, there is a lot of research on the implementation of NTRU both on networks, hardware, and the Internet of Things (IoT) [8]. Several security tests were also carried out on NTRU ANTS'98, such as algebraic cryptanalysis using Witt vectors and Grobner bases by Bourgeois and Faugere in 2009 [9], algebraic cryptanalysis using the method of solving equations in real numbers by Ding and Schmidt in 2012 [10], and lattice cryptanalysis experiments conducted by Bi and Han in 2021 [11], and side channel attack by Askeland and Ronjom [12]. In this research, algebraic cryptanalysis is carried out on the NTRU algorithm which has been updated and submitted by NTRU Inc. in round 3 of the NIST PQC standardization process. The purpose of this research is to determine the algebraic cryptanalysis process on the NTRU-HPS and NTRU-HRSS algorithms and to determine the resistance of the NTRU-HPS and NTRU-HRSS algorithms to algebraic cryptanalysis.

2. RESEARCH METHODS

The research methods used in the research correspond to the methods used in algebraic cryptanalysis. The main principle of algebraic cryptanalysis is simple, which is to turn the problem of attacking a cryptographic system (such as finding the secret key) into solving a system of polynomial equations [13]. This basic idea is then mapped into two stages in performing algebraic cryptanalysis as follows.

2.1 Forming a System of Equation

In a public-key cryptosystem, the private key is a key that is only owned by the key owner and is the parameter used to provide confidentiality and non-repudiation services in the public-key cryptosystem. The power of the public key cryptosystem lies in the private key. A public-key cryptosystem is said to be vulnerable if the private key is compromised.

NTRU-HPS dan NTRU-HRSS are used for encryption and key exchange management. The private key in the NTRU is used in the decryption function. Therefore, in this research, the decryption function is utilized to form a system of equations. It is assumed that the cryptanalyst has access to the decryption machine so that it can get the corresponding plaintext and ciphertext pairs without knowing the private key.

After obtaining the corresponding plaintext and ciphertext pairs, the cryptanalyst represents the decryption function in the form of an algebraic function. Cryptanalyst then enters the ciphertext value and the unknown private key variable into the function, thus forming equations of degree $(n-1)^2$ that represents the plaintext value.

2.2 Finding The Solution of A System of Equations

There are several commonly used methods to find solutions to polynomial equations, including the Grobner Basis, F4, F5, and XL algorithm [14]. In this research, the method of solving the system of equations used is linearization and Gaussian elimination. The polynomial multiplication rules in NTRU-HPS and NTRU-HRSS make the equations formed in Section 2.1 have a degree of n, but the private keys in NTRU-HPS and NTRU-HRSS have degree of n. A pair of plaintext and ciphertext can generate a system containing n equations. Therefore, n pairs of plaintext and ciphertext are generated to produce n^2 equations so that Gaussian elimination can be applied. There are no special rules on scalar multiplication and polynomial subtraction in NTRU-HPS and NTRU-HRSS, but when applying Gaussian elimination the operations must be performed in modulus q.

3. RESULTS AND DISCUSSION

NTRU-HPS (Hoffstein, Pipher, Silverman) and NTRU-HRSS (Hulsing, Rijnveld, Schanck, Schwabe) are two types of NTRU algorithms proposed in the NIST PQC standardization process. Algebraic cryptanalysis was performed on NTRU-HPS with $n = 509$ and $q = 2048$ (ntruhps2048509) also on NTRU-HRSS with $n = 701$ and $q = 4096$ (ntruhrss701). This research determines n plaintexts encrypted using the same public key to produce n corresponding ciphertexts. These n pairs of plaintexts and ciphertexts are used to generate n^2 polynomial equations according to the steps described in Section 3.1. The amount of n^2 polynomial equations is determined because the combination of n monomials of f and n monomials of f_p will produce n^2 monomials for every equation formed in Section 3.1.

3.1 System of Polynomial Equations

This research utilizes the decryption function in NTRU to generate a polynomial equation that represents the ciphertext bits. The decryption function on the NTRU consists of the following two operations

$$v = c \cdot f\ (mod\ q, \Phi_1\Phi_n) \tag{1}$$

$$m = v \cdot f_p(mod\ p, \Phi_n). \tag{2}$$

Below is a brief explanation of the symbols in the decryption function.

- $\Phi_1 = x - 1$.
- $\Phi_n = (x^n - 1)/(x - 1) = x^{n-1} + x^{n-2} \ldots + 1$.
- $\Phi_1\Phi_n = x^n - 1$.
- m is the plaintext, represented as a polynomial in $(mod\ p, \Phi_n)$.
- c is the ciphertext, represented as a polynomial in $(mod\ q, \Phi_1\Phi_n)$.
- f is the private key, represented as a polynomial in $(mod\ p, \Phi_n)$.
- f_p is the private key, $f_p \equiv f^{-1}\ (mod\ p, \Phi_n)$.
- v is the polynomial product of c and f, represented as a polynomial in $(mod\ q, \Phi_1\Phi_n)$.

