New Public-Key Cryptosystem
(First Version)

Dieaa I. Nassr, M. Anwar, Hatem M. Bahig
Department of Mathematics, Faculty of Science, Ain Shams University, Egypt
dieaa.nassr@sci.asu.edu.eg (diaa.rsa@gmail.com)
mohmmed.anwar@hotmail.com
hmbahig@sci.asu.edu.eg (h.m.bahig@gmail.com)

January 25, 2021

Abstract

In this article, we propose a new public key cryptosystem, called **NAB**. The most important features of NAB are that its security strength is no easier than the security issues of the NTRU cryptosystem [8] and the encryption/decryption process is very fast compared to the previous public key cryptosystems RSA [17], Elgamal [6], NTRU [8]. Since the NTRU cryptosystem [8] is still not known to be breakable using quantum computers, NAB is also the same. In addition, the expansion of the ciphertext is barely greater than the plaintext and the ratio of the bit-size of the ciphertext to the bit-size of the plaintext can be reduced to just over one. We suggest that NAB is an alternative to RSA [17], Elgamal [6] and NTRU [8] cryptosystems.

Keywords: public-key cryptosystem, lattice-based, the shortest vector problem, the closest vector problem

1 Introduction

The public key cryptosystems currently in use have many weaknesses in previous years, due to their old age. The RSA [17] and Elgamal [6] cryptosystems were announced in 1978 and 1985 respectively. RSA and Elgamal have became the most widely used public key cryptosystems that due to the simplicity of their mathematical background, constitute only a simple background in number theory. However, the security of the RSA depends on the difficulty of solving the factoring problem, as well as the security of Elgamal depends on the difficulty of solving the discrete logarithm problem. In [19], Shor et al. showed that these two problems can be solved using quantum computers. Additionally, the security of RSA and Elgamal depends on the quality of the choice of the parameters used in these two systems to avoid some of the weaknesses may be used to break the systems [3, 7]. Therefore, the designers of these cryptosystems must be sufficiently aware of all aspects of that weaknesses have appeared in these cryptosystems in recent years. As for performance, RSA and Elgamal require heavy calculations (power) during the encryption/decryption process. In RSA, the plaintext m and ciphertext c are two integers in Z_n. Therefore, RSA has $1/1$ message expansion and this is one of the most advantage of RSA. However, in Elgamal, the plaintext is an integer $m \in Z_p$, while the ciphertext has two integers $c_1, c_2 \in Z_p$. Therefore, message expansion in Elgamal cryptosystem is $2/1$.
NTRU is a relatively new public key cryptosystem that appears to be more efficient than the currently more

widely used public key cryptosystems, RSA [17] and ElGamal [6]. NTRU is still not known to be breakable using quantum computers. The security of NTRU relies on the difficulty of finding a non-zero vector of a smallest norm in some given lattice, solving *the shortest vector problem* [1, 13, 9]. Also, its security depends on the difficulty of getting a vector in a given lattice that is a closest to some known vector, solving *the closest vector problem (CVP)* [4, 9, 12]. The operations of NTRU are taken place over the polynomial rings $R = Z[x]/(x^n - 1)$, $R_p = Z_p[x]/(x^n - 1)$ and $R_q = Z_q[x]/(x^n - 1)$ with assumptions [8, 10, 9], p is very small relatively to q and $GCD(p, q) = 1$. NTRU has message expansion $\log q/\log p$ and this is one of the drawbacks of the NTRU.
Any new proposed public key cryptosystem must be secure even if quantum computers are used in the future, just as it should be more efficient than its predecessors. Because NTRU cryptosystem is still not breakable using quantum computers, we therefore rely on what was presented during the study of the security of NTRU. We demonstrate that the security of the proposed public key cryptosystem NAB is no less powerful than that of the NTRU. The message expansion in the proposed public key cryptosystem NAB is just over one. The necessary mathematical background of NAB is simple, as it depends on a simple background to add and multiply matrices (vectors), and therefore the speed of encryption/decryption process in NAB is very fast compared to RSA, Elgamal and NTRU cryptosystems, also it can be easy implemented in different platforms. But the downside of NAB is that its key size is larger than that of RSA, Elgamal and NTRU.
The paper is organized as follows. Section 2 presents definitions and basic concepts that are needed to NAB and describes the NTRU cryptosystem. We present NAB in Section 3. We study the security of NAB in Section 4. Section 5 contains the experimental results. Finally, Section 6 includes the conclusion.

2 Preliminaries

Since we show that NAB cryptanalysis is no less difficult than NTRU cryptanalysis, we provide a review of NTRU. Also, through the key creation and ecnryption/decryption processes of NAB we need to perform some matrix operations, therefore, we give some facts (rules) on matrix operations.

2.1 NTRU

We specify Z_p as a ring of integers modulo p that contains the integers in the interval $(-p/2, p/2]$. Similarly for Z_q. Let R, R_p and R_q be the rings of all polynomials with a degree less than n and integral coefficients in Z, Z_p, and Z_q, respectively. i.e., $R = Z[x]/(x^n - 1)$, $R_p = Z_p[x]/(x^n - 1)$ and $R_q = Z_q[x]/(x^n - 1)$. It is generally useful to use the polynomial $f = \sum_{i=0}^{n} f_i x^i \; inR$ (R_p or R_q) as the vector of its coefficients $(f_0, f_1, ..., f_n) \in Z^n$ (Z_p^n or Z_q^n).
The NTRU assumptions [8, 9] are that p is very small relative to q and $\gcd(p, q) = 1$. Let $T(d_1, d_2)$ be the set of all polynomials with coefficients in $\{-1, 0, 1\}$, where each polynomial has d_1 coefficients equal to 1, d_2, coefficients equal to -1 and all other coefficients equal to 0. For some selected security parameters d_f and d_g, a private key is generated by selecting two private polynomials $f \in T(d_f + 1, d_f)$, $g \in T(d_g, d_g)$, where the polynomial f has inverses f_p, f_q in R_p and R_q, respectively, i.e., $f \cdot f_p \equiv 1 \pmod p$ and $f \cdot f_q \equiv 1 \pmod q$. Then (f, g) is the private-key and the corresponding public-key is given by $h = f_q \cdot g \pmod q$.
Let $m \in R_p$ be the plaintext, the corresponding ciphertext c is computed by $c = E_h(m) = ph \cdot r + m \pmod q$, where r is chosen randomly in $T(d_m, d_m)$, for some chosen security parameters d_m.
The pliantext m can be restored from c as follows:

1. set $t = c \cdot f \pmod q$.

