A New Encryption Scheme Based On Subset Identifying Problem

Muhammad Rezal Kamel Ariffin

Institute for Mathematical Research,
Universiti Putra Malaysia (UPM), Selangor, Malaysia
rezal@upm.edu.my

Abstract. In this article we put forward an encryption mechanism that dwells on the problem of identifying the correct subset of primes from a known set. By utilizing our specially constructed public key when computing the ciphertext equation, the decryption mechanism can correctly output the shared secret parameter. The scheme has short key length, no decryption failure issues, plaintext-to-ciphertext expansion of one-to-two as well as uses "simple" mathematics in order to achieve maximum simplicity in design, such that even practitioners with limited mathematical background will be able to understand the arithmetic. Due to in-existence of efficient algorithms running upon a quantum computer to obtain the roots of our ciphertext equation and also to retrieve the private key from the public key, our encryption mechanism can be a probable candidate for seamless post quantum drop-in replacement for current traditional asymmetric schemes.

Keywords: post quantum cryptosystem, subset identifying problem, combination problem

1 Introduction

We begin our introduction with background of our motivation.

1.1 Motivation

Hard mathematical problems that can be proven or to a lesser degree believed unable to be solved in polynomial time by classical computers, have long been a source for asymmetric cryptosystems. Popular hard mathematical problems include the integer factorization, discrete logarithm and the elliptic curve discrete logarithm problems. As examples, the integer factorization problem was successfully deployed through the RSA cryptosystem, discrete logarithm problem through the El-Gamal cryptosyetem and the elliptic curve discrete logarithm problem by the elliptic curve cryptosystem. On special occasions, the ciphertext has its own source of "security". As an example the RSA ciphertext is based on the e-th root problem. It can be proven that the e-th root problem is reduced to the integer factorization problem. However, the converse is unknown. On the other hand, the notion of having hard mathematical problems seems vague in the symmetric case. As an example is the symmetric key cryptosystem AES [12]. Its security is not based on any known hard mathematical problem. Nevertheless, AES is believed secure, even with the presence of quantum computers.

Upon the discovery of Shor's algorithm in 1994 ([14]) which could solve the integer factorization problem as well as discrete logarithm based problems upon a quantum computer in polynomial time, cryptographers scrambled to find new hard mathematical problems which could resist Shor's algorithm and at the same time is able to provide asymmetric security (i.e. to be able to be used to design asymmetric cryptosystems that are quantum resistant). A compendium of potential hard problems was developed. Pioneering work can be traced to the code based cryptosystem by McEliece in 1978 ([10]). Lattice based cryptosystems which employs the short vector problem or the closest vector problem were also popular to be utilized. Among them the NTRU cryptosystem in 1995 ([8]) and LWE cryptosystem in 2005 ([13]). Since then we have had (not limited to) schemes based on multivariate quadratic equations such as the Rainbow cryptosystem in 2005 ([4]) and the UOV cryptosystem in 2010 ([3]).

An efficient candidate for post quantum cryptography should have short key length (approximately the length of RSA and discrete logarithm based algorithms for at least 128-bit security), decrypts correctly 100% of the time, plaintext ciphertext expansion ratio is kept to a minimum, is simple and fast (i.e. has complexity running time at most $O(n^3)$, where n is the length of the input). These targets motivate this research, as we have seen that most candidates for post quantum asymmetric cryptography have undesirable parameters or characteristics on the points mentioned above.

