12-2017

VKSE-MO: Verifiable keyword search over encrypted data in multi-owner settings

Yinbin MIAO
Xidian University

Jianfeng MA
Xidian University

Ximeng LIU
Singapore Management University, xmliu@smu.edu.sg

Junwei ZHANG
Xidian University

Zhiquan LIU
Xidian University

Published in SCIENCE CHINA Information Sciences, 2017 April, Volume 60
http://doi.org/10.1007/s11432-016-0540-x

VKSE-MO: verifiable keyword search over encrypted data in multi-owner settings

Yinbin MIAO[1], Jianfeng MA[1*], Ximeng LIU[2], Junwei ZHANG[1] & Zhiquan LIU[1]

[1]*School of Cyber Engineering, Xidian University, Xi'an* 710071*, China;*
[2]*School of Information Systems, Singapore Management University, Singapore* 178902*, Singapore*

Received September 30, 2016; accepted November 24, 2016; published online April 24, 2017

Abstract Searchable encryption (SE) techniques allow cloud clients to easily store data and search encrypted data in a privacy-preserving manner, where most of SE schemes treat the cloud server as honest-but-curious. However, in practice, the cloud server is a semi-honest-but-curious third-party, which only executes a fraction of search operations and returns a fraction of false search results to save its computational and bandwidth resources. Thus, it is important to provide a results verification method to guarantee the correctness of the search results. Existing SE schemes allow multiple data owners to upload different records to the cloud server, but these schemes have very high computational and storage overheads when applied in a different but more practical setting where each record is co-owned by multiple data owners. To address this problem, we develop a verifiable keyword search over encrypted data in multi-owner settings (VKSE-MO) scheme by exploiting the multisignatures technique. Thus, our scheme only requires a single index for each record and data users are assured of the correctness of the search results in challenging settings. Our formal security analysis proved that the VKSE-MO scheme is secure against a chosen-keyword attack under a random oracle model. In addition, our empirical study using a real-world dataset demonstrated the efficiency and feasibility of the proposed scheme in practice.

Keywords chosen-keyword attack, efficiency and feasibility, multi-owner settings, result verification, searchable encryption

Citation Miao Y B, Ma J F, Liu X M, et al. VKSE-MO: verifiable keyword search over encrypted data in multi-owner settings. Sci China Inf Sci, 2017, 60(12): 122105, doi: 10.1007/s11432-016-0540-x

1 Introduction

In the new cloud computing [1] epoch, many cloud clients are attracted to outsourcing data to cloud service provider (CSP) for storage because of its useful features, such as location-independent resource pooling and ubiquitous network access. Data outsourcing [2] can relieve data owners (DOs) of the heavy burden of local data management and maintenance, especially for the storage resource-limited entities (e.g., mobile terminal devices and sensor nodes). However, data outsourcing actually deprives the DO of ultimate control over the encrypted data, which may lead to a range of internal and external security breaches affecting sensitive data. For example, a malicious CSP can forge false search results or an

* Corresponding author (email: jfma@mail.xidian.edu.cn)

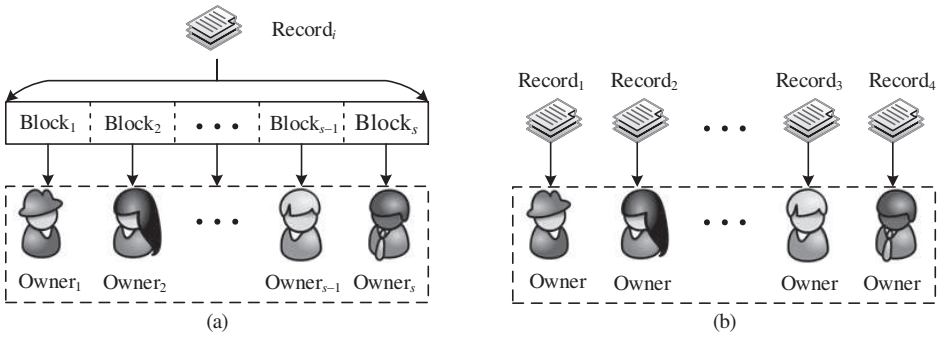

Figure 1 Settings considered by various schemes. (a) Settings in our scheme; (b) settings in previous SE schemes.

internal adversary can access sensitive information. Hence, data security and privacy concerns remain significant barriers to the adoption of cloud storage.

Encryption is considered a straightforward and efficient method for ensuring data security, but it makes it difficult to retrieve encrypted data from a remote cloud server. A naive solution is to download the entire encrypted data and decrypt it locally, but this wastes computational and bandwidth resources. The searchable encryption (SE) technique [3–8] allows data users (DUs) to search securely and selectively retrieve files of interest according to user-specified keywords, thereby addressing the conflicts between data privacy and usability, and SE has been studied widely in academic and industrial fields. However, in practice, CSP is a semi-honest-but-curious third-party that selfishly conducts only a fraction of the search operations and returns a fraction of false search results to save computational and bandwidth resources. Thus, the security of data may be at risk due to the various motivations of CSPs (e.g., discarding rarely accessed data for monetary reasons and hiding data loss incident to maintain the reputation).

Thus, practical SE schemes should be equipped with a results verification mechanism [9–14] to guarantee the correctness of the search results. Furthermore, the results verification overheads should be minimal and affordable in a broad range of practical applications, especially for resource-limited entities. However, none of the aforementioned SE schemes can be applied in a different but more practical multi-owner scenario where each record is co-owned by a fixed number of DOs (also called as data-subjects) rather than there exist multiple DOs contributing different records to CSP, as shown in Figure 1. To the best of our knowledge, the applications of multiple DOs are quite common in the context of cloud storage. For example, the contents of each personal health record (PHR) may be controlled by the patient and various medical staffs. Subsequently, an attending physician may need to access both the patient's name and medical records. However, the previously proposed multi-owner schemes [15–17] need to build multiple indexes for each block of a specific patient's PHR, which inevitably incurs high computational and space overheads. It should be noted that each block owned by different data-owners might not be the same size, and our proposed scheme allows DUs to access the required information without submitting multiple trapdoors for each PHR entry. The challenging multi-owner setting considered for our proposed scheme may be excessively rigid, but our scheme can still be applied in the single data-owner settings without incurring a high computational burden.

To address the problems described above, we propose a secure and efficient cryptographic primitive called the Verifiable Keyword Search over Encrypted data in the Multi-Owner settings (VKSE-MO) scheme. Using the multisignatures technique [18] and data auditing technique [19, 20], our scheme can guarantee data security in challenging settings. Based on a formal security analysis, we prove that our scheme is secure against chosen-keyword attack in a random oracle model. We also conducted empirical experiments using a real-world dataset, which demonstrated the feasibility and efficiency of our proposed scheme in practice. In particular, our main contributions can be summarized as follows.

(1) Supporting challenging multi-owner settings. In contrast to previous SE schemes, our scheme can be applied in a challenging multi-owner setting where each record is co-owned by multiple DOs.

(2) Supporting result verification. The results verification mechanism allows our scheme to precisely guarantee the correctness of the search results and to restore the confidence of cloud clients in the overall

search process.

(3) More secure and practical. First, our secure analysis formally proves that our scheme can resist chosen-keyword attack in a random oracle model. Second, our experimental results obtained using a real-world dataset demonstrate its efficiency and feasibility in practice.

The remainder of this paper is organized as follows. Section 2 briefly reviews the previous research related to our scheme. In Section 3, we give the preliminaries associated with our scheme. The system and threat models, design goals, scheme definition and security model are presented in Section 4. In Section 5, we describe the detailed construction of our scheme. In Section 6, we present the correctness, security and performance analyses. Finally, we give our conclusion in Section 7.

2 Related work

SE can achieve data confidentiality and usability, and thus it is very popular with many cloud clients and it has gradually become a fundamental solution for addressing the problem of secure search over encrypted data. According to different cryptography primitives, existing SE schemes can be roughly divided into two categories, e.g., public key SE (PKSE) [21–23] and symmetric SE (SSE) [24–26]. Song et al. [6] proposed the first SSE scheme, where the search time increases with the size of the data collection. Boneh et al. [7] constructed a PKSE scheme that providers a stronger security model. Many other SE schemes with different functionalities have also been proposed, such as conjunctive keyword search [23,27,28] and ranked keyword search [24,25,29]. Moreover, in these schemes, the CSP is assumed an honest-but-curious entity that follows the established protocols and that aims to find valuable information.

However, this assumption is usually incorrect in practice because a semi-honest-but-curious CSP may intentionally return incorrect search results for various reasons. Thus, the data security can be compromised and incorrect search results also lead to a poor user search experience. Therefore, a results verification mechanism should be provided to guarantee the accuracy of the returned results. Hence, Chai et al. [9] proposed the first verifiable SSE scheme to verify the correctness of the search results. In addition, the verifiable scheme developed by Zheng et al. [10] using attribute-based encryption can efficiently grant search capabilities to DUs. However, these schemes only support single keyword search and they cannot be applied to dynamic databases. Furthermore, Sun et al. [11] proposed a verifiable conjunctive keyword search method for large dynamic encrypted cloud data. Unfortunately, these schemes cannot be implemented in challenging multi-owner scenarios where each record is co-owned by multiple DOs.

