IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 26, 2019, accepted July 5, 2019, date of publication July 12, 2019, date of current version August 21, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2928455

A Keyword-Searchable ABE Scheme From Lattice in Cloud Storage Environment

LIHUA LIU[1,2], SHANGPING WANG[1], BINTAO HE[1,2], AND DUO ZHANG[1]
[1]Shaanxi Key Laboratory for Network Computing and Security Technology, Xi'an University of Technology, Xi'an 710054, China
[2]School of Mathematics and Computer Science, Shaanxi University of Technology, Hanzhong 723001, China

Corresponding author: Shangping Wang (spwang@mail.xaut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61572019 and in part by the Shaanxi Provincial Natural Science Foundation under Grant 2019GY-028.

ABSTRACT Currently, the security situation of data security and user privacy protection is increasingly serious in cloud environment. Ciphertext data storage can prevent the risk of user's privacy disclosure. But how to search keyword on ciphertext data without revealing keyword information becomes a new challenge. Searchable encryption (SE) is put forward for this reason, which can be used to realize ciphertext-search directly. In terms of multi-user data sharing, public-key encryption with keywords search (PEKS) is more widely used than symmetric searchable encryption. PEKS has been widely studied and developed by researchers in recent years. However, the existing PEKS scheme often lacks flexible access policy. Therefore, combining the advantage of policy control on attribute-based encryption (ABE), and as that the bilinear pairing related assumption is fragile in post-quantum era, lattice-based cryptography is considered as one of secure encryption technology against quantum attack. With this background, in this paper, we give a keyword-searchable ABE scheme based on the hardness of lattice problems, our scheme supports flexible attribute control policy by integrating ABE and PEKS, and the security of new scheme is proved under the learning with errors (LWE) assumption. As lattice-based cryptographic technology is currently thought to be resistant to quantum attacks, so the new scheme has stronger security in a quantum era.

INDEX TERMS Lattices, attribute-based encryption (ABE), public-key encryption with keyword search (PEKS), learning with errors (LWE).

I. INTRODUCTION

Now, the flexibility of the cloud platform has led to the cloud security situation extremely complex. Users' privacy protection and data security are more and more widely concerned in cloud storage environment. With the view of privacy protection and data security, the users usually choose to encrypt the data before uploading. How to search for encrypted data on a cloud without divulging the content of the data has always been a requirement for user privacy protection. But the traditional encryption technology cannot adapt to the cloud storage environment. In order to solve this problem, the searchable encryption technology was put forward in 1996. The ciphertext search technology with hidden user access mode was firstly given in literature [1]. In 2000, Song and Wagner [2] gave the practical techniques for searching on encrypted data. Furtherly, in 2004, asymmetric SE scheme was presented by Boneh *et al.* [3], named public-key encryption with keyword search (PEKS), which is an extension and promotion of searchable encryption (SE) in the symmetric environment. SE allows users to delegate search capability to a third party without providing it to decrypt. Its special security advantages include four aspects: provably security, controlled searching, hidden query and query isolation. The searchable encryption technology is used to ensure the privacy information and personal data security of users.

However, PEKS focuses on the search of ciphertext data, which is not well support fine-grained access control of encrypted files. In the general case, the ciphertext which contains a certain keyword can be decrypted only by the user with special privileges. Attribute-based encryption (ABE) [14], [15] is a cryptosystem that may better implemented fine-grained access control in cloud storage environment. In ABE scheme, user identity information is expressed by a set of attributes. The decryption capability is determined by the degree of matching between the access structure and

The associate editor coordinating the review of this manuscript and approving it for publication was Kuo-Hui Yeh.

the property set. It can flexibly describe cryptography system access control policy. As is well-known, ABE technology can better support multi-user mode, where the ciphertext will be selectively decrypted by multiple authorized users. Therefore, ABE technology is obviously suitable to solve the problem of fine-grained access control in PEKS scheme. Now, the combination of ABE and PEKS is a hot topic in the safe search and privacy protection in cloud application. And more research results that appear in the literature [16]–[21] will be given in detail below.

Most of the schemes mentioned above are designed on the bilinear Diffie-Hellman (BDH) intractability assumptions [22]. But with the advent of the post-quantum era, the super parallel computing capability of quantum computer will challenge the security of most existing public key cryptography. In 1994, Shor [23] proposed a polynomial time algorithm for discrete logarithms and factoring on a quantum computer, called Shor algorithm, which is suitable for solving difficult mathematical problems such as large integer factorization and discrete logarithm. It can effectively attack RSA and ECC public key cryptography which are widely used today. Hence, the bilinear pairing operations are vulnerable to quantum attacks. In recent years, researchers engaged in cryptography are looking for more secure algorithms in post-quantum era, post-quantum cryptography (PQC) [24] becomes a research hotspot. In this regard, the National Institute of Standards and Technology (NIST) began to standardize post-quantum cryptography [25] in 2016. Among all anti-quantum cryptography algorithms [26], lattice-based cryptography is considered as a candidate of post-quantum cryptography.

With the above background, we propose the mathematical model of keyword-searchable ABE encryption scheme from lattices in cloud environment. Thus, the new scheme has the property of resisting quantum attack.

A. RELATED WORKS

- PEKS: Searchable encryption (SE) is an encryption theory that enables users to search for keywords in ciphertext. It can make the utmost of the huge computing resources of the cloud server to search ciphertext with keyword. PEKS is an extended SE technique in multi-user search model. In the aspect of multi-user data sharing, the application scene of PEKS are broader than that of symmetric searchable encryption (SSE). At first, in 1996, the concept of SE was proposed by Goldreich and Ostrovsky [1]. In 2000, Song and Wagner [2] designed a symmetrical SE scheme that introduced provable security theory with undistinguished security objectives, and could retrieved ciphertext information disclosure any original files. In 2004, Boneh *et al.* [3] firstly presented public-key encryption with keyword search (PEKS), and constructed a PEKS scheme on bilinear Diffie-Hellman (BDH) assumption. In the same year, Waters [4] presented a new method for constructing searchable encrypted audit logs, which can be combined with any existing logging scheme for tamper-proof purposes. In 2005, Park *et al.* [5] presented the public key encryption scheme with conjunctive field keyword search. It extended PEKS to allow conjunctive field keyword search on a set of public keys. In 2006, Curtmola *et al.* [6] defined SSE in the multi-user modeling, and presented a highly efficient scheme. Subsequently, an extension was given in literature [7]. In 2008, Wang [8] presented a threshold privacy preserving keyword search scheme and proofed security under the difficulty hypothesis of discrete logarithm. In 2012, Nishioka [9] presented a new security definition for non-interactive PEKS, named perfect keyword privacy (PKP). Same year, Siad [10] extended the notion of anonymous IBE, and gave a method for converting the blind anonymous IBE to a threshold PEKS with oblivious keyword search. In 2015, Rhee and Dong [11] proposed PEKS with keyword updatability, where a tagged keyword can be updated upon the receiver's request. In recent years, many literatures discussed the safety of combination of PEK and PEKS, in 2016, Chen *et al.* [12] first formalized the stronger security notion for the scheme of PKE–PEKS, and then gave two simple design of PKE–PEKS schemes. In 2019, Suzuki *et al.* [13] extended the work by compositing secure PEKS and PKE, and gave an actual example of EMRs in cloud storage.
- Integrate ABE and PEKS: Regarding data access control, firstly, in 2005, Sahai and Waters [14] presented an identity encryption scheme based on the biological characteristics of information, called fuzzy identity encryption, which is often seen as a ''prototype'' of identity encryption scheme. In 2006, Goyal *et al.* [15] gave the extension of ABE, which includes two types: ciphertext-policy ABE scheme and key-policy ABE scheme. In ABE scheme, each user owns a set of sub-attributes and a predicate function of a set of attributes. Only if users' attributes satisfy the policy can be allowed to decrypt the ciphertext. ABE is considered as the most promising cryptographic technique to support fine-grained access. Nearly years, the integration of ABE and PEKS is a research hotspot in the secure search and privacy protection in cloud application. In 2013, Boneh *et al.* [16] proposed function-private IBE by providing predicate privacy in SE. In 2014, Khader [17], [18] presented a secure design of an ABSE on bilinear maps, and gave an application example of medical records. The same year, Li *et al.* [19] presented a single-keyword searchable encryption scheme on KP-ABE strategy. Han *et al.* [20] proposed a method to transform KP-ABE to ABEKS, and its security was proved on bilinear Diffie-Hellman (BDH) assumption. Sun *et al.* [21] presented an attribute-based PESK scheme with efficient user revocation that enables extensible fine-grained search authorization. On this basis, more practical applications scheme of the

integration of ABE and PEKS in the cloud storage environment had been presented in the literature [21], [29]–[31], [37]–[39].

- Lattice-based cryptographic scheme: Lattice-based cryptographic theory is recognized as the most powerful competitor of the post-quantum cryptography algorithm standard. In recent years, it has gotten rapid development and many excellent research results have appeared. At first, in 1996, Ajtai [32] gave a random class of lattices whose importance was demonstrated in cryptography. In 1997, Ajtai and Dwork [33] presented a lattice-based public-key cryptosystem firstly. In 1999, Ajtai [34] proposed a method for constructing random lattices and their short lattices. Subsequently, in 2005, Regev [35] proved that LWE was related to lattice difficulty problems (such as gap-SVP), and gave a public key cryptography scheme on LWE assumption. Furtherly, in 2008, Gentry *et al.* [36] designed cryptosystem to construct a variety of ''trapdoor'' on lattice difficulty problems. It was the further practical development of based-lattice cryptography.

Yet, the research on the PEKS scheme from lattice are still rare. In 2016, Kuchta *et al.* [37] constructed a PK-ABE searchable encryption on lattices. In 2018, Yang *et al.* [38] suggested a fuzzy information retrieval scheme support multi-user based on the lattice difficulty assumption. The Same year, Rouzbeh *et al.* [39] proposed a lattice-based PEKS scheme, and the scheme was implemented and deployed on Amazon Web Services. Other relevant research literature is shown in appendix [46]–[48].

B. OUR CONTRIBUTIONS

The integrating ABE and PEKS can support flexible attribute control policy. But the design of existing schemes is mostly based on bilinear pairing and cannot resist quantum attacks. In this paper, we propose a new keyword-searchable attribute-based encryption scheme on hardness of standard lattice problems. Advantages of the scheme are as following:

- Lattice-based cryptographic technology is currently thought to be resistant to quantum attacks. Thus, the new scheme has the property of quantum attack resistance.
- New scheme supports flexible attribute control policy by integrating ABE and PEKS. In the new encryption system, each user has a set of attributes, which associating with the private key of them. Different user has independent private key. It can avoid the risk of leaks that are caused by multiple users share.
- A complete application scene of new scheme is given in cloud storage environment, including encrypted original data by user attribute set, decrypted data by access control policy, searched data by keyword index, and stocked data by index structure on cloud server and so on. It covers the whole process of encryption, decryption, search and storage. So, the scheme can be more easily applied to personal health network, wireless sensor network, vehicle network, etc.

TABLE 1. Symbol description.

Symbol	Symbol Description
Z	the set of the integers
R	the set of real numbers
$\Lambda_q^{\perp} \in Z^m$	an m- dimensional lattice
W	a set of keywords $W = \{kw_1, kw_2, ...\}$
$(\mathbf{M}, \rho)$	an access control policy
$[n]$	the set of positive integers $\{1, 2, \ldots, n\}$
U =	a universal attribute set
S	an authorization attribute set
I_W	an index structure of the set of keywords
T_{kw_j}	a keyword trapdoor of the keyword kw_j
H	a secure hash function $H: \{0,1\}^* \rightarrow Z_q$
l	the number of attributes on a policy $(\mathbf{M}, \rho)$

II. PRELIMINARY

In this section, we give the algorithm definition of KP-ABE scheme and PEKS scheme separately. The corresponding security model is given as follows.

A. NOTATION

By convention, a vector be denoted as bold lower-case letter, e.g. $\mathbf{x}$, and i-th element of vector $\mathbf{x}$ be denoted as x_i. A matrix be denoted as bold capital letter, e.g. $\mathbf{X}$, and i-th vector of matrix $\mathbf{X}$ be denoted as $\mathbf{x}_i$. The norm of matrix $\mathbf{X}$ be defined as $||\mathbf{X}||_2$, $\tilde{\mathbf{X}}$ be denoted as the Gram-Schmidt orthogonal basis of $\mathbf{X}$. In this paper, other symbol definitions are shown in the table 1.

B. LATTICE_BASED THEORY

1) CONCEPT LATTICE

Lattice is the integer linear combinations of some linearly independent basis vectors $\mathbf{B} = \{b_1, b_2, \ldots, b_k\} : \mathcal{L} = \mathcal{L}(\mathbf{B}) := \mathbf{B} \cdot \mathbb{Z}^k = \{\sum z_i \mathbf{b}_i : z_i \in \mathbb{Z}\}$, where k is called the rank of the basis, and is an invariant of the lattice.

In order to reduce duplication, more conclusions of lattice are given in Appendix.

2) LEARNING WITH ERRORS (LWE)

The average-case learning with errors (LWE) problem was introduced by Regev [35] in 2005, which is the ''encryption-enabling'' analogue of the SIS problem.

a: LWE DISTRIBUTION

For a vector $\mathbf{s} \in \mathbb{Z}_q^n$ called the secret, the LWE distribution $A_{\mathbf{s},\chi}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ is sampled by choosing $\mathbf{a} \in \mathbb{Z}_q^n$ uniformly at random, choosing $e \leftarrow \chi$, and outputting $(\mathbf{a}, b = \langle \mathbf{s}, \mathbf{a} \rangle + e \bmod q)$.

b: SEARCH-LWE$_{n,q,\chi,m}$

Given m independent samples $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ drawn from $A_{\mathbf{s},\chi}$ for a uniformly random $\mathbf{s} \in \mathbb{Z}_q^n$ (fixed for all samples), find $\mathbf{s}$.

c: DECISION-LWE$_{n,q,\chi,m}$

Given m independent samples $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ where every sample is distributed according to either: (1) $A_{\mathbf{s},\chi}$ for a uniformly random $\mathbf{s} \in \mathbb{Z}_q^n$ (fixed for all samples), or (2) the uniform distribution, distinguish which is the case (with non-negligible advantage).