1.2 Our Cryptosystem

The basic design principle of the scheme rests upon the difficulty of identifying the selected factors of N_1 (i.e. N_2) even when the all the factors of N_1 are known. The integer N_1 can be identified as a public key while N_2 is the private key. To construct the public key N_1 we use a list of public primes larger than 2, $\mathbf{P}=\{p_i\}_{i=1}^{k}$. That is, $N_1 = \prod_{i=1}^{k} p_i$. We then generate the secret parameter given by $N_2 = \prod_{i=1}^{k} p_i^{a_i}$ where $\{a_i\}$ is randomly selected from $\{0, 1\}$. Let $\phi(\cdot)$ be the Euler totient function. Next, choose d co-prime to $\phi(N_1)$. Then, compute the parameters $e_1 \equiv d^{-1} \pmod{\phi(N_1)}$ and $\varepsilon \equiv d^{-1} \pmod{\phi(N_2)}$. Then choose g and h co-prime to N_1 and both are also co-prime to each other. Compute $e_2 \equiv g^{\varepsilon} h^{\phi(N_2)} \pmod{N_1}$. Let (e_1, e_2, g, N_1) and (d, ε, N_2) be public and private parameters respectively. To obtain a "workable" private key one has to identify a set $\mathbf{P}_1 \subset \mathbf{P}$ that constructs a value N_2' such that from $d' \equiv e_1^{-1} \pmod{\phi(N_2')}$, one can have $e_2^{d'} \equiv g \pmod{N_2'}$. Furthermore, one can also view it as the process of upon obtaining ε from e_2, one can have $e_1 - \varepsilon \equiv 0 \pmod{\phi(N_2')}$. The raw complexity for both cases would be $O(2^k)$.

The ciphertext utilized is of the relation $c \equiv e_2^x y^{e_1} \pmod{N_1}$ where $m, x \approx N_2$ and $y \equiv mg^{-x} \pmod{N_1}$ are secret and the tuple (e_1, e_2, g, N_1) is public. The complexity to identify the parameter $\lambda \equiv m \pmod{N_1}$ lies within Hermann and May's results in 2008 ([7]) on the modular multivariate linear equation that states when the product of the upper bounds of the unknown roots of the equation is larger than the modulus, one cannot reduce the exponential time strategy to solve the problem. That is, under this scenario, to obtain the unknown roots one is only left with exponential running time strategies. Furthermore, the complexity of such ciphertext equation is also based on the fact that the LLL algorithm is unable to retrieve the vector $\mathbf{V}_0$ within a lattice when $||\mathbf{V}_0||$ is much larger than the Gaussian heuristic and the upper bound of vectors able to be output by the LLL algorithm. Based on this fact, it is hard to extract the parameter $\lambda \equiv m \pmod{N_1}$. The secret roots can be interpreted as vectors on a lattice where the LLL algorithm is unable to identify it (see [7] for discussion on both items). At the same time the decryption procedure still extracts the desirable parameters.

1.3 Organisation of the Paper

The remainder of this paper is organized as follows. In Section 2, we discuss multivariate equations and the Minkowski theorem. In Section 3, we state the modular exponentiated variable root problem (MEVRP). Then in Section 4, we put forward the KAZ encryption mechanism. In Section 5, we discuss the KAZ key problem. One-wayness of KAZ is presented in Section 6. We conclude in Section 7.

2 Preliminaries

Throughout this article, an n-bit integer a will be denoted as $a \approx 2^n$ unless mentioned otherwise. We also denote when two integers a and b are of the same bit length as $a \approx b$ unless mentioned otherwise.

2.1 Multivariate Equation

The polynomial defined as $f(x_1, x_2, ..., x_k) = a_1x_1 + a_2x_2 + ... + a_kx_k$ is known as a multivariate linear polynomial. To find the root of such polynomial is the task to find the set of solutions $(y_1, y_2, ..., y_k) \in \mathbb{Z}^k$ where we have the equation $f(x_1, x_2, ..., x_k) = 0$. The equivalent task in its modular form is to find the set of solutions $(y_1, y_2, ..., y_k) \in \mathbb{Z}_N^k$ such that $f(x_1, x_2, ..., x_k) \equiv 0 \pmod{N}$.

2.2 Minkowski's Theorem

The Minkowski Theorem which relates the length of the shortest vector in a lattice to the determinant (see [9]) provides initial information to formulate our scheme. It is as follows.