Previous SE schemes can be applied in this challenging setting if each block of the record is viewed as an independent file, but this will inevitably yield multiple indexes where the computational and space overheads are greatly increased. In the present study, in contrast to previous schemes [15,16], we consider a multi-owner setting where each record is co-owned by multiple DOs. Layouni et al. [30] considered a scenario where each message is owned by multiple DOs, and Wang et al. [20] proposed an efficient public audit verification on the integrity of a multi-owner data scheme. To significantly reduce the time and space overhead, we extend this type of multi-owner scenario to SE schemes by building a single index for each whole record. In contrast to previous SE schemes, our proposed scheme can support both secure search based on keywords and search results verification in challenging multi-owner settings, as shown in Table 1. Our proposed scheme can significantly reduce the high computational burden imposed by building multiple indexes in challenging multi-owner settings, as well as improving the user search experience by ensuring the correctness of the search results.

3 Preliminaries

Let G_1, G_2 be two multiplicative cyclic groups of prime order p, g be a generator of group G_1, and e be the bilinear mapping $G_1 \times G_1 \to G_2$. Given a set X, the symbol $x \in_R X$ is defined as choosing an element x uniformly at random from the set X, and $[1, n]$ is denoted as an integer set $\{1, 2, \ldots, n\}$.

Table 1 Comparison of the functionalities of various schemes

Scheme	Keyword search	Result verification	Multi-owner setting
VABKS [10]	Yes	Yes	No
VCKS [11]	Yes	Yes	No
Re-dtPECK [23]	Yes	No	No
ABKS-UR [15]	Yes	Yes	No
VKSE-MO	Yes	Yes	Yes

Let an integer k be the security level and $(\mathcal{F}, \mathcal{W})$ be the file and keyword space, respectively. Next, we represent a group of cryptographic concepts used in our scheme as follows.

Definition 1 (Computational Diffie-Hellman (CDH) assumption)**.** Let G_1 be a multiplicative cyclic group of order p and g be a generator of G_1. Given random elements $g, g^a, g^b \in_R G_1$, $a, b \in_R Z_p^*$, it is computationally infeasible to compute $g^{ab} \in_R G_1$ for any probabilistic time adversary $\mathcal{A}$ with a negligible advantage ϵ, where the advantage of $\mathcal{A}$ is defined as $\Pr[\mathcal{A}_{\mathrm{CDH}}(g, g^a, g^b) = g^{ab}] \leqslant \epsilon$.

Definition 2 (Decision bilinear Diffie-Hellman (DBDH) assumption)**.** Given the bilinear map parameters (G_1, G_2, p, g, e) and elements $a, b, c, z \in_R Z_p^*$, the DBDH assumption states that no probabilistic time adversary $\mathcal{A}$ can distinguish the tuple $(g, g^a, g^b, g^c, e(g,g)^{abc})$ from the tuple $(g, g^a, g^b, g^c, e(g,g)^z)$ with a non-negligible advantage, where the advantage of adversary $\mathcal{A}$ is defined as $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{DBDH}}(1^k) = |\Pr[\mathcal{A}(g, g^a, g^b, g^c, e(g,g)^{abc}) = 1] - \Pr[\mathcal{A}(g, g^a, g^b, g^c, e(g,g)^z) = 1]|$. Then we say that the DBDH assumption relative to the generator G_1 holds if the advantage $\mathrm{Adv}_{\mathcal{A}}^{\mathrm{DBDH}}(1^k)$ is negligible for all probabilistic time adversaries.

Definition 3 (Truncated q-augmented bilinear Diffie-Hellman exponent (q-ABDHE) assumption)**.** Given the bilinear map parameters (G_1, G_2, p, g, e), a polynomial $q = q(k)$, and elements $a', b', c' \in_R Z_p^*$, the truncated q-ABDHE assumption states that no probabilistic time adversary $\mathcal{A}$ can distinguish the tuple $(g, g^{a'}, \ldots, g^{a'^q}, g^{b'}, g^{b'a'^{q+2}}, e(g,g)^{b'a'^{q+1}})$ from tuple $(g, g^{a'}, \ldots, g^{a'^q}, g^{b'}, g^{b'a'^{q+2}}, e(g,g)^{c'})$, where the advantage of adversary $\mathcal{A}$ is set as $\mathrm{Adv}_{\mathcal{A}}^{q-\mathrm{ABDHE}}(1^k) = |\Pr[\mathcal{A}(g, g^{a'}, \ldots, g^{a'^q}, g^{b'}, g^{b'a'^{q+2}}, e(g,g)^{b'a'^{q+1}}) = 1] - \Pr[\mathcal{A}(g, g^{a'}, \ldots, g^{a'^q}, g^{b'}, g^{b'a'^{q+2}}, e(g,g)^{c'}) = 1]|$ and is non-negligible. Then, the truncated q-ABDHE assumption relative to the generator G_1 holds if the advantage $\mathrm{Adv}_{\mathcal{A}}^{q-\mathrm{ABDHE}}(1^k)$ is negligible for all probabilistic time adversaries.

4 Problem formulation

To better understand our scheme, the terms search token (trapdoor) and file (record) are both used interchangeably throughout this study.

4.1 System and threat models

In this study, our scheme considers a cloud data storage system that mainly involves four entities, i.e., a CSP, multiple DOs, DU and private audit server (PAS), as shown in Figure 2. DOs upload indexes and signatures to CSP. The authorized DU issues search queries by submitting search tokens to CSP, PAS is responsible for verifying the correctness of the search results before returning them to the DU. CSP offers data storage and retrieval services to cloud clients (including DOs and DU). Note that the encryption of each record is achieved using the traditional public key encryption algorithm, which is outside the scope of our current discussion. Furthermore, the system initialization parameters and secret keys are generated by a fully trusted third-party, and we omit details of this process from the present study.

A fully credible PAS honestly ensures the accuracy of search results. Similar to the previous verifiable SE scheme, CSP is considered to be semi-honest-but-curious. We also assume that only the authorized DUs can issue search queries over encrypted data.

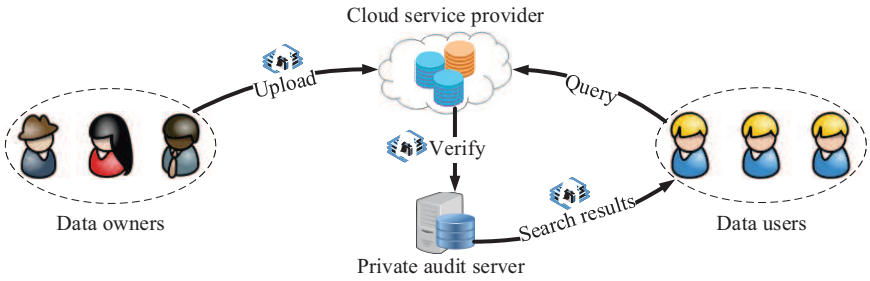

Figure 2 (Color online) System model of our scheme.

4.2 Design goals

In this section, we describe the design goals needed to guarantee the security and efficiency of our scheme as follows.

(1) Data security. To guarantee data security, sensitive data should be encrypted before outsourcing to a CSP. There should also be a results verification mechanism to be furnished to guarantee the correctness of the search results provided by the semi-trusted CSP.

(2) Feasibility and efficiency. To enhance its feasibility and practicality in real-world applications, our scheme should not incur a high computational burden in challenging multi-owner settings.

(3) Security goals. To address security concerns, our scheme should guarantee data security and privacy. Furthermore, our scheme should resist chosen-keyword attacks under a random oracle model.

4.3 Solution framework and security model

Our scheme is a tuple of six algorithms (**Setup**,**KeyGen**,**Enc**,**Trap**,**Search**,**Verify**), which are described as follows.

(1) **Setup**(1^k). Given the security parameter k, this deterministic algorithm outputs the global parameters $\mathcal{GP}$ and the public/secret key pair (PK,SK) for the traditional public key algorithm.

(2) **KeyGen**$(\mathcal{GP}, \mathrm{CSP}, \mathcal{O}, \mathcal{U})$. After inputting $\mathcal{GP}$, this probabilistic algorithm outputs the public/secret key pairs of $(\mathrm{pk}_{\mathcal{O}_t}, \mathrm{sk}_{\mathcal{O}_t})$, $(\mathrm{pk}_u, \mathrm{sk}_u)$, and $(\mathrm{pk}_S, \mathrm{sk}_S)$ for each DO $(\mathcal{O}_t \in \mathcal{O}, 1 \leqslant t \leqslant s)$, a specified DU $(u \in \mathcal{U})$, and CSP, respectively.

(3) **Enc**$(\mathcal{GP}, \mathrm{PK}, F, W, \mathrm{ID}, \mathrm{sk}_{\mathcal{O}_t}, \mathrm{pk}_S, \mathrm{pk}_u)$. For the file set F and keyword set W, the DOs first run this algorithm to generate the ciphertext set $C = \{c_i\}$ and index set $I = \{I_w\}$, before each DO $(\mathcal{O}_t)$ generates the signature $\mathrm{sig}_{t,i}$ for each ciphertext $c_i \in C$, where $1 \leqslant i \leqslant n$, $1 \leqslant t \leqslant s$, $w \in W$.

(4) **Trap**$(\mathcal{GP}, w', \mathrm{sk}_u)$. Given the keyword w', a specific DU (u) performs this probabilistic algorithm to output the trapdoor $T_{w'}$ utilizing the secret key sk_u, where $u \in \mathcal{U}$.

