High-Speed High-Security Public Key Encryption with Keyword Search

Rouzbeh Behnia, Attila Altay Yavuz, Muslum Ozgur Ozmen

High-Speed High-Security Public Key Encryption with Keyword Search

Rouzbeh Behnia, Attila Altay Yavuz, and Muslum Ozgur Ozmen

School of Electrical Engineering and Computer Science
Oregon State University
{behniar,attila.yavuz,ozmenmu}@oregonstate.edu

Abstract. Data privacy is one of the main concerns for clients who rely on cloud storage services. Standard encryption techniques can offer confidentiality; however, they prevent search capabilities over the encrypted data, thereby significantly degrading the utilization of cloud storage services. Public key Encryption with Keyword Search (PEKS) schemes offer encrypted search functionality to mitigate the impacts of privacy versus data utilization dilemma. PEKS schemes allow any client to encrypt their data under a public key such that the cloud, using the corresponding trapdoor, can later test whether the encrypted records contain certain keywords. Despite this great functionality, the existing PEKS schemes rely on extremely costly operations at the server-side, which often introduce unacceptable cryptographic delays in practical applications. Moreover, while data outsourcing applications usually demand long-term security, existing PEKS schemes do not offer post-quantum security.

In this paper, we propose (to the best of our knowledge) the first post-quantum secure PEKS scheme that is also significantly more computationally efficient than the existing (non-post-quantum) PEKS schemes. By harnessing the recently developed tools in lattice-based cryptography, the proposed scheme significantly outperforms the existing PEKS schemes in terms of computational overhead. For instance, the test (search) operation per item at the cloud side is approximately 36× faster than that of the most prominent pairing-based scheme in the literature (for 192-bit security). The proposed PEKS scheme also offers faster encryptions at the client side, which is suitable for mobile devices.

Keywords: Public Key Encryption with Keyword Search, Cloud Storage, Privacy, Lattice-Based Cryptography.

1 Introduction

The emergence of cloud storage and computing services has revolutionized the IT industry. One of the most prominent cloud services is data storage outsourcing [3], which can drastically reduce the cost of data management via continuous service, expertise and maintenance for the resource-limited clients (e.g., small/medium size businesses). Despite its merits, data outsourcing raises significant privacy concerns for users. Traditional data encryption techniques (e.g.,

Fig. 1: Potential applications of PEKS schemes

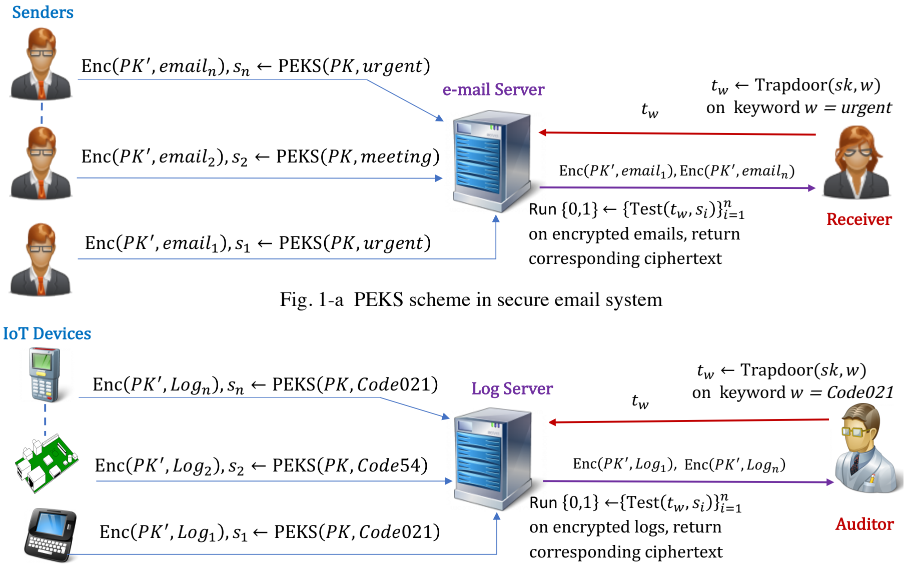

Senders
$\text{Enc}(PK', email_n), s_n \leftarrow \text{PEKS}(PK, urgent)$
$\text{Enc}(PK', email_2), s_2 \leftarrow \text{PEKS}(PK, meeting)$
$\text{Enc}(PK', email_1), s_1 \leftarrow \text{PEKS}(PK, urgent)$
e-mail Server
t_w
$t_w \leftarrow \text{Trapdoor}(sk, w)$
on keyword $w = urgent$
$\text{Enc}(PK', email_1), \text{Enc}(PK', email_n)$
Run $\{0,1\} \leftarrow \{\text{Test}(t_w, s_i)\}_{i=1}^{n}$
on encrypted emails, return
corresponding ciphertext
Receiver
IoT Devices
$\text{Enc}(PK', Log_n), s_n \leftarrow \text{PEKS}(PK, Code021)$
$\text{Enc}(PK', Log_2), s_2 \leftarrow \text{PEKS}(PK, Code54)$
$\text{Enc}(PK', Log_1), s_1 \leftarrow \text{PEKS}(PK, Code021)$
Log Server
t_w
$t_w \leftarrow \text{Trapdoor}(sk, w)$
on keyword $w = Code021$
$\text{Enc}(PK', Log_1)$, $\text{Enc}(PK', Log_n)$
Run $\{0,1\} \leftarrow \{\text{Test}(t_w, s_i)\}_{i=1}^{n}$
on encrypted logs, return
corresponding ciphertext
Auditor

Fig. 1-a PEKS scheme in secure email system

Fig. 1-b PEKS scheme for privacy-preserving audit logging system

symmetric ciphers) can be used to mitigate this concern. However, they abolish the data owner from performing any efficient search (and therefore retrieval) operation over the data that is remotely stored on the cloud. Various privacy enhancing technologies have been proposed towards addressing this limitation.

Searchable Encryption (SE) schemes allow a keyword-based search functionality over the encrypted data. SE schemes are generally applied to a client/server architecture, in which the client stores her encrypted data on a remote server. There are two main types of SE technologies: (i) Dynamic Symmetric SE (DSSE) (e.g., [41,43,30]) permits a client to perform encrypted search on her data via her own private key. DSSE is efficient as it relies on symmetric primitives, but it is rather suitable for a single client outsourcing/searching her own data. (ii) Public Key Encryption with Keyword Search (PEKS) [6] schemes allows *any* client to encrypt her data under a public key such that the server can later test whether the encrypted records contain certain keywords via search trapdoors produced by an entity holding the private key. PEKS is suitable for distributed applications (e.g., email, audit logging for Internet of Things), in which large number of entities generate encrypted data to be searched/analyzed by a particular auditor. *The focus of this paper is on PEKS schemes.* In Fig. 1, we provide some example applications of PEKS with their corresponding system model.

One of the potential application scenarios for PEKS schemes is illustrated in Fig. 1-a. Alice has a number of devices (e.g., desktop, pager), and her email gateway is supposed to route her emails based on the keywords associated with each email. For instance, when one of the originators (i.e., Bob) sends her an email with keyword "urgent" the email should be routed to her pager. To achieve this, Bob encrypts his email using a standard public key encryption and uses

PEKS algorithm to generate a searchable ciphertext of keyword $w =$ "*urgent*" to be associated with the email. Alice can then provide the server with trapdoor t_w computed for keyword w and enable the gateway to test whether any of the stored emails is associated with w via the Test algorithm of PEKS.

As depicted in Fig. 1-b, another possible scenario of employing PEKS schemes is for storing private log files on a remote server. In Internet of Things (IoT) applications, PEKS schemes can enable a set of heterogeneous devices to send their log files (encrypted under the auditor's public key) along with a searchable ciphertext of the keyword related to the logs to a storage server. To look for a specific event, the auditor can compute and send a trapdoor (of any keywords of his interest) to the server and receive all the files that contain the keyword.

1.1 Research Gap

There have been various PEKS schemes with additional features proposed in the literature [37,8,39,11]. We have identified two main research gaps that pose an obstacle towards the adoption of PEKS schemes in practice.

- Extreme Computational Overhead: Most of the proposed PEKS schemes are based on heavy pairing computations, and the schemes that are devised by pairing-free tools are even more costly than their pairing-based counterparts [15]. Despite their elegance, the existing constructions introduce a significant cryptographic delay as the server has to run a costly algorithm (i.e., `Test(.)` requiring at least one pairing operation per item) linear to the size of database.
- Lack of Long-term Security: When dealing with sensitive user data, most applications require long-term cryptographic security (e.g., sensitive medical data). However, due to algorithmic breakthroughs and the rise of powerful computers, the size of cryptographic keys are required to be steadily increased to ensure the same level of security. This causes the conventional cryptographic tools (e.g., RSA, ECC) to become increasingly inefficient. Therefore, there is a need for PEKS schemes that have a more efficient response against increasing key sizes. More importantly, with the predictions on the emergence of quantum computers, it is necessary to devise PEKS schemes that can achieve post-quantum security. However, current PEKS schemes are built on ECC or quadratic residuosity problems [15], which do not offer post-quantum security.