Regev[35] proved the following worst-case hardness theorem for LWE.

For any $m = ploy(n)$, any modulus $q \leq 2^{poly(n)}$, and any (discretized) Gaussian error distribution χ of parameter $\alpha q \geq 2\sqrt{n}$ where $0 < \alpha < 1$, solving the Decision-LWE$_{n,q,\chi,m}$ problem is at least as hard as quantumly solving GapSVP$_\gamma$ and SIVP$_\gamma$ on arbitrary *n-dimensional* lattices, for some $\gamma = \tilde{O}(n/\alpha)$.

3) LINEAR SECRET SHARING SCHEME (LSSS)

Boolean expression is used to describe access policy in attribute-based encryption scheme. Each monotonic Boolean expression can be converted into an equivalent LSSS secret sharing scheme [49]. The LSSS secret sharing scheme is defined as follows.

A linear secret sharing scheme Π over a set of parties P consists of an index map ρ and a share generating matrix $L \in \mathbb{Z}_q^{l\times\theta}$ where l is the number of share specified by the scheme, and θ depends on the structure of the scheme. For all $i \in 1, 2, \ldots, n$, the function ρ maps the i-th row of L to the corresponding party. For the matrix L maps an input θ-vector $v = (s, r_1, \ldots, r_\theta)$, where $s \in \mathbb{Z}_q$ is the secret to be shared and $r_1, \ldots, r_\theta \in \mathbb{Z}_q$ are random, into an output l-vector $L \cdot v = (s_1, \ldots, s_l)$ containing the share of the secret s according to Π. The share $s_i = (Lv)_i$ is assigned to party $\rho(i)$.

C. ORIGINAL SCHEME DEFINITION

1) KP-ABE SCHEME

We refer to the original definition of the ABE as given by Goyal [15]. A key-policy attribute-based encryption scheme (KP-ABE) includes four algorithms as follows:

- $Setup(\lambda) \rightarrow (Pk, Mk)$: It creates the system parameters that executed by attribute authority. It inputs an implicit security parameter λ. It outputs public key Pk and master key Mk.
- $KeyGen(Pk, Mk, (\mathbb{M}, \rho)) \rightarrow Sk$: It generates the users' private key that executed by attribute authority. It inputs public key Pk, master key Mk, and an access control policy $(\mathbb{M}, \rho)$. It outputs private key Sk.
- $Encrypt(Pk, \mathcal{S}, m) \rightarrow C$: It encrypts a message that executed by a data owner. It inputs public key Pk, a subset of attribute $\mathcal{S}$ and a message bit m. It outputs ciphertext C.
- $Decrypt(Pk, Sk, C) \rightarrow m$: It decrypts the ciphertext that executed by a common user. It inputs public key Pk, private key Sk and ciphertext C. It outputs the message bit m just that there is a match between the attributes set $\mathcal{S}$ and the access structure $(\mathbb{M}, \rho)$.

The IND-sCPA security game [50] for KP-ABE scheme: We define an IND-sCPA security model of KP-ABE scheme in the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{B}$.

- **Init.** $\mathcal{A}$ assigns a target attribute set $\mathcal{S}^*$ that wishes to be challenged upon.
- **Setup.** $\mathcal{B}$ invokes the algorithm *Setup* to create public parameter Pk and master key Sk, then sends Pk to $\mathcal{A}$.
- **Phase 1.** $\mathcal{A}$ issues adaptively private key queries on access structure $(\mathbb{M}, \rho)$, only if $\mathcal{S}^*$ does not satisfy $(\mathbb{M}, \rho)$.
- **Challenge.** $\mathcal{A}$ picks two random massages m_0, m_1 to encrypt. $\mathcal{B}$ selects $m_r (r = \{0, 1\})$ at random on the challenge attributes $\mathcal{S}^*$. If $r = 1$, $\mathcal{B}$ send the ciphertext to the adversary $\mathcal{A}$. Otherwise, if $r = 0$, $\mathcal{B}$ send a random element in the ciphertext space to $\mathcal{A}$.
- **Phase 2.** $\mathcal{A}$ may do more private key queries, Phase 1 is repeated.
- **Guess.** $\mathcal{A}$ must give a guess $r' = \{0, 1\}$of r. If $r = r'$, return 1, otherwise return 0. The advantage of an adversary $\mathcal{A}$ is defined as

$$Adv_{\mathcal{A}}^{KP-ABE}(\lambda) = \left|\Pr[r' = r] - \frac{1}{2}\right|$$

Definition 1: A KP-ABE scheme is IND-ABE-sCPA secure if all PPT adversaries have negligible advantage $Adv_{\mathcal{A}}^{KP-ABE}(\lambda)$ in the game above.

2) PEKS SCHEME DEFINITION

Public-key encryption with keywords search (PEKS) is a new type of encryption system, which can search encryption data by public-key cryptography on the keywords. It can ensure the privacy of data sender. There are a lot of definitions to describe PEKS, each of description method is not identical. Here, we refer to the definition of the PEKS functionality as given by Boneh *et al.* [3]. A PEKS scheme includes four algorithms as follows:

- $KeyGen(\lambda) \rightarrow (pk, sk)$: It creates the system parameters that is executed by authority agency. It inputs implied safety parameters λ of system. It outputs public parameter pk and private key sk. Let $\mathcal{W}$ be the set of all possible keywords.
- $Index\ (pk, kw) \rightarrow I_{kw}$: It creates a ciphertext structure of a file with keyword that is executed by a data owner. It inputs public parameters pk and a keyword $kw \in \mathcal{W}$ on data file. It outputs the keyword ciphertext I_{kw} of data and uploads the cloud server.
- $Trapdoor\ (sk, kw) \rightarrow T_{kw}$: It creates a search trapdoor of keyword that is executed by authority agency. It inputs the private key sk and a keyword $kw \in \mathcal{W}$ assigned by

the user. It outputs a trapdoor T_{kw} by using the private key sk.

- $Test(T_{kw}, C_{kw'}) \rightarrow b$: It tests the search result with the trapdoor that is executed by the cloud server. It inputs a trapdoor $T_{kw}$and a ciphertext $I_{kw'} \leftarrow Index(pk, kw')$. It outputs the test result $b = \{0, 1\}$. If $kw = kw'$, return $b = 1$, else return $b = 0$.

The IND-sCPA security game for PEKS scheme: We define an IND-sCPA security model of PEKS scheme in the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{B}$.

- **Init**. $\mathcal{A}$ assigns a challenge keyword set $\mathcal{W}^*$.
- **Setup.** $\mathcal{B}$ runs the algorithm $KeyGen$ to generate (pk, sk) and gives pk to $\mathcal{A}$.
- **Phase 1.** $\mathcal{A}$ can adaptively make trapdoor queries $\langle kw \rangle$ on the challenge keyword set $\mathcal{W}^*$. $\mathcal{B}$ responds with $T_{kw} \leftarrow Trapdoor(sk, kw)$, and returns the trapdoor T_{kw} with keyword kw to $\mathcal{A}$.
- **Challenge.** $\mathcal{A}$ chooses two distinct keywords $kw_0^*, kw_1^* \in \mathcal{W}^*$ at random. The only restriction is that the adversary never previously queried the trapdoor $T_{kw_0^*}$ and $T_{kw_1^*}$ in Phase 1. $\mathcal{B}$ picks a random bit $b \in \{0, 1\}$ and sends $C^*_{kw_b}$ to $\mathcal{A}$ as the challenge ciphertext.
- **Phase 2.** $\mathcal{A}$ may do more trapdoor queries $\langle kw \rangle$ and the only restriction is that $kw' \neq kw_0^*, kw_1^*$. $\mathcal{B}$ responds the same way as in Phase 1.
- **Guess.** $\mathcal{A}$ must give a guess $b' = \{0, 1\}$ of b. If $b = b'$, return 1, otherwise return 0. The advantage of an adversary $\mathcal{A}$ is defined as

$$Adv_{\mathcal{A}}^{PEKS}(\lambda) = \left| \Pr[b' = b] - \frac{1}{2} \right|$$

Definition 2: A PEKS scheme is IND-PEKS-sCPA secure if all PPT adversaries have negligible advantage $Adv_{\mathcal{A}}^{PEKS}(\lambda)$ *in the game above.*

III. SYSTEM MODEL IN CLOUD ENVIRONMENT

In ABE control scene, attribute authority manages user attribute and sends the private key for the users. The user is divided into data owner and common user. In key-policy attribute-based encryption (KP-ABE), a user private key is concerned with an access strategy over attributes, and ciphertext is concerned with a subset of attributes. Only when attribute set satisfies the access strategy, the receiver can encrypt file successfully.

In PEKS control scene, the system entities include authoritative agency, the cloud server, data owner and normal users. In authority agent system, authority agency generates public parameter and the main key for registered user. Data owner encrypts data file with the public parameter and uploads encrypted file to the cloud server. Once a general user puts forward a search requisition, authoritative agency creates a trapdoor with a set of keywords to the user. Then the cloud server performs the test algorithm. If the test is successful, the cloud server returns the ciphertext file containing the trapdoor to the user. Authority agent system is suitable for more complex user application scene, such as electronic medical record system.

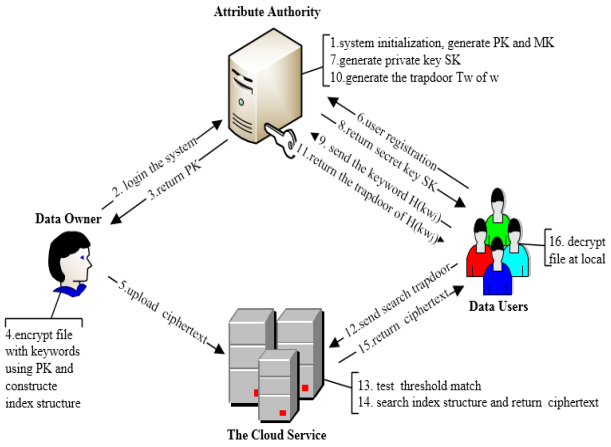


FIGURE 1. The relationship of system entity.

Therefore, combining the advantages of the control scene of ABE and PEKS, we gave a new keyword-searchable ABE scheme in cloud environment. The whole system is composed of four entity roles: attribute authority (AA), the cloud servicer (CS), data owner (DO) and date users (DU). Figure 1 shows the relationship of the entity role on actual cloud system.

a: ATTRIBUTE AUTHORITY (AA)

Attribute authority is seen as a trusted third-party, whose responsibility is to initialize system parameters, create private key of common users on their attribute, and generate a trapdoor of keywords for the users.

b: THE CLOUD SERVICE (CS)

The cloud service is honest but curious third-party data storage center. The cloud service provides data storage structure of encrypted data, which was upload by data owner. And it offers retrieval service on secure trapdoor, when search request was proposed by common users.

c: DATA OWNER (DO)

Data owner is a cloud data provider who needs to encrypt his/her data on some keywords (associating with attributes of the common users), and then upload the ciphertext to the cloud service.

d: DATA USERS (DU)

A common user is another cloud subscriber who has a specified attribute set corresponding private key. He/she wants to query encrypted data from the cloud service. Firstly, the user submits a trapdoor (created by AA) to the cloud service. If the test is successful, the cloud service returns the ciphertext to him/her. Only the users whose attributes satisfying the access policy can decrypt the ciphertext and read the original message.

The system model and workflow are shown in the figure1. The workflow of the system is explained as following.

1. AA initializes system and generates the system public parameter Pk and mast key Mk.
2. DO applies for Pk.
3. AA returns Pk to DO.
4. DO encrypts file with keywords using system public parameter Pk and an assigned attribute set $\mathcal{S}$, and construct the index structure of ciphertext.
5. DO uploads the ciphertext with index structure to CS.
6. DU submits a registration application to AA.
7. AA generates the private key Sk of $(\mathbb{M}, \rho)$ to DU.
8. AA returns the private key Sk to DU.
9. DU submits a keyword hash $H(kw_j)$ to AA.
10. AA creates the trapdoor T_{kw_j} of the hidden keyword $H(kw_j)$ to DU.
11. AA sends the trapdoor T_{kw_j} to DU.
12. DU submits a search requisition and sends the trapdoor T_{kw_j} to CS.
13. CS tests whether the trapdoor T_{kw_j} matches the index structure $\{I_{kw_j}\}_{kw_j \in \mathcal{W}}$. If it matches, the corresponding ciphertext will be labeled as to be determined. If it doesn't match, the algorithm stops.
14. DU verifies whether the attribute set $\mathcal{S}$ satisfies its own access policy $(\mathbb{M}, \rho)$. If satisfies, DU will send a yes signal to CS, CS will send the ciphertext (related to the attribute set $\mathcal{S}$) to DU; If not, the algorithm stops.
15. CS sends the attribute set $\mathcal{S}$ (related to the ciphertext to be determined) to DU.
16. DU decrypts the ciphertext C by the private key Sk at local, and gets the plaintext finally.

IV. ABE-PEKS SCHEME FROM LATTICES

In this section, we will give a keyword-searchable ABE scheme from lattices, which combines the virtue of ABE and PEKS, and the access structure in new scheme is implemented on LSSS.

A. NEW SCHEME DEFINITION

Combining the definition of the key-policy attribute-based encryption (KP-ABE) scheme (see section 2.2) and public-key encryption with keywords search (PEKS) (see section 2.3), we give the definition of the new scheme, as follows:

Definition 3: A keyword-searchable ABE scheme on lattice consists of seven algorithms: setup algorithm, keygen algorithm, encrypt algorithm, index algorithm, trapdoor algorithm, search algorithm, and decrypt algorithm.