(5) **Search**$(\mathcal{GP}, T_{w'}, C, I, \mathrm{pk}_S, \mathrm{sk}_S)$. Using the trapdoor $T_{w'}$ as an input, the CSP first matches it with the index set I, then returns the relevant encrypted records $C'_{w'} \subseteq C$ to PAS.

(6) **Verify**$(\mathcal{GP}, \mathrm{Sig}, C'_{w'}, \mathrm{pk}_{\mathcal{O}_t})$. After receiving the returned results $C'_{w'}$, PAS needs to verify the correctness of the search results by interacting with CSP. If $C'_{w'}$ passes the result verification, then PAS accepts it and sends it to DU (u); otherwise, PAS rejects it.

In the following, we provide the security definition for our scheme according to a similar definition given for a previous scheme [7]. Our scheme can resist chosen-keyword attacks by two types of attackers represented as Game 1 and Game 2, respectively. In particular, the CSP cannot distinguish that a certain index is encrypted by a specific keyword. In addition, without the private key of CSP, an outside attacker cannot make a decision about whether the indexes match with the trapdoor. Next, we describe the chosen-keyword attack games in Definition 4.

Definition 4 (IND-CKA game)**.** Let an integer k be the security level and $\mathcal{A}$ be an adversary, and we present the indistinguishability against chosen-keyword attack (IND-CKA) game between adversary $\mathcal{A}$ and simulator $\mathcal{B}$.

First, we assume that $\mathcal{A}$ is the CSP, and Game 1 is presented as follows.

Table 2 Notation definitions

Symbol	Description	Symbol	Description
$F = \{f_i\}_{1 \leqslant i \leqslant n}$	Data file set	$T_{w'}$	Trapdoor for $w' \in W$
$\mathrm{ID} = \{\mathrm{id}_i\}_{1 \leqslant i \leqslant n}$	Identity set for F	$C'_{w'} = \{c'_j\}_{1 \leqslant j \leqslant \#C_{w'}}$	Results containing $w' \in W$
$C = \{c_i\}_{1 \leqslant i \leqslant n}$	Ciphertext set for F	$\#C'_{w'}$	Number of ciphertext in $C'_{w'}$
$W = \{w_j\}_{1 \leqslant j \leqslant m}$	Keyword set	$\mathrm{ID}'_{w'} = \{\mathrm{id}'_j\}_{1 \leqslant j \leqslant \#C'_{w'}}$	Identity set of $C'_{w'}$
I_w	Index for $w \in W$	$\mathrm{Sig} = \{\mathrm{sig}_i\}_{1 \leqslant i \leqslant n}$	Signature set for F
I	Index set for W	$\mathrm{sig}_i = \{\mathrm{sig}_{t,i}\}_{1 \leqslant t \leqslant s}$	DOs' signature set for f_i

(1) Setup. Given the security parameter k, $\mathcal{B}$ first simulates the **Setup** and **KeyGen** algorithms to generate the global parameters $\mathcal{GP}$, and the public/secret key pairs $\{(\mathrm{pk}_S, \mathrm{sk}_S), (\mathrm{pk}_u, \mathrm{sk}_u)\}$ for the CSP and specific DU (u), respectively. Then, $\mathcal{B}$ sends $\{\mathrm{pk}_S, \mathrm{sk}_S, \mathrm{pk}_u\}$ to $\mathcal{A}$.

(2) Phase 1. $\mathcal{A}$ issues a number of search queries to the **Trap** oracle as follows.

• **Trap**. $\mathcal{B}$ first performs the **Trap** algorithm for each keyword $w' \in \mathcal{W}$ to generate the search token $T_{w'}$, and then responses to $\mathcal{A}$'s search queries.

(3) Challenge. After the Phase 1, $\mathcal{A}$ selects two target keywords w_0, w_1 on which to be challenged, where it requires that both keywords w_0, w_1 were not been queried in Phase 1. After receiving these keywords, $\mathcal{B}$ responds to $\mathcal{A}$ by selecting a random bit $b \in \{0, 1\}$. In addition, $\mathcal{B}$ generates the indexes $\{T_{w_b}\}$ for the target keyword w_b and sends them to $\mathcal{A}$.

(4) Phase 2. $\mathcal{A}$ issues a number of search queries, as in Phase 1. The only restriction here is that the target keywords w_0, w_1 cannot be used to query the **Trap** oracle.

(5) Guess. $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ and wins the game if $b = b'$.

$\mathcal{A}$'s advantage when resisting the attack in Game 1 is denoted as $\mathrm{Adv}^{\text{Game 1}}_{\text{IND-CKA},\mathcal{A}}(1^k) = |\Pr[b' = b] - \frac{1}{2}|$.

Next, we assume that $\mathcal{A}$ is the outside attacker, and Game 2 is described as follows.

(1) Setup. Using the security parameter k as an input, $\mathcal{B}$ first performs the **Setup** and **KeyGen** algorithms to output the global parameters $\mathcal{GP}$, and the public/secret key pairs $\{(\mathrm{pk}_S, \mathrm{sk}_S), (\mathrm{pk}_u, \mathrm{sk}_u)\}$ for the CSP and specific DU (u), respectively. Then, $\mathcal{B}$ sends $\{\mathrm{pk}_S, \mathrm{sk}_S, \mathrm{pk}_u\}$ to $\mathcal{A}$.

(2) Challenge. $\mathcal{A}$ outputs two target keywords w_0, w_1 on which to be challenged, and $\mathcal{B}$ responds by selecting a random bit $b \in \{0, 1\}$. Finally, $\mathcal{B}$ generates the search token $\{T_{w_b}\}$ for the target keyword w_b and sends it to $\mathcal{A}$.

(3) Guess. $\mathcal{A}$ outputs a guess bit $b' \in \{0, 1\}$ and wins the game if $b = b'$.

Then, $\mathcal{A}$'s advantage when resisting the attack in Game 2 is defined as $\mathrm{Adv}^{\text{Game 2}}_{\text{IND-CKA},\mathcal{A}}(1^k) = |\Pr[b' = b] - \frac{1}{2}|$. Therefore, our scheme is IND-CKA secure if $\mathrm{Adv}^{\text{Game } i}_{\text{IND-CKA},\mathcal{A}}(1^k) = |\Pr[b' = b] - \frac{1}{2}|$ is negligible, where $i \in \{1, 2\}$.

5 Proposed VKSE-MO scheme

In this section, we mainly focus on building indexes and generating signatures, whereas the file encryption process is beyond the scope of our discussion. Before describing the construction of our scheme, we first introduce some notations in Table 2.

Next, we describe the concrete construction of our scheme as follows.

Setup(1^k). Given a security parameter k, this deterministic algorithm first outputs the bilinear map parameters (G_1, G_2, e, p, g). Then, it chooses two hash functions $H_1 : \{0, 1\}^* \to_R G_1$, $H_2 : \{0, 1\}^* \to_R Z_p^*$ and generates the public/secret key pair (PK, SK) for the traditional public key encryption algorithm in order to encrypt/decrypt records, respectively. Finally, it publishes the global parameters $\mathcal{GP} = \{G_1, G_2, e, p, g, H_1, H_2, \mathrm{PK}\}$, where the secret key SK is shared among authorized cloud clients.

$$\mathcal{GP} = \{G_1, G_2, e, p, g, H_1, H_2, \mathrm{PK}\}. \tag{1}$$

KeyGen($\mathcal{GP}, \mathrm{CSP}, \mathcal{O}, \mathcal{U}$). Run the probabilistic algorithm to generate public/secret key pairs for cloud clients (including the DO set $\mathcal{O}$ and authorized DU set $\mathcal{U}$) and CSP. For each DO ($\mathcal{O}_t \in \mathcal{O} | 1 \leqslant t \leqslant s$),

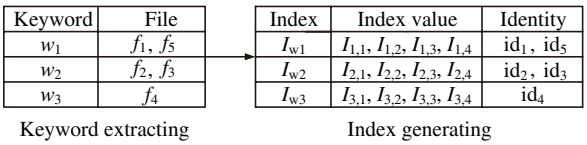

Figure 3 Example of an index structure.

this algorithm chooses elements $x_t, y_t \in_R Z_p^*$, $a_t \in_R G_1$ and calculates $g^{x_t}, g^{y_t} \in_R G_1$, before defining the public/secret key pair of DO ($\mathcal{O}_t$) as $\mathrm{pk}_{\mathcal{O}_t} = (a_t, g^{x_t}, g^{y_t})$, $\mathrm{sk}_{\mathcal{O}_t} = (x_t, y_t)$. In addition, for each authorized DU ($u \in \mathcal{U}$), this algorithm sets the public/secret key pair of DU (u) as $\mathrm{pk}_u = (a, g^y), \mathrm{sk}_u = y$, which is similar to the key generation of DO, where $y \in_R Z_p^*, a \in_R G_1$. For the CSP, this algorithm chooses elements $z \in_R Z_p^*, b \in_R G_1$ and computes $g^z \in_R G_1$ before outputting the public/secret key pair of CSP as $\mathrm{pk}_S = (g^z, b), \mathrm{sk}_S = z$.

$$(\mathrm{pk}_{\mathcal{O}_t}, \mathrm{sk}_{\mathcal{O}_t}) = \{(a_t, g^{x_t}, g^{y_t}), (x_t, y_t)\}; (\mathrm{pk}_u, \mathrm{sk}_u) = \{(a, g^y), y\}; (\mathrm{pk}_S, \mathrm{sk}_S) = \{(g^z, b), z\}. \tag{2}$$