1.2 Our Contribution

Towards addressing aforementioned problems, we introduce the first (to the best of our knowledge) NTRU-Based PEKS scheme which we refer to as NTRU-PEKS hereafter. In the following, we outline our contributions.

- A New PEKS via NTRU: In the initial proposal of PEKS [6], Boneh et al. showed how one could derive a PEKS scheme from an IBE scheme. Abdalla et al. [1] supported this claim and provided requirements for the underlying IBE scheme to ensure the security and correctness of the derived PEKS. This led to the proposal of a large number of PEKS schemes (e.g., [44,15,31]) based on different IBE schemes. In this paper, we rigorously prove that Ducas et al.'s IBE scheme [17] meets these requirements and put forth the first NTRU-based PEKS scheme by leveraging this IBE. Furthermore, we prove the security and

Table 1: Comparison of our NTRU-PEKS scheme with state-of-the-art

Schemes [†]		**Test** (ms)	**PEKS** (ms)	**Trapdoor** (ms)	**QC**[¶] **Resiliency**
BCO*	$\kappa = 80$	3.38	2.53	0.36	χ
	$\kappa = 192$	43.39	46.02	2.69	χ
ZI*	$\kappa = 80$	8.12	16.37	1.05	χ
	$\kappa = 192$	118.65	194.42	5.61	χ
NTRU-PEKS	$\kappa = 80$	**0.34**	**0.69**	5.15	χ
	$\kappa = 192$	**1.23**	**2.50**	17.35	✓

[†] Experimental setup and evaluation metrics are given is Section 5.
[*] BCO and ZI denote Boneh et al.'s scheme [6] and Zhang-Imai scheme [44], respectively.
[¶] QC stands for Quantum Computer.

consistency of our PEKS scheme and suggest parameter sizes to avoid potential decryption errors.

- High Efficiency: We devise a highly efficient PEKS scheme that significantly reduces the cryptographic delay by harnessing the latest advancements in lattice-based cryptography, ring-LWE [38] and fast arithmetic operations over polynomial rings $\mathbb{Z}[x]/(x^N + 1)$. We implement our scheme[1] for 80-bit and 192-bit security and compare its efficiency with the most prominent PEKS schemes. As it is shown in Table 1, our scheme has significantly more efficient `Test` and `PEKS` algorithms than those in in [6,44]. The efficiency of `Test` algorithm is of vital importance, since it is executed by the server linearly with the total number encrypted keywords to be searched. The efficiency of the `PEKS` algorithm facilitates the implementation of PEKS schemes on battery-limited devices.
- Long-term Security: We develop the first practical NTRU-based PEKS that offers long-term security, while being (currently) secure against quantum computers, thanks to the security guarantees of lattice-based cryptography.

2 Preliminaries

In this section, we provide definitions and notations that are used by our scheme. For the sake of compliance, we use the same notation as in [17].

Notations. $a \overset{\$}{\leftarrow} \mathcal{X}$ denotes that a is randomly selected from distribution $\mathcal{X}$. H_i for $i \in \{1, \ldots, n\}$ denotes a hash function which is perceived to behave as a random oracle in this paper. $\mathcal{A}^{\mathcal{O}_1 \ldots \mathcal{O}_n}(.)$ denotes algorithm $\mathcal{A}$ is provided with access to oracles $\mathcal{O}_1 \ldots \mathcal{O}_n$. We denote scalars in plain (e.g., x) and vectors in bold (e.g., $\mathbf{x}$). The norm of a vector $\mathbf{v}$ is denoted by $\|\mathbf{v}\|$. $\lceil x \rfloor$ rounds x to the closest integer. $x \overset{\Delta}{=} y$ means x is defined as y. The function $\gcd(x, y)$ returns the greatest common divisor of values x and y.

[1] The complete implementation can be found on `https://github.com/Rbehnia/NTRUPEKS.git`

2.1 NTRU-Based Cryptographic Tools

Ajtai [2] introduced the Short Integer Solution (SIS) problem and demonstrated the connection between average-case SIS problem and worst-case problems over lattices. Hoffstein et al. [26] proposed an efficient public key encryption scheme called NTRU-based on polynomial rings. Regev [38] introduced the Learning with Error (LWE) problem. The SIS and LWE problems have been used as the building blocks of many lattice-based schemes.

NTRU encryption works over rings of polynomials $\mathcal{R} \triangleq \mathbb{Z}[x]/(x^N + 1)$ and $\mathcal{R}' \triangleq \mathbb{Q}[x]/(x^N + 1)$ which are parametrized with N as a power-of-two integer. $(x^N + 1)$ is irreducible, therefore, $\mathcal{R}'$ is a cyclotomic field. For $f = \sum_{i=0}^{N-1} f_i x^i$ and $g = \sum_{i=0}^{N-1} g_i x^i$ as polynomials in $\mathbb{Q}[x]$, fg denotes polynomial multiplication in $\mathbb{Q}[x]$ while $f * g \triangleq fg \mod (x^N + 1)$ is referred to as convolution product. For an N-dimension anti-circulant matrix $\mathcal{A}_N$ we have $\mathcal{A}_N(f) + \mathcal{A}_N(g) = \mathcal{A}_N(f + g)$, and $\mathcal{A}_N(f) \times \mathcal{A}_N(g) = (f * g)$.

Definition 1. *For prime integer q and $f, g \in \mathcal{R}$, $h = g * f^{-1} \mod q$, the NTRU lattice with h and q is $\Lambda_{h,q} = \{(u, v) \in \mathcal{R}^2 | u + v * h = 0 \mod q\}$. $\Lambda_{h,q}$ is a full-rank lattice generated by $\mathcal{A}_{h,q} = \begin{pmatrix} \mathcal{A}_N(h) & I_N \\ qI_N & 0_N \end{pmatrix}$, where I is an identity matrix.*

Note that one can generate this basis using a single polynomial $h \in \mathcal{R}_q$. However, the lattice generated from $\mathcal{A}_{h,q}$ has a large orthogonal defect which results in inefficiency of standard lattice operations. As proposed by [25], another basis (which is much more orthogonal) can be efficiently [17] generated by selecting $F, G \in \mathcal{R}$ and computing $f * G - g * F = q$. The new base $\boldsymbol{B}_{f,g} = \begin{pmatrix} \mathcal{A}(g) & -\mathcal{A}(f) \\ \mathcal{A}(G) & -\mathcal{A}(F) \end{pmatrix}$ generates the same lattice $\Lambda_{h,q}$

Definition 2. *(Gram-Schmidt norm [22]) Given $\mathbf{B} = (b_i)_{i \in I}$ as a finite basis and $\tilde{\mathbf{B}} = (\tilde{b_i})_{i \in I}$ as its Gram-Schmidt orthogonalization, the Gram-Schmidt norm of $\mathbf{B}$ is $\left\|\tilde{\mathbf{B}}\right\| = \max_{i \in I} \|b_i\|$.*

Using Gaussian sampling, Gentry et al. [22] proposed a technique to use a short basis as trapdoor without disclosing any information about the short basis and prevent attacks similar as in [36].

Definition 3. *An n-dimensional Gaussian function $\rho_{\sigma,c} : \mathbb{R} \to (0, 1])$ is defined as $\rho_{\sigma,c}(x) \triangleq \exp(-\frac{\|x-c\|^2}{2\sigma^2})$. Given a lattice $\Lambda \subset \mathbb{R}^n$, the discrete Gaussian distribution over Λ is $D_{\Lambda,s,c}(\mathbf{x}) = \frac{\rho_{\sigma,c}(\mathbf{x})}{\rho_{\sigma,c}(\Lambda)}$ for all $\mathbf{x} \in \Lambda$.*

If we pick a noise vector over a Gaussian distribution with the radius not smaller than the *smoothing parameter* [34], and reduce the vector to the fundamental parallelepiped of our lattice, the resulting distribution is close to uniform. We formally define this parameter through the following definition.

