An improved scheme of searchable encryption algorithm based on NTRU

Sipei Cheng¹, Mengdong Li² and Yuwei Duan¹*

¹ Department of Electronics and Communication Engineering, Beijing Electronic Science & Technology Institute, Fengtai, Beijing, 100070, China
² Department of Cryptography and Technology, Beijing Electronic Science & Technology Institute, Fengtai, Beijing, 100070, China

Corresponding author's e-mail: a429137827@163.com

Abstract. R. Behnia et al proposed a NTRU-based searchable encryption scheme in 2017. Compared with the previous searchable encryption scheme, this scheme has a great improvement in security and efficiency, and can resist quantum computing. But this scheme will expose a lot of information to the server. If the server is not completely honest, an attacker can use the information to attack a user's keywords. This paper proposes an improvement of Rouzbeh's scheme, solves this problem and improves security.

1. Introduction

In recent years, cloud computing has developed rapidly, and people tend to store their data on cloud servers. For data owners, cloud services are not completely trusted. In order to ensure the security of the data, it must be encrypted before sending it to cloud. However, in this way, the existing plaintext-based keyword search technology cannot be used, so the concept of searchable encryption was proposed. Searchable encryption is a cryptographic primitive which supports users to search for keywords on ciphertext, the searchable encryption system work as follows: First, the data owner sends the encrypted data and related keyword ciphertext to the cloud server; then the authorized user generates a trapdoor and send it to the cloud server, the server searches for relevant data through the trapdoor and sends it to the user.

Searchable encryption is generally divided into symmetric searchable encryption and public key searchable encryption. The first symmetric searchable encryption scheme was proposed by Song et al[1] with the property of the symmetric encryption system, both the encrypted data in the symmetric searchable encryption system and the generated trapdoor must use the same key. Therefore, symmetric searchable encryption is suitable for use in scenes such as personal data storage.

On the contrary, public key searchable encryption can allow any authorized user getting the wanted ciphers. Public Encryption with Keyword Search (PEKS) was first proposed by Boneher al in 2004[2] and used for mail routing. Most public key searchable encryption is implemented through bilinear mapping constructs, its computational efficiency is lower than symmetric searchable encryption. But public searchable encryption uses the data owner's public key to encrypt the data. The data owner does not need to perform key agreement with the data sharer during the entire encryption scheme, which makes the scheme suitable for multi-user data sharing.
2. Rouzbeh Behnia's searchable encryption scheme

2.1 Notions.

Let \( a \leftarrow \chi \) denotes that \( a \) is randomly selected from distribution \( \chi \). \( H_i \) for \( i \in \{1,...,n\} \) denotes a hash function which is perceived to behave as a random oracle in this paper. The norm of a vector \( v \) is denoted by \( \|v\| \) \( [x] \) rounds \( x \) to the closest integer. The \( \gcd(x,y) \) returns the greatest common divisor of values \( x \) and \( y \).

Let \( B = [b_1|...|b_n] \in \mathbb{R}^{n \times n} \) be an \( n \times n \) matrix, the matrix's columns are independent vectors \( b_1,...,b_n \in \mathbb{R}^n \). The \( n \)-dimensional full-rank lattice \( \Lambda \) generated by \( B \) is set.

\[
\Lambda = \mathcal{P}(B) = \{ y \in \mathbb{R}^n : \exists s \in \mathbb{Z}^n, y = Bs = \sum_{i=1}^{n} s_i b_i \}
\]

Definition 1. Anticirculant matrix. An anticirculant matrix consisting of polynomials \( f \) is shown below:

\[
\mathcal{A}_N(f) = \begin{bmatrix}
  f_0 & f_1 & f_2 & \cdots & f_{N-2} & f_{N-1} \\
  -f_{N-1} & f_0 & f_1 & \cdots & f_{N-2} & f_{N-1} \\
  -f_1 & -f_2 & f_0 & \cdots & f_{N-2} & f_{N-1} \\
  \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
  \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
  \vdots & \vdots & \vdots & \vdots & \vdots & f_0
\end{bmatrix}
\]

Lemma [1]. Anticirculant matrices has a property. Let \( f, g \in R \). Then \( \mathcal{A}_N(f) + \mathcal{A}_N(g) = \mathcal{A}_N(f + g) \), and \( \mathcal{A}_N(f) \times \mathcal{A}_N(g) = \mathcal{A}_N(f \ast g) \).