Enc$(\mathcal{GP}, \mathrm{PK}, F, W, \mathrm{ID}, \mathrm{sk}_{\mathcal{O}_t}, \mathrm{pk}_S, \mathrm{pk}_u)$. This probabilistic algorithm first encrypts the file set F as ciphertext set C utilizing public key PK. For the ciphertext set $\{c_i \in C | 1 \leqslant i \leqslant n\}$ and corresponding identity set $\{\mathrm{id}_i\}_{1 \leqslant i \leqslant n}$, each DO $(\mathcal{O}_t | 1 \leqslant t \leqslant s)$ generates the signature $\mathrm{sig}_{t,i}$ for c_i as $\mathrm{sig}_{t,i} = (H_1(\mathrm{id}_i) g^{H_2(c_i)})^{x_t}$, and the signatures for c_i are then set as $\mathrm{sig}_i = \{\mathrm{sig}_{1,i}, \ldots, \mathrm{sig}_{s,i}\}$. Before building the index I_w for each keyword $w \in W$, it first selects elements $\alpha, \beta \in_R Z_p^*$ and computes $I_1 = g^\alpha, \gamma = H_2(e(g^z, b)^\alpha), I_2 = (g^y g^{-w})^{\beta/\gamma}, I_3 = e(g, g)^\beta, I_4 = e(g, a)^\beta$, then sends the signature set $\mathrm{Sig} = \{\mathrm{sig}_1, \ldots, \mathrm{sig}_n\}$ for file set F, index set $I = \{I_w | w \in W\}$ for keyword set W, and ciphertext set C to CSP, where $I_w = \{I_1, I_2, I_3, I_4\}$, $w \in_R Z_p^*$. Next, we provide an example of an index structure in Figure 3, which comprises five files and three keywords. For details of the specific encryption process, please refer to the Algorithm 1.

Algorithm 1 Ciphertext generation algorithm

Input: File set $F = \{f_i\}_{1 \leqslant i \leqslant n}$, identity set $\mathrm{ID} = \{\mathrm{id}_i\}_{1 \leqslant i \leqslant n}$, keyword set $W = \{w_j\}_{1 \leqslant j \leqslant m}$;
Output: Ciphertext set C, signature set Sig, index set I.
1: Given $\mathrm{sk}_{\mathcal{O}_t}, \mathrm{pk}_S, \mathrm{pk}_u, \mathrm{PK}$;
2: **for** $1 \leqslant i \leqslant n$ **do**
3: Encrypt f_i as c_i using PK; /* use the public key PK */
4: Generate signature sig_i for f_i; /* use the secret keys $\{\mathrm{sk}_{\mathcal{O}_t}\}$ of multiple DOs */
5: **end for**
6: Return Ciphertext set $C = \{c_i, \ldots, c_n\}$, signature set $\mathrm{Sig} = \{\mathrm{sig}_1, \ldots, \mathrm{sig}_n\}$;
7: **for** $1 \leqslant j \leqslant m$ **do**
8: Build index $I_{w_j} = \{I_1, I_2, I_3, I_4\}$; /* use the public keys $(\mathrm{pk}_S, \mathrm{pk}_u)$ of cloud server and DU */
9: **end for**
10: Return index set $I = \{I_{w_1}, \ldots, I_{w_m}\}$;
11: Send (C, I, Sig) to CSP.

Trap$(\mathcal{GP}, w', \mathrm{sk}_u)$. A specific DU ($u$) conducts this probabilistic algorithm to generate a search token for keyword w' by first selecting an element $\theta \in_R Z_p^*$ and setting $T_{w',1} = \theta, T_{w',2} = (ag^{-\theta})^{1/(y-w')}$ before submitting the search token $T_{w'} = \{T_{w',1}, T_{w',2}\}$ to CSP.

$$T_{w'} = \{\theta, (ag^{-\theta})^{1/(y-w')}\}. \tag{3}$$

Search$(\mathcal{GP}, T_{w'}, C, I, \mathrm{pk}_S, \mathrm{sk}_S)$. This deterministic algorithm is performed by CSP. It first computes $\gamma' = H_2(e(I_1, b)^z) = H_2(e(g^\alpha, b)^z)$, then checks whether Eq. (4) holds or not.

$$e(I_2^{\gamma'}, T_{w',2}) I_3^{T_{w',1}} = I_4. \tag{4}$$

If Eq. (4) holds, CSP returns the relevant ciphertext set $C'_{w'} = \{c'_j\}_{1 \leqslant j \leqslant \#C'_{w'}}$ and the corresponding identity set $\mathrm{ID}'_{w'} = \{\mathrm{id}'_j\}_{1 \leqslant j \leqslant \#C'_{w'}}$ to PAS; otherwise, it returns $\perp$. The specific search process is shown

Algorithm 2 Ciphertext search algorithm

Input: Search token $T_{w'} = \{T_{w',1}, T_{w',2}\}$, index set $I = \{I_w | w \in W\}$, public/secret key pair $(\mathrm{pk}_S, \mathrm{sk}_S)$, ciphertext set $C = \{c_i\}_{1 \leqslant i \leqslant n}$;

Output: Search results $C'_{w'}$ and corresponding identity set $\mathrm{ID}'_{w'}$.

1: Index $I_{w_j} = \{I_1, I_2, I_3, I_4\}$ for each keyword $w_j \in W$;
2: **for** $1 \leqslant j \leqslant m$ **do**
3: Compute $\gamma' = H_2(e(I_1, b)^z)$;
4: Check $e(I_2^{\gamma'}, T_{w',2}) I_3^{T_{w',1}} \stackrel{?}{=} I_4$ (1);
5: **if** Eq. (1) holds **then**
6: Output $C_{w_j} = C'_{w'}$, $\mathrm{ID}_{w_j} = \mathrm{ID}'_{w'}$; /* Eq. (1) holds, then $w_j = w'$ */
7: **else**
8: Output $\perp$;
9: **end if**
10: **end for**
11: Return $C'_{w'} = \{c'_j\}_{1 \leqslant j \leqslant \#C'_{w'}}$, $\mathrm{ID}'_{w'} = \{\mathrm{id}'_j\}_{1 \leqslant j \leqslant \#C'_{w'}}$; /* Return the ciphertext that contains w' */
12: Send $C'_{w'}$, $\mathrm{ID}'_{w'}$ to PAS.

in the Algorithm 2. First, CSP matches the trapdoor $T_{w'}$ with every keyword $w_j \in W$ $(1 \leqslant j \leqslant m)$ (Lines 2–10). Next, it returns the matched results $C'_{w'}$ and corresponding identity set $\mathrm{ID}'_{w'}$ to the PAS.

Verify$(\mathcal{GP}, \mathrm{Sig}, C_{w'}, \mathrm{pk}_{\mathcal{O}_t})$. After receiving the returned results $C'_{w'}$, PAS starts to verify the correctness of $C'_{w'}$ in order to ensure the data security. First, it chooses the elements $\tau_1, \ldots, \tau_{\#C'_{w'}} \in_R Z_p^*$, then sends the information $\{j, \tau_j\}_{j \in [1, \#C'_{w'}]}$ to CSP. Finally, the CSP returns the proof information as follows.

- **Step 1**. It first computes $\mu = \sum_{j=1}^{\#C'_{w'}} \tau_j H_2(c'_j)$, where $c'_j \in C'_{w'}$.
- **Step 2**. Then, it computes $\nu = \prod_{j=1}^{\#C'_{w'}} (\mathrm{sig}'_j)^{\tau_j}$, where $\mathrm{sig}'_j = \prod_{t=1}^{s} \mathrm{sig}'_{t,j}$, $\mathrm{sig}'_{t,j} = (H_1(\mathrm{id}'_j) g^{H_2(c'_j)})^{x_t}$.
- **Step 3**. Finally, it returns $\{\mu, \nu, \mathrm{id}'_{j \in [1, \#C'_{w'}]}\}$ to PAS.

After PAS obtains the proof information, it tests whether Eq. (5) holds,

$$e(\nu, g) = e\left(\prod_{j=1}^{\#C'_{w'}} H_1(\mathrm{id}'_j)^{\tau_j} \cdot g^{\mu}, \prod_{t=1}^{s} g^{x_t}\right). \tag{5}$$

If Eq. (5) holds, then it returns $C'_{w'}$ to DU (u); otherwise, it rejects the search results and returns $\perp$. After obtaining the search results, DU can decrypt them using the key SK. The detailed results verification is given in the Algorithm 3. First, CSP sends the proof information $\{\mu, \nu\}$ to PAS (Lines 2–5). Next, PAS verifies whether the search results $C'_{w'}$ are correct or not (Lines 6–16).

Remark. In our scheme, CSP returns the relevant ciphertext to DU when the submitted search token matches with the indexes, as shown in Eq. (4). CSP can honestly follow the established protocols to return relevant results, but guaranteeing whether CSP has tampered with or forged the encrypted records remains a challenge. To ensure the security of the data, our proposed **VKSE-MO** scheme use a verification mechanism to check the correctness of the returned results with Eq. (5).

6 Analysis of VKSE-MO scheme

6.1 Correctness

In this section, we show the correctness of our scheme if the aforementioned Eqs. (4) and (5) hold.