2. adjust the coefficients of t in the interval $(\frac{q}{2}, q]$.

3. get the plaintext $m = t \cdot f_p \pmod p$.

In [8], Hoffstein et al. studied various attacks on NTRU. They showed that brute force and that meet-in-the-middle attacks might be used to recover the private key or against a single message, but will not be successful in a reasonable time. Coppersmith and Shamir [5] explained the lattice attacks on NTRU. These attacks have mainly focused on the private-key recovery problem: find the private-key (f, g) giving only the public-key h and public information about how f and g are selected. Coppersmith and Shamir presented a lattice of dimension $2n \times 2n$ constructed from the rotation matrix of the public-key h and explained that the problem of recovering the private polynomials f and g from h is reduced to SVP. They declared that their attack is efficient to break the system if the NTRU parameters are poorly chosen. They showed that the smaller value of the ratio $c = \sqrt{\frac{2\pi e\|f\|\|g\|}{nq}}$ the greater chance to recover the private-key (f, g), where $\|\cdot\|$ is the Euclidean norm. May [11] showed that if g has a long zero pattern somewhere in its coordinate list, it is better to create the lattice generated by the rotation matrix of the public-key h after multiplying a few consecutive columns of the rotation matrix of h with an integer $\theta > q$. May [11] also shortens the lattice dimension in the attack by Coppersmith and Shamir from $2n$ to $(1+\delta)n$, $0 < \delta \leq 1$. With that lattice (he called it *zero-run lattice*), the time to restore the private-key from the public-key is reduced by a factor of 10 if the security level is low. Since the NTRU creator can skip using long consecutive zero coefficients in g, Silverman [20] made an improvement to the attack of May [11] and suggested to multiply k random columns of the rotation matrix of h by θ. Silverman [20] showed that this suggestion is better than multiplying k consecutive columns by θ. The lattice created by Silverman is called *forced-zero lattice*.
In [16], Nitaj studied NTRU with two different public-keys h and h', which are connected by two corresponding different private keys (f, g) and (f', g') for everyone. Nitaj showed that if the ratio $c' = \sqrt{\frac{2\pi e\|f\|\|g-g'\|}{nq}}$ is as small as possible, we have a great chance of breaking the system.
Bahig et al. [2] generalized the cryptanalysis of NTRU [11, 20] when g has one or more patterns of zero coefficients at various positions (not necessarily of the same length).

2.2 Matrix Operations

We give some facts (rules) on matrix operations. Let X_i and Y_i be matrices where each of dimension $k \times k$, their components in Z_q and $X_i \times Y_i = I \mod q$, $i = 1, \cdots, k$.

1. If $X = X_0 \times X_1 \times \cdots \times X_k \pmod q$ and $Y = Y_k \times Y_{k-1} \times \cdots \times Y_0 \pmod q$ then $X \times Y = I \pmod q$

2. let Y_i' be the transpose of Y_i, then the transpose of Y can be given by $Y = Y_0' \times Y_1' \times \cdots \times Y_k' \pmod q$

3. define $SwapTwoRows(X_i, n, m)$ to return the matrix X_i after swapping the n^{th} row with the m^{th}. Also, $SwapTwoCols(Y_i, n, m)$ to return the matrix Y_i after swapping the n^{th} column with the m^{th}. If $W_i = SwapTwoRows(X_i, n, m)$ and $Z_i = SwapTwoCols(Y_i, n, m)$, then $W_i \times Z_i = I \pmod q$.

4. Define $AddTwoRows(X_i, n, m)$ to return the matrix X_i after adding the the m^{th} row to the n^{th}. Also, $SubtractTwoCols(Y_i, n, m)$ to return the matrix Y_i after subtracting the the m^{th} row from the n^{th}. If $W_i = AddTwoRows(X_i, n, m)$ and $Z_i = SubtractTwoCols(Y_i, m, n)$, then $W_i \times Z_i = I \pmod q$.

Based on these facts, the following algorithm can be used to compute the inverse of a given square matrix A over Z_q where $GCD(|A|, q) = 1$. This algorithm can be used to speed up the creation of the NAB key.

Algorithm 2.1. ***Computing the inverse of a given matrix*** $\pmod q$
Input: *a matrix A on Z_q and of dimension $k \times k$.*
Output: *the inverse matrix B, where $A \times B = I \pmod q$.*
Begin

set B to be the identity matrix of dimension $k \times k$.
for $i = 0$ to $k - 1$ **do**
    find $j \geq i$ where $GCD(A_{ji}, q) = 1$, if there is no such j, then the algorithm fails to obtain the inverse.
    if $i \neq j$ **then** then
        $SwapTwoRows(A, i, j)$
        $SwapTwoRows(B, i, j)$
    end if
    $f = (A_{ii})^{-1} \pmod q$
    define $A(i)$ and $B(i)$ to be the i^{th} row of A and B, respectively
    $A(i) = fA(i) \pmod q$ ▷ *mutliply the i^{th} row of A by f*
    $B(i) = fB(i) \pmod q$ ▷ *mutliply the i^{th} row of B by f*
    for $j = i + 1$ to $k - 1$ **do**
        $g = A_{ji} \pmod q$
        $A(j) = A(j) - gA(i) \pmod q$
        $B(j) = B(j) - gB(i) \pmod q$
    end for
end for
for $i = k - 1$ downto 0 **do**
    $f = (A_{ii})^{-1} \pmod q$
    define $A(i)$ and $B(i)$ to be the i^{th} row of A and B, respectively
    $A(i) = fA(i) \pmod q$ ▷ *mutliply the i^{th} row of A by f*
    $B(i) = fB(i) \pmod q$ ▷ *mutliply the i^{th} row of B by f*
    for $j = i - 1$ downto 0 **do**
        $g = A_{ji} \pmod q$
        $A(j) = A(j) - gA(i) \pmod q$
        $B(j) = B(j) - gB(i) \pmod q$
    end for
end for
End

3 The Proposed public Key Cryptosystem NAB

The public key is summarized as two relatively prime integers p and q, $(p < q)$ and two public matrices $(k \times k)$ E_1 and E_2 their components in Z_q. The two matrices E_1 and E_2 are constructed from the random matrices B, T_1 and T_2 where $\gcd(|B|, q) = 1$ and the components of T_1 and T_2 in $\{-1, 0, 1\}$ with $\gcd(|T_1|, p) = \gcd(|T_2|, q) = 1$. We define $E_1 = B \times T_1 \pmod q$ and $E_2 = B \times T_2 \pmod q$. The private key is $(B, B^{-1} \pmod q, T_1, T_1^{-1} \pmod p, T_2, T_2^{-1} \pmod q)$ and hidden in E_1 and E_2. If p is chosen of bit-size l, then q is selected of bit-size greater than $2l + \log k - 1 + \epsilon$. The parameters l and k are determined according to the required level of the proposed system security.

