Journal of Information Security and Applications 58 (2021) 102709

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

Designated-ciphertext searchable encryption

Zi-Yuan Liu [a], Yi-Fan Tseng [a,*], Raylin Tso [a], Masahiro Mambo [b]

[a] Department of Computer Science, National Chengchi University, Taipei 11605, Taiwan
[b] Institute of Science and Engineering, Kanazawa University, Kakuma-machi, Kanazawa 920-1192, Japan

ARTICLE INFO

Keywords:
Designated-ciphertext
Insider-keyword-guessing attacks
Lattices
Quantum-resistant
Searchable encryption

ABSTRACT

Public-key encryption with keyword search (PEKS), proposed by Boneh et al. (2004), allows users to search encrypted keywords without losing data privacy. Although extensive studies have been conducted on this topic, only a few have focused on insider-keyword-guessing attacks (IKGA) that can reveal a user's sensitive information. In particular, after receiving a trapdoor used to search ciphertext from a user, a malicious insider (*e.g.*, a server) can randomly encrypt possible keywords using the user's public key, and then test whether the trapdoor corresponds to the selected keyword. This paper introduces a new concept called *designated-ciphertext searchable encryption* (DCSE), which provides the same desired functionality as a PEKS scheme and prevents IKGA. Each trapdoor in DCSE is designated to a specific ciphertext, and thus malicious insiders cannot perform IKGA. We further propose a generic DCSE construction that employs identity-based encryption and a key encapsulation mechanism. We provide formal proofs to demonstrate that the generic construction satisfies the security requirements. Moreover, we provide a lattice-based instantiation whose security is based on NTRU and ring-learning with errors assumptions; the proposed scheme is thus considered to be resistant to the quantum-computing attacks.

1. Introduction

With the development of the 5G and Internet of Things (IoT), the importance of cloud storage is increasing. However, because the cloud providers cannot be easily trusted, to avoid data leakage or abuse, data owners need to ensure the privacy of sensitive data. A straightforward method is encrypting data before the data is uploaded to cloud servers. However, encrypted data loses its processing flexibility and cannot be used for useful operations, such as sorting or searching. Specifically, search functionality is important for cloud storage. If a data owner wants to search for some specific files among large encrypted data sets, it becomes necessary to download and decrypt all the data to search, which is impractical and resource-consuming. To resolve this issue, Song et al. [1] proposed the first searchable encryption (SE) that allows the ciphertext to be searched using the corresponding trapdoor. However, because their construction is based on a symmetric key primitive, only the owner of a particular secret key can generate any corresponding ciphertext and trapdoor. Hence, as with a symmetric cryptosystem, their work faces the key distribution problem when it is deployed in public cloud environments.

1.1. Public-key encryption with keyword search

To circumvent the issue of the symmetric searchable encryption and allow multiple data owners to easily generate different ciphertexts for a single data receiver, Boneh et al. [2] proposed the first public-key encryption with keyword search (PEKS). The scheme, unlike Song et al.'s work [1], is built on a public-key cryptosystem. The PEKS scheme has three entities: data owner (Alice), data receiver (Bob), and cloud server. Consider the following scenario: Alice wants to store files that can be accessed and searched by Bob without any leakage of information to the cloud server. Therefore, in addition to encrypting files using Bob's public key pk, she also encrypts the related keywords of the files using a $PEKS$ algorithm that allows ciphertexts to be searched, *e.g.*, $Enc_{\mathsf{pk}}(\text{file}) \| PEKS(\mathsf{pk}, \text{“pkc”}) \| \cdots \| PEKS(\mathsf{pk}, \text{“crypto”}) \|$; she uploads the encrypted file along with the encrypted keywords to the cloud server. If Bob would like to request the cloud server to search for any encrypted files containing the keyword “crypto,” he first generates a trapdoor for “crypto” by using his private key, and then sends the trapdoor to the cloud server. Using this trapdoor, the cloud server can test the value of all encrypted keywords, determine which value is generated by the keyword “crypto,” and return the corresponding encrypted file to Bob.

* Corresponding author.
E-mail addresses: zyliu@cs.nccu.edu.tw (Z. Liu), yftseng@cs.nccu.edu.tw (Y. Tseng), raylin@cs.nccu.edu.tw (R. Tso), mambo@ec.t.kanazawa-u.ac.jp (M. Mambo).

https://doi.org/10.1016/j.jisa.2020.102709

Available online 15 February 2021

Compared with symmetric searchable encryption, PEKS is more suitable for purposes such as cloud services, IoT, and email system. In the first two decades of the twenty-first century, numerous PEKS schemes suitable for different scenarios that provide notable functionality were proposed.

1.2. Motivation

Even though numerous PEKS schemes have been advanced, their security precautions are inadequate. For instance, as most of the proposed schemes assume the insider (*e.g.*, cloud server, mail server, or IoT gateway) to be trustworthy and thus do not consider possible attacks from insiders. Byun et al. [3] first pointed out that it may cause problems in PEKS. In actual fact, because of the small number of commonly used keywords, a insider can guess some keywords from a trapdoor and obtain some useful information; this attack is called an *insider-keyword-guessing attacks* (IKGA). More concretely, after receiving a trapdoor from the authorized data receiver, a malicious insider can encrypt possible keywords using the data receiver's public key. Then, the insider can test whether the trapdoor corresponds to the selected keywords. As mentioned by Byun et al. the probability that malicious insider obtain the keyword is about $1/2^{18}$. Additionally, in certain applications with smaller keyword space (*e.g.,* email application), the probability of success will be higher. Since the keywords selected from the data sender are usually related to the encrypted file, if the malicious insider can get the keywords through IKGA, then the encrypted content can be guessed in advance. Therefore, if the ciphertext related keywords are guessed, the confidentiality of ciphertext can be said to be broken.

On the other hand, Shor [4,5] demonstrates the existence of quantum algorithms that can break some difficult assumptions in number theory (*i.e.,* the discrete logarithm assumption and the integer factoring assumption), the potential threat of quantum computers to modern cryptography is foreseeable. Arute et al. [6] recently proposed a 53-qubit quantum computer. Scholars believe that quantum computing will become mature in the twenty-first century. Although many studies on quantum-resistant PEKS [7–10] to resist quantum computing attacks have been proposed, only Mao et al. [9] is IKGA secure. The security of this scheme is based on the learning with errors assumption, which has been proven to be as difficult as solving worst-case lattice problems [11]. However, this scheme not sufficiently practical because of size constraints; public keys and private keys would thus be as large as hundreds of megabytes.

Actually, Abdalla et al. [12] proved that an IND-ANON-ID-CPA secure identity-based encryption (IBE) scheme can obtain a secure PEKS scheme. However, how to support IKGA security based on it is still unknown. In this paper, we consider the following question:

> *Can we instantiate a cryptographic primitive that is quantum-resistant and supports search functionality as well as the strength against IKGA using IND-ANON-ID-CPA secure IBE scheme as a building block?*

1.3. Our results

To answer our question, we first introduce a new cryptographic primitive, called "designated-ciphertext searchable encryption" (DCSE), to provide the same functionality as a PEKS scheme with additional strength against IKGA.

In contrast to PEKS, each trapdoor in DCSE is designated to a ciphertext. Thus, an adversarial insider cannot adaptively select keywords to generate various ciphertexts and then test these ciphertexts with the trapdoor received. In this paper, by combining a key encapsulation mechanism (KEM) with a pseudorandom generator, we use IBE to formalize a generic construction of our DCSE scheme under the standard model. Moreover, we implemented a lattice-based DCSE based on NTRU and ring-learning with errors (ring-LWE) assumptions; our implementation is more efficient, more secure, and more practical than other advanced schemes.

1.3.1. Designated-ciphertext searchable encryption

Attacks against conventional PEKS schemes succeed because insiders can adaptively generate ciphertext for any keyword. Therefore, our strategy, in DCSE, is to prevent insiders from producing valid ciphertexts themselves that can be effectively tested against trapdoor received from the authorized data receiver. Consider a scenario where Alice wants to encrypt some files and upload them to a cloud server, and she wants these encrypted files to be searchable within the cloud. In addition, she wants to avoid any IKGA. Alice first executes $DCSE(\mathsf{pk}, w_i)$ for keywords $w_1, \ldots, w_n$ to generate the pairs of a ciphertext c_i and a tag v_i of the ciphertext that hides some private information, where pk is Bob's public key and the $DCSE$ is discussed in later passages. Similar to the $PEKS$ algorithm, the $DCSE$ output is searchable using the trapdoors generated from the data receiver. However, to resist IKGA, $DCSE$ additionally generates a tag for the ciphertext, and only the specified data receiver (*i.e.,* Bob) can extract private information from the tag, which can be linked to a ciphertext. In addition to sending $Enc_{\mathsf{pk}}(\text{file})||c_1||\cdots||c_n$ to the cloud server, Alice publicly sends the tags $(v_1, \ldots, v_n)$ to Bob. Bob first uses his private key to retrieve private information from the tag v_i, and then uses this information to generate a trapdoor t_i for the ciphertext corresponding to the tag. In other words, we can consider a trapdoor as having been *designated* to a ciphertext. Bob then submits (v_i, t_i) to the cloud server for searching. Using the trapdoor, the cloud server returns the matched encrypted files to Bob. Because the cloud server cannot randomly select a keyword to generate a ciphertext matching this trapdoor, the design provides no further method to identify the keyword the data receiver is searching for. Furthermore, unlike PEKS, must test every ciphertext, the cloud server in the proposed scheme can quickly find all the matching ciphertexts by using the tag v_i as the index value.

For DCSE, we define two security models: indistinguishability under chosen-keyword attacks (IND-CKA) and indistinguishability under insider-keyword-guessing attacks (IND-IKGA). In particular, IND-CKA security and IND-IKGA security ensure that no adversary can retrieve any information about the keyword from the ciphertext and the trapdoor, even if that adversary can query a polynomial-time trapdoor oracle. Because the ability of a malicious insider exceeds that of a malicious outsider, we only consider IND-IKGA security.

1.3.2. Generic formulation and its security

We present a generic formulation for DCSE with a pseudorandom generator $F(\cdot)$, an IND-ANON-ID-CPA secure IBE scheme $\mathcal{IBE} = (KeyGen, Extract, Enc, Dec)$, and an IND-CCA2 secure KEM scheme $\mathcal{KEM} = (KeyGen, Encaps, Decaps)$. The high-level idea is as follows: To encrypt a keyword w for an authorized data receiver, the data owner first uses KEM to generate a random key k and its corresponding encapsulation e using the data receiver's public key, that is $(k, e) \leftarrow \mathcal{KEM}.Encaps(\mathsf{pk})$. The data owner then defines the output of pseudorandom generator $f \leftarrow F(k \parallel w)$ as an identity and use it to encrypt another random message r, that is $\mathsf{ct} \leftarrow \mathcal{IBE}.Enc(\mathsf{pk}, f, r)$. Here, we can view the encapsulation e as a tag of the ciphertext ct. Only the authorized data receiver can decaps e using her/his private key to obtain the private information, key k. That data receiver then generates the trapdoor t for the "identity" $f \leftarrow F(k \parallel w)$. In this way, the trapdoor actually links to a tag related to a ciphertext. Therefore, a malicious insider cannot randomly generate ciphertext to test the trapdoor.