When the submitted search token matches with the index (namely, $w' = w$), DU can obtain the required records. For Eq. (4), we first have

$$\gamma' = H_2(e(I_1, b)^z) = H_2(e(g^{\alpha}, b)^z) = H_2(e(g^z, b)^{\alpha}) = \gamma.$$

Then we obtain

$$e(I_2^{\gamma'}, T_{w',2}) = e((g^y g^{-w})^{\beta\gamma'/\gamma}, (ag^{-\theta})^{1/(y-w')}) = e(g^{(y-w)\beta}, (ag^{-\theta})^{1/(y-w')}) = e(g, a)^{\beta} e(g, g)^{-\beta\theta},$$

Algorithm 3 Results verification algorithm

Input: Signature set $\mathrm{Sig} = \{\mathrm{sig}_1, \ldots, \mathrm{sig}_n\}$, search results $C'_{w'} = \{c'_j\}_{1 \leqslant j \leqslant \#C'_{w'}}$, identity set $\mathrm{ID}'_{w'} = \{\mathrm{id}'_j\}_{1 \leqslant j \leqslant \#C'_{w'}}$;
Output: "Accept" or "Reject".

Send challenge information $\{\tau_j\}_{1 \leqslant j \leqslant \#C'_{w'}}$ to CSP;
for $1 \leqslant j \leqslant \#C'_{w'}$ **do**
    Compute $\mu = \sum_{j=1}^{\#C'_{w'}} \tau_j H_2(c'_j)$;
    Compute $\nu = \prod_{j=1}^{\#C'_{w'}} (\mathrm{sig}'_j)^{\tau_j}$;
    Send $\{\mu, \nu\}$ to PAS; /* Proof information returned by cloud server */
    Compute $\prod_{j=1}^{\#C'_{w'}} H_1(\mathrm{id}'_j)^{\tau_j}$;
end for
for $1 \leqslant t \leqslant s$ **do**
    Compute $\prod_{t=1}^{s} g^{x_t}$;
end for
Check $e(\nu, g) \stackrel{?}{=} e(\prod_{j=1}^{\#C'_{w'}} H_1(\mathrm{id}'_j)^{\tau_j} \cdot g^{\mu}, \prod_{t=1}^{s} g^{x_t})$ (2);
if Eq. (2) holds **then**
    Output "Accept"; /* Cloud server returns the correct search results */
else
    Output "Reject"; /* Cloud server returns the false search results */
end if
Send the correct search results to DU.

$$I_3^{T_{w',1}} = (e(g,g)^{\beta})^{\theta} = e(g,g)^{\beta\theta}.$$

Finally, we can check the correctness of Eq. (4) when $w' = w$, as follows:

$$e(I_2^{\gamma'}, T_{w',2}) I_3^{T_{w',1}} = e(g,a)^{\beta} e(g,g)^{-\beta\theta} e(g,g)^{\beta\theta} = e(g,a)^{\beta} = I_4.$$

For Eq. (5), we have

$$\begin{aligned} e(\nu, g) &= e\left(\prod_{j=1}^{\#C'_{w'}} (\mathrm{sig}'_j)^{\tau_j}, g\right) = e\left(\prod_{j=1}^{\#C'_{w'}} \left(\prod_{t=1}^{s} \mathrm{sig}'_{t,j}\right)^{\tau_j}, g\right) = e\left(\prod_{j=1}^{\#C'_{w'}} \prod_{t=1}^{s} (H_1(\mathrm{id}'_j) g^{H_2(c'_j)})^{x_t \tau_j}, g\right) \\ &= e\left(\prod_{j=1}^{\#C'_{w'}} (H_1(\mathrm{id}'_j))^{\tau_j} g^{H_2(c'_j)\tau_j}, \prod_{t=1}^{s} g^{x_t}\right) = e\left(\prod_{j=1}^{\#C'_{w'}} (H_1(\mathrm{id}'_j))^{\tau_j} \cdot g^{\sum_{j=1}^{\#C'_{w'}} H_2(c'_j)\tau_j}, \prod_{t=1}^{s} g^{x_t}\right) \\ &= e\left(\prod_{j=1}^{\#C'_{w'}} (H_1(\mathrm{id}'_j))^{\tau_j} \cdot g^{\mu}, \prod_{t=1}^{s} g^{x_t}\right). \end{aligned}$$

Thus, we can verify that the Eq. (5) holds.

6.2 Security

In terms of security, we formally prove the security property of our scheme and show that it meets the design goals. The security of our scheme can be guaranteed by Theorem 1 and Theorem 2. Theorem 1 can guarantee data confidentiality by proving that our proposed scheme can resist the chosen-keyword attack considered by most of existing SE schemes. Theorem 2 can protect data security by checking the correctness of the search results.

Theorem 1. Our scheme is IND-CKA (indistinguishability against chosen-keyword attack) secure under a random oracle model given that DBDH problem and q-ABDHE problem are intractable.

Proof. The proof of Theorem 1 relies on the proofs of Lemma 1 and Lemma 2.

Lemma 1. Our scheme is secure against IND-CKA in Game 1 under a random oracle model given that q-ABDHE problem is intractable. Where $q \geqslant \psi_k$, ψ_k is denoted as the number of search token queries.

Proof. Let $\mathcal{A}$ be a polynomial-time adversary that can attack our scheme in Game 1 under a random oracle model, and $\mathcal{A}$'s advantage when attacking our scheme is ϵ, and ψ_k is the number of search token queries. Then, we can build a simulator $\mathcal{B}$ that plays the q-ABDHE game as follows.

Assuming that the bilinear parameters (G_1, G_2, e, g) are first set by the challenger, then $\mathcal{B}$ outputs a q-ABDHE instance $(g, g^{a'}, \ldots, g^{a'^q}, g^{b'}, g^{b'a'^{q+2}}, \phi)$. Finally, $\mathcal{B}$ needs to distinguish $\phi = e(g,g)^{b'a'^{q+1}}$ from a random element in G_2.

(1) Setup. Given a security parameter k, $\mathcal{B}$ first outputs $\mathcal{GP} = \{G_1, G_2, e, p, g, H_2, \mathcal{W}\}$, where $H_2 : \{0,1\}^* \to_R Z_p^*$ is a hash function, and $\mathcal{W} \in_R Z_p^*$ is the keyword space. Then he selects two elements $z \in_R Z_p^*$, $b \in_R G_1$, sets $\mathrm{pk}_S = (\varpi, b), \mathrm{sk}_S = z$ as the CSP's public key and private key, respectively, where $\varpi = g^z$. In addition, $\mathcal{B}$ chooses a random degree q polynomial $f(\varpi)$ and defines $\mathrm{pk}_u = (g^{a'}, a)$ as the public key of DU (u), where $a = g^{f(a')}$. Finally, $\mathcal{B}$ sends $(\mathrm{pk}_S, \mathrm{sk}_S, \mathrm{pk}_u)$ to $\mathcal{A}$.

(2) Query phase 1. $\mathcal{A}$ issues the search token queries as follows.

Step 1. $\mathcal{A}$ uses the keyword w^* to query the **Trap** oracle.

Step 2. $\mathcal{B}$ sets $T_{w^*,1} = f(w^*)$ and computes $T_{w^*,2} = g^{(f(a')-f(w^*))/(a'-w^*)}$.

Step 3. $\mathcal{B}$ sends $T_{w^*} = (T_{w^*,1}, T_{w^*,2})$ to $\mathcal{A}$. Note that $T_{w^*,1}$ is a random element from the viewpoint of $\mathcal{A}$ when $q \geqslant \psi_k$ beacuse $f(\varpi)$ is a random degree q polynomial.

(3) Challenge. When the Query phase 1 is over, $\mathcal{A}$ submits two target keywords w_0^*, w_1^*, and $\mathcal{B}$ responds according to the following steps.

Step 1. $\mathcal{B}$ first selects a random bit $\varrho \in \{0,1\}$ and sets $T_{w_\varrho^*,1} = f_k(w_\varrho^*)$. Next, $\mathcal{B}$ computes $T_{w_\varrho^*,2} = g^{(f(a')-f(w_\varrho^*))/(a'-w_\varrho^*)}$.

Step 2. $\mathcal{B}$ selects an element $\alpha^* \in_R Z_p^*$ and computes $I_1^* = g^{\alpha^*}, \gamma^* = H_2(e(\varpi, b)^{\alpha^*})$.

Step 3. $\mathcal{B}$ first defines the degree $q+1$ polynomial $F^*(\varpi) = (\varpi^{q+2} - (w_\varrho^*)^{q+2})/(\varpi - w_\varrho^*) = \sum_{i=0}^{q+1}(F_i^*\varpi^i)$, and then computes $I_2^* = (g^{b'a'^{q+2}}(g^{b'})^{-(w_\varrho^*)^{q+2}})^{1/\gamma^*}$, $I_3^* = \phi^{F_{q+1}^*} e(g^{b'}, \prod_{i=0}^{q}(g^{a'^i})^{F_i^*})$, $I_4^* = e((I_2^*)^{\gamma^*}, T_{w_\varrho^*,2})(I_3^*)^{T_{w_\varrho^*,1}}$. Finally, the index $I^* = (I_1^*, I_2^*, I_3^*, I_4^*)$ is sent to $\mathcal{A}$. Set $\beta^* = b'F^*(a')$, and if $\phi = e(g,g)^{b'a'^{q+1}}$, then $I_2^* = g^{(a'-w_\varrho^*)(b'(a'^{q+2}-(w_\varrho^*)^{q+2})/(a'-w_\varrho^*))1/\gamma^*} = g^{(a'-w_\varrho^*)\phi/\gamma^*}$, $I_3^* = e(g,g)^{\beta^*}$, $I_4^* = e(g,a)^{\beta^*}$.