Algorithm 3.1. ***Key Creation***
Input:

$k > 2$ is the number of rows (columns) in the generated public key.

l is a positive integer (security parameter).

Output: *The public and private keys of NAB cryptosystem.*
Begin

1: *generate randomly a positive integer q of bit-size greater than $2l + \log k - 1 + \epsilon$*
2: *generate randomly a positive integer p of bit-size l with $GCD(p, q) = 1$.*
3: *generate two random matrices T_1 and T_2 each of dimension $k \times k$ where their components are selected from $\{-1, 0, 1\}$ and $GCD(|T_1|, p) = GCD(|T_2|, q) = 1$.*
4: *compute $T_1^{-1} \pmod p$ and $T_2^{-1} \pmod q$*
5: *generate a random matrix B of dimension $k \times k$ where their components are selected from Z_q and $GCD(|B|, q) = 1$.*
6: *compute $B^{-1} \pmod q$*
7: *compute $E_1 = B \times T_1 \pmod q$ and $E_2 = B \times T_2 \pmod q$.*
8: *the public key e is (E_1, E_2, p, q).*
9: *the private key d is $(B, B^{-1} \pmod q, T_1, T_1^{-1} \pmod p, T_2, T_2^{-1} \pmod q)$*

End

The plaintext $m = (m_0, \ldots, m_{k-1})$ is a vector of k components in Z_q, where each component is of bit-size $2(l-2)$. The plaintext m can written as $m = u + v2^{l-2}$, where u and v are two vectors their components of biz-size $l-2$ and belong to Z_p. The ciphertext is just computed by the matrix-vector multiplication modulo q as:

$$c = pE_2 \times v + E_1 \times (u + v) \pmod q.$$

The overall complexity of the matrix-vector multiplication is $\theta(k^2)$. Note that matrix-vector multiplication can be implemented in parallel to reduce the time complexity of NAB encryption.

Algorithm 3.2. ***Encryption***
Input:

A public key of a NAB cryptosystem $e = (E_1, E_2, p, q)$, where q and p are relatively prime and of bit-size $2l + \log k - 1 + \epsilon$ and l, respectively.

$m = (m_0, \ldots, m_{k-1})$ is the plaintext, where m_i is of bit-size $2(l-2)$ and written as $m_i = u_i + v_i(2^{l-2})$, (note that $0 \leq u_i, v_i < 2^{l-2} < p$).

Output: *The ciphertext c.*
Begin

1: *define $u = (u_0, u_1, \ldots, u_{k-1})$ and $v = (v_0, v_1, \ldots, v_{k-1})$.*
2: *compute $c = pE_2 \times v + E_1 \times (u + v) \pmod q$.*

End

If the ciphertext c is the output of Algorithm 3.2, we simply write $c = Enc_e(m)$.
The ciphertext $c = (c_0, \ldots, c_{k-1})$ is a vector of k components in Z_q. Obtaining the plaintext $m = u + v2^{l-2}$ from c begins by multiplying c by B^{-1} modulo q to get $R_1 = pT_1 \times v + T_2 \times (u + v) \pmod q$. We select q of

bit-size $2l + \log k - 1 + \epsilon$ to make the center lift of R_1 (make the components of R_1 in the interval $(-q/2, q/2]$) gives $pT_1 \times v + T_2 \times (u + v)$ (Over Z the set of all integers). Then, getting u and v becomes simple and described in Algorithm 3.3 that has a time complexity $\theta(k^2)$.

Algorithm 3.3. ***Decryption***
Input:

The private key of the NAB cryptosystem $d = (B, B^{-1} \pmod q, T_1, T_1^{-1} \pmod p, T_2, T_2^{-1} \pmod q)$.

$c = (c_0, c_1, \ldots, c_k)$ *is the ciphertext, where* $c_0, c_1, \ldots, c_{k-1} \in Z_q$.

Output: *The plaintext* m.
Begin
1: *define* $B' = B^{-1} \pmod q$ *and* $T_1' = T_1^{-1} \pmod p$ *and* $T_2' = T_2^{-1} \pmod p$
2: $R_1 = B' \times c \pmod q$, *and center lift* R_1 *in* Z_q *(make the components of* R_1 *in the interval* $(-q/2, q/2]$.
3: $R_2 = R_1 \pmod p$
4: $R_3 = T_1' R_2 \pmod p$
5: $v = (pE_2)^{-1}(c - E_1 \times R_3) \pmod q$
6: $u = R_3 - v$
7: *the plaintext is* $m = u + v(2^{l-2})$
End

If the plaintext m is the output of Algorithm 3.3, we simply write $m = Dec_d(c)$. In the following theorem, we prove the correctness of NAB cryptosystem.

Theorem 3.4. *Let* $e = (E_1, E_2, p, q)$ *and* $d = (T_1, T_2, B)$ *be the public and private keys of a NAB public key cryptosystem. Then* $m = Dec_d(Enc_e(m))$ *for every* $m = (m_0, \ldots, m_{k-1})$*, where* m_i *is of bit-size* $2(l-2)$.

Proof. Since m_i is of bit-size $2(l-2)$, it can be written as $m_i = u_i + v_i(2^{l-2})$, where u_i and v_i are of bit-size $l-2$. Define $u = (u_0, u_1, \ldots, u_{k-1})$, and $v = (v_0, v_1, \ldots, v_{k-1})$, therefore, the plaintext m can be written as $m = u + v(2^{l-2})$.
Define $c = Enc_e(m)$, therefore, $c = pE_2 \times v + E_1 \times (u + v) \pmod q$.
We need to show that $m = Dec_d(c)$. In Algorithm 3.3,

$$\begin{aligned} R_1 &= B' \times c \pmod q \\ &= pT_2 \times v + T_1 \times (u + v) \pmod q. \end{aligned}$$