The polynomials f and f_p are in $(mod\ \Phi_n)$ where $\Phi_n \in \Phi_1\Phi_n$, so Equation (1) and Equation (2) can be merged into one algebraic equation as follow

$$m = c \cdot f \cdot f_p\ (mod\ q, \Phi_1\Phi_n)(mod\ p, \Phi_n). \tag{3}$$

The polynomial product in NTRU is a cyclic convolution product defined as

$$H_k = \sum_{i=0}^{k} F_i G_{k-i} + \sum_{i=k+1}^{n-1} F_i G_{n+k-i} = \sum_{i+j \equiv k (\mathrm{mod}\ n)} F_i G_j$$

where F, G, H are any polynomials [4].

Example 1. A sample in ntruhps2048509 ($q = 2048$, $n = 509$) contains plaintext $m = \{1_{127}, 0_{254}, -1_{127}\}$, ciphertext $c = \{1209,230,174,1154, \dots, -1335\}$, $f = \{f_0, f_1, \dots, f_{507}\}$, $f_p = \{f_{p_0}, f_{p_1}, \dots, f_{p_{507}}\}$, $p = 3$. Polynomial m, c, f and f_p are represented as follows.

Table 1. Polynomial Representation of ntruhps2048509 Sample

Polynomial Representation	x^0	x^1	x^2	x^3	…	x^{507}	x^{508}
m	1	1	1	1	⋯	−1	−1
c	773	317	1865	1897	⋯	641	−755
f	f_0	f_1	f_2	f_3	⋯	f_{507}	0
f_p	f_{p_0}	f_{p_1}	f_{p_2}	f_{p_3}	⋯	$f_{p_{507}}$	0

Next, polynomial c, f and f_p are calculated as in **Equation (3)** using the cyclic convolution product with the process shown in **Table 2**.

Table 2. Cyclic Convolution Product in NTRU

Cyclic Convolution Product Output Index	$c \cdot f$	$m = c \cdot f \cdot f_p \pmod{q, \Phi_1 \Phi_n}$
0	$c_0 f_0 + c_{n-1} f_1 + c_{n-2} f_2 + \dots + c_2 f_{n-2}$	$(c_0 f_0 + c_{n-1} f_1 + c_{n-2} f_2 + \dots + c_2 f_{n-2}) \cdot f_{p_0} +$ $(c_{n-1} f_0 + c_{n-2} f_1 + c_{n-3} f_2 \dots + c_1 f_{n-2}) \cdot f_{p_1} +$ $(c_{n-2} f_0 + c_{n-3} f_1 + c_{n-4} f_2 + \dots + c_0 f_{n-2}) \cdot f_{p_2} + \dots +$ $(c_2 f_0 + c_1 f_1 + c_0 f_2 + \dots + c_4 f_{n-2}) \cdot f_{p_{n-2}}$
1	$c_1 f_0 + c_0 f_1 + c_{n-1} f_2 + \dots + c_3 f_{n-2}$	$(c_1 f_0 + c_0 f_1 + c_{n-1} f_2 + \dots + c_3 f_{n-2}) \cdot f_{p_0} +$ $(c_0 f_0 + c_{n-1} f_1 + c_{n-2} f_2 + \dots + c_2 f_{n-2}) \cdot f_{p_1} +$ $(c_{n-1} f_0 + c_{n-2} f_1 + c_{n-3} f_2 \dots + c_1 f_{n-2}) \cdot f_{p_2} + \dots +$ $(c_3 f_0 + c_2 f_1 + c_1 f_2 + \dots + c_5 f_{n-2}) \cdot f_{p_{n-2}}$
2	$c_2 f_0 + c_1 f_1 + c_0 f_2 + \dots + c_4 f_{n-2}$	$(c_2 f_0 + c_1 f_1 + c_0 f_2 + \dots + c_4 f_{n-2}) \cdot f_{p_0} +$ $(c_1 f_0 + c_0 f_1 + c_{n-1} f_2 + \dots + c_3 f_{n-2}) \cdot f_{p_1} +$ $(c_0 f_0 + c_{n-1} f_1 + c_{n-2} f_2 + \dots + c_2 f_{n-2}) \cdot f_{p_2} + \dots +$ $(c_4 f_0 + c_3 f_1 + c_2 f_2 + \dots + c_6 f_{n-2}) \cdot f_{p_{n-2}}$
⋮	⋮	⋮
$n-2$	$c_{n-2} f_0 + c_{n-3} f_1 + c_{n-4} f_2 + \dots + c_0 f_{n-2}$	$(c_{n-2} f_0 + c_{n-3} f_1 + c_{n-4} f_2 + \dots + c_0 f_{n-2}) \cdot f_{p_0} +$ $(c_{n-3} f_0 + c_{n-4} f_1 + c_{n-5} f_2 \dots + c_{n-1} f_{n-2}) \cdot f_{p_1} +$ $(c_{n-4} f_0 + c_{n-5} f_1 + c_{n-6} f_2 + \dots + c_{n-2} f_{n-2}) \cdot f_{p_2} + \dots$ $+$ $(c_0 f_0 + c_{n-1} f_1 + c_{n-2} f_2 + \dots + c_2 f_{n-2}) \cdot f_{p_{n-2}}$
$n-1$	$c_{n-1} f_0 + c_{n-2} f_1 + c_{n-3} f_2 \dots + c_1 f_{n-2}$	$(c_{n-1} f_0 + c_{n-2} f_1 + c_{n-3} f_2 \dots + c_1 f_{n-2}) \cdot f_{p_0} +$ $(c_{n-2} f_0 + c_{n-3} f_1 + c_{n-4} f_2 + \dots + c_0 f_{n-2}) \cdot f_{p_1} +$ $(c_{n-3} f_0 + c_{n-4} f_1 + c_{n-5} f_2 \dots + c_{n-1} f_{n-2}) \cdot f_{p_2} + \dots +$ $(c_1 f_0 + c_0 f_1 + c_{n-1} f_2 + \dots + c_3 f_{n-2}) \cdot f_{p_{n-2}}$