Additionally, we provide rigorous proofs to demonstrate that this generic construction satisfies the criteria of IND-CKA and IND-IKGA security. Our main idea for proving security requirements is a sequence of games, which slightly modify our origin protocol so that the challenge message contains no information of the keyword in the final game. Consequently, the strategy by which an adversary wins the games can only be guessed at, that is, the adversary cannot gain any advantage by attacking our construction.

1.3.3. Lattice-based instantiation

We provide an instantiation utilizing two efficient and secure lattice-based constructions: the NTRU-based IBE [13] and the NTRU-based KEM [14]. The security of these constructions is based on the ring-LWE and NTRU assumptions that in turn makes our instantiation quantum-resistant. We also experimentally evaluated the performance of the instantiation on a modern laptop. Each encryption, trapdoor, test algorithm only required approximately 1, 0.3, 0.01 (ms), respectively. In comparison with other state-of-the-art schemes, our scheme is not only more efficient and practical, it also provides more robust security.

1.4. Paper organization

The remainder of this paper is organized as follows. In Section 2, we describe the related works of PEKS. In Section 3, we introduce some notations and preliminaries. In Section 4, we introduce three cryptographic building blocks: pseudorandom generator, KEM scheme, and IBE scheme. In Section 5, we introduce a new notion, "designated-ciphertext searchable encryption," and define its system model and security requirements. In Section 6, we formalize DCSE from the IBE scheme and KEM scheme, and provide its security proofs. In Section 7, an efficient lattice-based DCSE is proposed. Finally, in Section 8, we provide the conclusion.

2. Related works

Byun et al. [3] first reported IND-IKGA and indicated that Boneh et al.'s work [2] could not withstand such an attack. The schemes that are resist to IKGA can be separated into the following three categories.

2.1. Dual-server PEKS

Some studies first utilized another server to perform the test algorithm, that is, decentralizing the power of the insider so that when the two insiders do not collude, the schemes can fend off IKGA. Chen et al. [15–17] therefore proposed "dual server" public key searchable encryption, which can withstand IKGA if the two servers do not collude with each other. However, Tso et al. [18] recently showed that the security models of Chen et al.'s work [15–17] are lack of soundness and strength. As a result, they further give a generic construction of dual-server PEKS that can achieve stronger and sounder security. Additionally, Mao et al. [9] followed the idea of the designated tester to introduce the first lattice-based searchable encryption that protected against IKGA.

2.2. Public-key authenticated encryption with keyword search

By adding a test server or designating an additional server, the malicious insider cannot obtain the private information (*i.e.,* trapdoor) required for the test, so IKGA can be effectively avoided. However, adding another server may increase other communication overhead in an actual environment. Moreover, scholars do not presently know how to ensure that a designated server is trustworthy and will not collude with malicious insiders. Consequently, some studies have begun to investigate how to authenticate ciphertext let trapdoors only be valid for authenticated ciphertext.

Fang et al. [19,20] proposed the first public key searchable encryption using a one-time signature and proved its security without random oracle. Huang and Li [21] introduced a new notion called "public-key authenticated encryption with keyword search (PAEKS)," to resist IKGA. In their schemes, the data sender not only encrypts the keyword, but also authenticates it, and the trapdoor generated by the data receiver is only valid for the ciphertext authenticated by the data sender. Therefore, the malicious server cannot adaptively generate ciphertext to perform IKGA. However, Noroozi and Eslami [22] showed that Huang and Li's scheme even insecure to outsider keyword guessing attacks; thus, they provide an improvement without adding the cost complexity.

Based on the concept of authenticating the ciphertext, Zhang et al. [10] proposed a forward secure lattice-based keyword search to protect against IKGA. However, Liu et al. [23] recently demonstrated the security model in the work does not capture the IKGA, and thus it is insecure. Additionally, Pakniat et al. [24] proposed the first certificateless PAEKS scheme. Moreover, Qin et al. [25] and Li et al. [26] further consider the leakage of the information about the data receiver's query pattern. In other words, they ensure that only server can execute the test algorithm to avoid that an adversary can decide whether two ciphertexts share some identical keywords or not.

2.3. Witness-based searchable encryption

Ma et al. [27] introduced cryptographic primitive called "witness-based searchable encryption" in which the trapdoor is valid only when the ciphertext have a witness relation to the trapdoor. Chen et al. [28] further gave an improvement to reduce the complexity of the size of the trapdoor.

3. Preliminary

3.1. Notations

For simplicity and readability, we use the following notations throughout the paper. Let λ be the natural security parameter. We use standard notations, O and o, to classify the growth of functions. The notation $\mathbf{negl}(n)$ is denoted as an arbitrary function f is *negligible* in n, where $f(n) = o(n^{-c})$ for every fixed constant c. The notation $\mathbf{poly}(n)$ denotes an arbitrary function $f(n) = O(n^c)$ for some constant c. By $\mathbb{N}$ (resp. $\mathbb{Z}$ and $\mathbb{R}$) we denote the set of positive integers (resp. integers and reals). In addition, for a prime q, $\mathbb{Z}_q$ denotes a finite field (or Galois field) with order q. For a power-of-two n, $\mathcal{R} = \mathbb{Z}[x]/(x^n + 1)$ and $\mathcal{R}_q = \mathbb{Z}_q[x]/(x^n + 1)$. The PPT is short for probabilistic polynomial-time. For two string a, b, the concatenation of a and b is denoted as $a \parallel b$. Matrices are denoted by bold capital letters (*e.g.*, $\mathbf{X}$). For a vector x and a matrix $\mathbf{X}$, the Euclidean norm of x and $\mathbf{X}$ is denoted by $\|x\|$ and $\|\mathbf{X}\|$ respectively. For a finite set Q, $a \leftarrow Q$ denotes that a is sampled from Q with uniform distribution. For two vectors a, b, the inner product of a and b is denoted as $\langle a, b \rangle$.

3.2. Lattices

The formalization of our instantiation is based on the NTRU lattices. In this section, we first briefly introduce lattice theory, and then review some lattice hardness assumptions.

A m-dimension lattice Λ is an additive discrete subgroup of $\mathbb{R}^m$. Basically, a lattice is the set of all the integer combinations of some linearly independent vectors, called the basis of the lattice. The formal definition of a lattice is as follows.

Definition 1 (*Lattice*)**.** Let $\mathbf{B} = [b_1|\cdots|b_n] \in \mathbb{R}^{m\times n}$ be an $m \times n$ matrix, where $b_1, \ldots, b_n \in \mathbb{R}^m$ are n linear independent vectors. The m-dimensional lattice Λ generated by $\mathbf{B}$ is the set,

$$\Lambda(\mathbf{B}) = \Lambda(b_1, \ldots, b_n) = \left\{ \sum_{i=1}^{n} b_i a_i | a_i \in \mathbb{Z} \right\}.$$

In addition, we call a lattice full-rank when $n = m$.

3.2.1. Hardness assumptions

Regev [11] introduced a new lattice hardness assumption, called learning with errors (LWE); he demonstrated that several worst-case lattice problems can be reduced to the LWE problem. In addition, he proposed the first public-key cryptosystem based on the hardness of the LWE assumption.

Definition 2 (*LWE Assumption*)**.** Given $n, m \in \mathbb{N}$, q as a prime, a probability distribution χ over $\mathbb{Z}_q$. Suppose there exists an oracle $\mathcal{O}_s^n$ that outputs m samples of the form $(a, \langle a, s\rangle + e)$ where $a \in \mathbb{Z}_q^n$ and $e \in \chi$ are chosen freshly at random for each sample, and $s \leftarrow \mathbb{Z}_q^n$ is the same for every sample. The search-LWE assumption is to find the s. In addition, let $\mathcal{O}_r$ be an oracle that outputs samples $(a, b) \leftarrow (\mathbb{Z}_q^n \times \mathbb{Z}_q^n)$ uniformly at random. The decision-LWE assumption is to guess whether you are interacting with $\mathcal{O}_s^n$ or $\mathcal{O}_r$.

After Regev's seminal work, many LWE-based cryptosystems have been proposed [29–33]. However, these cryptosystems encountered practical problems, because of their overly large key sizes and inefficiency. To solve the issue, in 2009, Lyubashevsky et al. [34] introduced a new algebraic variant of the LWE assumption from [35], and called it ring-LWE. The ring-LWE assumption is the LWE assumption specifically for polynomial rings over finite fields that can also be stated in "search" version and "decision" version that are defined as follows.

Definition 3 (*Ring-LWE Assumption*)**.** Given $n, m \in \mathbb{N}$, let q be a prime, a probability distribution χ over $\mathcal{R}_q$. Suppose there exists an oracle $\mathcal{O}_s$ that outputs m samples of the form $(a, \langle a, s\rangle + e)$ where $a \in \mathcal{R}_q$ and $e \in \chi$ is chosen freshly at random for each sample, and $s \leftarrow \mathcal{R}_q$ is the same for every sample. The search-Ring-LWE assumption is to find the s. In addition, let $\mathcal{O}_r$ be an oracle that outputs samples $(a, b) \leftarrow (\mathcal{R}_q \times \mathcal{R}_q)$ uniformly at random. The decision-Ring-LWE assumption is to guess whether the user is interacting with $\mathcal{O}_s$ or $\mathcal{O}_r$.

Another lattice hardness assumption is the NTRU assumption, defined in [36].

Definition 4 (*NTRU Assumption*)**.** Let χ be a probability distribution over $\mathcal{R}_q$. The NTRU assumption is to distinguish the following two distributions. The first distribution sample is a polynomial $h = g/f$, where $f, g \leftarrow \chi$ and f is invertible, and the second distribution uniformly samples a polynomial h over $\mathcal{R}_q$.

3.3. Public-key encryption with keyword search

In this section we introduce the system model of the PEKS that was proposed by Boneh et al. [2]. A PEKS scheme consists of a set of four-tuple PPT algorithms $\mathcal{PEKS} = (KeyGen, PEKS, Trapdoor, Test)$, described as follows:

- $KeyGen(1^\lambda)$: Taking the security parameter λ as input, this algorithm outputs a master private key msk and a master public key mpk.
- $PEKS(\mathsf{mpk}, w)$: Taking the master public key mpk and a keyword w, this algorithm produces a searchable encryption c of w.
- $Trapdoor(\mathsf{msk}, w)$: Taking the master private key msk and a keyword w, this algorithm generate a trapdoor t.
- $Test(\mathsf{mpk}, c, t)$: Taking the master public key mpk, a searchable encryption $c = PEKS(\mathsf{mpk}, w)$, and a trapdoor $t = Trapdoor(\mathsf{msk}, w')$, this algorithm output 1 if $w = w'$ and 0 otherwise.

Definition 5 (*Correctness of PEKS*)**.** Let λ be a security parameter. We say that a PEKS scheme is correct if:

$$\mathbf{Pr}[Test(\mathsf{mpk}, PEKS(\mathsf{mpk}, w), Trapdoor(\mathsf{msk}, w)) = 1] = 1 - \mathbf{negl}(\lambda),$$

where $(\mathsf{msk}, \mathsf{mpk}) \leftarrow KeyGen(1^\lambda)$.

4. Cryptographic building blocks

In this section, we recall three crucial cryptographic primitives used as building blocks in our generic construction. They are the pseudorandom generator, IBE, and KEM.

4.1. Pseudorandom generator

In our generic construction, we use a pseudorandom generator to generate a "pseudorandom." Informally, we say that a distribution D is pseudorandom if any polynomial-time distinguisher that can distinguish a string $s \leftarrow D$ from a string s chosen randomly and uniformly does not exist. We recall the definition of the pseudorandom generator in [37] Definition 3.15.