Definition 4. *(Smoothing Parameter [34]) For any n-dimensional lattice Λ, its dual Λ^* and $\epsilon > 0$, the smoothing parameter $\eta_\epsilon(\Lambda)$ is the smallest $s > 0$ such that $\rho_{1/s\sqrt{2\pi},0}(\Lambda^* \setminus 0) \leqslant \epsilon$. A scaled version of the smoothing parameter is defined in [17] as $\eta_\epsilon^{'} = \frac{1}{\sqrt{2\pi}}\eta_\epsilon(\Lambda)$.*

Gentry et al. [22] defined a requirement on the size of σ related to the smoothing parameter. In [17], Ducas et al. showed that using Kullback-Leibler divergence, the required width of σ can be reduced by factor of $\sqrt{2}$. Based on [18,22,17], for positive integers n, λ, $\epsilon \leqslant 2^{-\lambda/2}/(4\sqrt{2N})$, any basis $\mathbf{B} \in \mathbb{Z}^{N\times N}$ and any target vector $\mathbf{c} \in \mathbb{Z}^{1\times n}$, the algorithm ($\mathbf{v}_0 \leftarrow$ `Gaussian-Sampler`$(\mathbf{B}, \sigma, \mathbf{c})$) as defined in [22,17] is such that $\Delta(D_{\Lambda(\mathbf{B}),\sigma,\mathbf{c}}, \mathbf{v}_0)$ is less than $2^{-\lambda}$.

In this paper, we will use the same algorithm in our `Trapdoor` algorithm.

Definition 5. *(Decision LWE Problem) Given $\mathcal{R} \triangleq \mathbb{Z}[x]/(x^N+1)$ and an error distribution $\mathcal{X}$ over $\mathbb{R}$. For s as a random secret ring element, uniformly random a_i's $\in \mathcal{R}$ and small error elements $e_i \in \mathcal{X}$, the decision LWE problem asks to distinguish between samples of the form $(a_i, a_i s + e_i)$ and randomly selected $(a_i, b_i) \in \mathcal{R} \times \mathcal{R}$.*

Definition 6. *(A tool for computing Gram-Schmidt norm [17]) Let $f \in \mathcal{R}'$, we denote $\bar{f}$ as a unique polynomial in $f \in \mathcal{R}'$ such that $\mathcal{A}(f)^T = \mathcal{A}(\bar{f})$. If $f(x) = \sum_{i=0}^{N-1} f_i x^i$, then $\bar{f}(x) = f_0 - \sum_{i=1}^{N-1} f_{N-i} x^i$.*

2.2 Identity-Based Encryption

Definition 7. *An IBE scheme is a tuple of four algorithms IBE =* (`Setup`, `Extract`,`Enc`,`Dec`) *defined as follows.*

- *$(mpk, msk) \leftarrow$ `Setup`(1^k): On the input of the security parameter(s), this algorithm publishes system-wide public parameters params, outputs the master public key mpk and the master secret key msk.*
- *$sk \leftarrow$ `Extract`(id, msk, mpk):On the input of a user's identity $id \in \{0,1\}^*$, mpk, and msk, this algorithm outputs the user's private key sk.*
- *$c \leftarrow$ `Enc`(m, id, mpk): On the input of a message $m \in \{0,1\}^*$, identity id, and mpk, this algorithm outputs a ciphertext c.*
- *$m \leftarrow$ `Dec`(c, sk): On the input of a ciphertext c, the receiver's private key sk and mpk, this algorithm recovers the message m from the ciphertext c.*

Following the work of [1], the following definition defines anonymity in the sense of [24].

Definition 8. *Anonymity under chosen plaintext attack (IBE-ANO-RE-CPA) for an IBE scheme is defined as follows. Given an IBE scheme, we associate a bit $b \in \{0,1\}$ to the adversary $\mathcal{A}$ in the following experiment.*
Experiment $Exp_{IBE,\mathcal{A}}^{IBE\text{-}ANO\text{-}RE\text{-}CPA\text{-}b}(1^k)$

```
$idSet \leftarrow \emptyset$, $(mpk, msk) \xleftarrow{\$} \mathtt{Setup}(1^k)$
$for\ a\ random\ oracle\ H$
$(id_0, id_1, m) \leftarrow \mathcal{F}^{\mathtt{KeyQuery}(.),H}(find, mpk)$
$c \leftarrow \mathtt{Enc}^H(m, id_b, mpk)$
$b' \leftarrow \mathcal{F}^{\mathtt{KeyQuery}(.),H}(guess, c)$
if $\{id_0, id_1\} \cap idSet = \emptyset$ return $b'$ else, return 0

$\mathtt{KeyQuery}(id)$
    $idSet \leftarrow idSet \cup id$
    $sk \leftarrow \mathtt{Extract}(id, msk, mpk)$
    $return\ sk$
```

$\mathcal{A}$'s advantage in the above experiment is defined as $Adv_{IBE,\mathcal{A}}^{IBE\text{-}ANO\text{-}RE\text{-}CPA}(1^k) = \Pr[Exp_{IBE,\mathcal{A}}^{IBE\text{-}ANO\text{-}RE\text{-}CPA\text{-}1}(1^k) = 1] - \Pr[Exp_{IBE,\mathcal{A}}^{IBE\text{-}ANO\text{-}RE\text{-}CPA\text{-}0}(1^k) = 0]$.

2.3 Public Key Encryption with Keyword Search

A PEKS scheme consists of the following algorithms.

Definition 9. *A PEKS scheme is a tuple of four algorithms PEKS* = (`KeyGen`, `PEKS`, `Trapdoor`, `Test`) *defined as follows.*

- $(pk, sk) \leftarrow \mathtt{KeyGen}(1^k)$: *On the input of the security parameter(s), this algorithm outputs the public and private key pair* (pk, sk)*.*
- $s_w \leftarrow \mathtt{PEKS}(pk, w)$:*On the input of user's public key pk and a keyword* $w \in \{0,1\}^*$*, this algorithm outputs a searchable ciphertext* s_w*.*
- $t_w \leftarrow \mathtt{Trapdoor}(sk, w)$: *On the input of a user's private key sk and a keyword* $w \in \{0,1\}^*$*, this algorithm outputs a trapdoor* t_w*.*
- $b \leftarrow \mathtt{Test}(t_w, s_w)$: *On the input of a trapdoor* $t_w = \mathtt{Trapdoor}(sk, w')$ *and a searchable ciphertext* $s_w = \mathtt{PEKS}(pk, w)$*, this algorithm outputs a bit* $b = 1$ *if* $w = w'$*, and* $b = 0$ *otherwise.*

Definition 10. *Keyword indistinguishability against an adaptive chosen-keyword attack (IND-CKA) is defined as follows. Given a PEKS scheme, we associate a bit* $b \in \{0,1\}$ *to the adversary* $\mathcal{A}$ *in the following experiment.*

Experiment $Exp_{PEKS,\mathcal{A}}^{PEKS\text{-}IND\text{-}CKA\text{-}b}(1^k)$

```
$wSet \leftarrow \emptyset$, $(pk, sk) \leftarrow \mathtt{KeyGen}(1^k)$
for a random oracle $H$
$(w_0, w_1) \leftarrow \mathcal{A}^{\mathtt{TdQuery}(.),H}(find, pk)$
$s_w \leftarrow \mathtt{PEKS}^H(pk, w_b)$
$b' \leftarrow \mathcal{A}^{\mathtt{TdQuery}(.),H}(guess, s_w)$
$if\ \{w_0, w_1\} \cap wSet = \emptyset\ return\ b'\ else,\ return\ 0$

$\mathtt{TdQuery}(w)$
    $wSet \leftarrow wSet \cup w$
    $sk \leftarrow \mathtt{Extract}(w, sk, pk)$
    $return\ sk$
```

$\mathcal{A}$'s advantage in the above experiment is defined as $Adv_{PEKS,\mathcal{A}}^{PEKS\text{-}IND\text{-}CKA}(1^k) = \Pr[Exp_{PEKS,\mathcal{A}}^{PEKS\text{-}IND\text{-}CKA\text{-}1}(1^k) = 1] - \Pr[Exp_{PEKS,\mathcal{A}}^{PEKS\text{-}IND\text{-}CKA\text{-}0}(1^k) = 0]$.

2.4 Consistency of PEKS

Due to the properties of NTRU-based encryption scheme, and following the work of [15], we investigate the consistency of our scheme from two aspects, namely, right-keyword consistency and adversary-based consistency [1]. Right-keyword consistency implies the success of a search query to retrieve records associated with keyword w for which the $\texttt{PEKS}$ algorithm had computed a searchable ciphertext. On the other hand, adversary-based consistency [1] ensures the inability of an adversary to generate two distinct keywords that the $\texttt{Test}$ algorithm returns 1 on the input of a trapdoor for one keyword, and the searchable ciphertext of the other. We define the adversary-based consistency [1] as follows.