To simplify the calculation, the unknown variables of the private keys f and f_p in the equations are sorted from $f_0 f_{p_0}$, $f_1 f_{p_0}$, …, $f_{n-1} f_{p_0}$ to $f_0 f_{p_{n-1}}$, $f_1 f_{p_{n-1}}$, …, $f_{n-1} f_{p_{n-1}}$ as shown in **Table 3**.

Table 3. Polynomial m in NTRU with Sorted Unknown f and f_p

m	$f_0 f_{p_0}$	$f_0 f_{p_1}$	…	$f_0 f_{p_{n-2}}$	$f_1 f_{p_0}$	…	$f_{n-2} f_{p_0}$	$f_{n-2} f_{p_1}$	…	$f_{n-2} f_{p_{n-2}}$
m_0	c_0	c_{n-1}	…	c_2	c_{n-1}	…	c_2	c_1	…	c_4
m_1	c_1	c_0	…	c_3	c_0	…	c_3	c_2	…	c_5
m_2	c_2	c_1	…	c_4	c_1	…	c_4	c_3	…	c_6
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
m_{n-2}	c_{n-2}	c_{n-3}	…	c_0	c_{n-3}	…	c_0	c_{n-1}	…	c_2
m_{n-1}	c_{n-1}	c_{n-2}	…	c_1	c_{n-2}	…	c_1	c_0	…	c_3

The polynomial m in **Table 3** then modulo Φ_n with the results listed in **Table 4**.

Table 4. Polynomial $m(mod\ q, \Phi_1\Phi_n)(mod\ \Phi_n)$ in NTRU

m	$f_0f_{p_0}$	$f_1f_{p_0}$	...	$f_{n-2}f_{p_0}$	$f_0f_{p_1}$	...	$f_0f_{p_{n-2}}$	$f_1f_{p_{n-2}}$	...	$f_{n-2}f_{p_{n-2}}$
m_0	c_0-c_{n-1}	$c_{n-1}-c_{n-2}$	...	c_2-c_1	$c_{n-1}-c_{n-2}$	...	c_2-c_1	c_1-c_0	...	c_4-c_3
m_1	c_1-c_{n-1}	c_0-c_{n-2}	...	c_3-c_1	c_0-c_{n-2}	...	c_3-c_1	c_2-c_0	...	c_5-c_3
m_2	c_2-c_{n-1}	c_1-c_{n-2}	...	c_4-c_1	c_1-c_{n-2}	...	c_4-c_1	c_3-c_0	...	c_6-c_3
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
m_{n-2}	$c_{n-2}-c_{n-1}$	$c_{n-3}-c_{n-2}$	...	c_0-c_1	$c_{n-3}-c_{n-2}$	...	c_0-c_1	$c_{n-1}-c_0$	...	c_2-c_3

The values in **Table 1** are then entered into the variables in **Table 4** to produce the values in **Table 5**.

Table 5. Polynomial $m(mod\ q, \Phi_1\Phi_n)(mod\ \Phi_n)$ in ntruhps2048509

m	$f_0f_{p_0}$	$f_1f_{p_0}$	...	$f_{n-2}f_{p_0}$	$f_0f_{p_1}$	...	$f_0f_{p_{n-2}}$	$f_1f_{p_{n-2}}$	...	$f_{n-2}f_{p_{n-2}}$
m_0	1528	652	...	1548	652	...	1548	1592	...	759
m_1	1072	132	...	1580	132	...	1580	1092	...	1835
m_2	572	1724	...	291	1724	...	291	1124	...	1552
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
m_{507}	1396	1061	...	454	1061	...	454	520	...	2016

Based on the parameters in Example 1 and the values in **Table 5**, below is the illustration of polynomial equations that represent bits of the ntruhps2048509 plaintext sample.