Theorem 1. *In an ω-dimensional lattice $\mathcal{L}$, there exists a non-zero vector $\boldsymbol{V}$ with*

$$\parallel \boldsymbol{V} \parallel \leq \sqrt{\omega}\ det(\mathcal{L})^{\frac{1}{\omega}}$$

We note here that in lattices with fixed small dimension we can efficiently find the shortest vector, but for arbitrary dimensions, the problem of computing the shortest vector is known to be **NP**-hard under randomized reductions (see [1]). In order to find an approximation of the shortest vector, the LLL algorithm is able to compute in polynomial time such approximations up to a multiplicative factor of 2^{ω}, and this is sufficient for many applications. We use information from Theorem 1, to ensure that our vector $\mathbf{V}$ cannot be found by the LLL algorithm.

We will now observe the following remark.

Remark 1. The Gaussian heuristic says that a $\mathbf{V}_{short}$ will satisfy $||\mathbf{V}_{short}|| \approx \sigma(\mathcal{L})$ where $\sigma(\mathcal{L}) = \sqrt{\frac{\omega}{2\pi \mathrm{e}}}\det(M_{\mathcal{L}})^{\frac{1}{\omega}}$. This is preeminently if $||\mathbf{V}_{short}|| < \sigma(\mathcal{L})$ of a particular lattice $\mathcal{L}$, then the lattice reduction algorithm LLL is likely easy to find the shortest vector when the dimension of the lattice is small.

3 Modular Exponentiated Variable Root Problem (MEVRP)

Before focusing on MEVRP, we look into the multivariate case first and the Hermann and May remarks pertaining to it.

3.1 Uniqueness of Modular Multivariate Linear Equation Solutions

We now put forward the two Hermann and May remarks found in [7] which motivated our work.

Remark 2. Let $f(x_1, x_2, ..., x_k) = a_1x_1 + a_2x_2 + ... + a_kx_k$ be a multivariate linear polynomial. One can hope to solve the modular linear equation $f(x_1, x_2, ..., x_k) \equiv 0 \pmod N$, that is to be able to find the set of solutions $(y_1, y_2, ..., y_k) \in \mathbb{Z}_N^k$, when the product of the unknowns are smaller than the modulus. More precisely, let X_i be upper bounds such that $|y_i| \leq X_i$ for $i = 1, ..., k$. Then one can roughly expect a unique solution whenever the condition $\prod_i X_i \leq N$ holds (see [7]). It is common knowledge that under the same condition $\prod_i X_i \leq N$ the unique solution $(y_1, y_2, ..., y_k)$ can heuristically be recovered by computing the shortest vector in an k-dimensional lattice by the LLL algorithm. In fact, this approach lies at the heart of many cryptanalytic results (see [2],[5] and [11]).

Remark 3. If in turn we have $\prod_i X_i \geq N^{1+\epsilon}$ then the modular linear equation given by $f(x_1, x_2, ..., x_k) = \sum_{i=1}^{k} a_ix_i \equiv 0 \pmod N$ usually has N^{ϵ} many solutions, which is exponential in the bit-size of N. As a result, there is no hope to find efficient algorithms that in general improve on this bound, since one is not able to output all roots in polynomial time.

3.2 MEVRP

We now proceed to define the modular exponentiated variable root problem (MEVRP). Let n be an integer where we agree that 2^n is exponentially large and $p \approx 2^n$. Suppose we have the equation given by $c \equiv b^u v^a \pmod{p}$ where $v \approx p$, $u, w \approx p^{0.5}$ and $w \equiv g^u v \pmod{p}$. Let the tuple (a, b, g, p) be public and (u, v, w) secret. Now let $c' \equiv b^{u_0} v_0^a \pmod{p}$ where $v_0 \approx p$, $u_0, w_0 \approx p^{0.5}$ and $w_0 \equiv g^{u_0} v_0 \pmod{p}$. The MEVRP is to identify the private parameter given by $\lambda \equiv w_0 \pmod{p}$ when (g, p, c', a, b) is given.

3.3 The MEVRP Assumption

The advantage of any probabilistic polynomial time adversary running in time poly(n) in attempting to solve MEVRP is at least $O(p^{-0.5}) = O(2^{-\frac{n}{2}})$.