- AA.$Setup(\lambda, \mathbb{U}) \rightarrow (Pk, Mk)$: Setup algorithm is run by AA. It takes as input the implicit security parameter λ, and a universal attribute set $\mathbb{U}$. It outputs the public parameter Pk and master key Mk.
- AA.$KeyGen((\mathbb{M}, \rho), Pk, Mk) \rightarrow Sk_{\rho(i)}$: Key algorithm is an interactive operation run by AA and DU. It inputs the public parameter Pk, master key Mk, and an access control policy $(\mathbb{M}, \rho)$ appointed by DU. It outputs private key $Sk_{\rho(i)}$ for the DU who owns attribute $\rho(i)$.
- DO.$Encrypt(\mathcal{S}, Pk, \mathbf{m}) \rightarrow C$: Encryption algorithm is run by DO. It inputs the public parameter Pk, a subset of attribute $\mathcal{S}$, and a message array $\mathbf{m} \in \{0, 1\}^t$. It outputs ciphertext C with $\mathcal{S}$.
- DO.$Index(\mathcal{W}, Pk) \rightarrow I_{kw_j \in \mathcal{W}}$: Index algorithm is run by DO. It inputs the public parameter Pk and a keyword $kw_j \in \mathcal{W}$. It outputs the index $I_{kw_j \in \mathcal{W}}$.
- AA.$Trapdoor(H(kw_j), Pk, Mk) \rightarrow T_{kw_j}$: Trapdoor algorithm is an interactive operation run by AA and DU. Firstly, DU puts forward a request for trapdoor of a hidden keyword $H(kw_j)$ to AA. Then, AA creates the trapdoor T_{kw_j} of the hidden keyword $H(kw_j)$ by the public parameters Pk and master key Mk. Lastly, AA sends the trapdoor T_{kw_j} to DU.
- CS.$Search(T_{kw_j}, D\_stru) \rightarrow C$: The search algorithm is run by CS. Firstly, DU sends the keyword trapdoor T_{kw_j} to CS. Then CS tests the trapdoor on storage structure D_stru. If the test succeeds, sends the ciphertext C to DU. If not, return search fail to DU.
- DU.$Decrypt(Pk, Sk_{\rho(i)}, C) \rightarrow \mathbf{m}$: The decryption algorithm is run by DU and decrypts the ciphertext C on local. It inputs the public parameter Pk, private key $Sk_{\rho(i)}$ and ciphertext C. It outputs the message array $\mathbf{m} \in \{0, 1\}^t$.

B. OUR CONSTRUCTION

We will illustrate system parameters to construct algorithms in this subsection. According to the lemma based on the lattice assumption [35]–[40] (See appendix), the following parameters are defined.

Suppose λ be an implied security parameter, select n, m, q with λ, $n > \Omega(\lambda)$ ($\Omega$is a bounds and asymptotic function). Let m be a lattice base dimension, and q be a prime modulus. Let $H : \{0, 1\}^* \rightarrow \mathbb{Z}_q^{n \times m}$ be a secure hash function. $(\mathbb{M}, \rho)$ represent an access control policy on LSSS structure, where $\mathbb{M} \in \mathbb{Z}^{l \times \theta}$ is a secret sharing matrix, l is the number of attributes, and ρ is an mapping function $\rho : [l] \rightarrow \mathcal{U}$, the i-th row of $\mathbb{M}_i$ will be assigned to a attribute $\rho(i) \in \mathcal{U}$, where $[l] = \{1, 2, \cdots, l\}$.

In new scheme the users can be allowed to retrieve containing certain critical information without the need for decrypting ciphertext. Meanwhile, it has the fine-grained control features upon manage permissions in system. The new scheme includes 7 algorithms as follow.

1) AA.$Setup(\lambda, \mathbb{U}) \rightarrow (Pk, Mk)$

The algorithm is executed by AA. It initializes system based on an implied security parameter λ and attribute set $\mathbb{U} = \{att_1, att_2, \cdots, att_N\}$, do the following:

- Generate a pair $(\mathbf{A}_0 \in \mathbb{Z}_q^{m \times m}, \mathbf{B}_0 \in \mathbb{Z}_q^{m \times m})$ by invoking the algorithm $TrapGen(n, m, q, \sigma)$ (see [36, Theorem 3.2]), where $\mathbf{B}_0 \in \mathbb{Z}_q^{m \times m}$ be a short basis for

$\Lambda_q^{\perp}(\mathbf{A}_0)$ with satisfying the gaussian parameters $\sigma \geq \left\|\tilde{\mathbf{B}}_0\right\| \cdot \omega\left(\sqrt{\log m}\right)$.

- For each attribute $att_k \in \mathbb{U}$, select a random matrix $\mathbf{A}_{att_k} \in \mathbb{Z}_q^{n\times m}$, for all $k = 1, 2, \cdots, N$.
- Select a random integer $s \in \mathbb{Z}_q$ and a random vector $\hat{\mathbf{u}} = \{u_1, u_2, \ldots, u_n\} \in \mathbb{Z}_q^n$, construct a new vector $\mathbf{u} = (s, 0, \ldots, 0) + \hat{\mathbf{u}} = \{(s + u_1), u_2, \cdots, u_n\} \in \mathbb{Z}_q^n$.
- Select a hash function $H : \{0, 1\}^* \to \mathbb{Z}_q^{n\times m}$.
- Return the public parameter $Pk = \{\mathbf{A}_0, \{\mathbf{A}_{att_k}\}_{att_k\in\mathbb{U}}, \mathbf{u}, H\}$ and master key $Mk = \{\mathbf{B}_0, s, \hat{\mathbf{u}}\}$.

AA sends the public parameter Pk to system users, and keeps secret of the master key Mk. The step of algorithm is clarified clearly in **Algorithm 1.**

Algorithm 1 System Initialization

AA.$Setup(\lambda, \mathbb{U}) \to (Pk, Mk)$

Input: an implied security parameter λ and attribute set $\mathbb{U} = \{att_1, att_2, \cdots, att_N\}$

Output: the public parameter Pk and the master key Mk

1. $(\mathbf{A}_0, \mathbf{B}_0) \leftarrow TrapGen(n, m, q, \sigma)$, where Gaussian parameter $\sigma \geq \|\mathbf{B}_0\| \cdot \omega(\sqrt{\log m})$
2. For each attribute $att_k \in \mathbb{U}$, $\mathbf{A}_{att_k} \xleftarrow{R} \mathbb{Z}_q^{n\times m}$, $k = 1, 2, \cdots, N$
3. $S \xleftarrow{R} \mathbb{Z}_q$, $\hat{\mathbf{u}} \xleftarrow{R} \mathbb{Z}_q^n$
4. Set $\mathbf{u} = (s, \mathbf{0}^{n-1}) + \hat{\mathbf{u}} = \{(s + u_1), u_2, \cdots, u_n\} \in \mathbb{Z}_q^n$
5. $H : \{0, 1\}^* \to \mathbb{Z}_q^{n\times m}$
6. $Pk = \{\mathbf{A}_0, \{\mathbf{A}_{att_k}\}_{att_k\in\mathbb{U}}, \mathbf{u}, H\}$
7. $Mk = \{\mathbf{B}_0, s, \hat{\mathbf{u}}\}$
8. Return $\{Pk, Mk\}$

2) AA.$KeyGen((\mathbb{M}, \rho), Pk, Mk) \to Sk_{\rho(i)}$

The algorithm is executed by AA. It creates the private key $Sk_{\rho(i)}$ with the access structure $(\mathbb{M}, \rho)$. The private key generation algorithm is realized on the LSSS secret sharing principle.

- Firstly, AA converts an attribute access control policy to $(\mathbb{M}, \rho)$.
- To select the random integers $\omega_2, \cdots, \omega_\theta$, and construct a vector $\omega = (s, \omega_2, \cdots, \omega_\theta)^T \in \mathbb{Z}_q^\theta$. For each of $\rho(i)(i \in [l])$ upon $(\mathbb{M}, \rho)$, let $(\lambda_1, \cdots, \lambda_l)^T = \mathbb{M} \cdot \omega$, and $p_i = (\lambda_i, 0^{n-1})^T \in \mathbb{Z}_q^n$.
- For all of $\rho(i)$, sample $\zeta_{\rho(i)}$ using the algorithm $LeftSample(\mathbf{A}_0, \mathbf{A}_{\rho(i)}, \mathbf{B}_0, p_i, \sigma)$ (see [40, Theorem 3]). The probability distribution of $\zeta_{\rho(i)}$ be statistically close to $D_{\Lambda_q^{p_i}(\mathbf{A}_0||\mathbf{A}_{\rho(i)}),\sigma}$, such that $[\mathbf{A}_0||\mathbf{A}_{\rho(i)}]\zeta_{\rho(i)} = p_i$, where $\zeta_{\rho(i)} \in \mathbb{Z}_q^{2m}$ satisfying $\left\|\zeta_{\rho(i)}\right\| \leq \sigma\sqrt{2m}$, $\sigma \geq \left\|\tilde{\mathbf{B}}_0\right\| \cdot \omega\left(\sqrt{\log 2m}\right)$.
- In the same way, sample $\eta_{\rho(i)}$ using the algorithm $LeftSample(\mathbf{A}_0, \mathbf{A}_{\rho(i)}, \mathbf{B}_0, \hat{\mathbf{u}}, \sigma)$. The probability distribution $\eta_{\rho(i)}$ be statistically close to of $D_{\Lambda_q^{\hat{u}}(\mathbf{A}_0||\mathbf{A}_{\rho(i)}),\sigma}$, such that $[\mathbf{A}_0||\mathbf{A}_{\rho(i)}]\eta_{\rho(i)} = \hat{\mathbf{u}}$, where$\eta_{\rho(i)} \in \mathbb{Z}_q^{2m}$ satisfying $\left\|\eta_{\rho(i)}\right\| \leq \sigma\sqrt{2m}$, $\sigma \geq \left\|\tilde{\mathbf{B}}_0\right\| \cdot \omega\left(\sqrt{\log 2m}\right)$.
- Finally, AA sends the private key $Sk_{\rho(i)} = \{\zeta_{\rho(i)}, \eta_{\rho(i)}\}$ to the DU for the attribute $\rho(i)$ on the access structure $(\mathbb{M}, \rho)$.

AA generates the private key $Sk_{\rho(i)}$upon$(\mathbb{M}, \rho)$. The step of algorithm is clarified clearly in **Algorithm 2**.

Algorithm 2 Private Key Generation

AA.$KeyGen((\mathbb{M}, \rho), Pk, Mk) \to Sk_{\rho(i)}$

Input: an access policy $(\mathbb{M}, \rho)$, the public parameter Pk, the master key Mk

Output: the private key $Sk_{\rho(i)}$ with $(\mathbb{M}, \rho)$

1. $\omega_2, \cdots, \omega_\theta \xleftarrow{R} \mathbb{Z}_q$, construct $\omega = (s, \omega_2, \cdots, \omega_\theta)^T$
2. For all of $\rho(i)$ upon the access structure $(\mathbb{M}, \rho)$, let $(\lambda_1, \ldots, \lambda_l)^T = \mathbb{M} \cdot \omega$, and $p_i = (\lambda_i, 0^{n-1})^T$
3. $\zeta_{\rho(i)} \leftarrow LeftSample(\mathbf{A}_0, \mathbf{A}_{\rho(i)}, \mathbf{B}_0, p_i, \sigma)$, $\sigma \geq \left\|\tilde{\mathbf{B}}_0\right\| \cdot \omega\left(\sqrt{\log 2m}\right)$
4. $\eta_{\rho(i)} \leftarrow LeftSample(\mathbf{A}_0, \mathbf{A}_{\rho(i)}, \mathbf{B}_0, \hat{\mathbf{u}}, \sigma)$, $\sigma \geq \left\|\tilde{\mathbf{B}}_0\right\| \cdot \omega\left(\sqrt{\log 2m}\right)$
5. $Sk_{\rho(i)} = \{\zeta_{\rho(i)}, \eta_{\rho(i)}\}$
6. Return $Sk_{\rho(i)}$ with the policy $(\mathbb{M}, \rho)$

3) DO.$Encrypt(\mathcal{S}, Pk, \mathbf{m}) \to C$

The encryption algorithm is executed by DO. It encrypts a message array $\mathbf{m} \in \{0, 1\}^t$, and constructs ciphertext C with a subset of attribute $\mathcal{S}=\{att'_1, att'_2, \ldots, att'_{n1}\}$, where $\mathbf{m}$ be seen as symmetric encryption key file. Each bit $m_k(k \in [t])$ of the message array $\mathbf{m}$ will be encrypted respectively.

- Firstly, select a uniform matrix $\mathbf{a} \in \mathbb{Z}_q^n$ at random.
- For each bit $m_k(k \in [t])$ of the message array $\mathbf{m}$, randomly select an error scalar $\chi_1 \leftarrow \bar{\Psi}_\alpha \in \mathbb{Z}_q$, compute $C_{1,k} = \mathbf{a}^T\mathbf{u} + \chi_1 + m_k \cdot \left\lfloor \frac{q}{2} \right\rfloor \bmod q$, for all $k \in [t]$.
- Select an error matrix $\chi_2 \leftarrow \bar{\Psi}_\alpha \in \mathbb{Z}^{2lm}$ at random, compute $C_2 = \mathbf{a}^T \left[\begin{array}{c} \underbrace{\mathbf{A}_0||\tilde{\mathbf{A}}_1, \mathbf{A}_0||\tilde{\mathbf{A}}_2, \cdots, \mathbf{A}_0||\tilde{\mathbf{A}}_l}_{\begin{array}{c}if\ att'_i \in \mathcal{S}, \tilde{\mathbf{A}}_i = \mathbf{A}_{att_i} \\ else\ att'_i \notin \mathcal{S}, \tilde{\mathbf{A}}_i = 0\end{array}} \end{array}\right] + \chi_2 \bmod q$.
- Lastly, return the ciphertext $C = \left\{\left\{C_{1,k}, C_2\right\}_{k\in[t]}, \mathcal{S}\right\}$ with the subset of attribute $\mathcal{S}$.