(4) Query phase 2. $\mathcal{A}$ issues search queries as the processes in Query phase 1.

(5) Guess. $\mathcal{A}$ returns the guess bit $\varrho' \in \{0,1\}$, and if $\varrho' = \varrho$, then $\mathcal{B}$ outputs 1 to show that $\phi = e(g,g)^{b'a'^{q+1}}$; otherwise, $\mathcal{B}$ outputs 0 to show that $\phi = e(g,g)^{\beta}$.

If the equation $\phi = e(g,g)^{b'a'^{q+1}}$ holds, then $\mathcal{A}$ can accurately guess the bit ϱ with an advantage $\frac{1}{2} + \epsilon$; otherwise, ϕ is a random element in G_1, the tuple (I_2^*, I_3^*) is a random and independent element, and the inequality $I_3^* \neq e(g, (I_2^*)^{\gamma^*})^{1/(a'-w_\varrho^*)}$ holds with an advantage $1 - \frac{1}{p}$. If the inequality holds, then the value of $I_4^* = e((I_2^*)^{\gamma^*}, a^{1/(a'-w_\varrho^*)})(I_3^*/e(g,(I_2^*)^{\gamma^*})^{1/(a'-w_\varrho^*)})^{T_{w_\varrho^*,1}}$ is uniformly random and independent from the viewpoint of $\mathcal{A}$ (except for the value I_4^*) because $T_{w_\varrho^*,1}$ is a random and independent element from the viewponit of $\mathcal{A}$ (except for the value I_3^*) when $q \geqslant \psi_k, T_{w_\varrho^*,1} = f(w_\varrho^*)$. In addition, as α^* is an element in ${}_RZ_p^*$, so $I_1^* = g^{\alpha^*}$ is uniformly random and independent of (I_2^*, I_3^*, I_4^*). Thus, the tuple $(I_1^*, I_2^*, I_3^*, I_4^*)$ leaks no valuable information regarding the bit ϱ. This completes the proof of Lemma 1.

Lemma 2. Our scheme is secure against IND-CKA in Game 2 under a random oracle model given that DBDH problem is intractable.

Proof. Let $\mathcal{A}$ be a polynomial-time adversary which can attack our scheme in Game 2 under a random oracle model, then we build a simulator $\mathcal{B}$ to play the DBDH game as follows.

Given the bilinear map parameters (G_1, G_2, p, e, g), $\mathcal{B}$ outputs a DBDH tuple $(g, g^{a'}, g^{b'}, g^{c'}, \phi)$, and must then determine whether $\phi = e(g,g)^{a'b'c'}$ or an element in ${}_RG_2$.

(1) Setup. After inputting the security parameter k, output the parameters $\mathcal{GP} = (G_1, G_2, p, e, g, H_2, \mathcal{W})$, where $H_2 : \{0,1\}^* \to_R Z_p^*$ is a hash function and $\mathcal{W} \in_R Z_p^*$ is the keyword space. $\mathcal{B}$ first sets $g^z = g^{a'}, b = g^{b'}$ and denotes the public key of CSP as $\mathrm{pk}_S = (g^z, b)$ before selecting two elements $y \in_R Z_p^*, a \in_R G_1$ and denoting the public key and private key of DU (u) as $\mathrm{pk}_u = (g^y, a), \mathrm{sk}_u = y$, respectively. Finally, $\mathcal{B}$ sends the parameters $(\mathrm{pk}_S, \mathrm{pk}_u, \mathrm{sk}_u)$ to $\mathcal{A}$.

(2) Challenge. $\mathcal{A}$ issues two target keywords w_0', w_1' and $\mathcal{B}$ responds it with the following steps.

Step 1. $\mathcal{B}$ first outputs a random bit $\varrho \in \{0,1\}$ and computes $I_1^* = g^{c'}, \gamma^* = H_2(\phi)$.

Step 2. Then, $\mathcal{B}$ chooses $\beta^* \in_R Z_p^*$ and computes $I_2^* = (g^y g^{-w_\varrho'})^{\beta^*/\gamma^*}$, $I_3^* = e(g,g)^{\beta^*}$, $I_4^* = e(g,a)^{\beta^*}$.

Step 3. Finally, $\mathcal{B}$ sends the index $I^* = (I_1^*, I_2^*, I_3^*, I_4^*)$ to $\mathcal{A}$.

(3) Guess. $\mathcal{A}$ returns the guess bit ϱ', and if $\varrho' = \varrho$, then $\mathcal{B}$ outputs 1 meaning $\phi = e(g,g)^{a'b'c'}$; otherwise, $\mathcal{B}$ outputs 0 and thus $\phi = e(g,g)^{\beta^*}$.

Assuming that $\mathcal{A}$ has an advantage ϵ when breaking our scheme under a random oracle model, then the probability of $\mathcal{B}$ is shown as follows.

If the equation $\phi = e(g,g)^{a'b'c'}$ holds, then the advantage of $\mathcal{A}$ satisfies $|\Pr[\varrho' = \varrho] - \frac{1}{w}| \geqslant \epsilon$. If ϕ is uniformly random in G_2, then $\Pr[\varrho' = \varrho] = \frac{1}{2}$. Thus, we can find that $|\Pr[\mathcal{B}(g, g^{a'}, g^{b'}, g^{c'}, e(g,g)^{a'b'c'}) = 1] - \Pr[\mathcal{B}(g, g^{a'}, g^{b'}, g^{c'}, e(g,g)^{\beta^*}) = 1]| \geqslant |(\frac{1}{2} \pm \epsilon) - \frac{1}{2}|$ when a', b', c' are uniformly random in Z_p^* and ϕ is uniformly random in G_2. This completes the proof of Lemma 2.

The analysis above completes the proof of Theorem 1.

Theorem 2. For a semi-trusted CSP, it is computationally infeasible to forge a valid result verification proof for adversary $\mathcal{A}$ under the CDH and DL assumptions.

Proof. To the best of our knowledge, $\mathcal{A}$ can forge a valid results verification proof using one of the following two methods:

(1) First, assuming that $\mathcal{A}$ can forge a valid multisignature based on each encrypted record, then he can forge a valid result verification proof based on the forged multisignatures using the ciphertext set. By contrast, if $\mathcal{A}$ can generate a valid forgery, then we can solve the CDH problem in G_1, which contradicts the CDH assumption. Moreover, although $\mathcal{A}$ can corrupt with up to $(s-1)$ DOs and independently forge their corresponding public/secret key pairs, it is still computationally infeasible to forge a valid multisignature, as proved previously [20]. Thus, it is infeasible to forge a valid result verification via this way.

(2) Second, $\mathcal{A}$ is able to directly forge the valid result verification proof based on the whole ciphertext set if he breaks the following security game.

We present the details of security game as follows.

Step 1. First, PAS sends the challenge information $\{j, \tau_j\}_{j\in[1,\#C'_{w'}]}$ to the CSP, and CSP should return the proof information $\{\mu, \nu, \mathrm{id}'_{j\in[1,\#C'_{w'}]}\}$ based on the correct encrypted data $C'_{w'}$. In addition, $\mathcal{A}$ outputs the forgery of the result verification proof information $\{\mu', \nu, \mathrm{id}''_{j\in[1,\#C''_{w'}]}\}$ based on the corrupted $C''_{w'}$, where $\mu' = \sum_{j=1}^{\#C''_{w'}} \tau_j H_2(c''_j)$, $c''_j \in C''_{w'}$, $j \in [1, \#C''_{w'}]$ and $C''_{w'} \neq C'_{w'}$.

Step 2. Second, if we let $\triangle\mu = \mu' - \mu (\triangle\mu \neq 0)$, then we can say that $\mathcal{A}$ may successfully win the game if the forged proof information $\{\mu', \nu, \mathrm{id}''_{j\in[1,\#C''_{w'}]}\}$ can pass the result verification mechanism; otherwise, he fails. Suppose that $\mathcal{A}$ wins the security game, then we can have $e(\nu, g) = e(\prod_{j=1}^{\#C''_{w'}} H_1(\mathrm{id}''_j)^{\tau_j} \cdot g^{\mu'}, \prod_{t=1}^{s} g^{x_t})$. We also obtain $e(\nu, g) = e(\prod_{j=1}^{\#C'_{w'}} H_1(\mathrm{id}'_j)^{\tau_j} \cdot g^{\mu}, \prod_{t=1}^{s} g^{x_t})$ according to the valid result verification proof information $\{\mu, \nu, \mathrm{id}'_{j\in[1,\#C'_{w'}]}\}$. Therefore, we further have $g^{\mu'} = g^{\mu}, g^{\triangle\mu} = 1$.

Given two elements $\varpi_1, \varpi_2 \in_R G_1$, then a certain element $x \in_R Z_p^*$ exists such that $\varpi_2 = \varpi_1^x$ because G_1 is a multiplicative cyclic group. Without any loss of generality, the generator g can be expressed as $g = \varpi_1^{\lambda_1} \varpi_2^{\lambda_2}$, where $\lambda_1, \lambda_2 \in_R Z_p^*$. Thus, we can have the following equation:

$$(\varpi_1^{\lambda_1} \varpi_2^{\lambda_2})^{\triangle\mu} = 1 \Longleftrightarrow \varpi_1^{\lambda_1 \triangle\mu} \varpi_2^{\lambda_2 \triangle\mu} = 1.$$

Step 3. Based on the Step 2, a solution exists to the DL problem (given $g, g^{a'} \in_R G_1$, output $a' \in_R Z_p^*$). Given ϖ_1, ϖ_1^x, we can deduce that $\varpi_2 = \varpi_1^{-\frac{\lambda_1 \triangle\mu}{\lambda_2 \triangle\mu}} = \varpi_1^x$, $x = -\frac{\lambda_1 \triangle\mu}{\lambda_2 \triangle\mu}$ provided that $\lambda_2 \triangle \mu \neq 0$. As mentioned above, $\triangle\mu \neq 0$ and λ_2 is a random element in Z_p^*, the probability of $\triangle\mu \neq 0$ is $1 - \frac{1}{p}$. In other words, we can solve the DL problem if $\mathcal{A}$ breaks the security game, which contradicts to the DL assumption. This completes the proof of Theorem 2.