The advantage $O(p^{-0.5})$ is by way of brute forcing $\lambda \equiv w_0 \pmod{p}$.

3.4 Hermann and May Remarks and MEVRP

In order to appreciate the existence of the Hermann and May characteristics within MEVRP, we view the $c \equiv b^{u_0} v_0^a \pmod{p}$ as follows:

$$b^{u_0} v_0^a + z_0 p \equiv 0 \pmod{c} \tag{1}$$

Now let the bounds be $b^{u_0} < B \approx p^{\sqrt{p}}$, $v_0^a < V \approx p^a$ and $z_0 < Z \approx p^{a+\sqrt{p}-1}$. Thus, the product of the upper bounds for the solutions is $\approx p^{2(a+\sqrt{p})}$. It is clear that $p^{2(a+\sqrt{p})} \gg c \approx p$. To this end, Remark 3 can be observed within c of MEVRP.

3.5 Lattice based analysis upon MEVRP

To further analyse the intractability of MEVRP, the conventional way to solve multivariate equations is to employ lattices as well as the LLL algorithm. This is due to the ability to interpret the solutions of a modular equation as vectors on a lattice. Consider the lattice $\mathcal{L}$ with the matrix

$$M_{\mathcal{L}} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -p \\ 0 & 0 & c \end{bmatrix}$$

Let $\mathbf{V}_0$ be a vector of $\mathcal{L}$. Then there exists $(u_1, u_2, u_3) \in \mathbb{Z}^3$ such that

$$\mathbf{V}_0 = (u_1, u_2, u_3) M_{\mathcal{L}} = (u_1, u_2, -u_1 - p u_2 + c u_3)$$

is on the lattice $\mathcal{L}$. More precisely the vector solution $\mathbf{V}_0 = (b^{u_0} v_0^a, z_0, 0)$. Observe that the length of $\mathbf{V}_0$ is given by

$$||\mathbf{V}_0|| = \sqrt{(b^{u_0} v_0^a)^2 + (z_0)^2} \approx p^{a+\sqrt{p}}$$

On the other hand, the determinant of the matrix $M_{\mathcal{L}}$ is $\det(M_{\mathcal{L}}) = c \approx p$ and the Gaussian heuristics for the lattice $\mathcal{L}$ asserts that the length of its shortest non-zero vector is usually approximately $\sigma(\mathcal{L})$ (see [9]) where

$$\sigma(\mathcal{L}) = \sqrt{\frac{\omega}{2\pi \mathrm{e}}}\det(M_{\mathcal{L}})^{\frac{1}{\omega}} = \sqrt{\frac{3}{2\pi \mathrm{e}}}\det(M_{\mathcal{L}})^{\frac{1}{3}} = \sqrt{\frac{3}{2\pi \mathrm{e}}}c^{\frac{1}{3}} \approx \sqrt{\frac{3}{2\pi \mathrm{e}}}p^{\frac{1}{3}}$$

and $\omega = \dim(\mathcal{L})$ (see [9]). In this case $\omega = 3$. Thus, $||\mathbf{V}_0|| > \sigma(\mathcal{L})$. Then referring to Remark 1, the use of the LLL algorithm is insignificant.

Furthermore, the LLL algorithm outputs a reduced basis where the norm of the shortest vector is less than

$$2^{\frac{\omega-1}{4}}\det(\mathcal{L})^{\frac{1}{\omega}} = 2^{\frac{1}{2}}\det(\mathcal{L})^{\frac{1}{3}} = 2^{\frac{1}{2}}c^{\frac{1}{3}} \approx 2^{\frac{1}{2}}p^{\frac{1}{3}}$$

Obviously, $||\mathbf{V}_0|| > 2^{\frac{1}{2}}p^{\frac{1}{3}}$. Thus, LLL will output a vector shorter than our desired vector $\mathbf{V}_0$.