Definition 11. *Adversary-based consistency of a PEKS scheme is defined in the following experiment.*
Experiment $\underline{Exp_{PEKS,\mathcal{A}}^{PEKS\text{-}Consist}(1^k)}$

$(pk, sk) \leftarrow \texttt{KeyGen}(1^k)$
for a random oracle H
$(w_0, w_1) \leftarrow \mathcal{A}^H(pk)$, $s_{w_0} \leftarrow \texttt{PEKS}^H(pk, w_0)$
$t_{w_1} \leftarrow \texttt{Trapdoor}^H(pk, w_1)$
if $w_0 \neq w_1$ and $[\texttt{Test}^H(pk, t_{w_1}, s_{w_0}) = 1]$ return 1 else, return 0
$\mathcal{A}$'s advantage in the above experiment is defined as $Adv_{PEKS,\mathcal{A}}^{PEKS\text{-}Consist}(1^k) = \Pr[Exp_{PEKS,\mathcal{A}}^{PEKS\text{-}Consist}(1^k) = 1]$.

3 Proposed Scheme

In this section, we present our scheme that consists of the following algorithms.

$\underline{(h, B) \leftarrow \texttt{KeyGen}(q, N)}$: Given a power-of-two integer N and a prime q, this algorithm works as follows.

1. Compute $\sigma_f \leftarrow 1.17\sqrt{\frac{q}{2N}}$ and select $f, g \leftarrow \mathcal{D}_{N,\sigma_f}$ to compute $\left\|\tilde{\mathbf{B}}_{f,g}\right\|$ and $Norm \leftarrow \max(\|(g, -f)\|, \left\|(\frac{q\bar{f}}{f*\bar{f}+g*\bar{g}}, \frac{q\bar{g}}{f*\bar{f}+g*\bar{g}})\right\|)$. If $Norm < 1.17\sqrt{q}$, proceed to the next step. Otherwise, if $Norm \geq 1.17\sqrt{q}$, this process is repeated by sampling new f and g.
2. Using extended euclidean algorithm, compute $\rho_f, \rho_g \in \mathcal{R}$ and $\mathcal{R}_f, \mathcal{R}_g \in \mathbb{Z}$ such that $\rho_f \cdot f = \mathcal{R}_f \mod (x^N + 1)$ and $\rho_g \cdot g = \mathcal{R}_g \mod (x^N + 1)$. Note that if $\gcd(\mathcal{R}_f, \mathcal{R}_g) \neq 1$ or $\gcd(\mathcal{R}_f, q) \neq 1$, start from the previous step by sampling new f and g.
3. Using extended euclidean algorithm, compute $u, v \in \mathbb{Z}$ such that $u \cdot \mathcal{R}_f + v \cdot \mathcal{R}_g = 1$. Compute $F \leftarrow q \cdot v \cdot \rho_g$, $G \leftarrow q \cdot u \cdot \rho_f$ and $k \leftarrow \lfloor \frac{F*\bar{f}+G*\bar{g}}{f*\bar{f}+g*\bar{g}} \rceil \in \mathcal{R}$ and reduce F and G by computing $F \leftarrow F - k * f$ and $G \leftarrow G - k * g$.
4. Finally, compute $h = g * f^{-1} \mod q$ and $\mathbf{B} = \begin{pmatrix} \mathcal{A}(g) & -\mathcal{A}(f) \\ \mathcal{A}(G) & -\mathcal{A}(F) \end{pmatrix}$ and output $(pk \leftarrow h, sk \leftarrow \mathbf{B})$.

$\underline{s_w \leftarrow \texttt{PEKS}(pk, w)}$: Given cryptographic hash functions $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q^N$ and $H_2 : \{0,1\}^N \times \{0,1\}^N \rightarrow \mathbb{Z}_q^N$, the receiver's public key pk and a keyword $w \in \{0,1\}^*$ to be encrypted, the sender performs as follows.

1. Compute $t \leftarrow H_1(w)$ and pick $r, e_1, e_2 \xleftarrow{\$} \{-1, 0, 1\}^N$, $k \xleftarrow{\$} \{0,1\}^N$.
2. Compute $A \leftarrow r * h + e_1 \in \mathcal{R}_q$ and $B \leftarrow r * t + e_2 + \lfloor \frac{q}{2} \rfloor k \in \mathcal{R}_q$.
3. Finally, the algorithm outputs $s_w = \langle A, B, H_2(k, B) \rangle$.

$t_w \leftarrow \texttt{Trapdoor}(sk, w)$: Given the receiver's private key sk, and a keyword $w \in \{0,1\}^*$, the receiver computes $t \leftarrow H_1(w)$ and using the sampling algorithm $\texttt{Gaussian-Sampler}(B, \sigma, (t, 0))$, samples s and t_w such that $s + t_w * h = t$.

$b \leftarrow \texttt{Test}(pk, t_w, s_w)$: On the input of a receiver's public key pk, a trapdoor t_w and a searchable ciphertext $s_w = \langle A, B, H_2(k, B)\rangle$, this algorithm computes $y \leftarrow \lfloor \frac{B - A * t_w}{q/2} \rceil$ and outputs $b = 1$ if $H_2(y, B) = H_2(k, B)$ and $b = 0$, otherwise.

3.1 Completeness and Consistency

In this section, we show the completeness and consistency of our scheme.

Lemma 1. *Given a public-private key pair $(h, B) \leftarrow \texttt{KeyGen}(q, N)$, a searchable ciphertext $s_w \leftarrow \texttt{PEKS}(pk, w)$, and a trapdoor generate by the receiver $t_w \leftarrow \texttt{Trapdoor}(sk, w)$ our proposed scheme is complete.*

Proof. *To show the completeness of our scheme for $s_w = \langle A, B, H_2(k, B)\rangle$, the* $\texttt{Test}$ *algorithm should return 1 when $\lfloor \frac{B - A * t_w}{q/2} \rceil = k$. To affirm this, we work as follows.*

$$\begin{aligned} B - A * t_w &= (r * t + e_2 + \left\lfloor \frac{q}{2} \right\rfloor k) - (r * h + e_1) * t_w \in \mathcal{R}_q \\ &= r * s + r * h * t_w + e_2 + \lfloor \frac{q}{2} \rfloor k - r * h * t_w - t_w * e_1 \in \mathcal{R}_q \\ &= r * s + e_2 + \lfloor \frac{q}{2} \rfloor k - t_w * e_1 \in \mathcal{R}_q \end{aligned}$$

Given r, e_1, e_2, t_w and s are all short vectors (due to the parameters of our sampling algorithm), all the coefficients of $r * s + e_2 - t_w * e_1$ will be in $(-\frac{q}{4}, \frac{q}{4})$, and therefore, $\lfloor \frac{B - A * t_w}{q/2} \rceil = k$. □

To address right-keyword consistency issues related to the decryption error of encryption over NTRU lattices, we need to make sure that all the coefficients of $z = r * s + e_2 - e_1 * tw$ are in the range $(-\frac{q}{4}, \frac{q}{4})$ and $q \approx 2^{24}$ for $\kappa = 80$ and $q \approx 2^{27}$ for $\kappa = 192$.

Theorem 1. *The NTRU-PEKS scheme is consistent in the sense of Definition 11.*

Proof. Upon inputting q and N, the challenger $\mathcal{C}$ initiates the experiment $(h, \mathbf{B}) \leftarrow \texttt{KeyGen}(q, N)$. It passes h to the adversary $\mathcal{A}$ and keeps $\boldsymbol{B}$ secret.

$(w_0, w_1) \leftarrow \mathcal{A}^{H_1}(pk)$: $\mathcal{A}$ sends $\mathcal{C}$ two keywords (w_0, w_1).

$s_{w_0} \leftarrow \texttt{PEKS}^H(pk, w_b)$: $\mathcal{C}$ computes $A = r * h + e_1$ and $B = r * H(w_0) + e_2 + \left\lfloor \frac{q}{2} \right\rfloor k$ for a random selection of $r, e_1, e_2 \xleftarrow{\$} \{-1, 0, 1\}^N$, $k \xleftarrow{\$} \{0, 1\}^N$, and sends $\langle A, B, H_2(k, B)\rangle$ to $\mathcal{A}$.

$\underline{t_{w_1} \leftarrow \mathtt{Trapdoor}^H(pk, w_b)}$: $\mathcal{C}$ samples short vectors s, t_w such that $s + t_w * h = H(w_1)$ and returns t_w to $\mathcal{A}$.

Following Definition 11, $\mathcal{A}$ wins when $w_0 \neq w_1$, and the `Test` algorithm outputs 1 (i.e, $H_2(k, B) = H_2(y, B)$).