1. $1528f_0f_{p_0}+652f_1f_{p_0}+987f_2f_{p_0}+1688f_3f_{p_0}+166f_4f_{p_0}+\cdots+759f_{507}f_{p_{507}}=1$
2. $1072f_0f_{p_0}+132f_1f_{p_0}+1639f_2f_{p_0}+627f_3f_{p_0}+1854f_4f_{p_0}+\cdots+1835f_{507}f_{p_{507}}=1$
3. $572f_0f_{p_0}+1724f_1f_{p_0}+1119f_2f_{p_0}+1279f_3f_{p_0}+793f_4f_{p_0}+\cdots+1552f_{507}f_{p_{507}}=1$
4. $604f_0f_{p_0}+1224f_1f_{p_0}+663f_2f_{p_0}+759f_3f_{p_0}+1445f_4f_{p_0}+\cdots+1761f_{507}f_{p_{507}}=1$
5. $1363f_0f_{p_0}+1256f_1f_{p_0}+163f_2f_{p_0}+303f_3f_{p_0}+925f_4f_{p_0}+\cdots+219f_{507}f_{p_{507}}=1$
6. $391f_0f_{p_0}+2015f_1f_{p_0}+195f_2f_{p_0}+1851f_3f_{p_0}+469f_4f_{p_0}+\cdots+585f_{507}f_{p_{507}}=1$
7. $108f_0f_{p_0}+1043f_1f_{p_0}+954f_2f_{p_0}+1883f_3f_{p_0}+2017f_4f_{p_0}+\cdots+847f_{507}f_{p_{507}}=1$
8. $317f_0f_{p_0}+760f_1f_{p_0}+2030f_2f_{p_0}+594f_3f_{p_0}+1f_4f_{p_0}+\cdots+717f_{507}f_{p_{507}}=1$
9. $823f_0f_{p_0}+969f_1f_{p_0}+1747f_2f_{p_0}+1670f_3f_{p_0}+760f_4f_{p_0}+\cdots+1273f_{507}f_{p_{507}}=1$
10. $1189f_0f_{p_0}+1475f_1f_{p_0}+1956f_2f_{p_0}+1387f_3f_{p_0}+1836f_4f_{p_0}+\cdots+1180f_{507}f_{p_{507}}=1$
11. $1451f_0f_{p_0}+1841f_1f_{p_0}+414f_2f_{p_0}+1596f_3f_{p_0}+1553f_4f_{p_0}+\cdots+1497f_{507}f_{p_{507}}=1$
12. $1321f_0f_{p_0}+55f_1f_{p_0}+780f_2f_{p_0}+54f_3f_{p_0}+1762f_4f_{p_0}+\cdots+171f_{507}f_{p_{507}}=1$
13. $1877f_0f_{p_0}+1973f_1f_{p_0}+1042f_2f_{p_0}+420f_3f_{p_0}+220f_4f_{p_0}+\cdots+1584f_{507}f_{p_{507}}=1$
14. $1784f_0f_{p_0}+481f_1f_{p_0}+912f_2f_{p_0}+682f_3f_{p_0}+586f_4f_{p_0}+\cdots+1201f_{507}f_{p_{507}}=1$
15. $53f_0f_{p_0}+388f_1f_{p_0}+1468f_2f_{p_0}+552f_3f_{p_0}+848f_4f_{p_0}+\cdots+2021f_{507}f_{p_{507}}=1$
16. $775f_0f_{p_0}+705f_1f_{p_0}+1375f_2f_{p_0}+1108f_3f_{p_0}+718f_4f_{p_0}+\cdots+301f_{507}f_{p_{507}}=1$
17. $140f_0f_{p_0}+1427f_1f_{p_0}+1692f_2f_{p_0}+1015f_3f_{p_0}+1274f_4f_{p_0}+\cdots+1255f_{507}f_{p_{507}}=1$
18. $1805f_0f_{p_0}+792f_1f_{p_0}+366f_2f_{p_0}+1332f_3f_{p_0}+1181f_4f_{p_0}+\cdots+706f_{507}f_{p_{507}}=1$
19. $577f_0f_{p_0}+409f_1f_{p_0}+1779f_2f_{p_0}+6f_3f_{p_0}+1498f_4f_{p_0}+\cdots+1046f_{507}f_{p_{507}}=1$
20. $905f_0f_{p_0}+1229f_1f_{p_0}+1396f_2f_{p_0}+1419f_3f_{p_0}+172f_4f_{p_0}+\cdots+1979f_{507}f_{p_{507}}=1$

⋮

507. $409f_0f_{p_0}+1421f_1f_{p_0}+194f_2f_{p_0}+1260f_3f_{p_0}+2038f_4f_{p_0}+\cdots+468f_{507}f_{p_{507}}=-1$
508. $1396f_0f_{p_0}+1061f_1f_{p_0}+360f_2f_{p_0}+1882f_3f_{p_0}+1426f_4f_{p_0}+\cdots+2016f_{507}f_{p_{507}}=-1$

Below is the illustration of polynomial equations that represent bits of ntruhrss701 plaintext sample with plaintext $=\{1,0_{700}\}$, ciphertext $=\{1363,2145,4414,5577,\ldots,3025,-7599\}$, $q=8192$, $n=701$.