Remark 4. Under the assumption that integer factorization can be done in polynomial time, if $b^{u_0}v_0^a$ is retrieved from $\mathbf{V}_0$, then one can obtain the pair (b^{u_0}, v_0^a). By computing the logarithm base b on the first parameter, the value u_0 can be obtained. By taking the a-th root on the second parameter, the value v_0 can be obtained. Finally $w_0 \equiv g^{u_0}v_0 \pmod{p}$ can be retrieved.

4 The KAZ Cryptosystem

We now put forward our scheme, the KAZ cryptosystem mechanism.

4.1 The KAZ System Parameters

This section provides information regarding the key generation procedure. Let $\ell(\cdot)$ be a function that outputs length of binary string of input. From the given security parameter, κ determine k (see section 5.2 and table 2). Next generate a list of k primes greater than 2, $\mathbf{P} = \{p_i\}_{i=1}^{k}$. Then compute $N_1 = \prod_{i=1}^{k} p_i$. The KAZ system parameters are $(\mathbf{P}, N_1, k)$.

4.2 The KAZ Key Generation Algorithm

Algorithm 1 : KAZ.Key_Gen algorithm

```
Input: System parameters, $(\mathbf{P}, N_1, k)$
Output: Public keys $(e_1, e_2, g)$; private key $N_2$; private parameters $(\varepsilon, h)$ and public
    parameter $(d, n_2)$.
1: Compute integer $N_2 = \prod_{i=1}^{k} p_i^{a_i}$ where $a_i$ is chosen randomly from $\{0, 1\}$.
2: Calculate $n_2 = \ell(N_2)$.
3: Generate random $d \approx 2^{n_2-1}$ co-prime to $\phi(N_1)$.
4: Compute $e_1 \equiv d^{-1} \pmod{\phi(N_1)}$.
5: Compute $\varepsilon \equiv d^{-1} \pmod{\phi(N_2)}$.
6: Generate $g$ and $h$ co-prime to $N_1$ and both $g$ and $h$ are co-prime to each other.
7: Compute $e_2 \equiv g^{\varepsilon} h^{\phi(N_2)} \pmod{N_1}$.
8: Output public keys $(e_1, e_2, g)$, private key $N_2$, private parameters $(\varepsilon, h)$ and public
    parameter $(d, n_2)$.
```

4.3 KAZ Encryption and Decryption Algorithms

The following algorithm upon execution does the task of encrypting the secret message $m \approx 2^{n_2-1}$.

Algorithm 2 : KAZ.Enc algorithm

```
Input: System parameters $N_1$, public keys $(e_1, e_2)$ and public parameter $n_2$.
Output: The ciphertext, $c$.
1: Choose message $m \approx 2^{n_2-1}$.
2: Generate random $x \approx 2^{n_2-1}$.
3: Compute $y \equiv mg^{-x} \pmod{N_1}$.
4: Compute $c \equiv e_2^x y^{e_1} \pmod{N_1}$.
5: Output ciphertext $c$.
```

The following algorithm upon execution does the task of decrypting the message $m \approx 2^{n_2-1}$ from the ciphertext c.

Algorithm 3 : KAZ.Dec algorithm

```
Input: Ciphertext $c$, public parameter $d$ and private key $N_2$.
Output: $m \approx 2^{n_2-1}$.
1: Compute $Y = c^d \pmod{N_2}$.
2: Output $m = Y$.
```

Proposition 1. *The KAZ.Dec algorithm decrypts correctly and without failure.*

Proof. Via Fermat's Little Theorem, $Y = c^d \equiv e_2^{xd} y^{e_1 d} \equiv g^{x\varepsilon d} h^{\phi(N_2)d} y^{e_1 d} \equiv g^x y \equiv m \pmod{N_2}$. Together with the condition $m < N_2$ we obtain $m = Y$ without modular reduction (i.e. $m \in \mathbb{Z}$). That is, we obtain the value m without failure. ■

4.4 Toy Example

We will use the first $k = 10$ primes. That is $\mathbf{P} = \{3, 5, ..., 31\}$.