Since the positive components of $pT_2 \times v + T_1 \times (u + v)$ is of bit-size at most $2l + \log k - 2 + \epsilon$, these positive components are smaller than $q/2$. Similarly the negative components of $pT_2 \times v + T_1 \times (u + v)$ are greater than $-q/2$. Therefore, the center left of R_1 in Z_q gives $pT_2 \times v + T_1 \times (u + v)$. i.e., R_1 in Algorithm 3.3 gives $pT_2 \times v + T_1 \times (u + v)$.
Therefore, $R_2 = R_1 \pmod p = T_1 \times (u + v) \pmod p$. Thus, $R_3 = T_1' R_2 \pmod p = u + v \pmod p$.Since the components u_i and v_i are smaller than $p/2$, we ensure that $u + v \pmod p$ is itself $u + v$. i.e., $R_3 = u + v$. This leads to

$$\begin{aligned} v &= (pE_2)^{-1}(c - E_1 \times R_3) \pmod q \\ u &= R_3 - v. \end{aligned}$$

We get the plaintext $m = u + v(2^{l-2})$. □

We need to process the plaintext before the encryption where we aim to make the ciphertext appear to be not fixed and random for each plaintext. In addition, we use a checksum function to simply ensure the integrity of the plaintext. Therefore, we consider the original plaintext just as a vector of $k-2$ components in Z_q, where each component is of bit size $2(l-2)$, and pad this plaintext with two additional components, the first is random and the second is a checksum of all other plaintext components. i.e., if the plaintext is $m = (m_0, m_1, ..., m_{k-3})$, then we pad the two components m_{k-2} and m_{k-1} where m_{k-2} is selected to be a random value of size $2(l-2)$-bit and m_{k-1} is defined as $checksum(m_0, m_1, ..., m_{k-2})$,

$$checksum(m_0, m_1, ..., m_{k-2}) = \left(\sum_{i=0}^{k-2} (i+1+p) m_i^2 \pmod q \right) \pmod{2^{2(l-2)}} \tag{1}$$

Therefore, we give the following encryption algorithm with padding.

Algorithm 3.5. ***Encryption with Padding***
Input:

A public key of a NAB cryptosystem $e = (E_1, E_2, p, q)$, *where* q *and* p *are relatively prime and of bit-size* $2l + \log k - 1 + \epsilon$ *and* l, *respectively.*

$(m_0, \ldots, m_{k-3})$ *is the plaintext, where* m_i *is of bit-size* $2(l-2)$.

Output: *The ciphertext* c.
Begin

1: *select a random value* m_{k-2} *of bit-size* $2(l-2)$
2: *define* $m_{k-1} = checksum(m_0, m_1, ..., m_{k-2})$ *(Equation 1)*
3: *pad* m_{k-2} *and* m_{k-1} *to the end of the plaintext. i.e., define* $m = (m_0, \ldots, m_{k-3}, m_{k-2}, m_{k-1})$ *where each* m_i *is written as* $m_i = u_i + v_i(2^{l-2})$, *(note that* $0 \le u_i, v_i < 2^{l-2} < p$*).*
4: *define* $u = (u_0, u_1, \ldots, u_{k-1})$ *and* $v = (v_0, v_1, \ldots, v_{k-1})$.
5: *compute* $c = pE_2 \times v + E_1 \times (u+v) \pmod q$.

End

Through the decryption process, we can ensure the integrity of the decryption result by verifying that the k^{th} component of the decryption result is the checksum of the first $k-1$ components. Therefore, we give the following decryption algorithm to ensure the integrity of plaintext.

Algorithm 3.6. ***Decryption with plaintext integrity***
Input:

The private key of the NAB cryptosystem $d = (B, B^{-1} \pmod q, T_1, T_1^{-1} \pmod p, T_2, T_2^{-1} \pmod q)$.

$c = (c_0, c_1, \ldots, c_k)$ *is the ciphertext, where* $c_0, c_1, \ldots, c_{k-1} \in Z_q$.

Output: *The plaintext* m *or plaintext error.*
Begin

1: *define* $B' = B^{-1} \pmod q$ *and* $T_1' = T_1^{-1} \pmod p$ *and* $T_2' = T_2^{-1} \pmod p$
2: $R_1 = B' \times c \pmod q$, *and center lift* R_1 *in* Z_q *(make the components of* R_1 *in the interval* $(-q/2, q/2]$.
3: $R_2 = R_1 \pmod p$
4: $R_3 = T_1' R_2 \pmod p$

Table 1: Performance Comparision

Runtime	NAB	NTRU	RSA
Encryption	$O(n^2)$	$O(n^2)$	$O(n^2)$
Decryption	$O(n^2)$	$O(n^2)$	$O(n^3)$
Message expansion	$1 + \frac{\log k + 3 + \epsilon}{2l-4} \approx 1$	$log(q)/log(p) >> 2$	1

$v = (pE_2)^{-1}(c - E_1 \times R_3) \pmod q$
$u = R_3 - v$
set $m = u + v(2^{l-2})$, *m is a vector of k components in* Z_q, *let* $m = (m_0, m_1, \ldots, m_{k-1})$,
if $m_{k-1} = checksum(m_0, m_1, \ldots, m_{k-2})$ ***then***
    return the plaintext m
else
    return plaintext error.
end if
End

Let n be the bit-size of plaintext. We give in Table 1 some of the performance characteristics of the RSA, NTRU and NAB cryptosystems. In each case n is the bit-size of plaintext.

4 Security

4.1 Brute Force Attack

One of the possible attacks on the proposed system is that an adversary tries to search for one of the private matrices T_1, T_2 or B_1. Obtaining one of them is therefore equivalent to obtaining the private key. Therefore, if an adversary uses brute force search, it is easier to search for T_1 or T_2 than to search for B because each component of T_1 and T_2 has only three possibilities, either 0, 1 or -1 while each component of B is a number in Z_q. Therefore, we assume that an adversary searches for all possibilities for T_1 (or T_2). We try to estimate a lower bound for the number of all possible matrices for T_1 (or T_2). For simplicity, let p be a prime number and $p \geq 3$. The number of all matrices (of dimension $k \times k$) whose component is either 0, 1 or -1 and invertible over Z_p is greater greater than or equal to the number of invertible matrices (of dimension $k \times k$) over Z_3. Note that the number of all matrices of dimension $k \times k$ and invertible over Z_3 is given by (see [15])

$$\prod_{i=0}^{k-1}(3^k - 3^i) > 3^{k(k-1)} > 2^{3k(k-1)/2}$$

Therefore, the number of all possible matrices for T_1 (or T_2) is greater than $2^{3k(k-1)/2}$. So if $k = 12$, the number of all possible matrices for T_1 (or T_2) will be greater than 2^{198}.