Note that in the above game, $\mathcal{A}$ wins when $w \neq w'$ and $H_2(z_1, z_1') = H_2(z_2, z_2')$. Let's assume $\mathcal{A}$ makes q_1 queries to H_1 and q_2 queries to H_2 oracles. Let E_1 be the event that there exist (x_1, x_2) such that $H_1(x_1) = H_1(x_2)$ and $x_1 \neq x_2$ and let E_2 be the event that there exist two pairs (z_1, z_1') and (z_2, z_2') such that $H_2(z_1, z_1') = H_2(z_2, z_2')$ for $z_1 \neq z_2$ and $z_1' \neq z_2'$. Then if $\Pr[\cdot]$ represents the probability of consistency definition,

$$Adv_{PEKS,\mathcal{A}}^{PEKS\text{-}Consist}(1^k) \leq \Pr[E_1] + \Pr[E_2] + \Pr[Exp_{PEKS,\mathcal{A}}^{PEKS\text{-}Consist} = 1 \wedge \bar{E_1} \wedge \bar{E_2}]$$

Given the domain of our hash functions, the first and second terms are upper bounded by ${(q_1+2)^2}/{N2^{\log_2 q}}$ and ${(q_2+2)^2}/{N2^{\log_2 q}}$, respectively. For the last term, if $H_1(x_1) \neq H_1(x_2)$, then in our scheme, the probability that $B_1 = B_2$ is negligible due to the decryption error. Therefore, $H_2(y_1, B_1) \neq H_2(y_2, B_2)$, hence, the probability of the last term is also negligible. □

3.2 Discussion on Alternative NTRU-based Constructions

Bellare et al. [5] proposed a new variation of public key encryption with search capability called Efficiently Searchable Encryption (ESE). The idea behind ESE is to store a deterministically computed "tag" along with the ciphertext. To respond to search queries, the server only needs to lookup for a tag in a list of sorted tags. This significantly reduces the search time on the server. For ESE to provide privacy, the keywords need to be selected from a distribution with a high min-entropy. To compensate for privacy in absence of high min-entropy distribution for keywords, the authors suggested truncating the output of the hash function to increase the probability of collisions. However, this directly affects the consistency of the scheme and shifts the burden of decrypting unrelated responds to the receiver. As compared to PEKS schemes, in ESE schemes, the tag can be computed from both the plaintext and ciphertext. This highly differentiates the applications of these two searchable encryption schemes.

In this paper, we focused on PEKS scheme as it does not have consistency issues or min-entropy distribution requirement, and fits better for our target real-life applications (as discussed in Section 1). Nevertheless, for the sake of completeness, to extend the advantages of NTRU-based encryption [42] to ESE, we also instantiated an NTRU-based ESE scheme based on the encrypt-with-hash transformation proposed in [5]. We compared it with its counterpart which was instantiated based on El-Gamal encryption. Our implementations of NTRU-based ESE and El-Gamal ESE (developed on elliptic curves) were run on an Intel i7 6700HQ 2.6GHz CPU with 12GB of RAM . We observed that encryption for NTRU-based ESE takes $0.011ms$ where encryption in El-Gamal ESE takes $2.595ms$. As for decryption, NTRU-based ESE takes $0.013ms$ and El-Gamal ESE takes $0.782ms$. The differences are substantial, since the NTRU-base ESE is 236× and 60× faster in encryption and decryption, respectively.

4 Security Analysis

In this section, we focus on analyzing the security of our proposed schemes.

The security of lattice-based schemes is determined by hardness of the underlying lattice problem (in our case, ring-LWE). Based on [21], the hardness of lattice problems is measured using the root Hermite factor. For a vector $\mathbf{v}$ in an N-dimension lattice that is larger than the n^{th} root of the determinant, the root Hermite factor is computed as $\gamma = \frac{\|\mathbf{v}\|}{\det(\Lambda_{h,q})^{1/n}}$. According to [16], for a short planted vector $\mathbf{v}$ in an NTRU lattice, the associated root Hermite factor is computed as $\gamma^n = \frac{\sqrt{N/(2\pi\mathbf{e})}\times\det(\Lambda)^{1/n}\|\mathbf{v}\|}{0.4\times\|\mathbf{v}\|}$. Based on [21,13], $\gamma \approx 1.004$ guarantees intractability and provides at least 192-bit security.

Lemma 2. *If an IBE scheme is IBE-IND-CPA and IBE-ANO-RE-CPA-secure, then it is also IBE-ANO-CPA-secure.*

Proof. Please see appendix.

Following Lemma 2, to establish the security of our NTRU-PEKS scheme, we need to rely on the security of the underlying IBE scheme. Ducas et al. provided the proof of IBE-IND-CPA of their scheme in [17]. Therefore, we are left to prove the anonymity of their scheme via Theorem 2.

Theorem 2. *The IBE scheme of Ducas et al. is anonymous in the sense of Definition 8 under the decision ring-LWE problem.*

Proof. Since the output of the PEKS algorithm of our scheme corresponds to the encryption algorithm of [32,33], for $\mathcal{A}$ to determine s_w corresponds to which keyword with any probability Pr $\geq \frac{1}{2} + \epsilon$ - for any non-negligible ϵ, it has to solve the decision ring-LWE. 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 $A = r * h + e_1$ (for a uniform $r, e_1 \xleftarrow{\$} \{-1, 0, 1\}^N$) and uses $H(w)$ to compute $B = r * H(w) + e_2 + \left\lfloor \frac{q}{2} \right\rfloor k$ that is also statistically close to uniform. Therefore, the adversary's view of $\langle A, B, H_2(A, k)\rangle$ is indistinguishable 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 [35]. □

Theorem 3. *If there exists an adversary $\mathcal{A}$ that can break IND-CKA of NTRU-PEKS scheme as in Definition 10, one can build an adversary $\mathcal{F}$ that uses $\mathcal{A}$ as subroutine and breaks the security of the IBE scheme as in Definition 8.*

Proof. The proof works by having adversaries $\mathcal{F}$ and $\mathcal{A}$ initiating the *find* phase as in Definition 8 and Definition 10 respectively.

Algorithm $\mathcal{F}^{\texttt{KeyQuery}(.),H}(\mathit{find}, \mathit{mpk})$

- $(\mathit{mpk}, \mathit{msk}) \xleftarrow{\$}$ `Setup`(q, N): $\mathcal{F}$ receives mpk and passes it to $\mathcal{A}$.

Algorithm $\mathcal{A}^{TdQuery(.),H}(\mathit{find}, pk)$

- Queries on `TdQuery`(.): Upon such queries, $\mathcal{F}$ queries `KeyQuery`(.) which keeps a list *idSet* maintaining all the previously requested queries and responses. If the submitted query exists, the same response is returned, otherwise, to sample short vectors s, t_w, the oracle uses msk to run $(s, t_w) \xleftarrow{\$}$ `Gaussian-Sampler`$(msk, \sigma, (H(w), 0))$ and passes t_w to $\mathcal{F}$. $\mathcal{F}$ sends t_w to $\mathcal{A}$.

After the *find* phase, a hidden fair coin $b \in \{0, 1\}$ is flipped.
Execute $(w_0, w_1) \leftarrow \mathcal{A}^{\mathtt{TdQuery}(.),H}(guess, pk)$

- Upon receiving (w_0, w_1), $\mathcal{F}$ selects a message $m \in \{0\}^N$ and calls `Enc`(m, w_0, w_1) that runs encryption on (w_b, m) which works as in Definition 7 and outputs $s_w = \langle A, B, H_2(k, B)\rangle$. $\mathcal{F}$ relays s_w to $\mathcal{A}$.

Finally, $\mathcal{A}$ outputs its decision bit $b' \in \{0, 1\}$. $\mathcal{F}$ also outputs b' as its response. Omitting the terms that are negligible in terms of q and N, the upper bound on *IND-CKA* of NTRU-PEKS is as follows.

$$Adv_{\mathcal{A}}^{\textit{PEKS-IND-CKA}}(q, N) \leq Adv_{\mathcal{F}}^{\textit{NTRU-IBE-ANO-CPA}}(q, N)$$

□

Secure channel requirement. Baek et al. [4] highlighted the requirement of a secure channel for trapdoor transmission between the receiver and the server and proposed the notion of Secure-Channel Free (SCF) PEKS schemes where the keywords are encrypted by both the server's and receiver's public key. Offline keyword-guessing attack, as introduced by Byun et al. [12], implies the ability of an adversary to find which keyword was used to generate the trapdoor. This inherent issue is due to low-entropy nature of the commonly selected keywords and public availability of the encryption key [10]. Since Byun et al.'s work [12], there have been many attempts in proposing schemes that are secure against keyword guessing attacks [27,20,28]. However, in all the proposals, once the trapdoor is received by the server, the keyword guessing attacks remain a perpetual problem [28]. Jeong et al. [28] showed the trade-off between the security of a PEKS scheme against keyword-guessing attacks and its consistency - by mapping a trapdoor to multiple keywords. For our scheme, we can assume a conventional or even post quantum secure [9] SSL/TLS connection between the receiver and the server. We believe such reliable protocols provide the best mean for communicating trapdoors to the servers. Establishing a secure line through SSL/TLS could be much more efficient than using any public key encryption as in SFC-PEKS. Since in such protocols, after the hand shake protocol, all communications are encrypted using symmetric encryption.