1. $N_1 = 100280245065$
2. $N_2 = 527527 = (7)(11)(13)(17)(31)$
3. $d = 559561$
4. $e_1 = 11684536441$
5. $\varepsilon = 146041$
6. $g = 126365502521$
7. $h = 568433$
8. $e_2 = 34295129126$
9. $m = 506449$
10. $x = 628319$
11. $y = 1681269899$
12. $c = 86045435419$

During decryption, $c^d \pmod{N_2} = 506449 \in \mathbb{Z}$.

5 KAZ Key Problem

This section discusses the problem of finding the private key from the public key. That is, the task to find the private key via identifying the set $\mathbf{P}_1$ from $\mathbf{P}$ that constructs $N_2 \approx 2^{n_2}$.

5.1 Combinatorial Approach -1

The following definition gives a fundamental outline on the situation.

Definition 1. *Let k be an integer where we agree that 2^k is exponentially large. Given a set $\boldsymbol{P}$ where it contains k elements and exactly half of the elements (without repetition) are probable candidates for the set $\boldsymbol{P}_1$. Assume with overwhelming probability that there is only one correct candidate for the set $\boldsymbol{P}_1$. The combinatorial approach to solve KAZ key problem is to identify the set $\boldsymbol{P}_1$ from $\boldsymbol{P}$.*

Under the assumption that N_2 is constructed via the sequence $a_i \in \{0, 1\}$ which was chosen at random 50-50, and since the set $\mathbf{P}$ has k elements, the fundamental number of guesses for $\mathbf{P}_1$ is 2^k.

Thus, to achieve 256-bit security level, we will need a prime list $\mathbf{P}$ which contains $k = 256$ primes. Refer to Table 1 for an illustrative entropy table. We denote the entropy as κ_0 where $\kappa_0 = \log_2 2^k = k$.

k	κ_0
80	80
128	128
256	256

Table 1. KAZ.Key_Gen κ_0-bit entropy (i.e. 2^{κ_0})

We note here that if the first 256 primes greater than 2 are used together with the KAZ.Key_Gen methodology, the public key N_1 will be of length 2300 bits and the private key N_2 length would be approximately 1150 bits.

We stress here that the complexity of KAZ is due to the value k (i.e. number of primes in the public list $\mathbf{P}$). The calculated entropy value κ_0 is to give an intuition of the difficulty to obtain the correct combination of primes to construct the private parameter N_2.

The length of the private parameters is an inherited value from the length of each prime in the list $\mathbf{P}$. As such, the length of the private parameter does not represent the "strength". The fact is, the length of KAZ private parameters is due to the combination of each prime decomposition in the public list $\mathbf{P}$. The complexity to obtain KAZ private key from its public key is via the number of elements in $\mathbf{P}$ and not the length of the private parameters. It is because of this fact that we have chosen the value of k such that 2^k is exponentially large.

5.2 Combinatorial Approach -2

The following methodology is due to a strategy which describes a more intuitionist combinatorial approach. That is, to view the problem as identifying the number of combinations that arises from the problem at hand. Observe the value

$$C_1 = \binom{\theta_1}{\theta_2}$$

where $\theta_1 = k$ the number of primes in the list $\mathbf{P}$ and $\theta_2 = \frac{k}{2}$ be the number of primes constructing N_2. The value $\theta_2 = \frac{k}{2}$ is due to the assumption that the choice of the sequence $a_i \in \{0, 1\}$ is random 50-50, which implies N_2 is constructed of $\frac{k}{2}$ primes from the list $\mathbf{P}$.

Thus, to achieve 256-bit security level, we will need a prime list $\mathbf{P}$ which contains $k = 260$ primes. We denote the entropy as $\kappa = [\log_2 C_1]$.

k	κ
84	80
132	128
260	256

Table 2. KAZ.Key_Gen κ-bit entropy (i.e. 2^κ)

From Table 2, one can deduce that, we need to utilize a set $\mathbf{P}$ with $k = 132$ primes to obtain 128-bits security. If the first 132 primes greater than 2 are used together with the KAZ.Key_Gen methodology, the public key N_1 will be of length 1037 bits while the private key N_2 length would be approximately 540 bits. If the first 260 primes greater than 2 are used together with the KAZ.Key_Gen methodology, the public key N_1 will be of length 2343 bits while the private key N_2 length would be approximately 1160 bits.