Definition 6 (*Pseudorandom Generator*)**.** Let $F : \{0,1\}^n \to \{0,1\}^{n'}$ be a deterministic polynomial-time algorithm, where $n' = \mathbf{poly}(n)$ and $n' > n$. We say that F is a pseudorandom generator if it satisfies the following two conditions:

- Expansion: For every n, it holds that $n' > n$;
- Pseudorandomness: For all PPT distinguishers D,

  $$\mathbf{Adv}_{F,D}^{\mathrm{PSE}}(n) = |\mathbf{Pr}[D(r) = 1] - \mathbf{Pr}[D(F(s)) = 1]| \le \mathbf{negl}(n),$$

  where r is chosen randomly and uniformly from $\{0,1\}^{n'}$, the seed s is chosen randomly and uniformly from $\{0,1\}^n$, and the probabilities depend on the random coins used by D and the choice of r and s.

4.2. Identity-based encryption (IBE)

IBE is an essential primitive of public-key encryption, in which the public key of a user is information that can identify the user (such as e-mail address, name, and social security number). Its concept was first proposed by Shamir [38] as early as 1984. However, the first construction was realized by Cocks [39] based on the quadratic residuosity problem. Later, Boneh and Franklin [40,41] proposed a more practical and secure IBE using the pairing technique.

An IBE scheme is a four-tuple of PPT algorithms $\mathcal{IBE} = (Setup, Extract, Enc, Dec)$, described as follows:

- $Setup(1^\lambda)$: Taking the security parameter λ as input, this algorithm outputs a master private key msk and master public key mpk.
- $Extract(\mathsf{msk}, \mathsf{id})$: Taking the master private key msk and an identity id as input, this algorithm outputs the corresponding private key $\mathsf{sk}_{\mathsf{id}}$ for the identity.
- $Enc(\mathsf{mpk}, \mathsf{id}, m)$: Taking the master public key, mpk, an identity id, and a message m as input, this algorithm outputs a ciphertext ct encrypted by id.
- $Dec(\mathsf{sk}_{\mathsf{id}}, \mathsf{ct})$: Taking a private key $\mathsf{sk}_{\mathsf{id}}$, and a ciphertext ct as input, this algorithm outputs a decrypted message m'.

Definition 7 (*Correctness of IBE*)**.** We say that an IBE scheme, $\mathcal{IBE}$, is correct if

$$\mathbf{Pr}[Dec(\mathsf{sk}_{\mathsf{id}}, Enc(\mathsf{mpk}, \mathsf{id}, m)) = m] = 1 - \mathbf{negl}(\lambda),$$

where $(\mathsf{mpk}, \mathsf{msk}) \leftarrow Setup(1^\lambda)$ and $\mathsf{sk}_{\mathsf{id}} \leftarrow Extract(\mathsf{msk}, \mathsf{id})$.

Moreover, an IBE scheme must satisfy the indistinguishability under any adaptive chosen-identity chosen-plaintext attack (IND-ID-CPA), defined using the following game between a challenger $\mathcal{B}$ and an adversary $\mathcal{A}$.

Game IND-ID-CPA:

- **Setup.** In this stage, $\mathcal{B}$ runs the $Setup(1^\lambda)$ algorithm to generate the master public key mpk and master private key msk. Then, $\mathcal{B}$ keeps msk secret, and sends mpk to $\mathcal{A}$.

- **Phase 1.** $\mathcal{A}$ makes a polynomially bounded number of queries to the $Extract$ oracle on any identity id, and $\mathcal{B}$ returns a private key $\mathsf{sk}_{\mathsf{id}}$ to $\mathcal{A}$.
- **Challenge.** In this stage, $\mathcal{A}$ sends two challenge messages, m_0, m_1, and a challenge identity, id^*, to $\mathcal{B}$, where id^* has never been queried to $Extract$ oracle. After receiving the messages and identity, $\mathcal{B}$ chooses a random bit, $b \leftarrow \{0,1\}$, and generates the challenge ciphertext, $\mathsf{ct}^* \leftarrow Enc(\mathsf{mpk}, \mathsf{id}^*, m_b)$. Finally, $\mathcal{B}$ returns ct^* to $\mathcal{A}$.
- **Phase 2.** $\mathcal{A}$ can continue to ask for the $Extract$ oracle the same as in **Phase 1**. The only restriction is that $\mathcal{A}$ cannot issue an $Extract$ query on the challenge identity id^*.
- **Guess.** $\mathcal{A}$ outputs its guess b'. The adversary is said to win the game if $b' = b$. The advantage of $\mathcal{A}$ is as follows:

$$\mathbf{Adv}^{\text{IND-ID-CPA}}_{\mathcal{IBE},\mathcal{A}}(\lambda) = \left|\mathbf{Pr}[b' = b] - \frac{1}{2}\right|.$$

Definition 8 (*IND-ID-CPA Secure IBE*)**.** We say that an IBE scheme $\mathcal{IBE}$ is IND-ID-CPA secure, if no PPT adversary $\mathcal{A}$ can win the aforementioned game with an advantage exceeding $\mathbf{negl}(\lambda)$.

Moreover, we say that an IBE scheme $\mathcal{IBE}$ is anonymous if it satisfies the following stronger notion of security:

Game IND-ANON-ID-CPA:

- **Setup.** In this stage, $\mathcal{B}$ runs the $Setup(1^\lambda)$ algorithm to generate the master public key mpk and master private key msk. Then $\mathcal{B}$ keeps msk secret, and sends mpk to $\mathcal{A}$.
- **Phase 1.** $\mathcal{A}$ makes a polynomially bounded number of queries to the $Extract$ oracle on any identity id, and $\mathcal{B}$ returns a private key $\mathsf{sk}_{\mathsf{id}}$ to $\mathcal{A}$.
- **Challenge.** In this stage, $\mathcal{A}$ sends a challenge message m, and two challenge identities $\mathsf{id}_0, \mathsf{id}_1$ to $\mathcal{B}$, where id_0 and id_1 have never been queried to $Extract$ oracle. After receiving the messages and identities, $\mathcal{B}$ chooses a random bit, $b \leftarrow \{0,1\}$, and generates the challenge ciphertext, $\mathsf{ct}^* \leftarrow Enc(\mathsf{mpk}, \mathsf{id}_b, m)$. Finally, $\mathcal{B}$ returns ct^* to $\mathcal{A}$.
- **Phase 2.** $\mathcal{A}$ can continue to ask for the $Extract$ oracle, the same as in **Phase 1**. The only restriction is that $\mathcal{A}$ cannot issue an $Extract$ query on the challenge identities id_0 and id_1.
- **Guess.** $\mathcal{A}$ outputs its guess b'.

We say that the adversary wins the game, if $b' = b$. The advantage of $\mathcal{A}$ is defined as follows:

$$\mathbf{Adv}^{\text{IND-ANON-ID-CPA}}_{\mathcal{IBE},\mathcal{A}}(\lambda) = \left|\mathbf{Pr}[b' = b] - \frac{1}{2}\right|.$$

Definition 9 (*IND-ANON-ID-CPA Secure IBE*)**.** We say that an IBE scheme $\mathcal{IBE}$ is IND-ANON-ID-CPA secure if no PPT adversary $\mathcal{A}$ can win the aforementioned game with an advantage exceeding $\mathbf{negl}(\lambda)$.

4.3. Key encapsulation mechanism (KEM)

KEM, first proposed by Cramer and Shoup [42], is a variant of the public-key encryption. Rather than encrypting a message, KEM "encaps" a random value using public key, and outputs an encapsulation. With the corresponding private key, anyone can "decaps" the encapsulation to obtain the same random value.

A KEM scheme is a three-tuple of PPT algorithms, $\mathcal{KEM} = (KeyGen, Encaps, Decaps)$, described as follows.

- $KeyGen(1^\lambda)$: Taking the security parameter λ as input, this algorithm outputs a public key pk and a private key sk.
- $Encaps(\mathsf{pk})$: Taking the public key pk as input, this algorithm outputs a key k and an encapsulation e.
- $Decaps(\mathsf{sk}, e)$: Taking the private key sk and an encapsulation e as input, this algorithm outputs the corresponding key k, or an invalid symbol $\perp$.

Definition 10 (*Correctness of KEM*)**.** We say that a KEM scheme, $\mathcal{KEM}$, is correct, if

$$\mathbf{Pr}[Decaps(\mathsf{sk}, e) = k : (k, e) \leftarrow Encaps(\mathsf{pk})] = 1 - \mathbf{negl}(\lambda),$$

where $(\mathsf{pk}, \mathsf{sk}) \leftarrow KeyGen(1^\lambda)$.

Indistinguishability under the adaptive chosen-ciphertext-attack (IND-CCA2) security of a KEM is defined using the following game between a challenger $\mathcal{B}$ and an adversary $\mathcal{A}$.

Game IND-CCA2:

- **KeyGen.** In this stage, $\mathcal{B}$ runs the $KeyGen(1^\lambda)$ algorithm to generate the public/private key pair $(\mathsf{pk}, \mathsf{sk})$. Then, $\mathcal{B}$ sends pk to $\mathcal{A}$.
- **Phase 1.** $\mathcal{A}$ makes a polynomially bounded number of queries to the $Decaps$ oracle on any encapsulation e; $\mathcal{B}$ returns a key k or invalid symbol $\perp$ to $\mathcal{A}$.
- **Challenge.** In this stage, $\mathcal{B}$ chooses a random bit $b \leftarrow \{0,1\}$. Then, $\mathcal{B}$ generates $(e^*, k_0^*) \leftarrow Encaps(\mathsf{pk})$, and randomly chooses k_1^* from the key space $\mathcal{K}$. Finally, $\mathcal{B}$ returns the challenge ciphertext (e^*, k_b^*) to $\mathcal{A}$.
- **Phase 2.** $\mathcal{A}$ can continue to ask for the $Decaps$ oracle, same as in **Phase 1**. The only restriction is that $\mathcal{A}$ cannot issue a $Decaps$ query on e^*.
- **Guess.** $\mathcal{A}$ outputs its guess b'.

We say that the adversary wins the game, if $b' = b$. The advantage of $\mathcal{A}$ is defined as

$$\mathbf{Adv}^{\text{IND-CCA2}}_{\mathcal{KEM},\mathcal{A}}(\lambda) = \left|\mathbf{Pr}[b' = b] - \frac{1}{2}\right|.$$

Definition 11 (*IND-CCA2 of KEM*)**.** We say that a KEM scheme $\mathcal{KEM}$ is IND-CCA2 secure, if there is no PPT adversary $\mathcal{A}$ that can win the aforementioned game with an advantage exceeding $\mathbf{negl}(\lambda)$.

5. Designated-ciphertext searchable encryption (DCSE)

In this section, we formalize the system model of a DCSE scheme and its security models.

5.1. System model

We extend the system model of Boneh et al.'s work [2]. In DCSE, the trapdoor is linked not only to a keyword, but also to a ciphertext. More specifically, each ciphertext has a corresponding tag that the data receiver uses along with the keyword to generate a trapdoor for the search.

Let λ be a security parameter, $\mathcal{W}$ be a keyword space, $\mathcal{C}$ be a ciphertext space, and $\mathcal{V}$ be a tag space. A DCSE scheme is a four-tuple of PPT algorithms $\mathcal{DCSE} = (KeyGen, DCSE, Trapdoor, Test)$, described as follows.