6.3 Performance

In this section, we present the performance evaluations of our scheme in terms of its computational complexity and its actual performance using a real-world dataset. The experiments were implemented on an Ubuntu 15.04 Server with an Intel Core i5 processor running at 2.3 GHz using C and the Paring Based Cryptography (PBC) Library. In the PBC Library, Type A is denoted as $E(F_q) : y^2 = x^3 + x$,

Table 3 Computational complexity of different algorithms in the two schemes

Different algorithm	VKSE-MO scheme	ABKS-UR [15] scheme
KeyGen	$(2s+1+\|\mathcal{U}\|)E$	$(2\|\mathcal{U}\|+2\|\mathcal{N}\|+1)E$
Enc	$(2m+2n+4)E+n\mathcal{O}_{H_1}+3P$	$n(\|\mathcal{N}\|+2)E$
Trap	$(R+1)E$	$(2\|\mathcal{N}\|+1)E$
Search	$(2+R)E+(1+R)P$	$(\|\mathcal{N}\|+1)P+E$
Verify	$(1+\#C'_{w'})E+2P+\#C'_{w'}\mathcal{O}_{H_1}$	Not considerated

G_1 is a subgroup of $E(F_q)$, and the cyclic group is a subgroup of $E(F_q)^2$, where q is a large prime number. The group order of G_1 is 160-bit and the base field is 512-bit. In terms of the computational complexity, we mainly considered several computational operations such as the exponentiation operation (E) in group G_1, pairing operation (P), and hash operation ($\mathcal{O}_{H_1}$) which maps a bit string to element in G_1. In Table 3, we present the computational overheads of our scheme compared with the state-of-the-art ABKS-UR scheme [15] which is based on the attribute-based encryption [31], where $|\mathcal{U}|$ denotes the number of DUs, $|\mathcal{N}|$ is the number of attributes in system, and R denotes the number of keywords submitted. In the comparison, we set $s \in [1,10]$, $|\mathcal{U}| \in [1,50]$, $R \in [1,50]$, $|\mathcal{N}| = 100$, $\#C_{w'} \in [1,100]$, $m \in [1,1000]$, and $n \in [1,10000]$.

Table 3 shows that our scheme is more efficient than the ABKS-UR scheme with respect to the **KeyGen**, **Enc**, **Trap**, **Search** algorithms but not the **verify** algorithm. Apparently, our scheme has lower computational costs than the ABKS-UR scheme for the **Trap** and **Search** algorithms because $R \ll |\mathcal{N}|$, where $|\mathcal{N}|$ denotes the number of total attributes in system and R is the number of keywords submitted in a single search query. For the **KeyGen** algorithm, the ABKS-UR scheme and our scheme require $2|\mathcal{U}|E$, $|\mathcal{U}|E$ for DUs, respectively. Furthermore, in practice, $s \ll |\mathcal{N}|$. Thus, our scheme is more efficient than the ABKS-UR scheme when considering the **KeyGen** algorithm. For the **Enc** algorithm, the ABKS-UR scheme needs $|\mathcal{N}|E$ for each record. When encrypting the total records, the ABKS-UR scheme needs $n|\mathcal{N}|E$, whereas our scheme only needs $(2m+2n+4)E+n\mathcal{O}_{H_1}$. Therefore, ABKS-UR incurs a greater computational burden than our scheme for the **Enc** algorithm. To the best of our knowledge, the ABKS-UR scheme cannot accurately verify the correctness of search results because the false positive rate caused by the Bloom Filter will incur high communication overheads. Thus, the **verify** algorithm in the ABKS-UR scheme is outside the scope of our discussion.

We also conducted an empirical study using a real-world dataset, where we employed the Enron email dataset[1)] containing half million records from 150 users in order to assess the actual performance of the aforementioned schemes. For convenience, we randomly selected 10000 records ($n = 10000$) from this dataset and performed the experiments 100 times, where we set the number of keywords as $W \in [1,1000]$.

As shown in Figure 4, we first analyzed the key generation time by varying the number of DUs ($|\mathcal{U}| \in [1,50]$), where we found that the computational overheads of **KeyGen** algorithm in both schemes increase almost linearly with the value of $|\mathcal{U}|$ when we set $|\mathcal{N}| = 100$, $s = 1, 5, 10$, respectively. The ABKS-UR scheme requires two exponentiation operations for each DU and multiple operations for attribute set $\mathcal{N}$, whereas our scheme only needs one for each DU. Thus, our scheme outperformed the ABKS-UR scheme in terms of **KeyGen** algorithm. In addition, the computational cost of our scheme increased for the **KeyGen** algorithm when we increased the value from $s \in [1,10]$. In practical applications, the number of DOs for each record is much less than $|\mathcal{U}|$, so our scheme is more efficient than the ABKS-UR scheme.

As shown in Figure 5, we evaluated the computational burden of the **Enc** algorithm in both schemes by varying the number of records from 1 to 10000 ($n \in [1,10000]$). Obviously, the computational costs of the two schemes increased with the value of n when we set $|\mathcal{N}| = 100$, $m = 100, 500, 1000$, respectively. The ABKS-UR scheme requires $|\mathcal{N}|$ exponentiation operations for each record, so its computational cost is proportional to the value of n, whereas that of our scheme only needs two exponentiation operations and one hash operation $\mathcal{O}_{H_1}$. Our scheme needs to encrypt keywords, but it does not incur additional overheads when the value of m increases because the hash operation $\mathcal{O}_{H_2}$ is much more efficient than

1) http://www.cs.cmu.edu/ enron/.

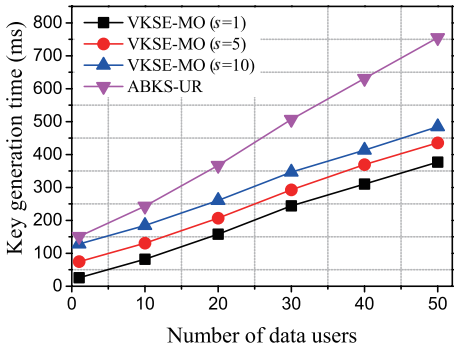

Figure 4 (Color online) **KeyGen** algorithm.

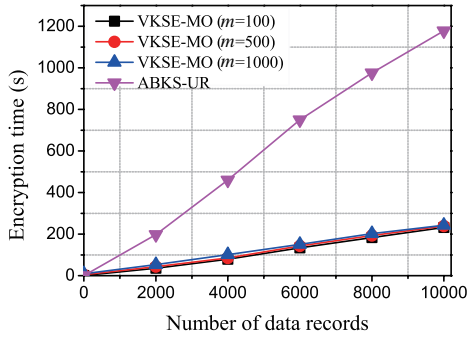

Figure 5 (Color online) **Enc** algorithm.

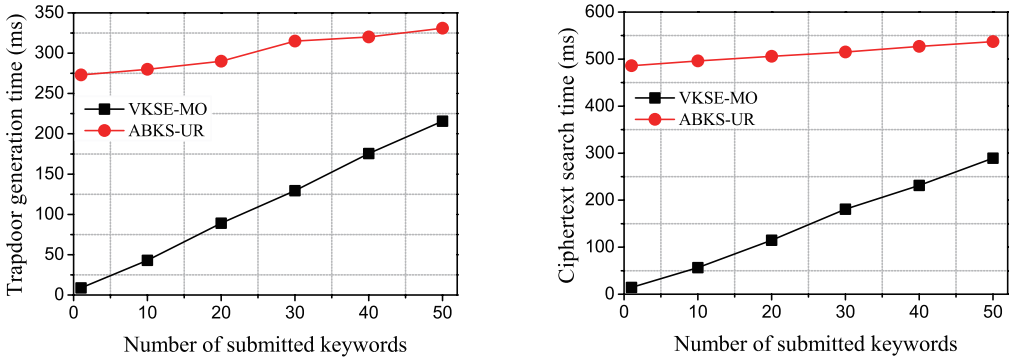

Figure 6 (Color online) Computational costs. (a) **Trap** algorithm; (b) **Search** algorithm.

other operations. Therefore, our scheme was much more efficient than the ABKS-UR scheme in **Enc** algorithm. In addition, the **Enc** algorithm does not affect the user search experience because it is only performed only in the initialization of the system. Thus, our scheme is still acceptable in practice.