DO sends the ciphertext C with $\mathcal{S}$ to CS. The step of algorithm is clarified clearly in **Algorithm 3.**

4) DO.$Index(\mathcal{W}, Pk) \to I_{kw_j\in\mathcal{W}}$

The index generation algorithm is executed by DO. It constructs system index structure for a keyword kw_j on the keyword set $\mathcal{W}$, where $kw_j = \{0, 1\}^*$ is seen as the lexicographical order synset by created WordNet library.

- For a keyword $kw_j \in \mathcal{W}$, select a random n-vector $\mathbf{b} \in \mathbb{Z}_q^n$.
- Compute the hash function $H(kw_j)$, and construct a new matrix $\mathbf{A}_{kw_j} = H(kw_j) \in \mathbb{Z}_q^{n\times(d_1+1)m}$.

Algorithm 3 Encryption Algorithm

DO.*Encrypt*$(\mathcal{S}, Pk, \mathbf{m}) \rightarrow C$

Input: a subset of attribute $\mathcal{S}$, the public parameter Pk, and a message array $\mathbf{m} \in \{0,1\}^t$,

Output: C with $\mathcal{S}$

1. $\mathbf{a} \xleftarrow{R} \mathbb{Z}_q^n$
2. For all $k \in [t]$, randomly select an error $\chi_1 \xleftarrow{\$} \bar{\Psi}_\alpha \in \mathbb{Z}_q$, compute $C_{1,k} = \mathbf{a}^T\mathbf{u} + \chi_1 + m_k \cdot \lfloor \frac{q}{2} \rfloor \bmod q$
3. Randomly select an error matrix $\chi_2 \xleftarrow{\$} \bar{\Psi}_\alpha \in \mathbb{Z}^{2lm}$, compute $C_2 = \mathbf{a}^T \left[\underbrace{\mathbf{A}_0||\tilde{\mathbf{A}}_1, \mathbf{A}_0||\tilde{\mathbf{A}}_2, \cdots, \mathbf{A}_0||\tilde{\mathbf{A}}_l}_{\substack{if\ att_i' \in \mathcal{S}, \tilde{\mathbf{A}}_i = \mathbf{A}_{att_i} \\ else\ att_i' \notin \mathcal{S}, \tilde{\mathbf{A}}_i = 0}} \right] + \chi_2 \bmod q$
4. Return$C = \left\{ \{C_{1,k}, C_2\}_{k\in[t]}, \mathcal{S} \right\}$

- Construct the matrix $\mathbf{F}_{kw_j}$=$[\mathbf{A}_0||\mathbf{A}_{kw_j}] \in \mathbb{Z}_q^{n\times 2m}$.
- Randomly select an error scalar $\chi_3 \leftarrow \bar{\Psi}_\alpha \in \mathbb{Z}_q$, compute $I_1 = \mathbf{b}^T\mathbf{u} + \chi_3$.
- Randomly select an error vector $\chi_4 \leftarrow \bar{\Psi}_\alpha \in \mathbb{Z}_q^{2m}$, compute $I_{2,j} = \mathbf{b}^T\mathbf{F}_{kw_j} + \chi_4$.
- Construct index structure$I_{kw_j\in\mathcal{W}} = \{I_1, I_{2,j}\}$.
- Lastly, build data index structure $D\_stru = \{\{I_{kw_j}\}_{kw_j\in\mathcal{W}}, C\}$ by combined the ciphertext C and index structure I_{kw_j} for $\mathcal{W}$.

DO unloads data storage structure D_stru to CS. The step of algorithm is clarified clearly in **Algorithm 4.**

Algorithm 4 Generation Index

DO.*Index*$(\mathcal{W}, Pk) \rightarrow I_{kw_j\in\mathcal{W}}$

Input: a keyword $kw_j \in \mathcal{W}$, the hash function H, the public key Pk

Output: data index structure D_stru with $\mathcal{S}$

1. Randomly select a vector $\mathbf{b} \xleftarrow{R} \mathbb{Z}_q^n$
2. For each keyword $kw_j \in \mathcal{W}$, compute the hash value $H(kw_j)$, construct a new matrix $\mathbf{A}_{kw_j} = H(kw_j) \in \mathbb{Z}_q^{n\times 2m}$, and let $\mathbf{F}_{kw_j} = [\mathbf{A}_0||\mathbf{A}_{kw_j}] \in \mathbb{Z}_q^{n\times 2m}$
3. Randomly select an error $\chi_3 \xleftarrow{\$} \bar{\Psi}_\alpha \in \mathbb{Z}_q$, compute $I_1 = \mathbf{b}^T\mathbf{u} + \chi_3$
4. $\chi_4 \xleftarrow{\$} \bar{\Psi}_\alpha \in \mathbb{Z}_q^{2m}$, compute $I_{2,j} = \mathbf{b}^T\mathbf{F}_{kw_j} + \chi_4$
5. Construct index structure $I_{kw_j\in\mathcal{W}} = \{I_1, I_{2,j}\}$
6. Return $D\_stru = \{\{I_{kw_j}\}_{kw_j\in\mathcal{W}}, C\}$

5) AA.*Trapdoor*$(H(kw_j), Pk, Mk) \rightarrow T_{kw_j}$

The trapdoor algorithm is executed by AA. Firstly, DU puts forward a request for trapdoor, and submits a hidden keyword $H(kw_j)$ to AA. AA creates the keyword trapdoor T_{kw_j} to DU.

- DU submits a hidden keyword $H(kw_j)$ to AA.
- Construct the matrix $\tilde{\mathbf{A}}_{kw_j} = H(kw_j) \in \mathbb{Z}_q^{n\times m}$.
- Generate $T_{kw_j} \in \mathbb{Z}_q^{n\times 2m}$ by invoking the algorithm $LeftSample(\mathbf{A}_0, \tilde{\mathbf{A}}_{kw_j}, \mathbf{B}_0, \mathbf{u}, \sigma)$, such that $[\mathbf{A}_0||\tilde{\mathbf{A}}_{kw_j}] \cdot T_{kw_j} = \mathbf{u}$, where the probability distribution of T_{kw_j} be statistically close to $D_{\Lambda_q^{\mathbf{u}}\left(\mathbf{A}_0||\tilde{\mathbf{A}}_{kw_j}\right),\sigma}$, and satisfying the condition $\sigma \geq \left\|\widetilde{\mathbf{B}}_0\right\| \cdot \omega\left(\sqrt{\log 2m}\right)$.
- Lastly, AA delivers the trapdoor T_{kw_j} of the hidden keyword $H(kw_j)$ to DU.

AA generate trapdoor T_{kw_j} to DU. The step of algorithm is clarified clearly in **Algorithm 5**.

Algorithm 5 Trapdoor Algorithm

AA.*Trapdoor*$(H(kw_j), Pk, Mk) \rightarrow T_{kw_j}$

Input: a hidden keyword $H(kw_j)$, the public key Pk, the master key Mk

Output: the trapdoor T_{kw_j} of hidden keyword $H(kw_j)$

1. Submits a hidden keyword $H(kw_j)$ to AA.
2. $T_{kw_j} \leftarrow LeftSample(\mathbf{A}_0, \tilde{\mathbf{A}}_{kw_j}, \mathbf{B}_0, \mathbf{u}, \sigma)$, $\sigma \geq \left\|\widetilde{\mathbf{B}}_0\right\| \cdot \omega\left(\sqrt{\log 2m}\right)$
3. Return trapdoorT_{kw_j}

6) CS.*Search*$(T_{kw_j}, D\_stru) \rightarrow C$

The search algorithm is executed by CS. Firstly, DU sends the keyword trapdoor T_{kw_j} to CS. Then CS tests the trapdoor T_{kw_j} on data storage structure D_stru. If the test succeeds, CS will label the ciphertext C to as to be determined by DU. If not, the algorithm will be terminated, where the tested index is $I = \{I_1, I_2\}$.

- For a trapdoor T_{kw_j} of $H(kw_j)$, CS computes $\eta = \left|I_2 T_{kw_j} - I_1\right| \in \mathbb{Z}_q$.
- If $\eta \leq \lfloor \frac{q}{4} \rfloor$, the test is successful, CS to label the ciphertext C, otherwise the search is failed.

The step of algorithm is clarified clearly in **Algorithm 6.**

Algorithm 6 Search Algorithm

CS.*Search*$(T_{kw_j}, D\_stru) \rightarrow C$

Input: a trapdoor T_{kw_j}, data storage structure D_stru

Output: C with a trapdoor T_{kw_j}

1. Search on $D\_stru = \{\{I_{kw_j}\}_{kw_j\in\mathcal{W}}, C\}$, where the tested index is $I = \{I_1, I_2\}$.
2. For each trapdoor T_{kw_j}, compute $\eta = \left|I_2 T_{kw_j} - I_1\right| \in \mathbb{Z}_q$
3. If $\eta \leq \lfloor \frac{q}{4} \rfloor$, to label the ciphertext C, otherwise the algorithm$\perp$.

7) DU.*Decrypt*$(Pk, Sk_{\rho(j)}, C) \rightarrow \mathbf{m}$

The decryption algorithm is executed by DU, and decrypts the ciphertext Con local, and ultimately gets the message array $\mathbf{m}$=$\{m_1, m_2, \ldots, m_t\} = \{0,1\}^t$.

- If $\mathcal{S}'$ is an authorized set on $(\mathbb{M}, \rho)$, that means it must exist a set of integer coefficients $\{g'_{\rho(j)} \in \mathbb{Z}_q\}$ to meet

the equation $\sum_{j\in\mathcal{S}} g'_{\rho(j)}\mathbb{M}_j = (1, 0, \ldots, 0)$. Each of attribute $\rho(j)$ in $\mathcal{S}'$ correspond to the attribute in $\mathbb{U}$.

- Compute
$v_k \hat{=} C_{1,k} - C_2[\boldsymbol{d}_1, \cdots, \boldsymbol{d}_l]^T - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}} C_2[\mathbf{e}_1, \cdots, \mathbf{e}_l]^T \bmod q$, where $d_j = \begin{cases} g'_{\rho(j)}\xi^T_{\rho(j)}, & j \in \mathcal{S}' \\ 0, & j \notin \mathcal{S}' \end{cases}$, $\mathbf{e}_j = \begin{cases} g'_{\rho(j)}\eta^T_{\rho(j)}, & j \in \mathcal{S}' \\ 0, & j \notin \mathcal{S}' \end{cases}$
- For each bit $m_k (k \in [t])$ of the message array $\mathbf{m}$, since $|v_k|$ must be an integer in $\left[-\lfloor \frac{q}{2} \rfloor, \lfloor \frac{q}{2} \rfloor\right]$.
- If $|v_k| \le \lfloor \frac{q}{4} \rfloor$, then $m_k = 0$, else $m_k = 1$.
- Lastly, return the message $\mathbf{m} = \{m_1, m_2, \ldots, m_t\} = \{0, 1\}^t$.

The users of the authorized set on $(\mathbb{M}, \rho)$ will successfully decrypt ciphertext, and get the message array $\mathbf{m} = \{m_1, m_2, \ldots, m_t\} = \{0, 1\}^t$ on local. The step of algorithm is clarified clearly in **Algorithm 7.**

Algorithm 7 Decryption Algorithm

DU.$Decrypt(Pk, Sk_{\rho(i)}, C) \rightarrow \mathbf{m}$

Input: the public parameter Pk, the private key $Sk_{\rho(i)}$, the ciphertext C.

Output: the message array $\mathbf{m} = \{m_1, m_2, \ldots, m_t\} = \{0, 1\}^t$

1. Compute
$v_k \hat{=} C_{1,k} - C_2[\boldsymbol{d}_1, \cdots, \boldsymbol{d}_l]^T - \frac{1}{\sum_{j\in J} g'_{\rho(j)}} C_2[\mathbf{e}_1, \cdots, \mathbf{e}_l]^T \bmod q$, where $d_j = \begin{cases} g'_{\rho(j)}\xi^T_{\rho(j)}, & j \in \mathcal{S}' \\ 0, & j \notin \mathcal{S}' \end{cases}$,
$\mathbf{e}_j = \begin{cases} g'_{\rho(j)}\eta^T_{\rho(j)}, & j \in \mathcal{S}' \\ 0, & j \notin \mathcal{S}' \end{cases}$.
2. For each bit $m_k (k \in [t])$ of message, to judge
if $|v_i| \le \lfloor \frac{q}{4} \rfloor$
$\mathbf{m}_i = 0$
else
$\mathbf{m}_i = 1$
end
3. Return $\mathbf{m} = \{m_1, m_2, \ldots, m_t\} = \{0, 1\}^t$

V. THE WORKFLOW OF NEW SCHEME

To explain the system workflow more clearly, we will give the six phase and the relationship of parameter between the four entities of the new scheme. The details are as follows.

1) The first phase, AA performs *Setup* algorithm to generate public key Pk and master key Mk. The public key Pk is distributed to CS and DO. The master key Mk is kept secret by AA.
2) In the second phase, the DO encrypts documents by invoking *Encrypt* algorithm, and constructs system index structure I_{kw_j} for each keyword by invoking *Index* algorithm. Then DO transmits the data storage structure D_stru to CS.