- $KeyGen(1^\lambda)$: Taking the security parameter λ as input, this algorithm outputs a public key pk and a private key sk.
- $DCSE(\mathsf{pk}, w)$: Taking a public key pk, and a keyword $w \in \mathcal{W}$, this algorithm outputs a searchable ciphertext $c \in \mathcal{C}$ and a tag $v \in \mathcal{V}$ of the ciphertext.
- $Trapdoor(\mathsf{sk}, w', v')$: Taking a private key sk, a keyword $w' \in \mathcal{W}$, and a tag $v' \in \mathcal{V}$ of the ciphertext, this algorithm outputs a trapdoor t.

- $Test(c, t)$: Taking a searchable ciphertext c, and a trapdoor t as input, this algorithm outputs 1, if t and c share the same keyword and the t is actually generated from the tag corresponding to c. Otherwise, it output 0.

Definition 12 (*Correctness of DCSE*)**.** Let λ be a security parameter, $\mathcal{W}$ be a keyword space, $(\mathsf{pk}, \mathsf{sk}) \leftarrow KeyGen(1^\lambda)$, and (c, v) be a pair of a searchable ciphertext and its corresponding tag generated from $DCSE(\mathsf{pk}, w)$, where $w \in \mathcal{W}$. We say that a DCSE scheme is correct if:

$$\mathbf{Pr}[Test(c, Trapdoor(\mathsf{sk}, w, v)) = 1] = 1 - \mathbf{negl}(\lambda).$$

5.2. Security models

We require that the proposed DCSE scheme satisfy the following two security requirements: indistinguishability under chosen-keyword attacks (IND-CKA) and indistinguishability under insider-keyword-guessing attacks (IND-IKGA), which are modeled through the following two games executed by an adversary $\mathcal{A}$ and a challenger $\mathcal{B}$. Note that because the ability of a malicious insider exceeds that of a malicious outsider, we only consider the IND-IKGA here.

5.2.1. Indistinguishability under chosen-keyword attacks

The IND-CKA security ensures that the adversary cannot obtain any information on the keyword from a ciphertext and its corresponding tag.

Game IND-CKA:

- **KeyGen.** In this stage, $\mathcal{B}$ runs the $KeyGen(1^\lambda)$ algorithm to generate the user's public key pk and private key sk. Then, $\mathcal{B}$ sends pk to $\mathcal{A}$.
- **Phase 1.** $\mathcal{A}$ makes a polynomially bounded number of queries to the $Trapdoor$ oracle. When $\mathcal{A}$ issues such a query on (w, v), $\mathcal{B}$ returns a trapdoor t to $\mathcal{A}$ using $Trapdoor$ algorithm with the private key sk.
- **Challenge.** $\mathcal{A}$ sends two challenge keywords $w_0, w_1 \in \mathcal{W}$, where w_0, w_1 have not been queried in **Phase 1**. $\mathcal{B}$ chooses a random bit $b \leftarrow \{0, 1\}$, generates the challenge ciphertext c^* and the corresponding challenge tag v^* from $DCSE(\mathsf{pk}, w_b)$, and returns (c^*, v^*) to $\mathcal{A}$.
- **Phase 2.** $\mathcal{A}$ can continue to ask for the $Trapdoor$ oracle, same as in **Phase 1**. The only restriction is that $\mathcal{A}$ cannot issue a $Trapdoor$ query on w_0 or w_1.
- **Guess.** $\mathcal{A}$ outputs its guess b'.

We say that the adversary wins the game, if $b' = b$. The advantage of $\mathcal{A}$ wins this game is defined as

$$\mathbf{Adv}^{\text{IND-CKA}}_{DCSE,\mathcal{A}}(\lambda) = \left|\mathbf{Pr}[b' = b] - \frac{1}{2}\right|.$$

Definition 13 (*IND-CKA of DCSE*)**.** We say that a DCSE scheme $DCSE$ is IND-CKA secure if there is no PPT adversary $\mathcal{A}$ that can win the aforementioned game with an advantage exceeding $\mathbf{negl}(\lambda)$.

5.2.2. Indistinguishability under insider-keyword-guessing attacks

The IND-IKGA security ensures that the adversary cannot obtain any information about the keyword from a trapdoor.

Game IND-IKGA:

- **KeyGen.** In this stage, $\mathcal{B}$ runs the $KeyGen(1^\lambda)$ algorithm to generate the user's public key pk and private key sk. Then, $\mathcal{B}$ sends pk to $\mathcal{A}$.
- **Phase 1.** $\mathcal{A}$ makes a polynomially bounded number of queries to the $Trapdoor$ oracle. When $\mathcal{A}$ issues such a query on (w, v), $\mathcal{B}$ returns a trapdoor t to $\mathcal{A}$ using $Trapdoor$ algorithm with private key sk.
- **Challenge.** $\mathcal{A}$ sends two challenge keywords $w_0, w_1 \in \mathcal{W}$, where w_0, w_1 have not been queried in **Phase 1**. $\mathcal{B}$ first randomly chooses a tag v^* from $\mathcal{V}$. Then, it chooses a random bit $b \leftarrow \{0, 1\}$ and generates the challenge trapdoor $t^* \leftarrow Trapdoor(\mathsf{sk}, w_b, v^*)$. Finally, $\mathcal{B}$ returns t^* to $\mathcal{A}$.
- **Phase 2.** $\mathcal{A}$ can continue to ask for the $Trapdoor$ oracle, same as in **Phase 1**. The only restriction is that $\mathcal{A}$ cannot issue a $Trapdoor$ query on w_0 or w_1.
- **Guess.** $\mathcal{A}$ outputs its guess b'.

We say that the adversary wins the game, if $b' = b$. The advantage of $\mathcal{A}$ wins this game is defined as

$$\mathbf{Adv}^{\text{IND-IKGA}}_{DCSE,\mathcal{A}}(\lambda) = \left|\mathbf{Pr}[b' = b] - \frac{1}{2}\right|.$$

Definition 14 (*IND-IKGA of DCSE*)**.** We say that a DCSE scheme $DCSE$ is IND-IKGA secure, if there is no PPT adversary $\mathcal{A}$, that can win the aforementioned game with an advantage exceeding $\mathbf{negl}(\lambda)$.

6. Efficient generic construction of DCSE

In this section, we first propose a generic construction of DCSE from an IND-ANON-ID-CPA secure IBE scheme and an IND-CCA2 secure KEM scheme. Then, we present rigorous proofs to demonstrate that this construction satisfies the correctness and security requirements defined in Section 5.

6.1. Generic construction

To construct a DCSE scheme $DCSE$, we first set the following parameters. Let $IBE = (Setup, Extract, Enc, Dec)$ be an IND-ANON-ID-CPA IBE scheme, and $KEM = (KeyGen, Encap, Decaps)$ be an IND-CCA2 secure KEM. Let $\mathcal{W}$ and $\mathcal{C}$ be the keyword space and ciphertext space of $DCSE$, respectively, and let $\mathcal{K}$ be the key space of KEM. Let $F : \mathcal{X} \to \mathcal{Y}$ be a pseudorandom generator with appropriate domain $\mathcal{X}$ and range $\mathcal{Y}$. Here, the domain $\mathcal{X}$ includes the set of any keyword $w \in \mathcal{W}$ concatenating any key $k \in \mathcal{K}$. That is, $\mathcal{X} = \{w \parallel k \mid w \in \mathcal{W} \wedge k \in \mathcal{K}\}$. Furthermore, let the range $\mathcal{Y}$ include an appropriate length of randomness used by the algorithm $IBE.Extract$. In addition, let H be a collision-resistant hash function defined on $\{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$. We then present a generic construction of DCSE from Algorithms 1 to 4.

Algorithm 1 $KeyGen(1^\lambda)$

Input: a security parameter λ
Output: user's key pair $(\mathsf{pk}, \mathsf{sk})$
1: $(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow KEM.KeyGen(1^\lambda)$
2: $(\mathsf{pk}_2, \mathsf{sk}_2) \leftarrow IBE.Setup(1^\lambda)$
3: Set public key $\mathsf{pk} = (\mathsf{pk}_1, \mathsf{pk}_2)$, private key $\mathsf{sk} = (\mathsf{sk}_1, \mathsf{sk}_2)$
4: Output a key pair $(\mathsf{pk}, \mathsf{sk})$

In this construction, the data receiver's public key and private key are generated from the $IBE.KeyGen(1^\lambda)$ and $KEM.KeyGen(1^\lambda)$. To generate a searchable ciphertext c for a keyword w, the data owner first uses KEM to randomly generate a key k and its corresponding encapsulation e using the public key of data receiver, $(e, k) \leftarrow KEM.Encaps(\mathsf{pk})$. He then runs $f \leftarrow F(w \parallel k)$ to obtain a pseudorandom which can be considered as an "identity." Next, he chooses a random value r and encrypts it using identity f, that is $\mathsf{ct} \leftarrow IBE.Enc(\mathsf{pk}, f, r)$, and computes a hash value $h = H(\mathsf{ct}, r)$. Finally, he outputs a searchable ciphertext $c = (\mathsf{ct}, h)$ and a tag $v = e$. Here, we note that if data owner wants to encrypt different keywords for the same data receiver, he can re-use the same key k without re-running the encapsulation algorithm to reduce the computation cost.

Algorithm 2 $DCSE(\mathsf{pk}, w)$

Input: data receiver's public key $\mathsf{pk} = (\mathsf{pk}_1, \mathsf{pk}_2)$ and a keyword $w \in \mathcal{W}$
Output: a ciphertext c and the tag v of the ciphertext

$(e, k) \leftarrow \mathcal{KEM}.Encaps(\mathsf{pk}_1)$
Randomly choose $r \leftarrow \{0,1\}^*$
$f \leftarrow F(w \| k)$
$\mathsf{ct} \leftarrow \mathcal{IBE}.Enc(\mathsf{pk}_2, f, r)$
Compute $h = H(\mathsf{ct}, r)$
Output a ciphertext $c = (\mathsf{ct}, h)$ and tag $v = e$

To generate a trapdoor to search a ciphertext encrypted by a keyword w, the data receiver first obtains the key hidden in the tag, $k \leftarrow \mathcal{KEM}.Decaps(\mathsf{sk}_1, v)$, and computes "identity" $f \leftarrow F(w \parallel k)$. He then generates a trapdoor t for the identity f, $t \leftarrow \mathcal{IBE}.Extract(\mathsf{sk}, f)$, and sends it to the server.

Algorithm 3 $Trapdoor(\mathsf{sk}, w, v)$

Input: user's private key $\mathsf{sk} = (\mathsf{sk}_1, \mathsf{sk}_2)$, a keyword $w \in \mathcal{W}$, and its corresponding tag $v = e$
Output: a trapdoor t for keyword w and tag v

$k \leftarrow \mathcal{KEM}.Decaps(\mathsf{sk}_1, v)$
if $k = \perp$ **then**
  Set trapdoor t to be an invalid symbol $\perp$
else
  $f \leftarrow F(w \| k)$
  Set trapdoor $t \leftarrow \mathcal{IBE}.Extract(\mathsf{sk}_2, f)$
end if
Output a trapdoor t

After receiving the trapdoor, because the ciphertext is actually encrypted by an identity, the server first decrypts the ciphertext to obtain the plaintext $r \leftarrow \mathcal{IBE}.Dec(t, \mathsf{ct})$. The can then check whether $H(\mathsf{ct}, r) = h$. If it matches, output 1. Otherwise, output 0. In a real-world scenario, the data receiver could not only sends the trapdoor but also sends an additional tag that she or he uses. The server can then use the tag as an index to quickly find any ciphertext that might need to be tested.