Definition 2. An \( N \)-dimensional Gaussian function \( \rho_{\sigma,\epsilon} : \mathbb{R} \to (0,1] \) is defined as \( x \in \Lambda \). Given a lattice \( \Lambda \subset \mathbb{R}^n \), the discrete Gaussian distribution over \( \Lambda \) is \( D_{\Lambda,\sigma,\epsilon}(x) = \frac{\rho_{\sigma,\epsilon}(x)}{\rho_{\sigma,\epsilon}(\Lambda)} \) for all \( x \in \Lambda \).

NTRU cryptosystem. Given an integer \( N \), \( q \) and \( p \). \( N \geq 1 \) and \( p, q \) are prime numbers. We begin by fixing an integer \( N \geq 1 \) and two moduli \( p \) and \( q \), and we let \( R, R_p, \) and \( R_q \) be the convolution polynomial rings: \( R = \mathbb{Z}[x]/(x^N-1) \), \( R_p = (\mathbb{Z}/p\mathbb{Z})[x]/(x^N-1) \), \( R_q = (\mathbb{Z}/q\mathbb{Z})[x]/(x^N-1) \). A polynomial \( a(x) \) belong to the \( R \), we can treat it as an element of \( R_p \) or \( R_q \) by reducing its coefficients modulo \( p \) or \( q \). Simultaneously, we use centered lifts to move elements from \( R_p \) or \( R_q \) to \( R \). We have some various assumption on \( N, p, q \). All of the parameters should be prime and \( \gcd(N,q) = \gcd(p,q) = 1 \).

Definition 3. Keyword indistinguishability against an adaptive chosen-keyword attack (IND-CKA) is define as follows. For a PEKS scheme, we define the TDQuery as below. In the following experiment, a bit \( b \in \{0,1\} \) was associated to the adversary \( \mathcal{A} \).

\[
\text{TDQuery}(w) \rightarrow wSet \leftarrow wSet \cup w
\]

\[
t_w \leftarrow \text{Trapdoor}(w, sk, pk)
\]

\[
\text{return } t_w
\]

\[
\text{Experiment } E_{\mathcal{A}}^{\text{PEKS-IND-CKA}}(1^k) \rightarrow
\]

\[
wSet \leftarrow \emptyset, (sk, pk) \leftarrow \text{GenKey}(1^k)
\]

\[
\text{for a random oracle } H
\]

\[
(w_0, w_1) \leftarrow \mathcal{A}^{\text{TDQuery}(w)}(\text{find, pk})
\]

\[
s_w \leftarrow \text{PEKS}^H(pk, w_h)
\]

\[
b' \leftarrow \mathcal{A}^{\text{TDQuery}(s_w)}(\text{guess, pk})
\]
If \( \{w_0, w_1\} \cap w \neq \emptyset \) return \( b' \) else, return 0

\( \mathcal{A} \)'s advantage in the above experiment is defined as

\[
\text{Adv}_{\text{PEKS}, \mathcal{A}}^{\text{IND-CKA-b}}(1^k) = \Pr[\text{Exp}_{\text{PEKS}, \mathcal{A}}^{\text{IND-CKA-b}}(1^k) = 1] - \Pr[\text{Exp}_{\text{PEKS}, \mathcal{A}}^{\text{IND-CKA-b}}(1^k) = 0]
\]

### 2.2 Rouzbeh Behnia's searchable encryption scheme

Rouzbeh Behnia's scheme consists of four steps.

1. \((pk, sk) \leftarrow \text{Genkey}(q, N)\): Input a power-of-two integer \( N \) and a prime \( q \), the output of this algorithm is a public-private key pair \((h, b)\). In this scheme, the public key will be a polynomial \( h \) and the secret key is consisted of a “nice base” for the \( 2N \)-dimension lattice generated by the rows of \( \mathcal{A}(h) = \begin{pmatrix} I_N & 0 \\ qI_N & O_N \end{pmatrix} \), where \( \mathcal{A}(h) \)'s \( i \)th row consist of the coefficients of the polynomial \(hx^i \mod x^N + 1\).

2. \( s_w \leftarrow \text{PEKS}(pk, w)\): Given cryptographic hash functions \( H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q^N \) and \( H_2 : \{0,1\}^N \times \{0,1\}^N \rightarrow \mathbb{Z}_q^N \), given the receiver's public key \( pk \) and an encrypted keyword \( w \in \{0,1\}^* \), this algorithm outputs a searchable ciphertext \( s_w \).

3. \( t_w \leftarrow \text{Trapdoor}(sk, w)\): Input the receiver's private key \( sk \) and a keyword \( w \in \{0,1\}^* \), this algorithm uses the algorithm Gaussian-Sampler \((B, \sigma, t, 0)\), and outputs a trapdoor \( t_w \). \( H \) is a publicly-known cryptographic hash function mapping into \( \mathbb{Z}_q^N \), and \( t = H(w) \). The trapdoor will be composed of a polynomial \( t_w \) such that \( s + t_w h = t \), where \( s_1 \) is another small polynomial.