1. $770f_0f_{p_0}+5760f_1f_{p_0}+7298f_2f_{p_0}+4844f_3f_{p_0}+1343f_4f_{p_0}+\cdots+1442f_{699}f_{p_{699}}=1$

2. $1552f_0f_{p_0} + 6530f_1f_{p_0} + 4866f_2f_{p_0} + 3950f_3f_{p_0} + 6187f_4f_{p_0} + \cdots + 7070f_{699}f_{p_{699}} = 0$
3. $3821f_0f_{p_0} + 7312f_1f_{p_0} + 5636f_2f_{p_0} + 1518f_3f_{p_0} + 5293f_4f_{p_0} + \cdots + 6462f_{699}f_{p_{699}} = 0$
4. $4984f_0f_{p_0} + 1389f_1f_{p_0} + 6418f_2f_{p_0} + 2288f_3f_{p_0} + 2861f_4f_{p_0} + \cdots + 1109f_{699}f_{p_{699}} = 0$
5. $6426f_0f_{p_0} + 2552f_1f_{p_0} + 495f_2f_{p_0} + 3070f_3f_{p_0} + 3631f_4f_{p_0} + \cdots + 6346f_{699}f_{p_{699}} = 0$
6. $3862f_0f_{p_0} + 3994f_1f_{p_0} + 1658f_2f_{p_0} + 5339f_3f_{p_0} + 4413f_4f_{p_0} + \cdots + 442f_{699}f_{p_{699}} = 0$
7. $3254f_0f_{p_0} + 1430f_1f_{p_0} + 3100f_2f_{p_0} + 6502f_3f_{p_0} + 6682f_4f_{p_0} + \cdots + 5638f_{699}f_{p_{699}} = 0$
8. $6093f_0f_{p_0} + 822f_1f_{p_0} + 536f_2f_{p_0} + 7944f_3f_{p_0} + 7845f_4f_{p_0} + \cdots + 4098f_{699}f_{p_{699}} = 0$
9. $3138f_0f_{p_0} + 3661f_1f_{p_0} + 8120f_2f_{p_0} + 5380f_3f_{p_0} + 1095f_4f_{p_0} + \cdots + 7146f_{699}f_{p_{699}} = 0$
10. $5426f_0f_{p_0} + 706f_1f_{p_0} + 2767f_2f_{p_0} + 4772f_3f_{p_0} + 6723f_4f_{p_0} + \cdots + 3111f_{699}f_{p_{699}} = 0$
11. $2430f_0f_{p_0} + 2994f_1f_{p_0} + 8004f_2f_{p_0} + 7611f_3f_{p_0} + 6115f_4f_{p_0} + \cdots + 4082f_{699}f_{p_{699}} = 0$
12. $890f_0f_{p_0} + 8190f_1f_{p_0} + 2100f_2f_{p_0} + 4656f_3f_{p_0} + 762f_4f_{p_0} + \cdots + 7455f_{699}f_{p_{699}} = 0$
13. $3938f_0f_{p_0} + 6650f_1f_{p_0} + 7296f_2f_{p_0} + 6944f_3f_{p_0} + 5999f_4f_{p_0} + \cdots + 2602f_{699}f_{p_{699}} = 0$
14. $8095f_0f_{p_0} + 1506f_1f_{p_0} + 5756f_2f_{p_0} + 3948f_3f_{p_0} + 95f_4f_{p_0} + \cdots + 4937f_{699}f_{p_{699}} = 0$
15. $874f_0f_{p_0} + 5663f_1f_{p_0} + 612f_2f_{p_0} + 2408f_3f_{p_0} + 5291f_4f_{p_0} + \cdots + 3825f_{699}f_{p_{699}} = 0$
16. $4247f_0f_{p_0} + 6634f_1f_{p_0} + 4769f_2f_{p_0} + 5456f_3f_{p_0} + 3751f_4f_{p_0} + \cdots + 4335f_{699}f_{p_{699}} = 0$
17. $7586f_0f_{p_0} + 1815f_1f_{p_0} + 5740f_2f_{p_0} + 1421f_3f_{p_0} + 6799f_4f_{p_0} + \cdots + 5652f_{699}f_{p_{699}} = 0$
18. $1729f_0f_{p_0} + 5154f_1f_{p_0} + 921f_2f_{p_0} + 2392f_3f_{p_0} + 2764f_4f_{p_0} + \cdots + 1259f_{699}f_{p_{699}} = 0$
19. $617f_0f_{p_0} + 7489f_1f_{p_0} + 4260f_2f_{p_0} + 5765f_3f_{p_0} + 3735f_4f_{p_0} + \cdots + 3900f_{699}f_{p_{699}} = 0$

$\vdots$

699. $1127f_0f_{p_0} + 6377f_1f_{p_0} + 6595f_2f_{p_0} + 912f_3f_{p_0} + 7108f_4f_{p_0} + \cdots + 6815f_{699}f_{p_{699}} = 0$
700. $2432f_0f_{p_0} + 894f_1f_{p_0} + 3348f_2f_{p_0} + 6849f_3f_{p_0} + 1882f_4f_{p_0} + \cdots + 7029f_{699}f_{p_{699}} = 0$

The system of equations generated in this research consists of n^2 polynomial equations, 508^2 for ntruhps2048509 and 700^2 for ntruhrss701.