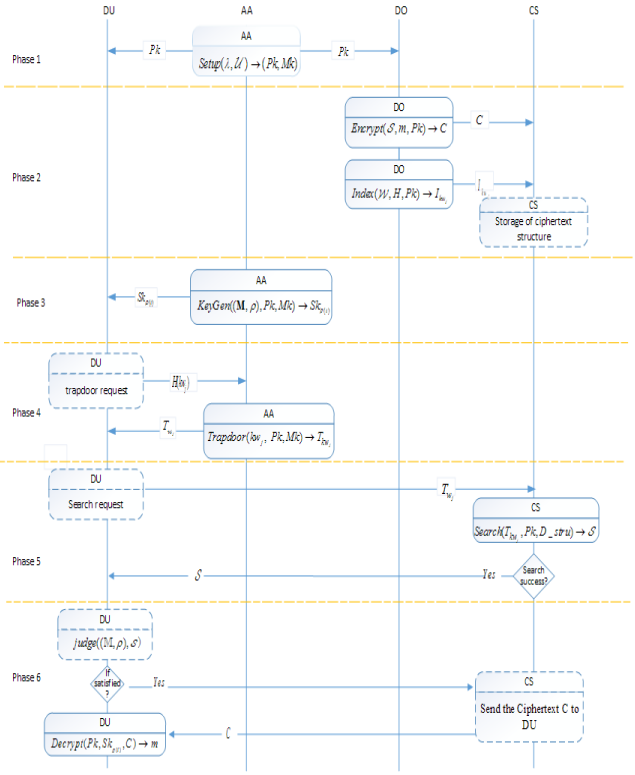


FIGURE 2. The workflow of new scheme.

3) In the third phase, AA generates private key $Sk_{\rho(i)}$ by *KeyGen* algorithm, and sends to the DU.
4) In the fourth phase, the DU firstly requests for the trapdoor of a keyword $H(kw_j)$ to AA. Then AA generates the trapdoor T_{kw_j} by invoking *Trapdoor* algorithm, and sends it to DU.
5) In the fifth phase, the DU firstly submits a search request to CS on the search trapdoor T_{kw_j}. Then CS executes *Search* algorithm with trapdoor T_{kw_j}. If it searches successfully, CS transmits the ciphertext C to DU.
6) In the last phase, the DU will decrypt ciphertext C by using own private key $Sk_{\rho(i)}$ on local.

Here, we give a detailed explanation of keyword search for encrypted files. The workflow of new scheme is shown below.

VI. SCHEME CORRECTNESS ANALYSIS

A. SEARCH CORRECTNESS

The search algorithm is executed by CS. CS tests the trapdoor T_{kw_j} on data storage structure D_stru.

$$\begin{aligned}\eta &= \left|I_2 T_{kw_j} - I_1\right| \\ &= \left|(\mathbf{b}^T\mathbf{F}_{kw_j} + \chi_4)T_{kw_j} - (\mathbf{b}^T\mathbf{u} + \chi_3)\right| \\ &= \left|(\mathbf{b}^T[\mathbf{A}_0||\mathbf{A}_{kw_j}] + \chi_4)T_{kw_j} - (\mathbf{b}^T\mathbf{u} + \chi_3)\right| \\ &= \left|(\mathbf{b}^T[\mathbf{A}_0||\mathbf{A}_{kw_j}]T_{kw_j} + \chi_4 \cdot T_{kw_j}) - (\mathbf{b}^T\mathbf{u} + \chi_3)\right|\end{aligned}$$

$$
\begin{aligned}
&= \left|(\mathbf{b}^T\mathbf{u} + \chi_4 \cdot T_{kw_j}) - (\mathbf{b}^T\mathbf{u} + \chi_3)\right| \\
&= \left|(\mathbf{b}^T\mathbf{u} + \chi_4 \cdot T_{kw_j} - \mathbf{b}^T\mathbf{u} - \chi_3)\right| \\
&= \left|\chi_4 \cdot T_{kw_j} - \chi_3\right| \\
&\approx 0 \bmod q
\end{aligned}
$$

B. DECRYPTION CORRECTNESS

If $\mathcal{S}'$ be authorized set upon the access policy $(\mathbb{M}, \rho)$, then there exists a proper vector $\{g'_{\rho(j)} \in \mathbb{Z}_q\}$ to satisfy $\sum_{j\in\mathcal{S}'} g'_{\rho(j)}\mathbb{M}_j = (1, 0, \ldots, 0)$. The authorized users can encrypt the ciphertext correctly by using the secret key $Key = \{\xi_{\rho(j)}, \eta_{\rho(j)}\}$. We will discuss decryption process as following, ε^*, as shown at the top of the next page.

Note that the error term must satisfy the condition:

$$
\left|\varepsilon^*\right| = \left| \chi_1 - \chi_2 \left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}} \chi_2 \left[g'_{\rho(1)}\eta^T_{\rho(1)}, \cdots, g'_{\rho(l)}\eta^T_{\rho(l)}\right]^T \right| \le \frac{q}{5}.
$$

Next, we will discuss the scheme parameters based on Lemma 1 below.

Lemma 1 (Correctness): For any prime $q = poly(n) \ge 2$ and $m \ge 5n\log q$, *if we can find a suitable Gaussian distribution parameter* $\alpha \ge 2\sqrt{m}/q$ *in* $LWE_{q,\chi}$ *(where* χ *is the noise distribution* $\bar{\Psi}^m_\alpha$*), such that the norm length of extraction private key* $key_i = \{\xi_i, \eta_i\}$ *satisfy* $||\xi_i|| \le \sigma\sqrt{m}$, $||\eta_i|| \le \sigma\sqrt{m}$ *(using the SamplePre algorithm (see literature [35]) with a Gaussian parameter* $\sigma \ge ||\tilde{\mathbf{B}}_0|| \cdot \omega\left(\sqrt{\log m}\right)$*), then the scheme can be correctly decrypt overwhelming probability.*

Proof:

$$
\begin{aligned}
\left|\varepsilon^*\right| &\hat{=} |\chi_1 - \chi_2 \left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T \\
&\quad - \frac{1}{\sum_{j\in[l]} g'_{\rho(j)}} \chi_2 \left[g'_{\rho(1)}\eta^T_{\rho(1)}, \cdots, g'_{\rho(l)}\eta^T_{\rho(l)}\right]^T | \\
&\le |\chi_1| + |\chi_2 \left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T \\
&\quad - \frac{1}{\sum_{j\in[l]} g'_{\rho(j)}} \chi_2 \left[g'_{\rho(1)}\eta^T_{\rho(1)}, \cdots, g'_{\rho(l)}\eta^T_{\rho(l)}\right]^T | \\
&\le |\chi_1| + |\chi_2 \left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T | \\
&\quad + |\frac{g'_{\rho(j)}}{\sum_{j\in[l]} g'_{\rho(j)}} \eta^T_{\rho(j)}\chi_2| \\
&\le |\chi_1| + |\sum_{j=1}^{l} g'_{\rho(j)}\xi^T_{\rho(j)}\chi_2| + |\sum_{j=1}^{l} g'_{\rho(j)}\eta^T_{\rho(j)}\chi_2| \\
&\le (q\alpha \cdot \omega(\sqrt{\log 2m}) + 1/2) + 2lt(m^{1.5} \cdot \omega(\sqrt{\log 2m}) \\
&\quad \cdot (q\alpha \cdot \omega(\sqrt{\log 2m}) + \sqrt{m}/2)) \\
&\le (q\alpha \cdot \omega(\sqrt{\log 2m}) + 2\sqrt{m})(1 + 2l(m^{1.5} \cdot \omega(\sqrt{\log 2m})) \\
&\le q\alpha(\omega(\sqrt{\log 2m}) + 1)(1 + 2l(m^{1.5} \cdot \omega(\sqrt{\log 2m}))
\end{aligned}
$$

With special attention, in the above derivation $k \in [\mathbf{m}]$ is the number of message array $\mathbf{m} = \{0, 1\}^t$.

If we let $|\varepsilon| \le q/5$, then $\alpha \le \frac{1}{5}[t(\omega(\sqrt{\log 2m}) + 1)(1 + 2l(m^{1.5} \cdot \omega(\sqrt{\log 2m}))]^{-1}$.

Because $q\alpha \ge 2\sqrt{m}$, we can get $q \ge 10\sqrt{m}[t(\omega(\sqrt{\log 2m}) + 1)(1 + l(m^{1.5} \cdot \omega(\sqrt{\log 2m}))]$.

Thus, for the above parameters, the ciphertext can be correctly decrypt with overwhelming probability.

VII. SCHEME SECURITY ANALYSIS

We give a keyword-searchable ABE scheme on lattice in section 4. It is clearly that new scheme includes ABE-ciphertext security and keyword-index security. The message encryption process is a KP-ABE sub-scheme (in which includes *Setup*, *KeyGen*, *Encrypt* and *Decrypt*), and the keyword search process is PEKS sub-scheme (in which includes *Index*, *Trapdoor* and *Search*). Next, we will give the security proof of ABE-ciphertext security and keyword-index security respectively.

A. ABE-CIPHERTEXT SECURITY

Here, we will give the ABE-ciphertext security of new scheme from lattice on the selective-security model by **definition 1**.

Theorem 1: For properly parameters n, m, q, α *by discussed in Lemma 1, the KP-ABE sub-scheme of new scheme is IND-ABE-sCPA secure under the* $LWE_{q,\chi}$ *assumption.*

Proof: Suppose an adversary $\mathcal{A}$ is able to break this scheme, a challenger $\mathcal{B}$ could solve decisional $LWE_{q,\chi}$ problem. The $LWE_{q,\chi}$ problem instance is provided as a sampling oracle $\mathcal{O}$ which can be either truly random $\mathcal{O}_\$$ or noisy pseudo-random $\mathcal{O}_s$ for some secret key $S \in \mathbb{Z}^n_q$. The challenger $\mathcal{B}$ uses the adversary $\mathcal{A}$ to distinguish its. There duction proceeds as follows.

- **Instance.** The challenger $\mathcal{B}$ requests from $\mathcal{O}$ and receives $(lm + 1)LWE_{q,\chi}$ samples that we denote as $(\mathbf{w}_0, v_0), \{(\mathbf{w}^1_1, v^1_1), (\mathbf{w}^2_1, v^2_1), \ldots, (\mathbf{w}^m_1, v^m_1)\}, \ldots, \{(\mathbf{w}^1_l, v^1_l), (\mathbf{w}^2_l, v^2_1), \ldots, (\mathbf{w}^m_l, v^m_l)\} \in \{\mathbb{Z}^n_q \times \mathbb{Z}_q\}^{(lm+1)}$.
- **Targeting.** The adversary $\mathcal{A}$ announces a target attribute set $Attrib^*$ that wishes to be challenged upon.
- **Setup.** The challenger $\mathcal{B}$ constructs the public parameters as follows:

  1. By using algorithm $TrapGen(n, m, q, \sigma)$, to generate a pair $\mathbf{A}_0 \in \mathbb{Z}^{n\times m}_q$ and $\mathbf{B}_0 \in \mathbb{Z}^{m\times m}_q$, such that $\mathbf{A}_0$ is statistically close to uniform, and $\mathbf{B}_0$ is a basis for $\Lambda^\perp_q(\mathbf{A}_0)$ with length satisfy $||\tilde{\mathbf{B}}_0|| \le m \cdot \omega(\sqrt{\log m})$.
  2. If $u_k \in Attrib^*$, $\mathbf{A}u_k$ be constructed using the LWE samples $\{(\mathbf{w}^1_i), (\mathbf{w}^2_i), \ldots, (\mathbf{w}^m_i)\}_{i\in[\mathcal{U}]}$.
  3. If $u_k \notin Attrib^*$, $\mathbf{A}u_k \in \mathbb{Z}^{n\times m}_q$ is selected as a uniform random matrix.
  4. The vector $\hat{\mathbf{u}} \in \mathbb{Z}^n_q$ is constructed from the LWE samples, set $\hat{\mathbf{u}} = \mathbf{w}_0$.