Algorithm 4 $Test(c, t)$

Input: a ciphertext $c = (\mathsf{ct}, h)$, and a trapdoor t
Output: 1 if t matches c or 0 otherwise

if $t = \perp$ **then**
  Output 0
else
  $r \leftarrow \mathcal{IBE}.Dec(t, \mathsf{ct})$
  Output 1 if $H(\mathsf{ct}, \mathsf{r}) = h$ and 0, otherwise
end if

6.2. Correctness and security proofs

Theorem 1. *The proposed construction is correct, according to Definition 12.*

Proof of Theorem 1. Let $(c = (\mathsf{ct}, h), v) \leftarrow DCSE(\mathsf{pk}, w)$ be a valid ciphertext and its corresponding tag, and let $t \leftarrow Trapdoor(\mathsf{sk}, w, v)$ be a valid trapdoor, where $(\mathsf{pk}, \mathsf{sk}) \leftarrow KeyGen(1^\lambda)$. Because t is actually the private key of identity $F(w \parallel k)$ in the IBE scheme, and ct is a ciphertext that encrypts a random value r using identity $F(w \parallel k)$. With the correctness of the IBE scheme (Definition 7), one can obtain $r \leftarrow \mathcal{IBE}.Dec(t, \mathsf{ct})$ with overwhelming probability. Therefore, $H(\mathsf{ct}, r) = h$; thus, we have $Test(c, t) = 1$. □

In the following, we prove that the proposed generic construction is IND-CKA secure and IND-IKGA secure. At a high level, our strategy is to use a series of games: we gradually modify the structure of the challenge phase so that the challenge does not contain any keywords in the final game. Therefore, the advantages of an attacker for winning the IND-CKA and IND-IKGA games are no higher than mere speculation.

Theorem 2. *The proposed scheme $DCSE$ is IND-CKA secure if the underlying KEM scheme $\mathcal{KEM}$ is IND-CCA2 secure, the IBE scheme $\mathcal{IBE}$ is IND-ANON-ID-CPA secure.*

Proof of Theorem 2. We prove Theorem 2 using a sequence of games, defined as follows.

- **Game**$_0$: This is the original IND-CKA game, as shown in Section 5.2.1.
- **Game**$_1$: We now make a minor change to the aforementioned game. Rather than obtain k from $\mathcal{KEM}.Encaps(\mathsf{pk}_1)$, we choose k' from the range of the output of $\mathcal{KEM}.Encaps(\mathsf{pk}_1)$ randomly.
- **Game**$_2$: We now transform **Game**$_1$ into **Game**$_2$. In this game, let $f = F(w \parallel k')$; we substitute the value $\mathsf{ct} \leftarrow \mathcal{IBE}.Enc(f, r)$ with $\mathsf{ct} \leftarrow \mathcal{IBE}.Enc(f', r)$, where f' is chosen randomly from $\mathcal{Y}$, and $\mathcal{Y}$ is the output range of F.

Let $\mathbf{Adv}_i$ denote the adversary's advantage for winning in **Game**$_i$. We have the following lemmas.

Lemma 1. *For all the PPT algorithms $\mathcal{A}_{01}$, $|\mathbf{Adv}_0 - \mathbf{Adv}_1|$ is negligible, if the underlying KEM scheme $\mathcal{KEM}$ is IND-CCA2 secure.*

Proof of Lemma 1. Suppose that there exists an adversary $\mathcal{A}_{01}$ such that $|\mathbf{Adv}_0 - \mathbf{Adv}_1|$ is non-negligible, then, there exists another challenger $\mathcal{B}_{01}$ that can win the IND-CCA2 game in the underlying KEM scheme $\mathcal{KEM}$ with non-negligible advantage.

- **KeyGen.** $\mathcal{B}_{01}$ first invokes the IND-CCA2 game of $\mathcal{KEM}$ to obtain pk_1. Next, $\mathcal{B}_{01}$ computes $(\mathsf{pk}_2, \mathsf{sk}_2) \leftarrow \mathcal{IBE}.Setup(1^\lambda)$. Finally, $\mathcal{B}_{01}$ sets the public key $\mathsf{pk} = (\mathsf{pk}_1, \mathsf{pk}_2)$, and sends pk to $\mathcal{A}_{01}$.
- **Phase 1.** In this phase, $\mathcal{A}_{01}$ can make polynomially many $Trapdoor$ queries with (pk, w, v), and $\mathcal{B}_{01}$ responds as follows. $\mathcal{B}_{01}$ first invokes $\mathcal{KEM}.Decaps$ oracle on v. The oracle returns an invalid symbol $\perp$ or a valid key k. If the oracle returns $\perp$, $\mathcal{B}_{01}$ also responds with $\perp$ to $\mathcal{A}_{01}$. Otherwise, $\mathcal{B}_{01}$ computes $f \leftarrow F(w \parallel k)$ and $t \leftarrow \mathcal{IBE}.Extract(\mathsf{sk}_2, f)$. Finally, t is returned to $\mathcal{A}_{01}$.
- **Challenge.** $\mathcal{A}_{01}$ sends two challenge keywords $w_0, w_1 \in \mathcal{W}$, where w_0, w_1 have not been queried in **Phase 1**. After receiving these challenge keywords, $\mathcal{B}_{01}$ chooses a random bit $b \leftarrow \{0,1\}$, and runs the following steps:
  - Invoke the Challenge phase of the IND-CCA2 game to obtain the challenge ciphertext (e^*, k^*).
  - Pick $r^* \leftarrow \{0,1\}^*$.
  - Compute $f^* \leftarrow F(w_b \parallel k^*)$.
  - Compute $\mathsf{ct}^* \leftarrow \mathcal{IBE}.Enc(f^*, r^*)$.
  - Compute $h^* = H(\mathsf{ct}^*, r^*)$.
  - Set $v^* = e^*$.

  Then, $\mathcal{B}_{01}$ returns $(c^* = (\mathsf{ct}^*, h^*), v^*)$ to $\mathcal{A}_{01}$.
- **Phase 2.** $\mathcal{A}_{01}$ can continue to make $Trapdoor$ queries, same as in **Phase 1**. The only restriction is that $\mathcal{A}_{01}$ cannot make a $Trapdoor$ query on w_0 or w_1.
- **Guess.** $\mathcal{A}_{01}$ outputs its guess b'. Then $\mathcal{B}_{01}$ outputs b'.

Note that, if k^* is a valid key, $\mathcal{B}_{01}$ gives the view of **Game**$_0$ to $\mathcal{A}_{01}$; if k^* is a random element, then $\mathcal{B}_{01}$ gives the view of **Game**$_1$ to $\mathcal{A}_{01}$. If $|\mathbf{Adv}_0 - \mathbf{Adv}_1|$ is non-negligible, $\mathcal{B}_{01}$ must also have non-negligible advantage against the IND-CCA2 game of the underlying KEM scheme. Therefore,

$$|\mathbf{Adv}_0 - \mathbf{Adv}_1| \leq \mathbf{Adv}^{\text{IND-CCA2}}_{\mathcal{KEM},\mathcal{B}_{01}}(\lambda). \quad \square$$

Lemma 2. *For all the PPT algorithms, $\mathcal{A}_{12}$, $|\mathbf{Adv}_1 - \mathbf{Adv}_2|$ is negligible, if the underlying IBE scheme $\mathcal{IBE}$ is IND-ANON-ID-CPA.*

Proof of Lemma 2. Suppose that there is an adversary $\mathcal{A}_{12}$ such that $|\mathbf{Adv}_1 - \mathbf{Adv}_2|$ is non-negligible, then, there exists another challenger $\mathcal{B}_{12}$ that can win the IND-ANON-ID-CPA game of the underlying IBE scheme $\mathcal{IBE}$ with non-negligible advantage. $\mathcal{B}_{12}$ constructs a hybrid game interacting with an adversary $\mathcal{A}_{12}$ as follows:

- **KeyGen.** $\mathcal{B}_{12}$ first invokes the IND-ANON-ID-CPA game of $\mathcal{IBE}$ to obtain pk_2; then, $\mathcal{B}_{12}$ computes $(\mathsf{pk}_1, \mathsf{sk}_1) \leftarrow \mathcal{KEM}.KeyGen(1^\lambda)$. Finally, $\mathcal{B}_{12}$ sets the public key $\mathsf{pk} = (\mathsf{pk}_1, \mathsf{pk}_2)$, and sends pk to $\mathcal{A}_{12}$.
- **Phase 1.** In this phase, $\mathcal{A}_{12}$ is able to make polynomially many $Trapdoor$ queries with the (pk, w, v), and $\mathcal{B}_{12}$ responds as follows. $\mathcal{B}_{12}$ first obtains $k \leftarrow \mathcal{KEM}.Decaps(v, \mathsf{sk}_1)$. If k is an invalid symbol $\perp$, $\mathcal{B}_{12}$ returns $\perp$ to $\mathcal{A}_{12}$. Otherwise, $\mathcal{B}_{12}$ invokes $\mathcal{IBE}.Extract$ oracle on $F(k \parallel w)$ to obtain a trapdoor t. Finally, $\mathcal{B}_{12}$ sends t to $\mathcal{A}_{12}$.
- **Challenge.** $\mathcal{A}_{12}$ sends two challenge keywords w_0, w_1, where w_0, w_1 have not been queried in **Phase 1**. $\mathcal{B}_{12}$ chooses a random bit $b \leftarrow \{0,1\}$, and performs the following steps:
  - Compute $(e^*, k^*) \leftarrow \mathcal{KEM}.Encaps(\mathsf{pk}_1)$.
  - Randomly choose k'^* from the range of the output of $\mathcal{KEM}.Encaps(\mathsf{pk}_1)$.
  - Randomly choose $f' \leftarrow \mathcal{Y}$.
  - Pick $r^* \leftarrow \{0,1\}^*$.
  - Invoke the Challenge phase of the IND-ANON-ID-CPA game using $F(w_b \parallel k'^*, r^*)$ and (f', r^*) to obtain the challenge ciphertext ct^*.
  - Compute $h^* = H(\mathsf{ct}^*, r^*)$.
  - Set $v^* = e^*$.

  Then, $\mathcal{B}_{12}$ returns $(c^* = (\mathsf{ct}^*, h^*), v^*)$ to $\mathcal{A}_{12}$.
- **Phase 2.** $\mathcal{A}_{12}$ can continue to make $Trapdoor$ queries, similar to **Phase 1**. The only restriction is that $\mathcal{A}_{12}$ cannot make a $Trapdoor$ query on w_0 or w_1.
- **Guess.** $\mathcal{A}_{12}$ outputs its guess b'. Then, $\mathcal{B}_{12}$ outputs b'.