3.2 Solving System of Polynomial Equations

The way to solve the system of polynomial equations in this research basically uses the concept of linearization in XL Algorithm [15] and Gaussian elimination solution method, but the concept of extending system of polynomial equations in XL Algorithm is not suitable with this research because it will increase the number of monomials in the polynomial equations. To fulfill the number of equations needed in performing Gaussian elimination, the option chosen is by generating n plaintext ciphertext pairs and converting n^2 plaintext bits into n^2 polynomial equations.

Next step is linearizing the system of polynomial equations that have been formed in Section 3.1. Linearization is carried out to convert polynomial equations into linear equations to facilitate the Gaussian elimination process. In this research, the monomials contained in the system of polynomial equations on ntruhps2048509 are 508^2, while on ntruhrss701 are 700^2. Linearization is performed by substituting the monomials $f_0f_{p_0}$, $f_0f_{p_1}$, ..., $f_nf_{p_n}$ in the equations into monomials of degree 1, namely M1, M2, M3, ..., M(n^2) respectively.

The linearized equation is converted into an $n^2 \times n^2$ matrix. Gaussian elimination is then performed on the matrix. The coefficients of the polynomial equations are in $\mathbb{Z}_q$ so that in this research the elementary row operations performed must pay attention to the rules of operation on the modulus q. The q values in NTRU-HPS and NTRU-HRSS are multiple of 2. Some coefficients will be not relatively prime with q that means they do not have an inverse in $\mathbb{Z}_q$. When the leading entry of a row has no inverse modulo q then it is difficult to convert the value to 1. Therefore, the elementary row operation in this research will only produce matrices that are close to the row echelon form. Below are the tricks in performing row echelon operations in q modulus in this research:

a. Look at the leading entry of the top row.
   1) If the leading entry is odd, calculate the inverse of the leading entry. Multiply all entries in the row by the inverse of its leading entry. Do multiplication and subtraction like row echelon operation in common to the all lower row. After that, do the step in point **h**.
   2) If the leading entry is even then do step in point **b**.

b. Look at the leading entry of the lower row.
   1) If the leading entry of the lower row is odd, swap it with the top row and do step in point **a.1**.
   2) If the leading entry is even then repeat step in point **b** until the lowest row.
   3) If the leading entries in all rows are even, continue to the step in point **c**.
c. Count the factor of the leading entries of this row and all lower rows. Move the row which has the smallest factor of 2 to the top.
d. Count gcd between leading entry of the top row and leading entries of all lower rows.
e. Divide each lower row by its gcd which obtained in point **d**, then calculate the inverse of this division.
f. Multiply each lower row by the value obtained in point **e**, and divide it by gcd in point **d**.
g. Subtract each entry in the row by the product of the top row and the value at point **f**.
h. If the top row in this step is not the second lowest row of this matrix, repeat all the steps form point **a**. If not then the calculation is complete.

Example 2**.** Below is matrix "A" which coefficient is in modulus $q = 8192$. The leading entries in the top row and all lower rows are even, so it is needed to carry out the steps in point **b** until **h**.

$$\begin{bmatrix} 1 & 1771 & 5292 & 7068 & 7371 \\ 0 & 1 & 3783 & 4032 & 6124 \\ 0 & 0 & 6142 & 398 & 2976 \\ 0 & 0 & 1700 & 6344 & 1776 \\ 0 & 0 & 6896 & 4644 & 7140 \end{bmatrix}$$

- Count the factor of the leading entries.

  $a_{3,3} = 6142 = 2 \cdot 3071 \rightarrow a_{3,3}$ becomes top row

  $a_{4,3} = 1700 = 2^2 \cdot 425$

  $a_{5,3} = 6896 = 2^4 \cdot 431$

- Count gcd between leading entry of the top row and leading entries of all lower rows.

  $gcd(a_{3,3}, a_{4,3}) = gcd(6142,1700) = 2$

  $gcd(a_{3,3}, a_{5,3}) = gcd(6142,6896) = 2$

- Divide each lower row by its gcd which obtained above, then calculate the inverse of this division.

  $$\left(\frac{a_{3,3}}{gcd(a_{3,3},a_{4,3})}\right)^{-1} (mod\ 8192) \equiv \left(\frac{6142}{2}\right)^{-1} (mod\ 8192) \equiv 5119$$

  $$\left(\frac{a_{3,3}}{gcd(a_{3,3},a_{5,3})}\right)^{-1} (mod\ 8192) \equiv \left(\frac{6142}{2}\right)^{-1} (mod\ 8192) \equiv 5119$$

- Multiply each lower row by the value obtained above, and divide it by the gcd.