5 Performance Evaluation

We first describe our experimental setup and evaluation metrics. We then provide a detailed performance analysis of our scheme by also comparing its efficiency with the pairing-based schemes proposed in [6,44] . To the best of our knowledge, and based on [10], the selected pairing-based counterparts are the most efficient schemes proposed in random oracle and standard models.

Table 2: Analytical performance analysis and comparison.

Schemes	**Computation**			**Storage**			
	`Test`	`PEKS`	`Trapdoor`	***PK Size***	**SK Size**	***SC Size***	***TD Size***
NTRU-PEKS	$Conv$	$2Conv^{\ddagger}$	$GSamp$	$N\lvert q\rvert$	$2N \log_2(2s\pi)^{\dagger}$	$3N\lvert q\rvert$	$N\lvert q\rvert$
BCO [6]	bp	$1bp + sm$	sm	$2\lvert q'\rvert$	$\lvert q'\rvert$	$2\lvert q'\rvert + \lvert q'\rvert$	$2\lvert q'\rvert$
ZI [44]	$ex + bp$	$2sm + 2ex+ 2bp$	$sm + 1pa$	$2\lvert q'\rvert$	$\lvert q'\rvert$	$2 \times 18\lvert q'\rvert +2\lvert q'\rvert$	$2\lvert q'\rvert + \lvert q'\rvert$

For 192-bit security, we set $N = 1024$ and $q \approx 2^{27}$ which gives us a root Hermite factor $\gamma = 1.0042$ for our scheme and for BCO and ZI schemes, we set $q' \approx 2^{192}$.

PK and SK denote public key and private key, respectively. SC and TD refer to the searchable ciphertext and trapdoor, receptively. $Conv$ denotes convolution product as defined in Section 2. $GSamp$ denotes a Gaussian Sampling function as in [17]. bp denotes a bilinear pairing operation [7], pa and sm denote point addition and scalar multiplication in $\mathbb{G}$, respectively, and ex denotes exponentiation in $\mathbb{G}_T$.

Public key, private key and SC are stored on the sender, receiver and server's machines, respectively. `PEKS`, `Trapdoor` and `Test` algorithms are run by the the senders, receiver and server machines, respectively.

‡ With a slight storage sacrifice, sender can pre-compute one of the convolution products.

† The value of s defines the norm of the Gram-Schmidt coefficient. In [17], the authors set the norm $s \approx \sqrt{\frac{q\mathbf{e}}{2}}$, where $\mathbf{e}$ is the base of natural logarithm.

Table 3: Parameter sizes of our scheme and its pairing-based counterparts

Schemes	***Public Key Size***	***Private Key Size***	***SC Size***	***TD Size***
NTRU-PEKS	27.2 Kb	32 Kb	52 Kb	27 Kb
BCO [6]	0.38 Kb	0.19 Kb	0.57 Kb	0.38 Kb
ZI [44]	0.76 Kb	0.19 Kb	0.89 Kb	0.57 Kb

All schemes are implemented for 192 bits of security.

5.1 Experimental Setup and Evaluation Metrics

We implemented our PEKS scheme in C++[2], using NTL [40] and GNU MP [23] libraries. The implementations of the pairing-based counterparts [6,44] were obtained from MIRACL library. We used the MIRACL suggested elliptic curves, MNT (with embedding degree $k = 6$) and KSS (with embedding degree $k = 18$) for 80-bit and 192-bit security, respectively. The implementations were done on an Intel Core i7-6700HQ laptop with a 2.6GHz CPU and 12GB RAM. Our evaluation metrics are computation, storage, and communication that are required by the sender, receiver and server.

5.2 Performance Evaluation and Comparisons

The `Test` algorithm of our scheme only requires one convolution product, which is much more efficient than the bilinear pairing operation required in all of the existing pairing-based PEKS schemes. Referring to Table 1, running the `Test` algorithm for one keyword and one record our scheme is 36× and 97× faster than BCO and ZI schemes, respectively. This gap significantly increases as the

[2] `https://github.com/Rbehnia/NTRUPEKS.git`

Fig. 2: Search Time of Server

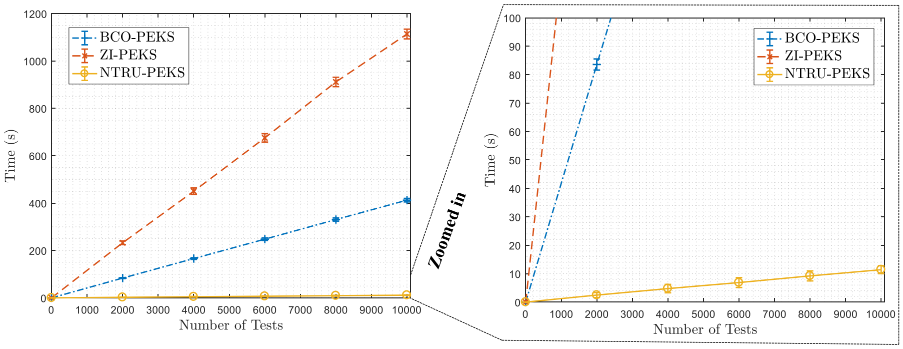
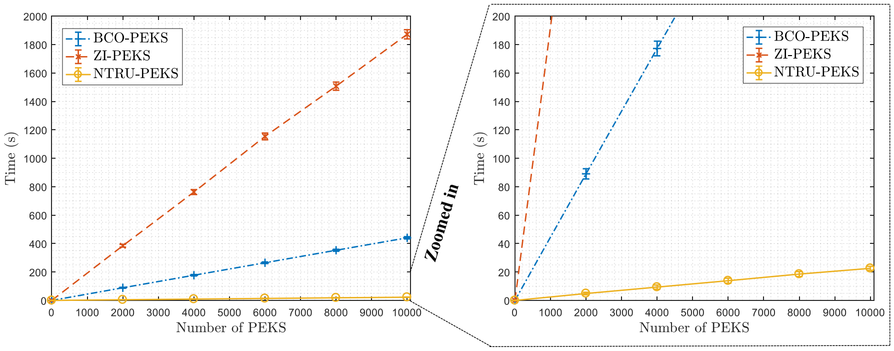


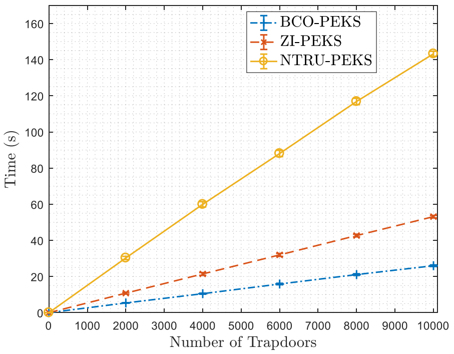
Fig. 3: Efficiency of PEKS algorithm



number of keywords/records increases, for instance, as depicted in Fig. 2, the search time for 10000 (with distinct keywords) records in database, is 10s in our scheme, and 400s and 1100s for BCO and ZI schemes, respectively. For 10000 records (which is rather small comparing to the number of records in actual databases), our scheme is 40 times faster than Boneh et al.'s scheme. For real-world cases with a large database, our scheme seems to be the only practical solution at this moment. We believe that this is one of the main aspects of our scheme which makes it an attractive candidate to be implemented for real-world applications. As it is shown in Table 2, the dominant operations of the PEKS algorithm in our scheme are two convolution products of form $x_1 * x_2$. However, since one of the operands has very small coefficients (i.e., $r \xleftarrow{\$} \{-1, 0, 1\}^N$), the convolution products can be computed very rapidly. Specifically, in our case, since N has been selected as a power-of-two integer, the convolution product can be computed in $N \log N$ operations by Fast Fourier Transform. In Fig. 3, we compare the efficiency of the PEKS algorithm of our scheme with ones in [6,44]. Generating one searchable encryption in our scheme is 19× and 78× faster than that of BCO and ZI schemes, respectively. Therefore, in our scheme, the sender can generate 2000 searchable encryptions with distinct keywords in 4s while this time is increased to 100s and 400s in BCO and ZI schemes, respectively. The sender needs to store the receiver's public key of size $N|q|$, referring to Table 3, for 192-bit security, it can be up to $27.2Kb$. The resulting searchable encryption of our `PEKS` algorithm is to be sent to the server is of size $52Kb$, based on Table 3. This is larger than the searchable encryption size of BCO and ZI scheme. Due to the structure of our `PEKS` algorithm, the computation of A in searchable ciphertext can be done prior to having knowledge of the keyword. Therefore, with a slight storage sacrifice (i.e., storing $N|q|$ bits), the `PEKS` algorithm can become twice as fast.