4. \( d \leftarrow \text{Test}(pk, t_w, s_w)\): On the input of a receiver's public key \( pk \), a trapdoor \( t_w \) and a searchable ciphertext \( s_w \), this algorithm computes \( y = \lfloor c_1 - c_0 \cdot t_w / q \rfloor \) and output \( d = 1 \) if \( H_2(y, c_1) = H_2(k, c_1) \) and \( d = 0 \), otherwise. Encryption and verify will proceed as in the Ring-LWE scheme of [6]. To encrypt a keyword \( w \), the sender chooses polynomials \( k, r, e_1, e_2 \) small coefficients and sends the ciphertext \( (u = rh + e_1, v = rt + e_2 + \left\lfloor \frac{q}{2} \right\rfloor k) \). To verify keyword, the server computes \( v - ut_w = r s + e_2 + \left\lfloor \frac{q}{2} \right\rfloor k \cdot t_w e_1 \). Compute \( r s + e_2 - t_w e_1 \). If the parameters are properly, its coefficients will be small, and so the coordinates \( k \) is 0 will be small, whereas the coordinates in which it is 1 will be close to \( q/2 \). In order to verify, the \( t_w \) should be as small as possible. Because it is essential that the polynomial \( r s + e_2 - t_w e_1 \) have small coefficients.

### 2.3 The defect of Rouzbeh Behnia's searchable encryption scheme

For searchable encryption, the user's keyword should be kept secret for the servers. So Behnia's scheme encrypt keyword with many random parameters. To verify if the keyword exists in the server, the receiver should generate a trapdoor \( t_w \) such that \( s + t_w h = t \). As can be seen from the above that \( t = H(w) \), and \( H \) is a public-known cryptographic hash function. Because the keyword space is limited. The adversary can use exhaustive method to compute \( s = t - t_w \cdot h \). If \( s \) and \( t_w \) obey the same distribution then the guessed keyword is likely to be the correctly.

### 3. Proposed schemes

#### 3.1 An improved scheme of searchable encryption algorithm based on NTRU

Our scheme consists of four algorithms. The first algorithm \((pk, sk) \leftarrow \text{GenKey}(q, N)\) and test algorithm are the same as the Behnias. We mainly improved the following parts.
s_w←PEKS(pk,w): Given cryptographic hash functions $H_1: \{0,1\}^* \to \mathbb{Z}_q^N$ and $H_2: \{0,1\}^N \times \{0,1\}^N \to \mathbb{Z}_q^N$. Given the receiver’s public key, and then encrypted a keyword $w \in \{0,1\}^*$. The sender performs as follows.

1. Pick $r, e_1, e_2 \leftarrow \{-1,0,1\}^N$, $k \leftarrow \{0,1\}^N$, $k_1 \leftarrow \{0,1\}^*$, Compute $t \leftarrow H_2(w+k_w)$.
2. Compute $c_0 = r \ast h + e_1 \in R_q$ and $c_1 = r \ast t + e_2 + \frac{q}{2} k \in R_q$.
3. Finally, the algorithm outputs $s_w = \langle c_0, c_1, H_2(k, c_1) \rangle$.

$t_w←\text{Trapdoor}(sk, w)$: Input the receiver’s private key $sk$, and a keyword $w \in \{0,1\}^*$, the receiver computes $t = H_2(w + k_w)$ and using the sampling algorithm Gaussian-Sampler($B, \sigma, (t, 0)$), samples $s$ and $t_w$ such that $s + t_w \ast h = t$.

### 3.2 Analysis of the scheme

From definition 3, we can know that the adversary $\mathcal{A}$ has ability to query the keyword. $H$ is some publicly-known cryptographic hash function. The adversary can map $w$ into $\mathbb{Z}_q[x]/(x^N+1)$ by $H$.

In the new scheme, we use a $k_w$ to protect the keyword. The sender encrypts the keyword with $k_w$ before the algorithm of $s_w←\text{PEKS}(pk, w)$. The encrypted keyword is $w'$. $K_w$ is secret for the server. Although the adversary can query the keyword, he doesn’t know the information of $k_w$. He can’t attack $w'$ either. Therefore, in the definition 3, our scheme can resistance the keyword guess attack.

**Lemma 2.** A scheme is complete if it contains a public-private key pair generated by the algorithm $(h, B)←\text{GenKey}(q, N)$, a searchable ciphertext $s_w←\text{PEKS}(pk, w)$, and a trapdoor generated by the receiver $t_w←\text{Trapdoor}(sk, w)$, our proposed scheme is complete.