5.3 Ad-hoc Cryptanalysis

In this section we discuss ad-hoc attempts to solve the KAZ key problem whose parameters are generated by the KAZ.Key_Gen methodology.

Attempt - 1

In an attempt to generate another private key for the given public key, one could generate $N_2' \neq N_2$ and use it to compute $e_2 \equiv h^{\phi(N_2)} g^{\varepsilon} \equiv g^{\delta\varepsilon} \equiv g^{\gamma} \pmod{N_2'}$. Upon solving the discrete logarithm problem on e_2 one obtains the exponent value γ. Then compute $d' \equiv \gamma^{-1} \pmod{\phi(N_2')}$. Observe that:

$$d' e_1 \equiv \gamma^{-1} \left(d^{-1} \pmod{\phi(N_1)} \right) \equiv 1 \pmod{\phi(N_2')}$$

will not occur with overwhelming probability.

Attempt -2

In an attempt to generate another private key for the given public key, one could generate $N_2' \neq N_2$ and use it to compute $d' \equiv e_1^{-1} \pmod{\phi(N_2')}$. Observe that

$$e_2^{d'} \equiv g^{\varepsilon d'} h^{\phi(N_2) d'} \equiv g \pmod{N_2'}$$

will not occur with overwhelming probability.

Attempt -3

One can attempt to use $d \equiv e_1^{-1} \pmod{\phi(N_1)}$ together with $N_2' \neq N_2$. Observe that

$$e_2^{d} \equiv g^{\varepsilon d} h^{\phi(N_2) d} \equiv g \pmod{N_2'}$$

will not occur with overwhelming probability.

5.4 The KAZ Key Equation and Grover's Algorithm

Grover's algorithm is a quantum algorithm that finds with high probability the unique input to a black box function that produces a particular output value, using just $O(\sqrt{N})$ evaluations of the function, where N is the size of the function's domain [6]. Thus, in order to achieve 128-bit post quantum security against Grover's algorithm, a total of 260 primes must be used from the list $\mathbf{P}$.

If $\mathbf{P}$ is the list of the first 260 primes larger than 2, then N_1 will be approximately 2343 bits and N_2 will be approximately 1160 bits. Since both KAZ encryption and decryption procedures has low computational complexity (i.e. power modulo), KAZ operates on a desirable speed.

6 One-wayness of KAZ

In this section, we provide the reader analytical reasoning as to why the KAZ Problem is sound and provides the one-wayness element within the KAZ ciphertext. We put forward reasoning analogous to the one-wayness property of the RSA problem embedded within the RSA ciphertext.

6.1 The KAZ Problem

We now formally define the KAZ Problem. Given:

1. KAZ public keys (e_1, e_2, g, N_1)
2. KAZ ciphertext c

one needs to output m, where the unknown variable size is as specified in Algorithm 2.

6.2 KAZ Problem reduces to MEVRP

From the KAZ ciphertext given by:

$$c \equiv e_2^x y^{e_1} \pmod{N_1}$$

we have the following proposition.

Proposition 2. *The KAZ Problem reduces to the MEVRP.*

Proof. Upon solving the MEVRP from c, the parameter $\lambda \equiv m \pmod{N_1}$ is obtained. Thus, KAZ Problem is reduced to the MEVRP.■

Remark 5. We remark here that the converse is still unknown.

6.3 KAZ Problem reduces to KAZ Key Problem

Proposition 3. *The KAZ Problem reduces to the KAZ Key Problem.*

Proof. Upon solving the KAZ Key Problem from (e_1, e_2, g, N_1), the private parameters (d, N_2) are obtained. Then one can obtain $c^d \equiv g^x y \equiv m \pmod{N_2}$. Thus, KAZ Problem is reduced to the KAZ Key Problem.■

Remark 6. We remark here that the converse is still unknown.