4.2 Chosen Ciphertext Attack (CCA)

In the chosen ciphertext attack (CCA), we assume that an adversary has access to an oracle machine that returns the plaintext for any suggested ciphertext except a given challenge ciphertext c. In order to avoid CCA's success on the proposed system, we have proposed Algorithms 3.5 and 3.6 for encryption and decryption, respectively. The encryption algorithm (Algorithm 3.5) causes the ciphertext to appear randomly each time. In addition, the decryption algorithm (Algorithm 3.6) ensures the integrity of the plaintext by using the checksum (Equation 1). Therefore, any modification of the ciphertext affects the integrity of the result of the decryption algorithm. Thus, CCA may be inefficient in the case of using Algorithms 3.5 and 3.6 for encryption and decryption, respectively.

4.3 Key Recovery Attack

We try to prove that the security strength (cryptanalysis) of NAB is no less difficult than the cryptanalysis of NTRU. Therefore, we review the problems on which the cryptanalysis of NTRU depends. In 1997, Coppersmith and Shamir [5] explained how NTRU private-key recovery problem 4.1 can be formulated as the shortest vector problem SVP [1, 9, 13] and how NTRU plaintext recovery problem 4.2 can be formulated as the closest vector problem CVP [4, 9, 12], problem in a certain special sort of lattice. Both SVP and CVP are computationally difficult as the lattice dimension grows and they are known to be NP-hard under a certain hypothesis, see [14]. In practice, CVP is considered to be a little bit harder than SVP, since CVP can often be reduced to SVP in a slightly higher dimension, see [18].

Problem 4.1. *(NTRU Private Key Recovery Problem) Given NTRU public key* $h \in \frac{Z_q[x]}{x^k-1}$ *(The quotient ring of* $Z_q[x]$ *modulo* (x^k-1)*): Find ternary polynomials* $f, g \in \frac{Z_q[x]}{x^k-1}$ *satisfying* $f * h \equiv g \pmod q$.

Problem 4.2. *(NTRU Plaintext Recovery Problem) Let* p *and* q *be two integers (modulus) used in a given NTRU public key* $h \in \frac{Z_q[x]}{x^k-1}$ *where* p *is small relatively to* q*. Given ciphertext* $c \in \frac{Z_q[x]}{x^k-1}$*: Find* $m, r \in \frac{Z_p[x]}{x^k-1}$ *(their coefficients in* Z_p*) satisfying* $c = p.h * r + m \pmod q$.

In the following two Problems 4.3, 4.4, we formulate the private key recovery and the plaintext recovery problems of NAB. Therefore, to confirm the security of our suggested cryptosystem, we prove in Theorems 4.5, 4.6 that the two Problems 4.3, 4.4 are not easier than Problems 4.1, 4.2, respectively.

Problem 4.3. *(NAB Private Key Recovery Problem) Let* $e = (E_1, E_2, p, q)$ *be a public key of a NAB cryptosystem where* E_1 *and* E_2 *are of dimension* $k \times k$*. Find a matrix* B *where* $T_1 = B \times E_1 \pmod q$ *and* $T_2 = B \times E_2 \pmod q$ *have their components in* $\{-1, 0, 1\}$.

Problem 4.4. *(NAB Plaintext Recovery Problem) Let* $e = (E_1, E_2, p, q)$ *be a public key of a NAB cryptosystem where* E_1 *and* E_2 *are of dimension* $k \times k$*. Given ciphertext* $c = (c_0, c_1, \ldots, c_{k-1})$*,* $c_i \in Z_q$*. Find* $u = (u_0, u_1, \ldots, u_k)$ *and* $v = (v_0, v_1, \ldots, v_k)$ *with* $0 \leq u_i, v_i < 2^{l-2} < p$ *satisfying* $c = pE_2 \times v + E_1 \times (u + v) \pmod q$.

In the following theorem, we prove that recovering NAB's private-key from its corresponding public-key is no easier than recovering NTRU's private-key from its public-key.

Theorem 4.5. *Problem 4.3 is not easier than Problem 4.1.*

Proof. Suppose that we have an algorithm (Algorithm-A) can be used to solve problem 4.3 in polynomial time. Therefore, by giving E_1 and E_2 modulo q, as an input to Algorithm-A, this algorithm gets a matrix $M = B^{-1} mod q$ in polynomial time such that the components of $T_1 = M \times E_1 \pmod q$ and $T_2 = M \times E_2 \pmod q$ are in $\{-1, 0, 1\}$.
We directly transform (in polynomial time) the parameters given in Problem 4.1 to be a special form of the parameters given in Problem 4.3. Let $h = (h_0, h_1, \ldots, h_{k-1}) \in \frac{Z_q[x]}{x^k-1}$ be an NTRU public key with its corresponding private key $f = (f_0, f_1, \ldots, f_{k-1})$ and $g = (g_0, g_1, \ldots, g_{k-1})$, where $f * h = g \pmod q$. Define H, F and G to be the rotation matrices of h, f and g, respectively. From the definition of $f * h = g \pmod q$, we have

$$F \times H = G \pmod q$$

Therefore, if we define the parameters $E_1 = H$ and $E_2 = I$ modulo q as an input to Algorithm-A, then this algorithm can find M such that the components of $T_1 = M \times H \pmod q$ and $T_2 = M \times I = M \pmod q$ are in $\{-1, 0, 1\}$. Therefore, each row of T_2 with its corresponding row in T_1 can be selected to represent two ternary polynomials f, g, respectively, satisfying $f * h = g \pmod q$. □

In the following theorem, we prove that recovering NAB's plaintext from its corresponding ciphertext is no easier than recovering NTRU's plaintext from its ciphertext.

Theorem 4.6. *Problem 4.4 is not easier than Problem 4.2.*

Proof. Suppose that we have an algorithm (Algorithm-B) can be used to solve problem 4.4 in polynomial time. Therefore, by giving a public key $e = (E_1, E_2, p, q)$ of a NAB cryptosystem and a ciphertext $c = (c_0, c_1, \ldots, c_{k-1})$, $c_i \in Z_q$ this algorithm gets $u = (u_0, u_1, \ldots, u_k)$ and $v = (v_0, v_1, \ldots, v_k)$ with $0 \leq u_i, v_i < 2^{l-2} < p$ satisfying $c = pE_2 \times v + E_1 \times (u + v) \pmod q$ in polynomial time.
We directly transform (in polynomial time) the parameters given in Problem 4.2 to be a special form of the parameters given in Problem 4.4.
Let p and q be the two integers (modulus) used in the encryption process with the NTRU public key $h = (h_0, h_1, \ldots, h_{k-1}) \in \frac{Z_q[x]}{x^k-1}$, where p is small relatively to q. Let $c \in \frac{Z_q[x]}{x^k-1}$ be the given ciphertext. From $c = p.h * r + m \pmod q$, define E_2 to be the rotation matrix of h and $E_1 = I$. Therefore, the ciphertext c can be expressed by $c = pE_2 \times r + E_1 \times m \pmod q$. Therefore, if we get the parameters c, E_1 and E_2 as inputs to Algorithm-B, then this algorithm can find $r = (r_0, r_1, \ldots, r_{k-1})$ and $m - r = (m_0 - r_0, m_1 - r_1, \ldots, m_{k-1} - r_{k-1})$ such that $m_i, r_i \in Z_p$. Therefore, $m = (m_0, m_1, \ldots, m_{k-1})$ and $r = (r_0, r_1, \ldots, r_{k-1})$ can be used to represent two polynomials in $\frac{Z_p[x]}{x^k-1}$, satisfying $c = p.h * r + m \pmod q$. □