$$
\begin{aligned}
\varepsilon^* \hat{=}\ & C_{1,k} - C_2\,[d_1, \cdots, d_t]^T - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}} C_2\,[\mathbf{e}_1, \cdots, \mathbf{e}_t]^T \\
=\ & C_{1,k} - C_2\left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}} C_2\left[g'_{\rho(1)}\eta^T_{\rho(1)}, \cdots, g'_{\rho(l)}\eta^T_{\rho(l)}\right]^T \\
=\ & C_{1,k} - \mathbf{a}^T\left[\mathbf{A}_0||\tilde{\mathbf{A}}_{\rho(1)}, \cdots, \mathbf{A}_0||\tilde{\mathbf{A}}_{\rho(l)}\right]\left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T \\
& - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}}\mathbf{a}^T\left[\mathbf{A}_0||\tilde{\mathbf{A}}_{\rho(1)}, \cdots, \mathbf{A}_0||\tilde{\mathbf{A}}_{\rho(l)}\right]\left[g'_{\rho(1)}\eta^T_{\rho(1)}, \cdots, g'_{\rho(l)}\eta^T_{\rho(l)}\right]^T \\
& - \chi_2\left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}}\chi_2\left[g'_{\rho(1)}\eta^T_{\rho(1)}, \cdots, g'_{\rho(l)}\eta^T_{\rho(l)}\right]^T \\
=\ & C_{1,k} - \mathbf{a}^T\left[\, g'_{\rho(1)}(\mathbf{A}_0||\tilde{\mathbf{A}}_{\rho(1)})\xi^T_{\rho(1)} + \cdots + g'_{\rho(l)}(\mathbf{A}_0||\tilde{\mathbf{A}}_{\rho(l)})\xi^T_{\rho(l)}\,\right] \\
& - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}}\mathbf{a}^T\left[\, g'_{\rho(1)}(\mathbf{A}_0||\tilde{\mathbf{A}}_{\rho(1)})\eta^T_{\rho(1)} + \cdots + g'_{\rho(l)}(\mathbf{A}_0||\tilde{\mathbf{A}}_{\rho(l)})\eta^T_{\rho(l)}\,\right] \\
& - \chi_2\left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}}\chi_2\left[g'_{\rho(1)}\eta^T_{\rho(1)}, \cdots, g'_{\rho(l)}\eta^T_{\rho(l)}\right]^T \\
=\ & C_{1,i} - \mathbf{a}^T\left[g'_{\rho(1)}p_1 + \cdots + g'_{\rho(l)}p_l\right] - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}}\mathbf{a}^T\left[g'_{\rho(1)}\hat{\mathbf{u}} + \cdots + g'_{\rho(l)}\hat{\mathbf{u}}\right] \\
& - \chi_2\left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}}\chi_2\left[g'_{\rho(1)}\eta^T_{\rho(1)}, \cdots, g'_{\rho(l)}\eta^T_{\rho(l)}\right]^T \\
=\ & C_{1,i} - \mathbf{a}^T\left[g'_{\rho(1)}(\lambda_1, 0^{n-1}) + \cdots + g'_{\rho(l)}(\lambda_l, 0^{n-1})\right] - \mathbf{a}^T\hat{\mathbf{u}} \\
& - \chi_2\left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}}\chi_2\left[g'_{\rho(1)}\eta^T_{\rho(1)}, \cdots, g'_{\rho(l)}\eta^T_{\rho(l)}\right]^T \\
=\ & C_{1,k} - \mathbf{a}^T\left[(g'_{\rho(1)}\lambda_1 + \cdots + g'_{\rho(l)}\lambda_l), 0^{n-1}\right] - \mathbf{a}^T\hat{\mathbf{u}} \\
& - \chi_2\left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}}\chi_2\left[g'_{\rho(1)}\eta^T_{\rho(1)}, \cdots, g'_{\rho(l)}\eta^T_{\rho(l)}\right]^T \\
=\ & C_{1,i} - \mathbf{a}^T\left[(g'_{\rho(1)}, \cdots, g'_{\rho(l)})M\cdot\omega, 0^{n-1}\right]^T - \mathbf{a}^T\hat{\mathbf{u}} \\
& - \chi_2\left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}}\chi_2\left[g'_{\rho(1)}\eta^T_{\rho(1)}, \cdots, g'_{\rho(l)}\eta^T_{\rho(l)}\right]^T \\
=\ & C_{1,k} - \mathbf{a}^T\left[s, 0^{n-1}\right]^T - \mathbf{a}^T\hat{\mathbf{u}} \\
& - \chi_2\left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}}\chi_2\left[g'_{\rho(1)}\eta^T_{\rho(1)}, \cdots, g'_{\rho(l)}\eta^T_{\rho(l)}\right]^T \\
=\ & \mathbf{m}_k\cdot\lfloor\tfrac{q}{2}\rfloor + \underbrace{\mathbf{a}^T\mathbf{u} - (\mathbf{a}^T\left[s, 0^{n-1}\right]^T + \mathbf{a}^T\hat{\mathbf{u}})}_{=0 \bmod q} \\
& \underbrace{+\ \chi_1 - \chi_2\left[g'_{\rho(1)}\xi^T_{\rho(1)}, \cdots, g'_{\rho(l)}\xi^T_{\rho(l)}\right]^T - \frac{1}{\sum_{j\in\mathcal{S}} g'_{\rho(j)}}\chi_2\left[g'_{\rho(1)}\eta^T_{\rho(1)}, \cdots, g'_{\rho(l)}\eta^T_{\rho(l)}\right]^T}_{\approx 0 \bmod q}
\end{aligned}
$$

5. Select a random integer $s \in \mathbb{Z}_q$, to construct a new vector $\mathbf{u} = (s, 0^{n-1}) + \hat{\mathbf{u}} = \{(s + u_1), u_2, \cdots, u_n\} \in \mathbb{Z}_q^n$.

The challenger $\mathcal{B}$ returns the public parameter $Pk = \{\mathbf{A}_0, \{\mathbf{A}_{u_k}\}_{u_k \in \mathbb{U}}, \mathbf{u}\}$ to the adversary $\mathcal{A}$. And $\{\mathbf{B}_0, s, \hat{\mathbf{u}}\}$ is the master key is protected by system.

- **Queries.** The adversary $\mathcal{A}$ can issue adaptive queries for a secret key $Sk_{\rho(i)}$ by submitting attribute $\rho(i)$ to the challenger $\mathcal{B}$. The challenger $\mathcal{B}$ constructs and returns the private-key $Sk_{\rho(i)}$ for each query policy $(\mathbb{M}, \rho)$ as follows, where $\rho(i)$ is an attribute on its choice policy $(\mathbb{M}, \rho)$, as long as the target attribute list $\mathcal{S}^*$ does not satisfy.
  1. Let $\kappa \hat{=} \{\rho(i_1), \rho(i_2), .., \rho(i_l)|, 1 \leq i_k \leq l\}$ denote the number of attribute on choice police $(\mathbb{M}, \rho)$, compute the user private key $Sk_{\rho(i)}$ for each attribute $\rho(i_k)$ on the query policy $(\mathbb{M}, \rho)$.
  2. Construct the encryption matrix $\mathbf{F} = [\mathbf{A}_0||\mathbf{A}_{\rho(i_1)}, \mathbf{A}_0||\mathbf{A}_{\rho(i_2)}, \cdots, \mathbf{A}_0||\mathbf{A}_{\rho(i_l)}]$, let $\mathbf{F}_{\rho(i_k)} = [\mathbf{A}_0||\mathbf{A}_{\rho(i_k)}]$.
  3. Since the target attribute set $\mathcal{S}^*$ do not satisfy the query policy $(\mathbb{M}, \rho)$, there must be at least one attribute $\rho(i_k) \in \kappa$ and $\rho(i_k) \notin \mathcal{S}^*$. The challenger invokes algorithm $ExtBasis(\mathbf{F}_{\rho(i_k)}, \mathbf{B}_0)$ to generate $\mathbf{T}_{\mathbf{F}_{\rho(i_k)}}$, which is a short base on orthogonal lattice $\Lambda_q^{\perp}\left(\mathbf{F}_{\rho(i_k)}\right)$.
  4. Generate a short base $\mathbf{T}_{\mathbf{F}}$ on the matrix $\mathbf{F} = [\mathbf{A}_0||\mathbf{A}_{\rho(i_1)}, \mathbf{A}_0||\mathbf{A}_{\rho(i_2)}, \cdots, \mathbf{A}_0||\mathbf{A}_{\rho(i_l)}]$ by invoking algorithm $ExtBasis(\mathbf{F}, \mathbf{T}_{\mathbf{F}_{\rho(i_k)}})$.
  5. Construct a vector $\omega = (s, \omega_2, \cdots, \omega_\theta)^T \in \mathbb{Z}_q^\theta$, where$\omega_2, \cdots, \omega_\theta$ is random integers. Let $(\lambda_1, \cdots, \lambda_l)^T = \mathbb{M} \cdot \omega, p_i = (\lambda_i, 0)^T$.
  6. Create a short vector $(\zeta_{\rho(i_1)}, \zeta_{\rho(i_2)}, \cdots, \zeta_{\rho(i_l)})^T$ by invoking algorithm $SamplePre(\mathbf{F}, \mathbf{T}_{\mathbf{F}}, p_{\rho(i)}, \sigma)$, such that satisfy $\mathbf{F} \cdot [\zeta_{\rho(i_1)}, \zeta_{\rho(i_2)}, \cdots, \zeta_{\rho(i_l)}]^T \hat{=} p_{\rho(i)}$.
  7. Similarly, create the short vectors $(\eta_{\rho(i_1)}, \eta_{\rho(i_2)}, \cdots, \eta_{\rho(i_l)})^T$ by invoking algorithm $SamplePre(\mathbf{F}, \mathbf{T}_{\mathbf{F}}, \hat{\mathbf{u}}, \sigma)$, such that it satisfies $\mathbf{F}[\eta_{\rho(i_1)}, \eta_{\rho(i_2)}, \cdots, \eta_{\rho(i_{num_0})}]^T \hat{=} \hat{\mathbf{u}}$.

The challenger returns the private-key pair $Sk = \{\zeta_{\rho(i_k)}, \eta_{\rho(i_k)}\}$ for each $\rho(i_k)$ in attribute list κ on the policy $(\mathbb{M}, \rho)$ to the adversary.

- **Challenge.**The adversary $\mathcal{A}$ gives a sign in readiness for accepting a challenge. The challenger $\mathcal{B}$ encrypts the message array with the subset of attributes $\mathcal{S}^*$. Let $l = [\mathcal{S}^*]$, then the challenger $\mathcal{B}$ returns a ciphertext $C^* = (c_1^*, c_2^*)$ constructed form the LWE samples, as follow:
  1. Select a random vector $\mathbf{a} \xleftarrow{R} \mathbb{Z}_q^n$, construct the matrix $\mathbf{F} = [\mathbf{A}_0||\mathbf{A}_{\rho(1)}, \mathbf{A}_0||\mathbf{A}_{\rho(2)}, \cdots, \mathbf{A}_0||\mathbf{A}_{\rho(l)}]$.
  2. For any one bit of the message array $m_i^* \in \{0, 1\}$, let $c_1^* = v_0 + \lfloor \frac{q}{2} \rfloor \cdot m_i^* = \mathbf{a}^T W_0 + \chi_1 + \lfloor \frac{q}{2} \rfloor \cdot m_i^*$, let $c_2^* = \mathbf{a}^T \mathbf{F} + \chi_2 = \left[v_1^1, \cdots, v_1^m, \cdots, v_l^1, \cdots, v_l^{2m}\right]$
  Lastly, the challenger$\mathcal{B}$ returns the ciphertext index $C^* = (c_1^*, c_2^*)^t$ to the adversary $\mathcal{A}$.
- **Phase 2.** The adversary $\mathcal{A}$ can continue to query the private key extraction as in phase 1 but the queried $\mathcal{S}^*$ does not satisfy the policy $(\mathbb{M}, \rho)$.
- **Guess.** The adversary $\mathcal{A}$ is required to output a guess of m_i^*. If the adversary $\mathcal{A}$ succeeds in guessing m_i^* with probability at least ε, then the challenger $\mathcal{B}$ will correctly guess the LWE oracle with probability at least $\varepsilon \big/ (l + 1)$.

B. KEYWORD-INDEX SECURITY

Next, we show that the selective keywords index security of new scheme from lattice on the selective-security model by **definition** 2.

Theorem 2: For the parameters described above, the PEKS sub-scheme of new scheme is IND-PEKS-sCPA secure under the $LWE_{q,\chi}$ assumption.

Proof: If there is an adversary $\mathcal{A}$ is able to break this scheme with advantage $\varepsilon > 0$, we now construct a challenger $\mathcal{C}$ that has with advantage $\varepsilon \big/ N_{H^*}$ to solve decisional $LWE_{q,\chi}$ problem. Let N_{H^*} be queriers times of H^*. There duction proceeds as follows.

- **Setup.** The challenger $\mathcal{C}$ prepares the system the public parameters that is same as above, returns the public parameter $Pk = \{\mathbf{A}_0, \{\mathbf{A}_{u_k}\}_{u_k \in \mathbb{U}}, \mathbf{u}, H^*\}$ to the adversary $\mathcal{A}$.
- **Phase 1:** The adversary $\mathcal{A}$ adaptively make the following queries.

  Hash queries. The adversary $\mathcal{A}$ make the j-th hash query with keyword kw_j.
  1. The challenger $\mathcal{C}$ invokes the algorithm $TrapGen(n, m, q, \sigma)$ to generate a basis $\mathbf{B}_{kw_j}$ for $\Lambda_q^{\perp}(\mathbf{A}_0||\mathbf{A}_{kw_j})$, in which $\mathbf{A}_{kw_j} = H^*(kw_j)$ is selected uniformly over $\mathbb{Z}_q^{n \times m}$.
  2. Then, the challenger $\mathcal{C}$ returns $\mathbf{A}_{kw_j}$to adversary $\mathcal{A}$, and stores the tuple $\langle kw_j, \mathbf{A}_{kw_j}, \mathbf{B}_{kw_j}\rangle$ in the list $\mathcal{H}$.

  Extract queries. The adversary $\mathcal{A}$ makes the trapdoor generation query on the keyword set W^*. It is assumed that adversary $\mathcal{A}$ has already made a hash query on the keyword kw_j.
  1. If the tuple $\langle kw_j, \mathbf{A}_{kw_j}, \mathbf{B}_{kw_j}\rangle$ be included in the list $\mathcal{H}$, the challenger $\mathcal{C}$ creates a properly distributed basis T_{kw_j} with $\mathbf{F}_{kw_j} = [\mathbf{A}_0||\mathbf{A}_{kw_j}]$ by sampling algorithm $T_{kw_j} \leftarrow LeftSample(\mathbf{F}_{kw_j}, \mathbf{B}_{kw_j}, \mathbf{u}, \sigma)$.
  2. If the generation is successful, then the challenger $\mathcal{C}$ returns T_{kw_j}. Otherwise, challenger $\mathcal{C}$ aborts.
- **Challenge.** The adversary $\mathcal{A}$ gives a sign in readiness for accepting a challenge. The adversary $\mathcal{A}$ sends the challenger $\mathcal{C}$ two keywords kw_0^*, kw_1^* on which he wants to be challenged. The restriction on then choice of kw_0^*, kw_1^* is that the trapdoors of Tkw_0^*, Tkw_1^* has not been queried in phase 1.

TABLE 2. Scheme comparison.

Scheme	Searchable Encryption	Attribute-based Encryption	Difficulty Hypothesis
literature [21]	PEKS	RABE (revocation ABE)	DBDH
literature [29]	PEKS	RABE (revocation ABE)	q-BDHE
literature [30]	TKS(temporary SE)	KP-ABE	DBDH
literature [31]	PEKS-SM(Shared Multi-owner PEKS)	CP-ABE	DBDH
literature [37]	DABSE	KP-MABE(Multi-Authority ABE)	LWE-Lattices
literature [38]	FPEKS(fuzzy PEKS)	IBE	LWE-Lattices
Literature [39]	PEKS	IBE	NTRU-Lattices
Ours	PEKS	KP-ABE	LWE-Lattices

1. The challenger $\mathcal{C}$ chooses a random $b = \{0, 1\}$, and constructs the index $I^*_{kw_b}$ of the challenge keyword kw^*_b. Then $\mathcal{C}$ responds with a ciphertext index $I^*_{kw_b} = (I^*_1, I^*_2)$ assembled form the LWE instance.
2. Randomly select a vector $\mathbf{b} \xleftarrow{\$} \mathbb{Z}^n_q$, construct the matrix $\mathbf{F}_{kw_j} = [\mathbf{A}_0||\mathbf{A}^*_{kw_b}] \in Z^{n\times 2m}_q$
   Set $I^*_1 = \mathbf{b}^T\mathbf{u} + \chi_3$
   Set $I^*_2 = \mathbf{b}^T\mathbf{F}_{kw_b} + \chi_4 = \left[v^1_1, \cdots, v^m_1, \cdots, v^1_t, \cdots, v^{2m}_t\right]$
   Return the ciphertext index $I^*_{kw_b} = (I^*_1, I^*_2)$ to the adversary $\mathcal{A}$.