  $$x_{4,3} \equiv \frac{a_{4,3}}{gcd(a_{3,3},a_{5,3})} \cdot \left(\frac{a_{3,3}}{gcd(a_{3,3},a_{5,3})}\right)^{-1} (mod\ 8192) \equiv \frac{1700}{2} \cdot 5119 (mod\ 8192) \equiv 1198$$

  $$x_{5,3} \equiv \frac{a_{5,3}}{gcd(a_{3,3},a_{5,3})} \cdot \left(\frac{a_{3,3}}{gcd(a_{3,3},a_{5,3})}\right)^{-1} (mod\ 8192) \equiv \frac{6896}{2} \cdot 5119 (mod\ 8192) \equiv 4744$$

- Subtract each entry in the row by the product of the top row and the value at point f

  $a'_{4,3} = a_{4,3} - a_{3,3} * x_{4,3}\ (mod\ q) = 1700 - 6142 * 1198\ (mod\ 8192) = 0$

  $a'_{5,3} = a_{5,3} - a_{3,3} * x_{5,3}\ (mod\ q) = 6896 - 6142 * 4744\ (mod\ 8192) = 0$

  The same process done to $a_{4,4}, a_{4,5}, a_{5,4}, a_{5,5}$.

The elementary row operation on the system of polynomial equations representing the NTRU-HPS and NTRU-HRSS first plaintext bits in Example 2 resulted in the matrices shown in **Table 6** and **Table 7**, respectively.

Table 6. Matrix of NTRU-HPS

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15	M16	M17	M18	…	M508	*ptx*
c_1	1	149	744	995	414	852	42	1750	1573	82	398	2034	623	190	1701	1875	13	2015	…	492	1
c_2	0	1	1295	752	1022	150	188	1826	1719	1099	1515	607	1196	706	2007	1952	137	298	…	316	0
c_3	0	0	1	1701	1609	1122	762	1500	705	1031	772	972	1307	229	1417	318	1234	362	…	1439	0
c_4	0	0	0	1	168	471	794	190	843	1483	926	1493	1696	698	1606	434	780	1103	…	1303	1
c_5	0	0	0	0	1	104	695	1754	222	2027	395	766	501	768	922	422	1202	1452	…	1379	1
c_6	0	0	0	0	0	1	341	59	1521	1049	679	1366	909	1748	509	303	370	1653	…	805	0
c_7	0	0	0	0	0	0	1	79	1260	369	221	1428	852	112	660	1916	105	1655	…	1363	-1
c_8	0	0	0	0	0	0	0	1	1604	314	933	1148	1195	566	1211	184	707	1479	…	1903	0
c_9	0	0	0	0	0	0	0	0	1	826	2040	63	185	664	1201	806	1041	780	…	1144	0
c_{10}	0	0	0	0	0	0	0	0	0	1	870	84	161	1811	714	1279	1652	499	…	1754	0
c_{11}	0	0	0	0	0	0	0	0	0	0	1	1578	1182	119	41	1128	1329	702	…	226	0
c_{12}	0	0	0	0	0	0	0	0	0	0	0	1	1414	1858	1955	725	1508	133	…	741	0
c_{13}	0	0	0	0	0	0	0	0	0	0	0	0	1	416	1936	519	1853	1434	…	900	0
c_{14}	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1954	1555	2014	1720	…	206	-1
c_{15}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1247	1689	1277	…	1553	0
c_{16}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1234	1167	…	452	0
c_{17}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	174	…	306	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
c_{508}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	…	1970	0

Table 7. Matrix of NTRU-HRSS

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15	M16	M17	M18	…	M700	*ptx*
c_1	1	3280	2452	4582	1217	2293	876	3844	2802	7917	3121	6873	7650	375	1333	7	1603	7053	…	5106	0
c_2	0	1	3130	7072	2025	7	2920	2559	5184	4872	5196	3501	1984	198	6929	2102	4615	4327	…	6446	0
c_3	0	0	1	6894	5547	8191	3934	1897	7869	76	1271	1114	6152	1354	1258	668	8136	2515	…	7758	0
c_4	0	0	0	1	1441	3381	826	8055	2995	1845	1099	3209	8157	6306	2646	5325	4590	6884	…	4187	0
c_5	0	0	0	0	1	7254	1953	7115	6558	2080	1221	5742	2841	4998	684	6753	782	3804	…	1107	-1
c_6	0	0	0	0	0	1	4736	4097	2373	7576	16	6517	1728	7161	6672	7204	4201	3672	…	4783	-1
c_7	0	0	0	0	0	0	1	2674	6913	3917	5112	1293	8141	1484	4389	4068	3110	4904	…	2690	1
c_8	0	0	0	0	0	0	0	1	1258	2525	2149	3868	5781	997	1140	1705	5100	50	…	2096	1
c_9	0	0	0	0	0	0	0	0	1	920	2765	2763	5200	1285	3885	1678	4993	5458	…	5771	1
c_{10}	0	0	0	0	0	0	0	0	0	1	1186	5424	1248	4846	6044	5019	7994	1618	…	6691	-1
c_{11}	0	0	0	0	0	0	0	0	0	0	1	7402	3306	4640	1568	2240	2991	1274	…	3607	0
c_{12}	0	0	0	0	0	0	0	0	0	0	0	1	3681	3666	5603	3142	3086	5467	…	4332	1
c_{13}	0	0	0	0	0	0	0	0	0	0	0	0	1	4661	3027	6917	1472	2939	…	6887	0
c_{14}	0	0	0	0	0	0	0	0	0	0	0	0	0	1	6054	1170	3817	6974	…	73	-1
c_{15}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2797	1713	4912	…	6076	1
c_{16}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	2928	6133	…	4972	-1
c_{17}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	6346	…	5454	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
c_{700}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	…	1	1