5 Experiments

We present in Table 2 some of suggested sets of parameters that may be used to setup NAB cryptosytems based on levels of required security. In Table 3, we give the speed benchmarks for our implementation of NAB. Our implementation is coded in C++, compiled with Microsoft Visual C++ .NET 2010 and ran on a GenuineIntel 2600 Mhz under windows 10 operating system. The size of data used for encryption is 100 MB. The time is given in seconds.

Table 2: NAB Parameter Sets

Set	l (bit-size of p)	k	bit-size of q	NAB key size (K.Byte)	NAB Message expansion
A	10	12	24	0.845	1.5
B	10	14	24	1.149	1.5
C	14	16	24	1.5	1.5
D	18	18	24	1.899	1.5

Table 3: NAB Run Time

set	Key Creationn	Encryption	Decryption
A	0	65.5403	121.0090
B	0.003	71.1960	135.8805
C	0.008	59.7874	108.7359
D	0.01	91.0616	154.4752

6 Conclusion

The advantage of NAB is that its performance (key creation, encryption, decryption) is very fast compared to RSA, Elgamal and NTRU cryptosystems. Also, NAB security strength is not easier than the security of NTRU. Therefore, if NTRU is considered secure against quntum computer, then NAB is the same. Additionally, NAB needs some simple mathematical background in matrix algebra for its implementation and it can be implemented simply in different platforms. Message expansion in NAB is smaller than message expansion in NTRU and Elgamal. The disadvantage of NAB is its message expansion is not optimal like RSA but can be optemized to be just over one. Also, its key-size depends on matrix storage, therefore, the key-size of NAB is greater than the key-size of RSA, Elgamal and NTRU. We suggest that NAB is an alternative public key cryptosystem for RSA, Elgamal and NTRU cryptosystems.

References

[1] M. Ajtai, *Optimal lower bounds for the korkine-zolotareff parameters of a lattice and for schnorr's algorithm for the shortest vector problem*, Theory of Computing **4** (2008), no. 1, 21–51.

[2] Hatem M. Bahig, Ashraf M. Bhery, and Dieaa I. Nassr, *Cryptanalysis of ntru where the private polynomial has one or more consecutive zero coefficients*, Journal of Discrete Mathematical Sciences and Cryptography **0** (2019), no. 0, 1–21.

[3] D. Boneh, *Twenty years of attacks on the RSA cryptosystem*, Notices of the AMS **46** (1999), 203–213.

[4] W. Chen and J. Meng, *The hardness of the closest vector problem with preprocessing over ℓ_∞ norm*, IEEE Trans. Information Theory **52** (2006), no. 10, 4603–4606.

[5] D. Coppersmith and A. Shamir, *Lattice attacks on NTRU*, Advances in Cryptology EUROCRYPT 97, Lecture Notes in Computer Science, vol. 1233, Springer, 1997, pp. 52–61.

[6] T. El-Gamal, *A public key cryptosystem and a signature scheme based on discrete logarithms*, IEEE Transactions on Information Theory **31** (1985), no. 4, 469–472.

[7] J. Hinek, *Cryptanalysis of RSA and its variants*, Chapman & Hall/CRC Cryptography and Network Security Series, CRC Press, 2009, [Online]. Available: http://books.google.com.eg/books?id=LAxAdqv1z7kC.

[8] J. Hoffstein, J. Pipher, and J. Silverman, *NTRU: A ring-based public key cryptosystem*, Algorithmic Number Theory, Lecture Notes in Computer Science, vol. 1423, Springer, 1998, pp. 267–288.

[9] ______, *Introduction to mathematical cryptography*, vol. Undergraduate texts in mathematics, Springer, New York, 2014.

[10] J. Hoffstein, J. Silverman, and W. Whyte, *Estimated breaking times for NTRU lattices*, 2003, NTRU Cryptosystems Technical Report.

[11] A. May, *Cryptanalysis of NTRU-107*, 1999.

[12] D. Micciancio, *Closest vector problem*, Encyclopedia of Cryptography and Security, 2nd Ed., 2011, pp. 212–214.

[13] ______, *Shortest vector problem*, Encyclopedia of Algorithms, 2016, pp. 1974–1977.

[14] D. Micciancio and S. Goldwasser, *Complexity of lattice problems: a cryptographic perspective*, The Kluwer International Series in Engineering and Computer Science, vol. 671, Kluwer Academic, 2002.

[15] Kent Morrison, *Integer sequences and matrices over finite fields*, Journal of Integer Sequences **9** (2006).

[16] A. Nitaj, *Cryptanalysis of NTRU with two public keys*, International J. Network Security **16, No.2** (2014), 112–117.

[17] R. Rivest, A. Shamir, and L. Adleman, *A method for obtaining digital signatures and public-key cryptosystems*, Communications of the ACM **21** (1978), 120–126.

[18] M. Schneider and J. Buchmann, *Extended lattice reduction experiments using the BKZ algorithm*, Sicherheit, LNI, vol. 170, GI, 2010, pp. 241–252.

[19] P. Shor and W.Peter, *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, SIAM J. Comput. **26** (1997), no. 5, 1484–1509.

[20] J. Silverman, *Dimension-reduced lattices, zero-forced lattices, and the ntru public key cryptosystem*, Technical report 13, 1999, [Online]. Available: http://www.ntru.com.