Note that if ct^* is generated from $(F(w_b \parallel k'^*), r^*)$, $\mathcal{B}_{12}$ gives the view of $\mathbf{Game}_1$ to $\mathcal{A}_{12}$; if ct^* is generated from (f', r^*), then $\mathcal{B}_{12}$ gives the view of $\mathbf{Game}_2$ to $\mathcal{A}_{12}$. If $|\mathbf{Adv}_1 - \mathbf{Adv}_2|$ is non-negligible, $\mathcal{B}_{12}$ must also have non-negligible advantage in the IND-ANON-ID-CPA game of the underlying IBE scheme. Therefore,

$$|\mathbf{Adv}_1 - \mathbf{Adv}_2| \leq \mathbf{Adv}^{\text{IND-ANON-ID-CPA}}_{\mathcal{IBE},\mathcal{B}_{12}}(\lambda). \quad \square$$

Lemma 3. $\mathbf{Adv}_2 = 0$.

Proof of Lemma 3. The proof of Lemma 3 is intuitive. Because the ciphertext c^* is irrelevant to the keywords w_0, w_1, the ciphertext reveals nothing about the information of the keywords. The adversary $\mathcal{A}_2$ can only return b' by guessing. Therefore,

$\mathbf{Adv}_2 = 0$. $\square$

By the Definition of Game IND-CKA, $\mathbf{Adv}_0 = \mathbf{Adv}^{\text{IND-CKA}}_{DCSE,\mathcal{B}}(\lambda)$. By combining Lemmas 1, 2, and 3, we have

$$\mathbf{Adv}^{\text{IND-CKA}}_{DCSE,\mathcal{B}}(\lambda) \leq \mathbf{Adv}^{\text{IND-CCA2}}_{\mathcal{KEM},\mathcal{B}_{01}}(\lambda) + \mathbf{Adv}^{\text{IND-ANON-ID-CPA}}_{\mathcal{IBE},\mathcal{B}_{12}}(\lambda).$$

Therefore, if the underlying KEM and IBE scheme are secure, the adversary's advantage is negligibly close to 0. This completes the proof of Theorem 2. $\square$

Theorem 3. *The proposed scheme is IND-IKGA secure, if the underlying KEM scheme $\mathcal{KEM}$ is IND-CCA2 secure, and pseudorandom generator F satisfies pseudorandomness.*

Proof of Theorem 3. We prove Theorem 3 through a sequence of games, defined as follows.

- $\mathbf{Game}_0$: This is the original IND-IKGA game, as shown in Section 5.2.2.
- $\mathbf{Game}_1$: This game is identical to $\mathbf{Game}_0$, except that k is randomly chosen from the output range of $\mathcal{KEM}.Encaps(\mathsf{pk}_1)$, rather than being computed from $\mathcal{KEM}.Encaps(\mathsf{pk}_1)$.
- $\mathbf{Game}_2$ This game is the same as $\mathbf{Game}_1$, except that f is chosen randomly from $\mathcal{Y}$, instead of being computed from $F(w_b \parallel k)$.

Let $\mathbf{Adv}_i$ denote the adversary's advantage in $\mathbf{Game}_i$. We have the following lemmas.

Lemma 4. *For all the PPT algorithms, $\mathcal{A}_{01}$, $|\mathbf{Adv}_0 - \mathbf{Adv}_1|$ is negligible, if the underlying KEM scheme $\mathcal{KEM}$ is IND-CCA2 secure.*

Proof of Lemma 4. Suppose that there exists an adversary $\mathcal{A}_{01}$ such that $|\mathbf{Adv}_0 - \mathbf{Adv}_1|$ is non-negligible, then, there exists another challenger $\mathcal{B}_{01}$ that can win the IND-CCA2 game of the underlying KEM scheme $\mathcal{KEM}$ with non-negligible advantage.

- **KeyGen.** $\mathcal{B}_{01}$ first invokes the IND-CCA2 game of $\mathcal{KEM}$ to obtain pk_1. Next, $\mathcal{B}_{01}$ computes $(\mathsf{pk}_2, \mathsf{sk}_2) \leftarrow \mathcal{IBE}.Setup(1^\lambda)$. Finally, $\mathcal{B}_{01}$ sets the public key $\mathsf{pk} = (\mathsf{pk}_1, \mathsf{pk}_2)$, and sends pk to $\mathcal{A}_{01}$.
- **Phase 1.** In this phase, $\mathcal{A}_{01}$ can make polynomially many $Trapdoor$ queries with (pk, w, v), and $\mathcal{B}_{01}$ responses as follows. $\mathcal{B}_{01}$ first invokes $\mathcal{KEM}.Decaps$ oracle on v. The oracle returns an invalid symbol $\perp$ or a valid key k. If the oracle returns $\perp$, $\mathcal{B}_{01}$ also responses $\perp$ to $\mathcal{A}_{01}$. Otherwise, $\mathcal{B}_{01}$ computes $f = F(w \parallel k)$ and $t \leftarrow \mathcal{IBE}.Extract(\mathsf{sk}_2, f)$. Finally, t is returned to $\mathcal{A}_{01}$.
- **Challenge.** $\mathcal{A}_{01}$ sends two challenge keywords $w_0, w_1 \in \mathcal{W}$, where w_0, w_1 have not been queried in **Phase 1**. $\mathcal{B}_{01}$ chooses a random bit $b \leftarrow \{0,1\}$, and runs the following steps:
  - Invoke the Challenge phase of the IND-CCA2 game to obtain the challenge (e^*, k^*).
  - Compute $f^* \leftarrow F(w_b \parallel k^*)$.
  - Compute $t^* \leftarrow \mathcal{IBE}.Extract(\mathsf{sk}_2, f)$.

  Then, $\mathcal{B}_{01}$ returns t^* to $\mathcal{A}_{01}$.
- **Phase 2.** $\mathcal{A}_{01}$ can continue to make $Trapdoor$ queries, same as in **Phase 1**. The only restriction is that $\mathcal{A}_{01}$ cannot make a $Trapdoor$ query on w_0 or w_1.
- **Guess.** $\mathcal{A}_{01}$ outputs its guess b'. Then, $\mathcal{B}_{01}$ outputs b'.

Note that if k^* is a valid key, $\mathcal{B}_{01}$ gives the view of $\mathbf{Game}_0$ to $\mathcal{A}_{01}$; if k^* is a random element, then, $\mathcal{B}_{01}$ gives the view of $\mathbf{Game}_1$ to $\mathcal{A}_{01}$. If $|\mathbf{Adv}_0 - \mathbf{Adv}_1|$ is non-negligible, $\mathcal{B}_{01}$ must also have non-negligible advantage in the IND-CCA2 game. Therefore,

$$|\mathbf{Adv}_0 - \mathbf{Adv}_1| \leq \mathbf{Adv}^{\text{IND-CCA2}}_{\mathcal{KEM},\mathcal{B}_{01}}(\lambda). \quad \square$$

Lemma 5. *For all the PPT algorithms, $\mathcal{A}_{12}$, $|\mathbf{Adv}_1 - \mathbf{Adv}_2|$ is negligible, if F is a secure pseudorandom generator.*

Proof of Lemma 5. We prove the lemma by describing a PPT reduction algorithm $\mathcal{B}_{12}$ that plays a pseudorandom generator security game. Given a challenge string $T \in \mathcal{Y}$ and the description of a pseudorandom generator F, $\mathcal{B}_{12}$ constructs a hybrid game, interacting with an adversary $\mathcal{A}_{12}$ as follows.

- **KeyGen.** $\mathcal{B}_{12}$ chooses the public parameters, as described in Section 6.1, except that, instead of choosing a proper pseudorandom generator from the pseudorandom generator family, $\mathcal{B}_{12}$ sets F as the public parameter. Then, $\mathcal{B}_{12}$ generates the key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow KeyGen(1^\lambda)$, and sends pk to $\mathcal{A}_{12}$. Note that $\mathcal{B}_{12}$ has full control of the private key sk.

Table 1
Comparison with related schemes on the basis of security properties.

Schemes	Quantum-resistance	IKGA security
[2]	✗	✗
[7]	✓	✗
[8]	✓	✗
[10]	✓	✗
[9]	✓	✓
Ours	✓	✓

Table 2
Comparison with related schemes on the basis of Key size, Trapdoor size, and Ciphertext size (in bytes). Note that $|ID|$ refers to the length of user identity.

Schemes	PK	SK	Trapdoor	Ciphertext
[2]	0.38	0.19	0.38	0.57
[7]	27.2	35	27	52
[8]	$\|ID\|$	560 128	113	113
[10]	3657.05	139 325.1	142.86	14.28
[9]	3657.42	139 325.1	71.42	57.14
Ours	31.88	59.98	38.98	23

Table 3
Time taken (operations per second) by different operations of KeyGen (key generation), Encryption (PEKS in [2,7] and DCSE in our scheme), Extract, and Test.

Scheme	KeyGen	Encryption	Extract	Test
[2]	84.88	186.48	17.41	100 908.17
[7]	0.10	349.28	67.42	174.64
Ours	26.56	3224.35	739.06	63 451.77

- **Phase 1.** In this phase, $\mathcal{A}_{12}$ can make polynomially many $Trapdoor$ queries using (pk, w, v). Due to the knowledge of sk, $\mathcal{B}_{12}$ answers the queries by simply running the $Trapdoor$ algorithm.
- **Challenge.** $\mathcal{A}_{12}$ sends two challenge keywords $w_0, w_1 \in \mathcal{W}$, where w_0, w_1 have not been queried in **Phase 1**. $\mathcal{B}_{12}$ chooses a random bit $b \leftarrow \{0,1\}$, and runs the following steps:
  - Set $f^* = T$.
  - Compute $t^* \leftarrow \mathcal{IBE}.Extract(\mathsf{sk}_2, f^*)$.

  Then, $\mathcal{B}_{12}$ returns t^* to $\mathcal{A}_{12}$.

Note that, if T is generated from F, $\mathcal{B}_{12}$ provides the view of $\mathbf{Game}_1$ to $\mathcal{A}_{12}$; if T is a random string sampled from $\mathcal{Y}$, then $\mathcal{B}_{12}$ provides the view of $\mathbf{Game}_2$ to $\mathcal{A}_{12}$. If $|\mathbf{Adv}_1 - \mathbf{Adv}_2|$ is non-negligible, $\mathcal{B}_{12}$ must also have non-negligible advantage against the pseudorandom generator security game. Therefore,

$$|\mathbf{Adv}_1 - \mathbf{Adv}_2| \leq \mathbf{Adv}^{\mathrm{PSE}}_{F,\mathcal{B}_{12}}(\lambda). \quad \square$$

Lemma 6. $\mathbf{Adv}_2 = 0$.

Proof of Lemma 6. The proof of Lemma 6 is intuitive. Because the trapdoor t^* is irrelevant to the keywords, w_0 and w_1, the trapdoor reveals nothing about the information of the keywords. The adversary $\mathcal{A}_2$ can only return b' by guessing. Therefore,

$$\mathbf{Adv}_2 = 0. \quad \square$$

By the Definition of Game IND-IKGA, $\mathbf{Adv}_0 = \mathbf{Adv}^{\text{IND-IKGA}}_{DCSE,\mathcal{B}}(\lambda)$. By combining Lemmas 4, 5, and 6, we have

$$\mathbf{Adv}^{\text{IND-IKGA}}_{DCSE,\mathcal{B}}(\lambda) \leq \mathbf{Adv}^{\text{IND-CCA2}}_{\mathcal{KEM},\mathcal{B}_{01}}(\lambda) + \mathbf{Adv}^{\mathrm{PSE}}_{F,\mathcal{B}_{12}}(\lambda).$$

Therefore, if the underlying KEM and pseudorandom generator are secure, the adversary's advantage is negligibly close to 0. This completes the proof of Theorem 3. $\square$

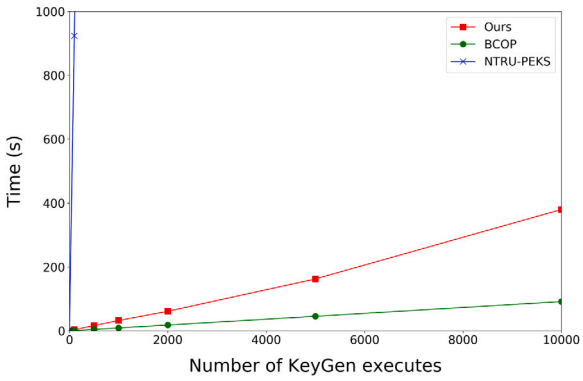

Fig. 1. Time taken by the key generation algorithm.