6.4 KAZ Ciphertext Equation and Grover's Algorithm

Based on the existing arguments within this document one needs to conduct exhaustive search for the ciphertext secret parameter m. That is one needs to conduct at most 2^{n_2} searches. For 128-bit security, with Grover's algorithm the complexity is reduced to $\approx 2^{0.5n_2}$. Through the KAZ.Key_Gen procedure, for 128-bit security, we have $n_2 \approx 540$. Thus, the complexity is reduced to $\approx 2^{270}$.

7 Conclusion

In this work we have utilized the modular exponentiated variable root problem (MEVRP) to design an encryption mechanism. It is proven analytically that all current strategies to either extract the private key from the public key or the secret information from the ciphertext will incur exponential running time complexity. We also show that KAZ can achieve 128-bit security with key length of approximately 1037 bits. We also can observe there is a 1-to-2 message expansion rate. Furthermore, we have proven there is no decryption failure. With complexity running time $O(n^3)$ (where n is the length of the input) for both encryption and decryption, KAZ has desirable speed for any practical application. We also point out again here that, KAZ utilizes "simple" mathematics in order to achieve maximum simplicity in design, such that even practitioners with limited mathematical background will be able to understand the arithmetic. Indeed, KAZ can be a seamless post quantum drop-in replacement for traditional asymmetric cryptosystems.

Acknowledgements

This research is funded by XXXX

References

1. Miklòs Ajtai. The Shortest Vector Problem in L_2 is **NP**-hard for Randomized Reductions. In *In ACM Symposium on Theory of Computing*, pages 10–19. ACM New York, NY, USA, 1998.
2. Daniel Bleichenbacher and Alexander May. New Attacks on RSA with Small Secret CRT-Exponents. In *In Public Key Cryptography*, pages 1–13. Springer, 2006.
3. Stanislav Bulygin, Albrecht Petzoldt, and Johannes Buchmann. Towards Provable Security of the Unbalanced Oil and Vinegar Signature Scheme under Direct Attacks. In *Progress in Cryptology - INDOCRYPT 2010*, pages 17–32. Springer, 2010.
4. Jintai Ding and Dieter Schmidt. Rainbow, a New Multivariable Polynomial Signature Scheme. In *Applied Cryptography and Network Security (ACNS) 2005*, pages 164–175. Springer, 2005.
5. Marc Girault, Philippe Toffin, and Brigitte Vallée. Computation of approximate l-th roots modulo n and application to cryptography. In *Crypto*, volume 88, pages 100–117. Springer, 1988.
6. Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219. ACM, 1996.
7. Mathias Herrmann and Alexander May. Solving linear equations modulo divisors: On factoring given any bits. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 406–424. Springer, 2008.
8. Jeffrey Hoffstein, Jill Pipher, and Joseph H Silverman. NTRU: A Ring Based Public Key Cryptosystem. In *Algorithmic Number Theory (ANTS III)*, pages 267–288. Springer, 1998.
9. Jeffrey Hoffstein, Jill Pipher, and Joseph H Silverman. *An introduction to mathematical cryptography*, volume 1. Springer, 2008.
10. Robert J McEliece. A Public-Key Cryptosystem Based On Algebraic Coding Theory. In *DSN Progress Report*, volume 44, pages 114–116, 1978.
11. Phong Q. Nguyen. Can We Trust Cryptographic Software? Cryptographic Flaws in GNU Privacy Guard v1.2.3. In *EUROCRYPT*, pages 555–570. Springer, 2004.
12. NIST. AES: Advanced Encryption Standard, http://csrc.nist.gov/publications/fips/fips197/ fips- 97.pdf, 2015.
13. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *The 37th ACM Symposium on Theory of Computing (STOC 2005)*, pages 84–93. Association of Computing Machinery, 2005.
14. Peter W Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing*, 26:1484–1509, 1994.