- **Phase 2.** The adversary $\mathcal{A}$ is allowed continuing to make further trapdoor queries as in phase 1 with the constrain that the queried keyword $kw_j \neq kw^*_0, kw^*_1$.
- **Guess.** Lastly, the adversary $\mathcal{A}$ is required to output a guess $b' = \{0, 1\}$. If the adversary $\mathcal{A}$ succeeds in guessing b' with probability at least $\varepsilon > 0$, then the challenger $\mathcal{C}$ will correctly guess the LWE oracle with probability at least $\varepsilon \big/ N_{H^*}$.

VIII. SCHEME COMPARISON

In this subsection, we will compare our scheme with some existing ones recently proposed in literatures [21], [29]–[31], [37]–[39].

In Table 2, three aspects are compared which are the model of searchable encryption, the type of attribute-based encryption, and difficulty hypothesis problem.

The difficulty hypothesis depends on the hardness of decisional bilinear Diffie Hellman (DBDH) in literatures [21], [29]–[31]. Literature [21] give a verifiable attribute-based keyword search scheme based on DBDH hypothesis. Literature [29] supports revocable attribute encryption combining with searchable encryption. Literature [30] gives a KP-ABE temporary keyword search scheme. Literature [31] supports a shared multi-owner setting (ABKS-SM) system.

The difficulty hypothesis depends on the hardness of lattice-based problem in literatures [37]–[39]. Literature [37] introduces a key-policy multi-authority attribute-based encryption (KP-MABE) and a key-policy distributed attribute-based searchable encryption (DABSE) on lattices. Literature [38] gives a fuzzy information retrieval scheme with identity-based control. Literatures [39] fully implements a NTRU-PEKS scheme.

Two schemes respectively in literature [37] and [38] rely on identity-based policy control rather than attribute-base control. Different from literature [39], our scheme is a combination PEKS and KP-ABE on lattices, but in literature [39], DABSE is an independent scheme rather than combining with KP-MABE.

IX. CONCLUSION

In this paper, we construct a new keyword-searchable ABE scheme upon the hardness of lattice-based problem hypothesis. New scheme has the property of quantum attack resistance and can support flexible private key management. It can avoid the risk of leaks that caused by multiple users share in a cloud storage environment.

The system model and application scene are introduced in cloud environment, new scheme can be applied to personal health network, vehicle network, wireless sensor network and other emerging network application services.

APPENDIX

A. LATTICES

Definition 1 (q-ary Lattice [26]): Given positive integers n, m and a prime q, a matrix $\mathbf{A} \in \mathbb{Z}^{n\times m}_q$ and a vector $\mathbf{u} \in \mathbb{Z}^n_q$, define:

$$\Lambda^{\perp}_q(\mathbf{A}) = \{\mathbf{e} \in \mathbb{Z}^m \quad s.t.\ \mathbf{Ae} = 0 \,(\text{mod } q)\}$$
$$\Lambda^{\mathbf{u}}_q(\mathbf{A}) = \{\mathbf{e} \in \mathbb{Z}^m \quad s.t.\ \boldsymbol{Ae} = \mathbf{u} \,(\text{mod } q)\}$$

observe that $\Lambda^{\mathbf{u}}_q(\mathbf{A})$ is a shift of $\Lambda^{\perp}_q(\mathbf{A})$. That is $\Lambda^{\mathbf{u}}_q(\mathbf{A}) = \Lambda^{\perp}_q(\mathbf{A}) + v$ for any v satisfying $Av = \mathbf{u} \bmod q$.

B. THE TRAPGEN ALGORITHM

Ajtai [34] showed how to sample an essentially uniform matrix $\mathbf{A} \in \mathbb{Z}^{n\times m}_q$ with an associated full-rank set $T_{\mathbf{A}} \subset \Lambda(\mathbf{A})$ of low-norm vectors. We will use an improved version of Ajtai's basis sampling algorithm due to Alwen and Peikert [41].

TrapGen(n, m, q, σ) *([41, Theorem 3.2]):* For any prime $q \geq 2$ and $m \geq 5n \log q$. There exists a probabilistic polynomial-time algorithm that outputs a pair ($\mathbf{A} \in \mathbb{Z}^{n\times m}_q$, $\mathbf{B} \in \mathbb{Z}^{m\times m}_q$), such that $\mathbf{A}$ is statistically close to uniform and $\mathbf{B}$ is a basis for $\Lambda^T_q(\mathbf{A})$ with length $L = ||\tilde{\mathbf{B}}|| \leq m \cdot \omega\left(\sqrt{\log m}\right)$ with all but $n^{-\omega(1)}$ probability.

C. DISCRETE GAUSSIANS AND SAMPLING ALGORITHM

The m-dimensional continuous Gaussian function centered at $c \in \mathbb{R}^m$ with paraments $\sigma > 0$ is

$$\rho_{\sigma,c}(\mathbf{x}) = \exp\left(-\pi \frac{\|\mathbf{x} - c\|^2}{\sigma^2}\right) \quad \text{for any } \mathbf{x} \in \mathbb{R}^m.$$

The discrete Gaussian distribution is defined as follows.

Definition 2 (Discrete Gaussian): Let $c \in \mathbb{R}^m$ and any positive parameter, define:

$$\forall \mathbf{x} \in \Lambda, D_{\Lambda,\sigma,c}(\mathbf{y}) = \frac{\rho_{\sigma,c}(\mathbf{x})}{\rho_{\sigma,c}(\Lambda)},$$

where $\rho_{\sigma,c}(\Lambda) = \sum \mathbf{x} \in \Lambda \rho_{\sigma,c}(\mathbf{x})$: The discrete integral of $\rho_{\sigma,c}$ over the lattice Λ, $D_{\Lambda,\sigma,c}$ be the discrete Gaussian distribution over Λ with center c and parameter σ.

Gentry *et al.* [36] construct the following algorithm for sampling from the discrete Gaussian $D_{\Lambda,\sigma,c}$, given a basis $\mathbf{B}$ for the lattice Λ with $\sigma \geq \left\|\tilde{\mathbf{B}}\right\| \cdot \omega\left(\sqrt{\log m}\right)$:

SampleGaussian $(\Lambda, \boldsymbol{B}, \sigma, c)$ *[36]:* On input lattice Λ, a basis $\mathbf{B}$ for Λ, a positive Gaussian parameter σ, and a center vector $c \in \mathbb{R}^m$, it outputs a fresh random vector $\mathbf{x} \in \Lambda$ drawn from a distribution statistically close to $D_{\Lambda,\sigma,c}$.

SamplePre $(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{u}, \sigma)$ *[36]:* Let n, q, m be positive integers with $q \geq 2$, $m \geq 2n \log q$. On input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ with 'short' trapdoor basis $\mathbf{B}$ for $\Lambda_q^T(\mathbf{A})$, a target image $\mathbf{u} \in \mathbb{Z}_q^n$ and a Gaussian parameter $\sigma \geq \left\|\tilde{\mathbf{B}}\right\| \cdot \omega\left(\sqrt{\log m}\right)$, outputs a sample $\mathbf{e} \in \mathbb{Z}^m$ from a distribution that is within negligible statistical distance $D_{\Lambda_q^{\mathbf{u}}(\mathbf{A}),\sigma}$.

LeftSample $(\boldsymbol{A}, \boldsymbol{A}_1, T\boldsymbol{A}, \boldsymbol{u}, \sigma)$ *[45, Theorem 3]:* Inputs a rank n matrix $\mathbf{A}$ in $\mathbb{Z}_q^{n \times m}$ and a matrix $\mathbf{A}_1$ in $\mathbb{Z}_q^{n \times m_1}$, a short basis $T\mathbf{A}$ of $\Lambda_q^T(\mathbf{A})$ and a vector $\mathbf{u} \in \mathbb{Z}_q^n$, a gaussian parameter $\sigma \geq ||\tilde{T}\mathbf{A}|| \cdot \omega\left(\sqrt{\log(m+m_1)}\right)$. Let $\mathbf{F}_1 := (\mathbf{A}|\mathbf{A}_1)$. Outputs a vector $\mathbf{e} \in \mathbb{Z}^{m+m_1}$ sampled from a distribution statistically close to $D_{\Lambda_q^{\mathbf{u}}(\mathbf{F}_1),\sigma}$. In particular, $\mathbf{e} \in \Lambda_q^{\mathbf{u}}(\mathbf{F}_1)$.

D. LATTICE BASIS DELEGATE ALGORITHM

Cash *et al.* [44] described how an arborist may extend its control of a lattice to an arbitrary higher-dimensional extension, without any loss of quality in the resulting basis.

ExtBasis $\left(S, A' = A \left\|\overline{A}\right.\right)$ *[44, Lemma 3.2]:* Given an arbitrary matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ whose columns generate the entire group $\mathbb{Z}_q^n$, an arbitrary basis $\mathbf{S} \in \mathbb{Z}^{m \times m}$ of $\Lambda^{\perp}(\mathbf{A})$, and an arbitrary $\bar{A} \in \mathbb{Z}_q^{n \times \overline{m}}$. Outputs a basis $\mathbf{S}'$ of $\Lambda^{\perp}(\mathbf{A}') \subseteq \mathbb{Z}^{m+\overline{m}}$, such that $\left\|\tilde{\mathbf{S}}\right\| = \left\|\tilde{\mathbf{S}}_0\right\|$. Moreover, the same holds even for any given permutation of the columns of $\mathbf{A}'$ (e.g., if columns of $\bar{A}$ are both appended and extended to$\mathbf{A}$).

RandBasis $(\boldsymbol{S}, r)$ *[44, Lemma 3.3]:* Input m-dimension lattice $\Lambda^{\perp}(\mathbf{A})$ with a basis $\mathbf{S} \in \mathbb{Z}^{m \times m}$, and a parameter $r \geq \left\|\tilde{\mathbf{S}}\right\| \cdot \omega\left(\sqrt{\log n}\right)$. With overwhelming probability, output a short basis $\mathbf{S}'$ of lattice$\Lambda^{\perp}(\mathbf{A})$, such that $\left\|\mathbf{S}'\right\| \leq r \cdot \sqrt{m}$. Moreover, for any two bases $\mathbf{S}_0, \mathbf{S}_1$ of the same lattice and any $r \geq \max\left\{\left\|\tilde{\mathbf{S}}_0\right\|, \left\|\tilde{\mathbf{S}}_1\right\|\right\} \cdot \omega\left(\sqrt{\log n}\right)$, the outputs of *RandBasis* $(\mathbf{S}_0, r)$ and *RandBasis* $(\mathbf{S}_1, r)$ are within $negl(n)$ statistical distance.

E. TWO LEMMAS TO BOUND NORMS

Next two lemmas will need to show that can guarantee decryption works correctly.

Lemma 1 [40, Lemma 4.4]: For any m-dimension lattice Λ, vector $c \in R^m$, and real $\varepsilon \in (0, 1)$, $s > \eta_{\varepsilon}(\Lambda)$, we have

$$\Pr_{\mathbf{x} \sim D_{\Lambda,s,c}}\left[\|\mathbf{x} - c\| > s\sqrt{n}\right] \leq \frac{1+\varepsilon}{1-\varepsilon} \cdot 2^{-m}$$

The lemma sates that for large enough s, almost the elements chosen from $D_{\Delta,s,c}$ are close to c.

Lemma 2 [36, Lemma 8.2]: Let $\mathbf{e}$ be some vector in $\mathbb{Z}^m$ and let $\mathbf{y} \leftarrow \bar{\Psi}_{\alpha}^m$. Then the quantity $\left|\mathbf{e}^T\mathbf{y}\right|$ treated as an integer in $[0, q-1]$ satisfies

$$\left|e^T\mathbf{y}\right| \leq \|\mathbf{e}\|\left(q\alpha \cdot \omega\left(\sqrt{\log m}\right) + \frac{\sqrt{m}}{2}\right)$$

with all but negligible probability in m. In particularly, if $x \leftarrow \bar{\Psi}_{\alpha}$ is treated as an integer in $[0, q-1]$ then $|x| \leq q\alpha \cdot \omega\left(\sqrt{\log m}\right) + 1/2$ with all but negligible probability in m.

F. LEARNING WITH ERROR

Definition 3 (Learning With Error [35]): Consider a prime q, a positive integer n, and a distribution $\chi \in \mathbb{Z}_q$, all public. An $\left(\mathbb{Z}_q, n, \chi\right)$-LWE problem instance consists of access to an unspecified challenge oracle $\mathcal{O}$, being, either, a noisy pseudo-random sampler $\mathcal{O}_{\mathbf{x}}$ carrying some constant random secret key $\mathbf{x} \in \mathbb{Z}_q^n$, or, a truly sampler $\mathcal{O}_{\$}$, whose behaviors are respectively as follows:

$\mathcal{O}_{\mathbf{x}}$: Output noisy pseudo-random samples of the form $(\mathbf{w}_i, v_i) = \left(\mathbf{w}_i, \mathbf{w}_i^T\mathbf{x} + \chi_i\right) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, where $\mathbf{x} \in \mathbb{Z}_q^n$ is a uniformly distributed persistent secret key that is invariant across invocations, $\chi_i \in \mathbb{Z}_q$ is a freshly generated ephemeral additive noise component with distribute χ, and $\mathbf{w}_i \in \mathbb{Z}_q^n$ is a fresh uniformly distributed vector revealed as part of the output.