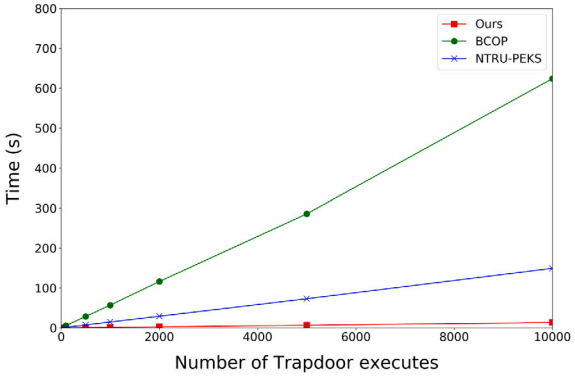

Fig. 2. Time taken by the extract algorithm.

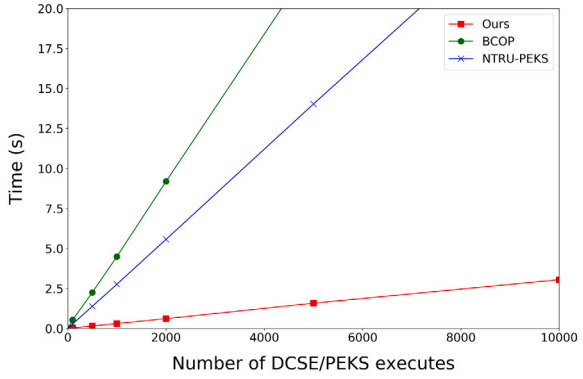

Fig. 3. Time taken by DCSE/PEKS algorithm.

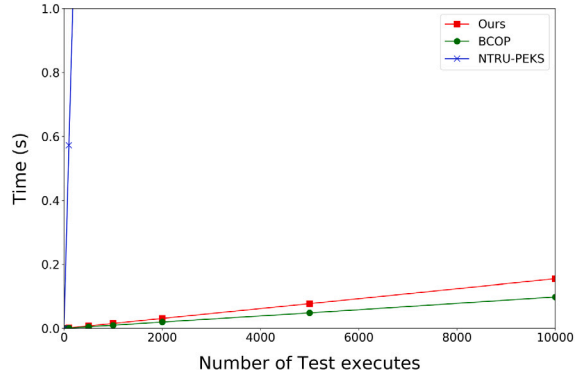

Fig. 4. Time taken by the test algorithm.

7. Efficient instantiation and comparison

In this section, we first propose an DCSE instantiation based on NTRU lattices. Then, we compare different aspects in our instantiation with other state-of-the-art schemes.

7.1. Efficient instantiation

Our instantiation utilizes the IBE of Ducas et al. [13] and KEM of Hülsing et al. [14] (hereafter, referred to as DLP-IBE and HRSS-KEM, respectively).

The DLP-IBE is the first lattice-based IBE scheme with practical parameters. Its security is based on the NTRU and Ring-LWE assumptions. In addition, Behnia et al. [7] have also proven that the DLP-IBE is IND-ANON-ID-CPA secure. The first implementation of the DLP-IBE was provided by Ducas,[1] written in C++ based on the NTL library [43]. Although this implementation is very efficient, it is merely a proof of concept without any optimization. To improve efficiency, McCarthy et al. [44] propose a practical implementation of the DLP-IBE, written in ANSI C, using the number theoretic transform (NTT) optimizations.

The HRSS-KEM is a candidate cryptographic KEM in the Round 2 of the National Institute for Standards and Technology's Post-Quantum Project.[2] In the work, Hülsing et al. first provide a OW-CPA secure NTRU-based encryption scheme with optimized parameters; then, they transform the scheme into a IND-CCA2 secure NTRU-based KEM under quantum-accessible random oracle model.

For concrete instantiation, we use SHA256 as a secure hash function, and symmetric encryption AES-256 as a pseudorandom generator. We used open source project software for DLP-IBE[3] and HRSS-KEM[4] to test the feasibility of our DCSE scheme on an Intel Core i7-8700 3.2-GHz CPU with 10G of RAM. For the DLP-IBE, we selected parameters $n = 1024, q \approx 2^{27}$ for 192-bit security level, and for HRSS-KEM, we selected parameters $n = 701, p = 3, q = 8192$ for 128-bit security level.

7.2. Comparison

To compare the proposed scheme with other state-of-the-art schemes, we set the parameters as follows. For the pairing-based PEKS scheme in [2], we chose the 160-bit group order and 2048-bit group elements $\mathcal{G}, \mathcal{G}_T$. For the NTRU-based PEKS scheme in [7], we chose $n = 1024, q = 2^{27}$ for 192-bit security level. For the LWE-based PEKS schemes in [8–10], we adopted the same secure parameter as in [8], that is $n = 256$, dimension $m = 9753$, and prime $q = 4093$. In addition, we set the number of distinct keywords $k = 1$ and unusual keywords $k' = 1$ for [9], and the security level $l = 10$ for [10].

Table 1 compares of our scheme with other schemes on the basis of its security properties. Only Mao et al.'s work [9] is quantum-resistant and IKGA secure. However, as illustrated in Table 2, the scheme's overly large key sizes make it impractical. We also note that Mao et al.'s scheme require another server to execute test algorithm; thus, the computation overhead is increased. Furthermore, compared with Mao's schemes [9], our public and private key sizes is 1/115 times smaller.

In Table 3, we further compare our instantiation with other two practical PEKS schemes [2,7] on the basis of efficiency. In particular, we adpot PBC library[5] and the open soucre[6] proposed by the authors of [7] to implement [2] and [7], respectively. Compared with [2], although our instantiation is 0.31x and 0.62x slower than that of the KeyGen and Test algorithms, respectively, our instantiation is 17x and 42x faster than those of the Encrypt and Extract algorithms, respectively. As for [7], our instantiation is 245x, 9x, 11x, and 363x faster than those of the KeyGen, Encrypt, Extract, and Test algorithms, respectively. Additionally, we carefully experimented with the time required for the algorithms under different execution times (100, 500, 1000, 2000, 5000, 10 000), the results are presented in Figs. 1 to 4.

[1] https://github.com/tprest/Lattice-IBE.

[2] https://csrc.nist.gov/projects/post-quantum-cryptography/round-2-submissions.

[3] https://github.com/safecrypto/libsafecrypto.

[4] https://github.com/ntru-hrss/ntru-hrss.

[5] https://crypto.stanford.edu/pbc/.

[6] https://github.com/Rbehnia/Full_PEKS.

8. Conclusions

This paper proposed a new cryptographic primitive, DCSE, to counter IKGA in public key searchable encryption. We first provided a generic formulation of DCSE using an IND-ANON-ID-CPA secure IBE and an IND-CCA2 secure KEM, and then proved its security in the standard model. Furthermore, we provided a quantum-resistant instantiation from NTRU lattices utilizing the DLP-IBE and HRSS-KEM. In conclusion, this paper provides a novel solution to IKGA in a searchable encryption. In addition to yielding interesting theoretical results, the proposed scheme is notably more efficient and safe compared with other state-of-the-art schemes.

CRediT authorship contribution statement

Zi-Yuan Liu: Methodology, Data curation, Software, Writing - original draft, Writing - review & editing. **Yi-Fan Tseng:** Methodology, Writing - original draft, Writing - review & editing. **Raylin Tso:** Conceptualization, Validation, Supervision, Funding acquisition. **Masahiro Mambo:** Conceptualization, Writing - review & editing, Validation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported by the Ministry of Science and Technology, Taiwan (ROC), under Project Numbers MOST 108-2218-E-004-001-, 108-2218-E-004-002-MY2, 109-2221-E-004-011-MY3, 109-2218-E-011-007-, and by Taiwan Information Security Center at National Sun Yat-sen University (TWISC@NSYSU).

References

[1] Song DX, Wagner D, Perrig A. Practical techniques for searches on encrypted data. In: Proceedings of the 2000 IEEE symposium on security and privacy - S&P'00. USA: IEEE; 2000, p. 44. http://dx.doi.org/10.1109/SECPRI.2000.848445.

[2] Boneh D, Di Crescenzo G, Ostrovsky R, Persiano G. Public key encryption with keyword search. In: Cachin C, Camenisch JL, editors. International conference on the theory and applications of cryptographic techniques - EUROCRYPT'04. Berlin, Heidelberg: Springer; 2004, p. 506–22. http://dx.doi.org/10.1007/978-3-540-24676-3_30.

[3] Byun JW, Rhee HS, Park H-A, Lee DH. Off-line keyword guessing attacks on recent keyword search schemes over encrypted data. In: Workshop on secure data management - SDM'06. Berlin, Heidelberg: Springer; 2006, p. 75–83. http://dx.doi.org/10.1007/11844662_6.

[4] Shor PW. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Rev 1999;41(2):303–32. http://dx.doi.org/10.1137/S0097539795293172.

[5] Shor PW. Algorithms for quantum computation: Discrete logarithms and factoring. In: Proceedings 35th annual symposium on foundations of computer science - FOCS'94. IEEE; 1994, p. 124–34. http://dx.doi.org/10.1109/SFCS.1994.365700.

[6] Arute F, Arya K, Babbush R, Bacon D, Bardin JC, Barends R, et al. Quantum supremacy using a programmable superconducting processor. Nature 2019;574(7779):505–10. http://dx.doi.org/10.1038/s41586-019-1666-5.

[7] Behnia R, Ozmen MO, Yavuz AA. Lattice-based public key searchable encryption from experimental perspectives. IEEE Trans Dependable Secure Comput 2018. http://dx.doi.org/10.1109/TDSC.2018.2867462, (early access).