As shown in Figure 6(a), the computational overheads of the **Trap** algorithm in our scheme depend mainly on the number of keywords ($R \in [1, 50]$) submitted, whereas those in the ABKS-UR scheme are affected by the factor $|\mathcal{N}|$. For comparison, we set $|\mathcal{N}| = 100$ and varied the value of R from 1 to 50 in this algorithm. The computational overheads of our scheme increased almost linearly with R, whereas those of the ABKS-UR scheme remained almost unchanged. Our scheme had much lower computational overheads in terms of the trapdoor generation time than the ABKS-UR scheme when we set $R \ll |\mathcal{N}|$ in practice. Similar to the **Trap** algorithm, we demonstrated the computational costs of the **Search** algorithm by varying the value of $R \in [1, 50]$ and the results are shown in Figure 6(b). The ABKS-UR scheme must perform $|\mathcal{N}|$ exponentiation operations for each query, whereas our scheme only needs to conduct R exponentiation operations. The computational costs incurred by our scheme with the **Search** algorithm increased almost linearly with respect to R, whereas those of ABKS-UR scheme remained about the same, but our scheme was still much more efficient than the ABKS-UR scheme in terms of the ciphertext search time. For example, with $R = 50$, $|\mathcal{N}| = 100$, the ABKS-UR scheme required 537 ms and our scheme needed 289 ms.

The ABKS-UR scheme cannot accurately verify the correctness of search results because of the high number of false positives generated by the inherent defects of bloom filter, so we only show the results verification time for our scheme with the **Verify** algorithm in Figure 7. The computational costs of **Verify** algorithm increased with the number of search results ($\#C'_{w'} \in [1, 100]$), which is consistent with the theoretical study in Table 3. In particular, when $\#C'_{w'} = 100$, the verification process only required 1445 ms. In addition, the process is conducted mainly by the PAS, which can provide a powerful computing capacity. Thus, the results verification operation will not impose a great computational

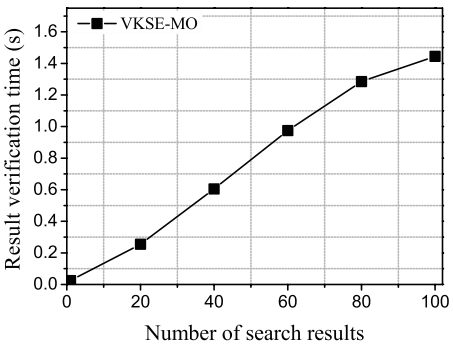

Figure 7 **Verify** algorithm.

burden on DUs, especially for the computation and bandwidth resource-limited DUs, such as sensor nodes and mobile terminals.

According to the results presented above, the performance evaluation based on a real-world dataset agreed completely with the computational complexity shown in Table 3. Compared with the state-of-the-art ABKS-UR scheme, we verified that our scheme is more efficient and feasible in practice.

7 Conclusion

In this study, for a challenging multi-owner setting, we proposed an efficient and feasible results verification scheme that allows DU to issue search queries and that also guarantees the accuracy of the search results simultaneously. The formal security analysis demonstrated that our scheme is secure against IND-CKA under a random oracle model. In addition, the results verification time is independent of the number of DOs, and our experimental results over real-world dataset demonstrated the practical efficiency and feasibility of our scheme.

Acknowledgements This work was supported by National High Technology Research and Development Program (863 Program) (Grant No. 2015AA016007), National Nature Science Foundation of China (Grant Nos. 61303221, 61472310, 61370078, 61309016), Science Foundation of Two sides of Strait (Grant Nos. U1405255, U1135002), and Shaanxi Science & Technology Coordination & Innovation Project (Grant No. 2016TZC-G-6-3).

Conflict of interest The authors declare that they have no conflict of interest.

References

1 Xia Z H, Wang X H, Zhang L G, et al. A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing. IEEE Trans Inf Foren Secur, 2016, 11: 2594–2608
2 Li Q, Ma J F, Li R, et al. Secure, efficient and revocable multi-authority access control system in cloud storage. Comput Secur, 2016, 59: 45–59
3 Li H W, Liu D X, Dai Y S, et al. Engineering searchable encryption of mobile cloud networks: when QoE meets QoP. IEEE Wirel Commun, 2015, 22: 74–80
4 Fu Z J, Sun X M, Ji S, et al. Towards efficient content-aware search over encrypted outsourced data in cloud. In: Proceedings of Annual IEEE International Conference on Computer Communications, San Francisco, 2016. 1–9
5 Li H W, Liu D X, Dai Y S, et al. Personalized search over encrypted data with efficient and secure updates in mobile clouds. IEEE Trans Emerg Topics Comput, 2015, in press. doi:10.1109/TETC.2015.2511457
6 Song D X, Wagner D, Perrig A. Practical techniques for searches on encrypted data. In: Proceedings of IEEE Symposium on Security and Privacy, California, 2000. 44–55
7 Boneh D, Crescenzo G D, Ostrovsky R, et al. Public key encryption with keyword search. In: Proceedings of International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, 2004. 506–522
8 Miao Y B, Ma J F, Liu Z Q. Revocable and anonymous searchable encryption in multi-user setting. Concurr Comp-Pract E, 2016, 28: 1204–1218
9 Chai Q, Gong G. Verifiable symmetric searchable encryption for semi-honest-but-curious cloud servers. In: Proceedings of IEEE International Conference on Communications, Ottawa, 2012. 917–922
10 Zheng Q J, Xu S H, Ateniese G. VABKS: verifiable attribute-based keyword search over outsourced encrypted data. In: Proceedings of IEEE Conference on Computer Communications, Toronto, 2014. 522–530

11 Sun W H, Liu X F, Lou W J, et al. Catch you if you lie to me: Efficient verifiable conjunctive keyword search over large dynamic encrypted cloud data. In: Proceedings of IEEE Conference on Computer Communications, Kowloon, 2015. 2110–2118
12 Fu Z J, Shu J G, Sun X M, et al. Smart cloud search services: verifiable keyword-based semantic search over encrypted cloud data. IEEE Trans Consum Electron, 2014, 60: 762–770
13 Cheng R, Yan J B, Guan C W, et al. Verifiable searchable symmetric encryption from indistinguishability obfuscation. In: Proceedings of ACM Symposium on Information, Computer and Communications Security, Singapore, 2015. 621–626
14 Li T, Liu Z L, Li P, et al. Verifiable searchable encryption with aggregate keys for data sharing in outsourcing storage. In: Proceedings of Australasian Conference on Information Security and Privacy, Melbourne, 2016. 153–169
15 Sun W H, Yu S C, Lou W J, et al. Protecting your right: verifiable attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud. IEEE Trans Parallel Distrib Syst, 2016, 27: 1187–1198
16 Li J G, Lin Y P, Wen M, et al. Secure and verifiable multi-owner ranked-keyword search in cloud computing. In: Proceedings of International Conference on Wireless Algorithms, Systems, and Applications, Qufu, 2015. 325–334
17 Zhang W, Lin Y P, Xiao S, et al. Privacy preserving ranked multi-keyword search for multiple data owners in cloud computing. IEEE Trans Comput, 2016, 65: 1566–1577
18 Lu S, Ostrovsky R, Sahai A, et al. Sequential aggregate signatures, multisignatures, and verifiably encrypted signatures without random oracles. J Cryptol, 2013, 26: 340–373
19 Ren Y J, Shen J, Wang J, et al. Mutual verifiable provable data auditing in public cloud storage. J Int Tech, 2015, 16: 317–323
20 Wang B Y, Li H, Liu X F, et al. Efficient public verification on the integrity of multi-owner data in the cloud. J Commun Netw, 2014, 16: 592–599
21 Chen R M, Mu Y, Yang G M, et al. Dual-server public-key encryption with keyword search for secure cloud storage. IEEE Trans Inf Foren Secur, 2016, 11: 789–798
22 Wang W, Xu P, Li H, et al. Secure hybrid-indexed search for high efficiency over keyword searchable ciphertexts. Future Gener Comput Syst, 2016, 55: 353–361
23 Yang Y, Ma M D. Conjunctive keyword search with designated tester and timing enabled proxy re-encryption function for E-health clouds. IEEE Trans Inf Foren Secur, 2016, 11: 746–759
24 Xia Z H, Wang X H, Sun X M, et al. A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data. IEEE Trans Parallel Distrib Syst, 2016, 27: 340–352
25 Fu Z J, Sun X M, Liu Q, et al. Achieving efficient cloud search services: multi-keyword ranked search over encrypted cloud data supporting parallel computing. IEICE Trans, 2015, 98: 190–200
26 Fu Z J, Ren K, Su J G, et al. Enabling personalized search over encrypted outsourced data with efficiency improvement. IEEE Trans Parallel Distrib Syst, 2016, 27: 2546–2559
27 Fu Z J, Wu X L, Guan C W, et al. Towards efficient multi-keyword fuzzy search over encrypted outsourced data with accuracy improvement. IEEE Trans Inf Foren Secur, 2016, 11: 2706–2716
28 Li H W, Yang Y, Luan T H, et al. Enabling fine-grained multi-keyword search supporting classified sub-dictionaries over encrypted cloud data. IEEE Trans Depend Secure, 2016, 13: 312–325
29 Li H W, Liu D X, Dai Y S, et al. Enabling efficient multi-keyword ranked search over encrypted mobile cloud data through blind storage. IEEE Trans Emerg Topics Comput, 2015, 3: 127–138
30 Layouni M, Yoshida M, Okamura S. Efficient multi-authorizer accredited symmetrically private information retrieval. In: Proceedings of International Conference on Information and Communications Security, Birmingham, 2008. 387–402
31 Attrapadung N, Libert B, Panafieu E D. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In: Proceedings of International Conference on Practice and Theory in Public Key Cryptography, Taormina, 2011. 90–108