$\mathcal{O}_{\$}$: Output truly random samples $(\mathbf{w}_i, v_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, drawn in independently uniformly at random in the entire domain $\mathbb{Z}_q^n \times \mathbb{Z}_q$.

The $\left(\mathbb{Z}_q, n, \chi\right)$-LWE problem statement, or LWE for short, allows an unspecified number of queries to be made to the challenge oracle $\mathcal{O}$, with no stated prior bound. We say that an algorithm A decides the $\left(\mathbb{Z}_q, n, \chi\right)$-LWE problem if $\left|\Pr[A^{\mathcal{O}_{\mathbf{x}}} = 1] - \Pr[A^{\mathcal{O}_{\$}} = 1]\right|$ is non-negligible for a random $\mathbf{x} \in \mathbb{Z}_q^n$.

The confidence in the hardness of the LWE problem stems in part of a result of Regev [35], which shows that the for certain noise distributions χ, the LWE problem is as hard as the worst-case SIVP and GapSVP under a quantum reduction. A classical reduction with related parameters was later obtained by Peikert [37], [38].

Proposition 2 [35], [43]: For an $\alpha \in (0, 1)$ and a prime $q > 2\sqrt{n}/\alpha$, let $\bar{\Psi}_{\alpha}$ denote the discrete distribution over $\mathbb{Z}_q$ of the random variable $\lfloor qX + 1/2 \rfloor \bmod q$ where the random variable X is a normal random variable with mean 0 and standard deviation $\alpha / \sqrt{2\pi}$. Then, if there exists an efficient, possibly quantum algorithm for deciding the $\left(\mathbb{Z}_q, n, \chi\right)$-LWE, there exists a quantum poly-time algorithm for approximating the SIVP and Gap-SVP problems, to within $\tilde{O}(n/\alpha)$ factors in the l_2 norm, in the worst case.

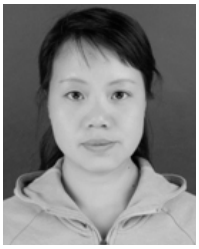
ACKNOWLEDGMENT

Thanks also go to the anonymous reviewers for their useful comments.

REFERENCES

[1] O. Goldreich and R. Ostrovsky, "Software protection and simulation on oblivious RAMs," *J. ACM*, vol. 43, no. 3, pp. 431–473, 1996.
[2] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proc. IEEE Secur. Privacy Symp.*, Berkeley, CA, USA, May 2000, pp. 44–55.
[3] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proc. EUROCRYPT*. Berlin, Germany: Springer-Verlag, 2004, pp. 506–522.
[4] B. R. Waters, D. Balfanz, and G. Durfee, D. K. Smetters, "Building an encrypted and searchable audit log," in *Proc. NDSS*. Berlin, Germany: Springer, 2004, pp. 5–6.
[5] D. J. Park, K. Kim, and P. J. Lee, "Public key encryption with conjunctive field keyword search," in *Proc. Inf. Secur. Appl.*, 2005, pp. 73–86.
[6] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: Improved definitions and efficient constructions," *J. Comput. Secur.*, vol. 19, no. 5, pp. 895–934, Jan. 2011.
[7] S. Kamara, C. Papamanthou, and T. Roeder, "Dynamic searchable symmetric encryption," in *Proc. ACM Conf. Comput. Commun. Secur.*, 2012, pp. 965–976.
[8] P. Wang, H. Wang, and J. Pieprzyk, "Threshold privacy preserving keyword searches," in *Proc. Int. Conf. Current Trends Theory Pract. Comput. Sci.* Berlin, Germany: Springer, 2008, pp. 646–658.
[9] M. Nishioka, "Perfect keyword privacy in PEKS systems," in *Proc. Int. Conf. Provable Secur.* Berlin, Germany: Springer, 2012, pp. 175–192.
[10] A. Siad, "A new approach for private searches on public-key encrypted data," in *Proc. 13th Int. Conf. Commun. Multimedia Secur. (CMS)*, Canterbury, U.K., Sep. 2012, pp. 160–173.
[11] H. S. Rhee and H. L. Dong, "Keyword updatable PEKS," in *Proc. Int. Workshop Inf. Secur. Appl. WISA*, 2015, pp. 96–109.
[12] Y. Chen, J. Zhang, D. Lin, and Z. Zhang, "Generic constructions of integrated PKE and PEKS," *Des. Codes Cryptogr.*, vol. 78, no. 2, pp. 493–526, Feb. 2016.
[13] T. Suzuki, K. Emura, and T. Ohigashi, "A generic construction of integrated secure-channel free PEKS and PKE," in *Proc. Int. Conf. Inf. Secur. Pract. Exper.* Cham, Switzerland: Springer, 2018, pp. 69–86.
[14] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2005, pp. 457–473.
[15] V. Goyal, O. Pandey, and A. Sahai, B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. 13th ACM Conf. Comput. Commun. Secur.*, 2006, pp. 89–98.
[16] D. Boneh, A. Raghunathan, and G. Segev, "Function-private identity-based encryption: Hiding the function in functional encryption," in *Advances in Cryptology* (Lecture Notes in Computer Science), vol. 8043, R. Canetti and J. A. Garay, Eds. Berlin, Germany: Springer, 2013, pp. 461–478.
[17] D. Khader, "Attribute based search in encrypted data: ABSE," in *Proc. ACM Workshop Inf. Sharing Collaborative Secur.*, 2014, pp. 31–40.
[18] D. Khader, "Introduction to attribute based searchable encryption," in *Proc. Int. Conf. Commun. Multimedia Secur. (IFIP)*. Berlin, Germany: Springer, 2014, pp. 131–135.
[19] S. Li and M. Xu, "Attribute-based public encryption with keyword search," *Chin. J. Comput.*, vol. 37, no. 5, pp. 1017–1024, Jun. 2014.
[20] F. Han, J. Qin, H. Zhao, and J. Hu, "A general transformation from KP-ABE to searchable encryption," *Future Gener. Comput. Syst.*, vol. 30, pp. 107–115, Jan. 2014.
[21] W. Sun, S. Yu, W. Lou, Y. T. Hou, and H. Li, "Protecting your right: Verifiable attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 4, pp. 1187–1198, Apr. 2016.
[22] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. IT-22, no. 6, pp. 644–654, Nov. 1976.
[23] P. W. Shor, "Polynomial time algorithms for discrete logarithms and factoring on a quantum computer," in *Proc. 1st Int. Symp. Algorithmic Number Theory (ANTS-I)*. Ithaca, NY, USA: Springer-Verlag, 1994, p. 289.
[24] D. J. Bernstein, "Post-quantum cryptography," in *Proc. 1st Int. Workshop PQCrypto*, Leuven, The Netherland, May 2006, pp. 1–127.
[25] L. Chen and S. Jordan, "Report on post-quantum cryptography," U.S. Dept. Commerce, Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep., 2016.
[26] G. Alagic, J. Alperin-Sheriff, D. Apon, D. Cooper, Q. Dang, Y.-K. Liu, C. Miller, D. Moody, R. Peralta, R. Perlner, A. Robinson, and D. Smith-Tone, "Status report on the first round of the NIST post-quantum cryptography standardization process," NIST, Gaithersburg, MD, USA, Tech. Rep. NISTIR 8240, Jan. 2019.
[27] J. Hayata, M. Ishizaka, Y. Sakai, G. Hanaoka, and K. Matsuura, "Generic construction of adaptively secure anonymous key-policy attribute-based encryption from public-key searchable encryption," in *Proc. Int. Symp. Inf. Theory Appl. (ISITA)*, Oct. 2018, pp. 707–711.
[28] Z. Liu, Z. L. Jiang, and X. Wang, "Offline/online attribute-based encryption with verifiable outsourced decryption," *Concurrency Comput. Pract. Exper.*, vol. 29, Apr. 2017, Art. no. e3915.
[29] S. P. Wang, D. Zhao, and Y. Zhang, "Searchable attribute-based encryption scheme with attribute revocation in cloud storage," *PLoS ONE*, vol. 12, no. 8, 2017, Art. no. e0183459.
[30] M. H. Ameri, M. Delavar, J. Mohajeri, and M. Salmasizadeh, "A key-policy attribute-based temporary keyword search scheme for secure cloud storage," *IEEE Trans. Cloud Comput.*, to be published.
[31] Y. Miao, X. Liu, K.-K. R. Choo, R. H. Deng, J. Li, H. Li, and J. Ma, "Privacy-preserving attribute-based keyword search in shared multi-owner setting," *IEEE Trans. Dependable Secure Comput.*, to be published.
[32] M. Ajtai, "Generating hard instances of lattice problems," in *Proc. 28th Annu. ACM Symp. Theory Comput.*, 1996, pp. 99–108.
[33] M. Ajtai and C. Dwork, "A public-key cryptosystem with worst-case/average-case equivalence," in *Proc. 29th Annu. ACM Symp. Theory Comput.* New York, NY, USA, 1997, pp. 284–293.
[34] M. Ajtai, "Generating hard instances of the short basis problem," in *Proc. 26th Int. Colloquium Automata, Lang. Program. (ICALP)*, J. Wiedemann, P. van Emde Boas, and M. Nielsen, Eds. Berlin, Germany: Springer, 1999, pp. 1–9.
[35] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," in *Proc. 37th Annu. ACM Symp. Theory Comput. (STOC)*, New York, NY, USA, 2005, pp. 84–93.
[36] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *Proc. 40th Annu. ACM Symp. Theory Comput.*, New York, NY, USA, 2008, pp. 197–206.
[37] V. Kuchta and O. Markowitch, "Multi-authority distributed attribute-based encryption with application to searchable encryption on lattices," in *Proc. Int. Conf. Cryptol. Malaysia*. Cham, Switzerland: Springer, 2016, pp. 409–435.
[38] Y. Yang, X. Zheng, V. Chang, S. Ye, and C. Tang, "Lattice assumption based fuzzy information retrieval scheme support multi-user for secure multimedia cloud," *Multimedia Tools Appl.*, vol. 77, no. 8, pp. 9927–9941, 2018.
[39] B. Rouzbeh, O. M. Ozgur, and Y. A. Altay, "Lattice-based public key searchable encryption from experimental perspectives," *IEEE Trans. Dependable Secure Comput.*, to be published.
[40] D. Micciancio and O. Regev, "Worst-case to average-case reductions based on Gaussian measures," *SIAM J. Comput.*, vol. 37, no. 1, pp. 267–302, 2007.
[41] J. Alwen and C. Peikert, "Generating shorter bases for hard random lattices," in *Proc. 26th Int. Symp. Theor. Aspects Comput. Sci. (STACS)*, 2009, pp. 75–86.
[42] C. Peikert, "Public-key cryptosystems from the worst-case shortest vector problem," in *Proc. 41st Annu. ACM Symp. Theory Comput. (STOC)*, Bethesda, Maryland, 2009, pp. 333–342.
[43] C. Peikert, "An efficient and parallel Gaussian sampler for lattices," in *Proc. Annu. Cryptol. Conf.* Berlin, Germany: Springer, 2010, pp. 80–97.
[44] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert, "Bonsai trees, or how to delegate a lattice basis," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2010, pp. 523–552.
[45] S. Agrawal, D. Boneh, and X. Boyen, "Efficient lattice (H) IBE in the standard model," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2010, pp. 553–572.
[46] S. P. Wang, L. H. Liu, J. Sun, and Y. Zhang, "Multi–party concurrent signatures scheme from lattice," *Int. J. Inf. Commun. Technol.*, vol. 7, nos. 2–3, pp. 247–262, 2015.

[47] L. H. Liu, S. P. Wang, and Q. Yan, "A multi-authority key-policy ABE scheme from lattices in mobile ad hoc networks," *Ad Hoc Sensor Wireless Netw.*, vol. 37, nos. 1–4, pp. 117–143, 2017.
[48] Y. L. Zhang, S. P. Wang, and Q. Du, "Revocable identity-based encryption scheme under LWE assumption in the standard model," *IEEE Access*, vol. 6, pp. 65298–65307, 2018.
[49] A. Beimel, "Secure schemes for secret sharing and key distribution," Ph.D. dissertation, Dept. Comput. Sci., Technion-Israel Inst. Technol., Haifa, Israel, 1996.
[50] M. M. Naor Yung, "Public-key cryptosystems provably secure against chosen ciphertext attacks," in *Proc. 22nd Annu. ACM Symp. Theory Comput.*, 1990, pp. 427–437.

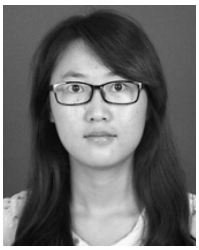


LIHUA LIU received the B.S. degree in computer engineering from Northwest University, Xi'an, China, in 1998, and the M.S. degree in computer software and theory from Shaanxi Normal University, Xi'an, in 2006. She is currently pursuing the Ph.D. degree with the Xi'an University of Technology. She is currently an Associate Professor with the Shaanxi University of Technology. Her current research interests include cryptography and information security.

SHANGPING WANG received the B.S. degree in mathematics from the Xi'an University of Technology, Xi'an, China, in 1982, the M.S. degree in applied mathematics from Xi'an Jiaotong University, Xi'an, in 1989, and the Ph.D. degree in cryptology from Xidian University, Xi'an, in 2003.

He is currently a Professor with the Xi'an University of Technology. His current research interests include cryptography and information security.

BINTAO HE received the B.S. degree from the School of Science, Northwest A&F University, Xianyang, China, in 2001, and the M.S. degree from the School of Science, Xi'an University of Technology, Xi'an, China, in 2014, where he is currently pursuing the Ph.D. degree with the School of Automation and Information Engineering. His research interests include information security and blockchain technology.



DUO ZHANG received the B.S. and M.S. degrees from the School of Science, Xi'an University of Technology, Xi'an, China, in 2014 and 2017, respectively, where she is currently pursuing the Ph.D. degree with the School of Automation and Information Engineering. Her research interests include information security and modern cryptography.