[8] Xu L, Yuan X, Steinfeld R, Wang C, Xu C. Multi-writer searchable encryption: An LWE-based realization and implementation. In: Proceedings of the 2019 ACM Asia conference on computer and communications security - AsiaCCS'19. ACM; 2019, p. 122–33. http://dx.doi.org/10.1145/3321705.3329814.
[9] Mao Y, Fu X, Guo C, Wu G. Public key encryption with conjunctive keyword search secure against keyword guessing attack from lattices. Trans Emerg Telecommun Technol 2019;30(11):e3531. http://dx.doi.org/10.1002/ett.3531, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/ett.3531.
[10] Zhang X, Xu C, Wang H, Zhang Y, Wang S. FS-PEKS: Lattice-based forward secure public-key encryption with keyword search for cloud-assisted industrial Internet of things. IEEE Trans Dependable Secure Comput 2019. http://dx.doi.org/10.1109/TDSC.2019.2914117, (early access).
[11] Regev O. On lattices, learning with errors, random linear codes, and cryptography. In: Proceedings of the thirty-seventh annual ACM symposium on theory of computing - STOC'05. New York, NY, USA: ACM; 2005, p. 84–93. http://dx.doi.org/10.1145/1060590.1060603.
[12] Abdalla M, Bellare M, Catalano D, Kiltz E, Kohno T, Lange T, et al. Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions. In: Shoup V, editor. Annual international cryptology conference - CRYPTO'05. Berlin, Heidelberg: Springer; 2005, p. 205–22. http://dx.doi.org/10.1007/11535218_13.
[13] Ducas L, Lyubashevsky V, Prest T. Efficient identity-based encryption over NTRU lattices. In: Sarkar P, Iwata T, editors. International conference on the theory and application of cryptology and information security - ASIACRYPT'14. Berlin, Heidelberg: Springer; 2014, p. 22–41. http://dx.doi.org/10.1007/978-3-662-45608-8_2.
[14] Hülsing A, Rijneveld J, Schanck J, Schwabe P. High-speed key encapsulation from NTRU. In: Fischer W, Homma N, editors. International conference on cryptographic hardware and embedded systems - CHES'17. pringer, Cham; 2017, p. 232–52. http://dx.doi.org/10.1007/978-3-319-66787-4_12.
[15] Chen R, Mu Y, Yang G, Guo F, Wang X. A new general framework for secure public key encryption with keyword search. In: Foo E, Stebila D, editors. Australasian conference on information security and privacy - ACISP'15. Springer, Cham; 2015, p. 59–76. http://dx.doi.org/10.1007/978-3-319-19962-7_4.
[16] Chen R, Mu Y, Yang G, Guo F, Wang X. Dual-server public-key encryption with keyword search for secure cloud storage. IEEE Trans Inf Forensics Secur 2015;11(4):789–98. http://dx.doi.org/10.1109/TIFS.2015.2510822.
[17] Chen R, Mu Y, Yang G, Guo F, Huang X, Wang X, et al. Server-aided public key encryption with keyword search. IEEE Trans Inf Forensics Secur 2016;11(12):2833–42. http://dx.doi.org/10.1109/TIFS.2016.2599293.
[18] Tso R, Huang K, Chen Y-C, Rahman SMM, Wu T-Y. Generic construction of dual-server public key encryption with keyword search on cloud computing. IEEE Access 2020;8:152551–64. http://dx.doi.org/10.1109/ACCESS.2020.3017745.
[19] Fang L, Susilo W, Ge C, Wang J. A secure channel free public key encryption with keyword search scheme without random oracle. In: Garay J, Miyaji A, Otsuka A, editors. International conference on cryptology and network security - CANS'09. Berlin, Heidelberg: Springer; 2009, p. 248–58. http://dx.doi.org/10.1007/978-3-642-10433-6_16.
[20] Fang L, Susilo W, Ge C, Wang J. Public key encryption with keyword search secure against keyword guessing attacks without random oracle. Inform Sci 2013;238:221–41. http://dx.doi.org/10.1016/j.ins.2013.03.008.
[21] Huang Q, Li H. An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks. Inform Sci 2017;403:1–14. http://dx.doi.org/10.1016/j.ins.2017.03.038.
[22] Noroozi M, Eslami Z. Public key authenticated encryption with keyword search: Revisited. IET Inf Secur 2018;13(4):336–42. http://dx.doi.org/10.1049/iet-ifs.2018.5315.
[23] Liu Z-Y, Tseng Y-F, Tso R. Cryptanalysis of "FS-PEKS: Lattice-based forward secure public-key encryption with keyword search for cloud-assisted industrial Internet of things". Cryptology ePrint Archive, Report 2020/651, 2020, https://eprint.iacr.org/2020/651.
[24] Pakniat N, Shiraly D, Eslami Z. Certificateless authenticated encryption with keyword search: Enhanced security model and a concrete construction for industrial IoT. J Inf Secur Appl 2020;53:102525. http://dx.doi.org/10.1016/j.jisa.2020.102525.
[25] Qin B, Chen Y, Huang Q, Liu X, Zheng D. Public-key authenticated encryption with keyword search revisited: Security model and constructions. Inform Sci 2020;516:515–28. http://dx.doi.org/10.1016/j.ins.2019.12.063.
[26] Li H, Huang Q, Shen J, Yang G, Susilo W. Designated-server identity-based authenticated encryption with keyword search for encrypted emails. Inform Sci 2019;481:330–43. http://dx.doi.org/10.1016/j.ins.2019.01.004.
[27] Ma S, Mu Y, Susilo W, Yang B. Witness-based searchable encryption. Inform Sci 2018;453:364–78. http://dx.doi.org/10.1016/j.ins.2018.04.012.
[28] Chen Y-C, Xie X, Wang PS, Tso R. Witness-based searchable encryption with optimal overhead for cloud-edge computing. Future Gener Comput Syst 2019;100:715–23. http://dx.doi.org/10.1016/j.future.2019.05.038.
[29] Gentry C. A fully homomorphic encryption scheme. (Ph.D. thesis), Stanford University; 2009, crypto.stanford.edu/craig.
[30] Gentry C, Peikert C, Vaikuntanathan V. Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the fortieth annual ACM symposium on theory of computing - STOC'08. Association for Computing Machinery; 2008, p. 197–206. http://dx.doi.org/10.1145/1374376.1374407.
[31] Agrawal S, Boneh D, Boyen X. Efficient lattice HIBE in the standard model. In: Gilbert H, editor. Annual international conference on the theory and applications of cryptographic techniques - EUROCRYPT'10. Berlin, Heidelberg: Springer; 2010, p. 553–72. http://dx.doi.org/10.1007/978-3-642-13190-5_28.
[32] Boyen X. Attribute-based functional encryption on lattices. In: Sahai A, editor. Theory of cryptography conference - TCC'13. Berlin, Heidelberg: Springer; 2013, p. 122–42. http://dx.doi.org/10.1007/978-3-642-36594-2_8.
[33] Brakerski Z, Vaikuntanathan V. Efficient fully homomorphic encryption from (standard) LWE. SIAM J Comput 2014;43(2):831–71. http://dx.doi.org/10.1137/120868669.
[34] Lyubashevsky V, Peikert C, Regev O. On ideal lattices and learning with errors over rings. In: Gilbert H, editor. Annual international conference on the theory and applications of cryptographic techniques - EUROCRYPT'10. Berlin, Heidelberg: Springer; 2010, p. 1–23. http://dx.doi.org/10.1007/978-3-642-13190-5_1.
[35] Stehlé D, Steinfeld R, Tanaka K, Xagawa K. Efficient public key encryption based on ideal lattices. In: Matsui M, editor. International conference on the theory and application of cryptology and information security - ASIACRYPT'09. Springer; 2009, p. 617–35. http://dx.doi.org/10.1007/978-3-642-10366-7_36.
[36] Hoffstein J, Pipher J, Silverman JH. NTRU: A ring-based public key cryptosystem. In: Buhler J, editor. International algorithmic number theory symposium - ANTS'98. Berlin, Heidelberg: Springer; 1998, p. 267–88. http://dx.doi.org/10.1007/BFb0054868.
[37] Katz J, Lindell Y. Introduction to modern cryptography. Chapman and Hall/CRC; 2014.
[38] Shamir A. Identity-based cryptosystems and signature schemes. In: Blakley G, Chaum D, editors. Workshop on the theory and application of cryptographic techniques - CRYPTO'84. Berlin, Heidelberg: Springer; 1984, p. 47–53. http://dx.doi.org/10.1007/3-540-39568-7_5.
[39] Cocks C. An identity based encryption scheme based on quadratic residues. In: Honary B, editor. IMA international conference on cryptography and coding - IMACC'01. Berlin, Heidelberg: Springer; 2001, p. 360–3. http://dx.doi.org/10.1007/3-540-45325-3_32.
[40] Boneh D, Franklin M. Identity-based encryption from the weil pairing. In: Kilian J, editor. Annual international cryptology conference - CRYPTO'01. Berlin, Heidelberg: Springer; 2001, p. 213–29. http://dx.doi.org/10.1007/3-540-44647-8_13.
[41] Boneh D, Franklin M. Identity-based encryption from the weil pairing. SIAM J Comput 2003;32(3):586–615. http://dx.doi.org/10.1137/S0097539701398521.
[42] Cramer R, Shoup V. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM J Comput 2003;33(1):167–226. http://dx.doi.org/10.1137/S0097539702403773.
[43] Shoup V, et al. NTL: A library for doing number theory. 2019, http://www.shoup.net/ntl. Accessed: December, 2019.
[44] McCarthy S, Smyth N, O'Sullivan E. A practical implementation of identity-based encryption over NTRU Lattices. In: O'Neill M, editor. IMA international conference on cryptography and coding - IMACC'17. Springer, Cham: Springer; 2017, p. 227–46. http://dx.doi.org/10.1007/978-3-319-71045-7_12.

Further reading

[1] Rhee HS, Park JH, Susilo W, Lee DH. Trapdoor security in a searchable public-key encryption scheme with a designated tester. J Syst Softw 2010;83(5):763–71. http://dx.doi.org/10.1016/j.jss.2009.11.726.

Zi-Yuan Liu received the B.E. degree in computer science from National Tseng Hua University, Taiwan in 2016 and the M.E. degree in computer science from National Chengchi University, Taiwan in 2018. He is currently pursuing the Ph.D. degree in computer science at National Chengchi University, Taiwan. His current research interests include post-quantum cryptography and blockchain technology.

Yi-Fan Tseng was born in Kaohsiung, Taiwan. He received the Ph.D. degree and MS degree in computer science and engineering from National Sun Yat-sen University, Taiwan, in 2014 and 2018, respectively. From 2018 to 2019, as a postdoctoral researcher, he joined the research group of Taiwan Information Security Center at National Sun Yat-sen University (TWISC@NSYSU). In 2019, he has joined the faculty of the Department of Computer Science, National Chengchi University, Taipei, Taiwan. His research interests include cloud computing and security, network and communication security, information security, cryptographic protocols, and applied cryptography.

Raylin Tso is currently a professor in the Department of Computer Science, National Chengchi University, Taiwan. He obtained his B.Eng. degree from National Tsing Hua University, Taiwan, in 1995. He received his M.Eng. and PhD degrees in Systems and Information Engineering from Tsukuba University, Japan, in 2004 and 2006, respectively. He has authored or coauthored over 60 papers in referred journals and conferences in the area of information security. His research interests are mainly in the areas of cryptography, IoT security, privacy-preserving data analysis, and blockchain technology.

Masahiro Mambo received the B.Eng. degree from Kanazawa University, Japan, in 1988, and the M.S.Eng. and Dr.Eng. degrees in electronic engineering from the Tokyo Institute of Technology, Japan, in 1990 and 1993, respectively. After working at the Japan Advanced Institute of Science and Technology, JAIST, Tohoku University, and the University of Tsukuba, he joined Kanazawa University, in 2011. He is currently a Professor with the Faculty of Electrical, Information and Communication Engineering, Institute of Science and Engineering. His research interests include information security, software protection, and privacy protection. He has served as the Co-Editor-in-Chief of the International Journal of Information Security (Springer), the Steering Committee Chair of the International Conference on Information Security, and the Chair of the Technical Committee on Information Security (ISEC), Engineering Sciences Society (ESS), Institute of Electronics, Information and Communication Engineers (IEICE).