Appendix

We give an example of NAB public/private key. The chosen parameters for this example $p = 997$, $q = 1569816417$, $k = 12$ and the two public matrices E_1 and E_2 are defined as follows:

$$E1 = \begin{pmatrix}
346378333 & 1506998010 & 1158592697 & 763237038 & 371322664 & 890771778 & 205041698 & 1477375343 & 1362167950 & 965148466 & 1408949995 & 1355443658 \\
1448448924 & 963645714 & 818715253 & 191197494 & 206785581 & 1464028532 & 1411094989 & 1415102743 & 387527256 & 938941991 & 125655093 & 757635453 \\
1090068744 & 27535634 & 633132532 & 729499765 & 906952983 & 479121430 & 1181577206 & 1415787567 & 623227470 & 219307795 & 1338673896 & 1317910750 \\
513018534 & 173346356 & 1274292116 & 1328991186 & 517513981 & 1564975556 & 130304344 & 651292759 & 81472167 & 52560322 & 917887418 & 242324450 \\
560658541 & 935720992 & 1278354437 & 1195933319 & 599350928 & 557850131 & 289344517 & 1082008659 & 241132847 & 1183007611 & 619038307 & 11907839 \\
985662470 & 320889076 & 1413540469 & 403759887 & 1075085644 & 1468853558 & 581475161 & 49793901 & 1194709672 & 1065413170 & 1443738060 & 1121552258 \\
463624907 & 190907440 & 74453960 & 1260783574 & 533316635 & 564071119 & 879830638 & 1498169604 & 857137306 & 1160699572 & 1162594458 & 1083542447 \\
74871099 & 1454624910 & 1200185496 & 810836025 & 960685183 & 204449626 & 86039933 & 1369025023 & 1070865891 & 319384926 & 26780244 & 420096508 \\
433677034 & 364407279 & 14867365 & 1483667512 & 1332885986 & 867710011 & 86130598 & 885623184 & 1163040432 & 593134817 & 302278440 & 1092185751 \\
158538057 & 1196925829 & 1383236813 & 552166463 & 383025120 & 971492194 & 1133591998 & 370591475 & 1038124520 & 679234204 & 1153287653 & 35075533 \\
1338809905 & 388894927 & 755239219 & 1207691077 & 256435631 & 1102501077 & 180956646 & 623977236 & 1454335612 & 279777424 & 1301080660 & 665257857 \\
619355059 & 969091848 & 554008868 & 100498089 & 705468446 & 921198463 & 732997298 & 339419965 & 917261532 & 212519999 & 949438992 & 459319391
\end{pmatrix}$$

$$E2 = \begin{pmatrix}
948836796 & 615103021 & 706260575 & 142259602 & 572563356 & 962093654 & 24558377 & 5498008 & 841413183 & 44583823 & 625977184 & 312395348 \\
744538730 & 409550130 & 479451745 & 1095277346 & 1515873877 & 1501651247 & 1180216840 & 1423919733 & 579654868 & 483423275 & 293770734 & 267408234 \\
1490085732 & 494775932 & 1234507854 & 1491959252 & 292360605 & 194606719 & 31626198 & 997867895 & 196642199 & 738187132 & 928286568 & 1136890584 \\
1337392927 & 1234915931 & 1228929352 & 1449836587 & 439784877 & 1130435083 & 294676109 & 1284711192 & 290260134 & 1129846830 & 291708300 & 1454795509 \\
540258069 & 330253603 & 323381288 & 738347634 & 1528183054 & 308295084 & 913175201 & 1567452467 & 1225516021 & 761663106 & 1352023103 & 573089872 \\
1479054880 & 921507251 & 1256686627 & 630149289 & 1055591009 & 625352812 & 1529483923 & 1106115452 & 342226583 & 1163942259 & 1100618208 & 1309716574 \\
667683303 & 677511941 & 1074765856 & 287036403 & 694053257 & 1202964046 & 245261465 & 1052105887 & 1325279631 & 325080109 & 1409954428 & 1483847545 \\
255095472 & 1348371068 & 1507239982 & 42775154 & 1384520577 & 1113218482 & 1113271416 & 378908655 & 711552876 & 1036574159 & 742624076 & 1224783652 \\
1333239492 & 292582481 & 1459946116 & 401370542 & 333925420 & 665299635 & 672431763 & 215428983 & 1338561338 & 1179800893 & 277599125 & 1460381405 \\
1324751585 & 92150361 & 477036161 & 1255635255 & 811353757 & 159699476 & 741795822 & 1408864210 & 1429535055 & 1007022628 & 1115059615 & 1339562016 \\
439530465 & 955485956 & 200858144 & 985727033 & 537158382 & 1474614124 & 1064970422 & 1177255335 & 665625437 & 513176893 & 1342363652 & 989307729 \\
601264177 & 1322278694 & 160631582 & 784384575 & 1027825831 & 815195463 & 1017564850 & 948009988 & 476352121 & 279804745 & 1149415978 & 214324897
\end{pmatrix}$$

The private matrices B, T_1, T_2, with their inverses $B^{-1} \pmod q$, $T_1^{-1} \pmod p$ and $T_2^{-1} \pmod q$ are defined as follows:

$B =$

1569280474	11501396	471959977	277474994	941096369	382486568	197271368	1183153892	520778386	861171260	1461015349	233124047
304348146	603689488	219438485	928033369	1117346764	990988177	1430769231	163587098	1323444363	1560560884	473751309	800400097
1453519647	445619114	1000052561	1416639823	1521214044	834111306	1541954661	636920411	1514703521	391437638	87299684	551527779
1325245673	1215327067	621592147	549887286	670282125	869588469	1035432673	1419148098	809273427	436378340	606248635	1047690416
967811593	857355873	592331673	1350482154	499380724	1358846141	1111258217	1293863975	1485407195	1223190968	1103649796	358098917
1158939524	13695754	238247038	441343282	522180823	1395078794	864691610	423621668	1077172009	1450015647	306623974	407316053
1150290497	302554171	1222287532	573793426	1387698258	430122760	354761580	1224414469	717250725	783646320	1146919664	977729827
672058606	1272227316	214520487	1201987118	146716893	1198191890	206594828	684655776	1069014031	9018565	1147063026	856555918
1224282453	1448676706	646631720	1281005884	1155669645	101465652	726169982	1208286228	1208514534	1318276053	406650194	4012807
1419582650	178637124	431922416	402425011	898176081	1492257684	33515371	44626906	1246919535	421390684	998168379	471973366
1057362051	1274491966	747211298	1495318427	1517934262	1221104248	401976617	683027253	846314939	1210094154	802860703	1157891382
849058700	414887321	357487556	325293457	1506770075	380221133	548147958	526544160	782230372	861823096	489345348	327008458