The `Trapdoor` algorithm in our scheme requires a Gaussian Sampling similar as in [22,17]. This is the most costly operation in our scheme. As it is shown in Table 1, for 192-bit security, one trapdoor generation is 6.4× and 3× slower than those of BCO and ZI schemes, respectively. In Fig. 4 we compare the efficiency of the `Trapdoor` algorithm of our scheme with the ones in [6,44]. This algorithm in Boneh et al's scheme only requires one scalar multiplication and consequently, it is the fastest. `Trapdoor` algorithm in BCO scheme is capable of generating 2000 trapdoors for distinct keywords in 5s, ZI scheme generates the same number of trapdoor in 10s comparing to our scheme which takes 30s. Each trapdoor in our scheme is $27Kb$ which is much larger than those in BCO and ZI schemes. With the sacrifice of storage, the receiver can pre-compute and securely store the trapdoors locally. As discussed in Section 4, the trapdoors are to be transmitted to the server via a secure channel.

Fig. 4: Efficiency of `Trapdoor` algorithm



5.3 Discussion

To the best of our knowledge, our scheme is the first post-quantum secure PEKS scheme in the literature. Except the `Trapdoor` algorithm, which is only run $\mathcal{O}(1)$ times for each keyword, our algorithm enjoys from a very efficient `PEKS` and `Test` algorithms. While the `PEKS` algorithm is run $\mathcal{O}(1)$ times for each keyword and record, the efficiency of our `Test` algorithm, which is run $\mathcal{O}(L)$ times (for a database of size L), significantly decreases the search time on the server and minimizes the cryptographic end-to-end delay. Note that achieving a low end-to-end delay is of great importance, since even small delays (e.g., a few milliseconds) could incur significant financial costs for companies like Amazon [19].

One limitation of our scheme is that its searchable ciphertext sizes are larger than its pairing-based counterparts, as our scheme relies on NTRU. This incurs a larger storage overhead on the server. However, given its significantly efficiency advantage for `PEKS` and especially critical algorithm `Test`, and also high storage capability of the modern cloud servers with a relatively low storage cost, this can be considered as a highly favorable trade-off. Moreover, as discussed, a faster response time (i.e., lower end-to-end delay) seems a much critical ecumenical parameter for modern cloud services than having a relatively higher storage.

6 Related Work

Searchable encryption can be instantiated from both symmetric or asymmetric key settings. Song et al. [41] proposed the first SE scheme that relies on symmetric key cryptography. Kamara et al. [29] proposed the first DSSE scheme to address the limitation of its static ancestors. While being highly efficient, symmetric SE schemes are more suitable for applications that involve a single client who outsources her own data to the cloud relying on her private key.

In this paper, given the target applications that need multiple heterogeneous entities to create searchable encrypted data, our focus is on SE schemes instanced in asymmetric settings. In particular, we concentrate on PEKS, as it requires neither specific probability distributions on keywords nor performance/consistency trade-offs as dictated by some other asymmetric alternatives (e.g., ESE as discussed in Section 3.2). In PKES, decryption and trapdoor generation take place using the private key of the receiver, while any user can use the corresponding public key to generate searchable ciphertext. With a few exceptions, all of the proposed PEKS schemes are developed using costly bilinear pairing operations. The first instance of pairing-free PEKS schemes is constructed by Crescenzo and Saraswat [15] based on the IBE scheme in [14], which is constructed using quadratic residue for a composite modulus. Khader [31] proposed the first instance of such schemes in the standard model based on a k-resilient IBE, she also put forth a scheme which supports multiple-keyword search. Nonetheless, due to their costly operations, the proposed schemes are not practical to be implemented in real-world applications.

7 Conclusion

In this paper, we proposed (to the best of our knowledge) the first NTRU-based PEKS scheme, which harnesses some of the most recent cryptographic tools in lattice-based cryptography, IBE scheme based on ring-LWE and efficient polynomial arithmetics at the same time. We formally proved that our scheme is secure and consistent in IND-CKA model, and also showed that our base IBE scheme achieves anonymity property required by our PEKS construction. Our theoretical and experimental analysis confirmed that our NTRU-based PEKS scheme is significantly more computationally efficient than its most efficient pairing-based counterparts at the server and sender side, which offer the lowest end-to-end cryptographic delay among the existing PEKS schemes. In addition to its efficiency, our PEKS scheme demonstrated a much smoother performance for increasing key sizes and inherits the (current) post-quantum security properties of its underlying NTRU primitives. The high efficiency and long-term security of NTRU-based PEKS are expected to pave a path towards potential consideration of PEKS schemes for real-life applications.

References

1. Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., Shi, H.: Searchable encryption revisited consistency properties, relation to anonymous IBE, and extensions. In: Advances in Cryptology – CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA. Proceedings. pp. 205–222. Springer Berlin Heidelberg (2005)
2. Ajtai, M.: Generating hard instances of lattice problems. In: Twenty-Eighth Annual ACM Symposium on Theory of Computing, Philadelphia, PA, USA. Proceedings. pp. 99–108. ACM (1996)
3. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I., et al.: Above the clouds: A berkeley view of cloud computing. Tech. rep., EECS Department, University of California, Berkeley (2009)
4. Baek, J., Safavi-Naini, R., Susilo, W.: Public key encryption with keyword search revisited. In: Computational Science and Its Applications – ICCSA, Perugia, Italy. Proceedings, Part I. pp. 1249–1259. Springer Berlin Heidelberg (2008)
5. Bellare, M., Boldyreva, A., O'Neill, A.: Deterministic and efficiently searchable encryption. In: Advances in Cryptology - CRYPTO 2007: 27th Annual International Cryptology Conference, Santa Barbara, CA, USA. Proceedings. pp. 535–552. Springer Berlin Heidelberg (2007)
6. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public Key Encryption with Keyword Search. In: Advances in Cryptology - EUROCRYPT 2004: International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland. Proceedings. pp. 506–522. Springer Berlin Heidelberg (2004)
7. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Advances in Cryptology — CRYPTO 2001: 21st Annual International Cryptology Conference, Santa Barbara, California, USA. Proceedings. pp. 213–229. Springer Berlin Heidelberg (2001)
8. Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: 4th Theory of Cryptography Conference, TCC, Amsterdam, The Netherlands. Proceedings. pp. 535–554. Springer Berlin Heidelberg (2007)
9. Bos, J.W., Costello, C., Naehrig, M., Stebila, D.: Post-quantum key exchange for the tls protocol from the ring learning with errors problem. In: IEEE Symposium on Security and Privacy S&P, Â Fairmont, San Jose, California. Proceedings. pp. 553–570. IEEE Computer Society (2015)
10. Bösch, C., Hartel, P., Jonker, W., Peter, A.: A survey of provably secure searchable encryption. ACM Comput. Surv. 47(2), 18:1–18:51 (2014)
11. Bringer, J., Chabanne, H., Kindarji, B.: Error-tolerant searchable encryption. In: IEEE International Conference on Communications, Dresden, Germany. Proceedings. pp. 1–6. IEEE (2009)
12. Byun, J.W., Rhee, H.S., Park, H.A., Lee, D.H.: Off-line keyword guessing attacks on recent keyword search schemes over encrypted data. In: Third VLDB Workshop, SDM 2006, Seoul, Korea. Proceedings. pp. 75–83. Springer Berlin Heidelberg (2006)
13. Chen, Y., Nguyen, P.Q.: BKZ 2.0: Better lattice security estimates. In: Advances in Cryptology – ASIACRYPT 2011: 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea. Proceedings. pp. 1–20. Springer Berlin Heidelberg (2011)