**Proof.** If our scheme for is complete for $s_w = \langle c_0, c_1, H_2(k, c_1) \rangle$, the Test algorithm should return 1 when $\left\lceil \frac{c_1 - c_0 \ast t_w}{q/2} \right\rceil = k$. To verify this, we do follow works.

\[
c_1 - c_0 \ast t_w = (r \ast t + e_2 + \frac{q}{2} k - (r \ast h + e_1) \ast t_w) \in R_q = r \ast s + e_2 + \frac{q}{2} - t_w \ast e_1
given r, e_1, e_2, t_w$ and $s$ are all short vectors, all the coefficients of $r \ast s + e_2 - t_w \ast e_1$ will be in $\langle -\frac{q}{4}, \frac{q}{4} \rangle$, and therefore, $\left\lceil \frac{c_1 - c_0 \ast t_w}{q/2} \right\rceil = k$.

The hardness of the lattice problem determines the lattice-based schemes’ security. Therefore, we evaluate the security of the NTRU-PEKS by the root Hermite factor. According to [4], for a short planted vector $v$ in a NTRU lattice, the Hermite factor is computed as $\gamma_n = \sqrt{N/ (2\Pi e) \times \det(A)^2 \|v\|}$. According to [5, 6], $\gamma \approx 1.004$ can provides approximately 192-bits security.

The output of of our scheme corresponds to the algorithm of [7, 8]. Our scheme works over the polynomial ring $\mathbb{Z}[x]/(x^N+1)$, for a power-of-two $N$ and a prime $q \equiv 1 \mod 2N$. The ring-LWE based PEKS algorithm computes a pseudorandom ring-LWE vector $c_0 = r \ast h + e_4$ (for a uniform $r, e_1 \leftarrow \{-1,0,1\}^N$) and uses $H(w)$ to compute $c_1 = r \ast t + e_2 + \frac{q}{2} k$. Therefore, the adversary’s view of $(c_0, c_1, H_2(k, c_1))$ is indistinguishable from uniform distinguishable from uniform distribution under the hardness of decision ring-LWE. The pseudorandomness is preserved when $t_w$ is chosen from the error distribution (by adopting the transformation to Hermite’s normal form) similar to the one in standard LWE [9].

### 4. Discussion
From definition 3, we can know that the adversary $\mathcal{A}$ has ability to query the keyword. $H$ is some publicly-known cryptographic hash function. The adversary can map $w$ into \( \mathbb{Z}_q[x]/(x^w + 1) \) by $H$.

In the new scheme, we use a $k_w$ to protect the keyword. The sender encrypts the keyword with $k_w$ before the algorithm of $s_w \leftarrow \text{PEKS}(p, w)$. The encrypted keyword is $w'$. $K_w$ is secret for the server.

Although the adversary can query the keyword, he doesn't know the information of $k_w$. He can't attack $w'$ either. Therefore, in the definition 3, our scheme can resistance the keyword guess attack.

References

[1] D. Song, D. Wagner, and A. Perrig. (2000) Practical Techniques for Searches on Encrypted Data. IEEE Symposium on Security and Privacy, pp.44-55.

[2] D. Boneh, G. D. Crescenzo, R. Ostrovsky. (2004) Public key encryption with keyword search. Cryptology Eurocrypt, pp. 506-522.

[3] R. Behnia, M. O. Ozemen and A. Attila. (2017) Lattice-Based Public Key Encryption with Keyword Search. Cryptology ePrint Archive: Report 2017/1215.

[4] L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky. Lattice Signatures and Bimodal Gaussians. Annual Cryptology Conference. Springer, Berlin, Heidelberg, 2013, pp. 40-56.

[5] N. Gama and P. Q. Nguyen. (2008) Predicting lattice reduction. Springer Berlin Heidelberg, pp. 31-51.

[6] Y. Chen and P. Q. Nguyen, (2011) Bkz 2.0: Better lattice security estimates. Springer Berlin Heidelberg, pp. 1-20.

[7] V. Lyubashevsky, C. Peikert, and O. Regev. (2010) On ideal lattices and learning with errors over rings. Springer Berlin Heidelberg, pp. 1-23.

[8] T. Johansson and P. Q. Nguyen. (2013) A toolkit for ring-lwe cryptography. Springer Berlin Heidelberg, pp. 25-54.

[9] D. Micciancio and O. Regev. (2009) Lattice-based cryptography. Springer Berlin Heidelberg, pp. 147-191.