$T_1 =$

-1	0	-1	-1	-1	0	1	0	-1	1	-1	1
1	-1	1	0	1	0	-1	0	1	1	1	1
1	1	1	1	-1	0	1	-1	1	-1	1	-1
0	0	1	-1	1	-1	-1	-1	1	0	1	0
-1	-1	-1	-1	1	1	0	1	1	1	0	-1
1	1	0	-1	1	-1	-1	-1	0	0	0	-1
0	-1	-1	0	1	0	1	-1	-1	-1	1	-1
0	-1	-1	1	-1	1	-1	1	-1	0	-1	1
-1	-1	1	0	-1	1	-1	-1	1	-1	-1	0
-1	-1	1	-1	1	-1	0	1	1	1	-1	-1
0	1	0	0	-1	0	-1	-1	1	-1	-1	-1
1	-1	-1	0	-1	-1	1	1	-1	1	-1	-1

$T_2 =$

-1	-1	0	-1	1	-1	1	0	-1	0	-1	0
-1	1	-1	0	-1	-1	0	1	0	-1	-1	1
-1	-1	1	0	0	0	0	0	1	1	0	-1
1	0	-1	-1	1	-1	-1	1	0	1	-1	0
0	-1	1	1	-1	-1	-1	0	-1	-1	1	-1
1	0	-1	1	0	-1	1	-1	0	1	0	0
1	0	1	-1	-1	0	1	0	-1	-1	-1	-1
-1	1	1	0	0	1	1	-1	-1	1	-1	-1
-1	0	1	1	0	-1	0	-1	1	-1	1	-1
-1	-1	1	0	0	0	-1	0	-1	1	-1	-1
0	1	-1	-1	-1	-1	-1	0	1	-1	1	0
0	1	1	1	-1	1	1	1	0	0	1	-1

$B^{-1} \pmod q =$

1196205829	−17825286	433150574	759735803	−941218143	−127865614	821067326	−846333260	−274734611	−737396754	−337813401	682410997
−453845630	595204738	−62287397	−854181085	55313868	255458248	−277429290	663256447	1379517453	1221920929	1554011913	−144534251
−155512527	303894757	−771286663	−153546268	49659778	152432913	332065590	1054417210	385410700	69015607	758449176	1192862930
−1250834626	−1393297181	225006722	1224568611	−1033124354	−130121719	264220971	−193153383	−518705613	1448565272	−20852778	−53028529
581382827	58316617	−929252828	−1243278125	530628714	−839682909	−408733767	954729640	−1226273254	−501032723	−596326815	−398596388
486604763	31963204	−1149574288	−202886860	481310886	−15891400	−1421463487	1134288117	154717474	−234151666	24623135	496805331
1415328370	757152808	−372714938	148810458	949584548	1269931339	−587487218	278908354	1149014348	−18973648	70898994	229219800
595298179	749586347	−223035044	492276725	746552324	653820302	−1171081743	−275699474	85302325	81837261	801771829	384005646
506061907	671565249	−1233037104	5982928	248849578	543805107	−764754987	357451075	718951257	−943453912	981319020	242668419
1094094234	−1440167958	−1104958114	1224880168	−1380784904	748616666	−236687117	−753033254	−277431509	698539015	−1344570987	−1038612948
−1064322949	1279665519	551279446	−117343631	120251624	−982400550	21767393	1230554734	1270528674	773491230	895647512	402370292
−584935551	719852622	1497029914	−534515516	1124813428	−121746494	437876904	1209017282	448143028	−1279251195	184084448	475522891

$T_1^{-1} \pmod p =$

754	-42	-425	-744	-834	-750	507	-782	-114	367	671	-603
393	592	937	435	707	520	-127	943	610	-175	-417	483
846	48	-513	-290	-139	-235	749	-628	-628	671	277	-544
334	-66	-662	-598	-266	-132	134	-462	-799	534	532	-334
-121	79	785	525	314	157	-610	640	308	-447	-628	30
725	-272	-725	634	242	-211	-393	-121	212	755	513	151
-212	387	-122	-327	-115	108	12	-44	-42	-617	230	-362
-787	211	119	-272	-151	422	120	-425	-90	-515	-695	-301
90	490	-92	54	-701	-684	-556	947	284	102	406	-936
395	393	-57	635	-755	-542	-392	880	-122	93	-484	-183
-755	-357	424	-851	236	949	290	-17	-683	-169	525	604
-31	-628	693	-218	477	736	894	-141	-471	-412	43	756

$T_2^{-1} \pmod{q} =$

$$
\begin{pmatrix}
1200447848 & 253940891 & -1246618919 & 69256607 & -23085535 & 46171071 & -1108105706 & -1500559810 & -669480531 & 969592492 & -115427678 & 692566066 \\
1266406521 & 811291678 & -435327241 & 1444494939 & -930017288 & 738737137 & -237451223 & -1022359430 & -59362806 & 1385132131 & -613415659 & 540861118 \\
435327241 & 1088318105 & -1081722234 & 521073518 & -98938006 & 646394994 & -1160872645 & -375964433 & -1075126371 & 1015763559 & 402347904 & 725545401 \\
-1015763564 & -1165819545 & 1477474274 & 288569194 & -750279906 & 323197498 & -300111962 & -496339015 & -173141516 & -1454388738 & -611766692 & 530967318 \\
-646394995 & -1027306332 & 369368568 & 611766692 & -334740266 & 1061934635 & -761822673 & -173141517 & -681023298 & -69256606 & -103884910 & 623309460 \\
725545403 & 1028955298 & -494690049 & -178088418 & 620011524 & -230855355 & -626607393 & 943209022 & 824483413 & 646394998 & 408943773 & -883846217 \\
-580436322 & -469955545 & 395752038 & 1202096815 & -1241672019 & 969592493 & -676076398 & -479849346 & 714002635 & -1223533384 & 967943526 & -1098211905 \\
-1042147033 & 141811147 & -573840455 & -1164170578 & -583734256 & 46171071 & -883846218 & -491392114 & 563946654 & -600223924 & 1117999507 & -428731374 \\
844271014 & 1325769327 & -290218161 & -1391728000 & 164896682 & -554052853 & 1411515602 & -943209024 & -824483412 & 923421422 & 375964436 & 883846218 \\
-1055338768 & 501285915 & -1207043714 & -811291677 & 382560305 & -877250351 & 982784227 & -587032188 & -1127893309 & 1200447846 & 118725611 & 857462748 \\
461710711 & -611766692 & 184684285 & 11542768 & 911878655 & 138513213 & -577138389 & 796450977 & 934964189 & -623309461 & -150055981 & 900335886 \\
-672778465 & -1289492056 & -897037950 & -56064871 & 1401621804 & 1569816416 & 1009167696 & 616713595 & -728843338 & 784908204 & -1513751546 & 1233427183
\end{pmatrix}
$$