The steps in Section 3.1 and Section 3.2 done to all plaintext ciphertext sample and form matrix with size $700^2 \times 700^2$.

3.3 Key Recovery

The monomials in the resulting matrices in Section 3.2 that are then converted back to their original form $(f_i f_{p_j})$. The difficulty is the value of each coefficient obtained in the matrices is the result of multiplication between f_i and f_{p_j}, which means it will result in many solutions. Further research is needed on efficient calculations in factorizing numbers in the q modulus. Therefore, in this research, algebraic cryptanalysis on the NTRU-HPS and NTRU-HRSS algorithms cannot be carried out until the key recovery stage.

4. CONCLUSIONS

The NTRU-HPS algorithm with $q = 2048$, $n = 509$ and NTRU-HRSS with $n = 701$ are resistant to algebraic cryptanalysis. However, there is still potential for algebraic cryptanalysis to be successfully performed on NTRU-HPS and NTRU-HRSS with further research in the future.

ACKNOWLEDGMENT

This research is funded by Dikti Research Grant No NKB-988/UN2.RST/HKP.05.00/2022.

REFERENCES

[1] M. Haart and C. Hoffs, "Quantum Computing: What it is, how we got here, and who's working on it.," Mar. 2019.
[2] K. Li, P. G. Yan and Q. Y. Cai, "Quantum computing and the security of public key cryptography," *Fundamental Research*, vol. 1, no. 1, pp. 85–87, Jan. 2021.
[3] K. S. Roy, "A survey on post-quantum cryptography for constrained devices," *International Journal of Applied Engineering Research*, vol. 14, pp. 2608-2615, May. 2019.
[4] D. Moody et al., "Status report on the second round of the NIST post-quantum cryptography standardization process," *National Institute of Standards and Technology*, 2020, doi: 10.6028/NIST.IR.8309.
[5] J. Hoffstein, J. Pipher and J. H. Silverman, "NTRU: A Ring-Based Public Key Cryptosystem," *Lecture Notes in Computer Science*, Berlin: Springer, 1998, 267-288.
[6] *IEEE Standard Specification for Public Key Cryptographic Techniques Based on Hard Problems over Lattices IEEE Computer Society*, IEEE Std 1363.1-2008, 2009.
[7] C. Chen et al., "NTRU algorithm specifications and supporting documentation," *NTRU Inc.*, 2019.
[8] Y. M. Agus, M. A. Murti, F. Kurniawan, N. D. W. Cahyani, and G. B. Satrya, "An Efficient Implementation of NTRU Encryption in Post-Quantum Internet of Things," in *2020 27th International Conference on Telecommunications (ICT)*, 2020, pp. 1–5. doi: 10.1109/ICT49546.2020.9239560.
[9] G. Bourgeois and J. C. Faugère, "Algebraic attack on NTRU using Witt vectors and Gröbner bases," *Journal of Mathematical Cryptology*, vol. 3, no. 3, pp. 205–214, Sep. 2009, doi: 10.1515/JMC.2009.011.
[10] J. Ding and D. Schmidt, "Algebraic attack on lattice-based cryptosystems via solving equations over real numbers" *Cryptology ePrint Archive*, p. 94, 2012.
[11] J. Bi and L. Han, "Lattice Attacks on NTRU Revisited," *IEEE Access*, vol. 9, pp. 66218–66222, 2021, doi: 10.1109/ACCESS.2021.3076598.
[12] H. Arabnezhad-Khanoki, B. Sadeghiyan, and J. Pieprzyk, "S-boxes representation and efficiency of algebraic attack," *IET Inf Secure*, vol. 13, no. 5, pp. 448–458, Sep. 2019, doi: 10.1049/iet-ifs.2018.5201.
[13] A. Askeland and S. Rønjom. "A Side-Channel Assisted Attack on NTRU," *IACR Cryptol*. ePrint Arch., p. 790, 2021.
[14] A. Abdel-Hafez, R. A. Elbarkouky and W. Hafez, "Comparative Study of Algebraic Attacks," *International Advanced Research Journal in Science, Engineering and Technology*, vol. 3, pp. 85-90, 2016.
[15] C. Mascia, E. Piccione and M. Sala, "An algebraic attack on stream ciphers with application to nonlinear filter generators and WG-PRNG," *arXiv*, Cornell University, Dec. 2021, doi: 10.48550/arXiv.2112.12268.