14. Cocks, C.: An identity based encryption scheme based on quadratic residues. In: Cryptography and Coding: 8th IMA International Conference, Cirencester. Proceedings. pp. 360–363. Springer Berlin Heidelberg (2001)
15. Di Crescenzo, G., Saraswat, V.: Public key encryption with searchable keywords based on jacobi symbols. In: Progress in Cryptology – INDOCRYPT: 8th International Conference on Cryptology, Chennai, India. Proceedings. pp. 282–296. Springer Berlin Heidelberg (2007)
16. Ducas, L., Durmus, A., Lepoint, T., Lyubashevsky, V.: Lattice signatures and bimodal gaussians. In: Advances in Cryptology – CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, CA, USA. Proceedings, Part I. pp. 40–56. Springer Berlin Heidelberg (2013)
17. Ducas, L., Lyubashevsky, V., Prest, T.: Efficient identity-based encryption over NTRU lattices. In: Advances in Cryptology – ASIACRYPT 2014: 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan. Proceedings, Part II. pp. 22–41. Springer Berlin Heidelberg (2014)
18. Ducas, L., Nguyen, P.Q.: Faster gaussian lattice sampling using lazy floating-point arithmetic. In: Advances in Cryptology – ASIACRYPT 2012: 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China. Proceedings. pp. 415–432. Springer Berlin Heidelberg (2012)
19. Eaton, K.: How One Second Could Cost Amazon $1.6 Billion In Sales (2012, Accessed: 2017), https://www.fastcompany.com/1825005/how-one-second-could-cost-amazon-16-billion-sales
20. Fang, L., Susilo, W., Ge, C., Wang, J.: Public key encryption with keyword search secure against keyword guessing attacks without random oracle. Information Sciences 238, 221 – 241 (2013)
21. Gama, N., Nguyen, P.Q.: Predicting lattice reduction. In: Advances in Cryptology – EUROCRYPT 2008: 27th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Istanbul, Turkey. Proceedings. pp. 31–51. Springer Berlin Heidelberg (2008)
22. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Fortieth Annual ACM Symposium on Theory of Computing, STOC, Victoria, Canada. Proceedings. pp. 197–206. ACM (2008)
23. Granlund, T., the GMP development team: GNU MP: The GNU Multiple Precision Arithmetic Library (2012), http://gmplib.org/
24. Halevi, S.: A sufficient condition for key-privacy. IACR Cryptology ePrint Archive 2005, 5 (2005)
25. Hoffstein, J., Howgrave-Graham, N., Pipher, J., Silverman, J.H., Whyte, W.: NTRUSign: Digital signatures using the ntru lattice. In: Topics in Cryptology – CT-RSA: The Cryptographers' Track at the RSA Conference, San Francisco, CA, USA. Proceedings. pp. 122–140. Springer Berlin Heidelberg (2003)
26. Hoffstein, J., Pipher, J., Silverman, J.H.: NTRU: A ring-based public key cryptosystem. In: Algorithmic Number Theory: Third International Symposiun, ANTS-III Portland, Oregon, USA. Proceedings. pp. 267–288. Springer Berlin Heidelberg (1998)
27. Hu, C., Liu, P.: A secure searchable public key encryption scheme with a designated tester against keyword guessing attacks and its extension. In: Advances in Computer Science, Environment, Ecoinformatics, and Education, CSEE, Wuhan, China, August 21-22, 2011. Proceedings, Part II. pp. 131–136. Springer Berlin Heidelberg (2011)

28. Jeong, I.R., Kwon, J.O., Hong, D., Lee, D.H.: Constructing PEKS schemes secure against keyword guessing attacks is possible? Computer Communications 32(2), 394 – 396 (2009)
29. Kamara, S., Lauter, K.: Cryptographic cloud storage. In: Financial Cryptography and Data Security, Canary Islands, Spain. Proceedings. pp. 136–149. Springer Berlin Heidelberg (2010)
30. Kamara, S., Papamanthou, C., Roeder, T.: Dynamic searchable symmetric encryption. In: ACM Conference on Computer and Communications Security, CCS, Raleigh, NC, USA. Proceedings. pp. 965–976. ACM (2012)
31. Khader, D.: Public key encryption with keyword search based on k-resilient ibe. In: International Conference on Computational Science and Its Applications, Kuala Lumpur, Malaysia. Proceedings, Volume Part III. pp. 1086–1095. ICCSA'07, Springer Berlin Heidelberg (2007)
32. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Advances in Cryptology - EUROCRYPT 2010: International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera. Proceedings. pp. 1–23. Springer Berlin Heidelberg (2010)
33. Lyubashevsky, V., Peikert, C., Regev, O.: A toolkit for ring-LWE cryptography. In: Advances in Cryptology - EUROCRYPT 2013: International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece. Proceedings. pp. 35–54. Springer Berlin Heidelberg (2013)
34. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. SIAM Journal on Computing 37(1), 267–302 (2007)
35. Micciancio, D., Regev, O.: Lattice-based cryptography. In: Post-Quantum Cryptography. pp. 147–191. Springer Berlin Heidelberg (2009)
36. Nguyen, P.Q., Regev, O.: Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. J. Cryptol. 22(2), 139–160 (2009)
37. Park, D.J., Kim, K., Lee, P.J.: Public key encryption with conjunctive field keyword search. In: 5th International Workshop of Information Security Applications WISA, Jeju Island, Korea. Proceedings. pp. 73–86. Springer Berlin Heidelberg (2004)
38. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Thirty-seventh Annual ACM Symposium on Theory of Computing, STOC, Hunt Valley, MD. Proceedings. pp. 84–93. ACM (2005)
39. Shi, E., Bethencourt, J., Chan, T.H.H., Song, D., Perrig, A.: Multi-dimensional range query over encrypted data. In: IEEE Symposium on Security and Privacy, S&P, Oakland, California, USA. Proceedings. pp. 350–364. IEEE Computer Society (2007)
40. Shoup, V.: NTL: A library for doing number theory. `http://www.shoup.net/ntl` (2003)
41. Song, D.X., Wagner, D., Perrig, A.: Practical techniques for searches on encrypted data. In: IEEE Symposium on Security and Privacy S&P, Berkeley, California, USA. Proceedings. pp. 44–55. IEEE Computer Society (2000)
42. Whyte, W., Howgrave-Graham, N., Hoffstein, J., Pipher, J., Silverman, J.H., Hirschhorn, P.S.: IEEE p1363. 1 Draft 10: Draft Standard for Public Key Cryptographic Techniques Based on Hard Problems over Lattices. IACR Cryptology EPrint Archive 2008, 361 (2008)
43. Yavuz, A.A., Guajardo, J.: Dynamic searchable symmetric encryption with minimal leakage and efficient updates on commodity hardware. In: Selected Areas in Cryptography - SAC 2015: 22nd International Conference, Sackville, NB, Canada. Proceedings. pp. 241–259. Springer Berlin Heidelberg (2015)

44. Zhang, R., Imai, H.: Generic combination of public key encryption with keyword search and public key encryption. In: Bao, F., Ling, S., Okamoto, T., Wang, H., Xing, C. (eds.) Cryptology and Network Security, CANS, Singapore. Proceedings. pp. 159–174. Springer Berlin Heidelberg (2007)

Appendix

The following proof is obtained from [1].

Proof of Lemma 1.

Let $\mathcal{A}$ be an adversary on an IBE-ANO-CPA-secure scheme. We can build another adversaries $\mathcal{A}_1$ and $\mathcal{A}_3$ attacking the IBE-IND-CPA, and another adversary $\mathcal{A}_2$ attacking ANO-RE-CPA of the IBE scheme such that

$$\Pr[Exp_{IBE,\mathcal{A}}^{IBE\text{-}ANO\text{-}CPA\text{-}1}(k) = 1] - \Pr[Exp_{IBE,\mathcal{A}}^{IBE\text{-}ANO\text{-}RE\text{-}1}(k) = 1] \leqslant Adv_{IBE,\mathcal{A}_1}^{IBE\text{-}IND\text{-}CPA}(1^k)$$

$$\Pr[Exp_{IBE,\mathcal{A}}^{IBE\text{-}ANO\text{-}RE\text{-}1}(k) = 1] - \Pr[Exp_{IBE,\mathcal{A}}^{IBE\text{-}ANO\text{-}RE\text{-}0}(k) = 1] \leqslant Adv_{IBE,\mathcal{A}_2}^{IBE\text{-}ANO\text{-}RE}(1^k)$$

$$\Pr[Exp_{IBE,\mathcal{A}}^{IBE\text{-}ANO\text{-}RE\text{-}0}(k) = 1] - \Pr[Exp_{IBE,\mathcal{A}}^{IBE\text{-}ANO\text{-}CPA\text{-}0}(k) = 1] \leqslant Adv_{IBE,\mathcal{A}_3}^{IBE\text{-}IND\text{-}CPA}(1^k)$$

Adding the above equations will conclude this proof. □