

9-2017

# VCKSM: Verifiable conjunctive keyword search over mobile e-health cloud in shared multi-owner settings

Yinbin MIAO  
*Xidian University*

Jianfeng MA  
*Xidian University*


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

Qi JIANG  
*Xidian University*

Junwei ZHANG  
*Xidian University*

*See next page for additional authors*

Follow this and additional works at: [https://ink.library.smu.edu.sg/sis\\_research](https://ink.library.smu.edu.sg/sis_research)

 Part of the [Databases and Information Systems Commons](#), [Information Security Commons](#), and the [Medicine and Health Sciences Commons](#)

---

## Citation

MIAO, Yinbin; MA, Jianfeng; LIU, Ximeng; JIANG, Qi; ZHANG, Junwei; SHEN, Limin; and LIU, Zhiquan. VCKSM: Verifiable conjunctive keyword search over mobile e-health cloud in shared multi-owner settings. (2017). *Pervasive and Mobile Computing*. 40, 205-219. Research Collection School Of Information Systems.

**Available at:** [https://ink.library.smu.edu.sg/sis\\_research/3677](https://ink.library.smu.edu.sg/sis_research/3677)

This Journal Article is brought to you for free and open access by the School of Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email [libIR@smu.edu.sg](mailto:libIR@smu.edu.sg).

---

**Author**

Yinbin MIAO, Jianfeng MA, Ximeng LIU, Qi JIANG, Junwei ZHANG, Limin SHEN, and Zhiqian LIU

# VCKSM: Verifiable conjunctive keyword search over mobile e-health cloud in shared multi-owner settings

Yinbin Miao<sup>a,\*</sup>, Jianfeng Ma<sup>a</sup>, Ximeng Liu<sup>b</sup>, Qi Jiang<sup>a,c</sup>, Junwei Zhang<sup>a</sup>,  
Limin Shen<sup>a</sup>, Zhiquan Liu<sup>a</sup>

<sup>a</sup> School of Cyber Engineering, Xidian University, Xi'an 710071, China

<sup>b</sup> School of Information Systems, Singapore Management University, 80 Stamford Road, Singapore, Republic of Singapore

<sup>c</sup> School of Computer and Software, Nanjing University of Information Science and Technology, Nanjing 210044, China

## Keywords:

Searchable encryption  
Multi-owner settings  
Conjunctive keyword  
Correctness  
Keyword guessing attacks

Searchable encryption (SE) is a promising technique which enables cloud users to conduct search over encrypted cloud data in a privacy-preserving way, especially for the electronic health record (EHR) system that contains plenty of medical history, diagnosis, radiology images, etc. In this paper, we focus on a more practical scenario, also named as the *shared* multi-owner settings, where each e-health record is co-owned by a fixed number of parties. Although the existing SE schemes under the *unshared* multi-owner settings can be adapted to this *shared* scenario, these schemes have to build multiple indexes, which definitely incur higher computational overhead. To save bandwidth and computing resources in cloud servers and guarantee the correctness of search results, we present a secure cryptographic primitive, namely verifiable conjunctive keyword search over mobile e-health cloud scheme, in the *shared* multi-owner settings by utilizing multisignatures technique. Formal security analysis proves that our scheme is secure against the keyword guessing attacks in standard model. Empirical study using a real-world dataset justifies that our scheme is efficient and feasible in practical applications.

## 1. Introduction

Cloud computing [1,2] enables cloud clients to remotely store their data in the cloud server so as to reduce the local data storage and management burden. However, it also introduces side effects such that the sensitive information contained in the data may be leaked to the cloud service provider (CSP) or the attackers. To protect data security and privacy [3,4], the sensitive data should be encrypted before outsourcing. Nevertheless, the encryption-before-outsourcing mechanism makes the retrieval [5–9] over encrypted data extremely difficult, especially for the EHR system [10] that contains a large number of medical diagnosis and treatment documents. With the prevalence of cloud computing, EHRs are commonly used by patients, doctors and other healthcare professionals in order to reduce the healthcare costs and provide efficient healthcare services. However, in the semi-trusted cloud environment, data security and privacy concerns will be enlarged as EHRs may contain sensitive information and are likely to suffer from potential data compromise and privacy disclosure. To balance the conflicts between data privacy and usability, the public key encryption with keyword search (PEKS) scheme [11], which enables data users to securely search and selectively retrieve records of interest according to the user-specified keywords, has been proposed and gained extensive research attentions.

\* Corresponding author.

E-mail address: [ybmiao2014@163.com](mailto:ybmiao2014@163.com) (Y. Miao).

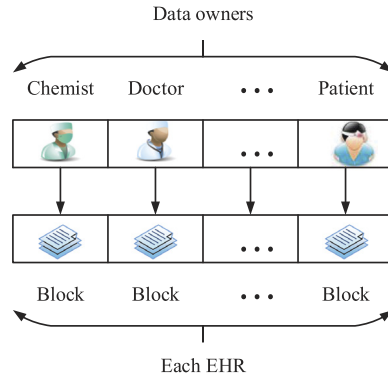


Fig. 1. Shared multi-owner settings.

Mobile cloud computing [12,13] which is a specific field of cloud computing always considers the resource constrained devices, i.e., mobile terminals and sensor nodes, and it usually includes the front-end users who possess mobile devices and the back-end cloud servers. By integrating the cloud computing and mobile devices, mobile cloud computing enables data users to access the large volume of cloud storage resources. Due to the limited resources of mobile devices, it is of prime importance to provide an efficient encrypted EHR search solution in mobile cloud computing.

In most of traditional SE schemes, the cloud server is always considered as a *honest-but-curious* entity which honestly follows the established protocols and returns the correct search results, but can deduce some sensitive information from access patterns and search patterns. However, in commercial cloud computing environment, the CSP is assumed to be a *semi-honest-but-curious* [14] third-party which may execute a fraction of search operations and return a fraction of false search results under various motivations. To overcome this problem, the PEKS schemes should be equipped with a verification mechanism such that the correctness of search results can be guaranteed without decrypting ciphertexts. Hence, various verifiable SE schemes [8,15–18] have been extensively studied among academical and industrial fields. Meanwhile, the computational overhead of results verification should be as small as possible such that the SE schemes can be widely used in mobile cloud computing.

Obviously, a practical search system in cloud has to support multiple data owners (or multi-owner) [19,20]. In fact, depending on whether each record is owned by a single party or co-owned by a fixed number of parties, multi-owner settings can be further classified into two scenarios, namely *unshared* multi-owner [19,20] and *shared* multi-owner [21,22], and most of the existing verifiable SE schemes only focus on the *unshared* multi-owner settings [18] rather than the *shared* one. Indeed, the *shared* multi-owner settings are more common in practical applications, especially in e-health systems. For example, each EHR is co-owned by a certain patient and several hospital staffs (i.e., surgeon, physician, chemist), and each of the staffs is responsible for the content of a particular part (i.e., block), as shown in Fig. 1. Notably, although the *unshared* multi-owner schemes can be applied if each block is viewed as an independent record, it will inevitably run into multiple indexes, each of which corresponds to a single block, such that the computational and space overhead is extremely expanded. Otherwise, the multiple data owners need to own the same random element, which is not feasible in the actual applications. Instead, a scheme specially designed for the *shared* multi-owner settings only requires a single index for the whole record, which significantly saves time and space costs.

On the other hand, to check the correctness of data stored in CSP, Wang et al. [23] proposed a public verification mechanism to audit the integrity of multi-owner data with multisignatures technique. Unfortunately, the scheme cannot guarantee data searchability. To the best of our knowledge, there are no schemes that support both searchability and verifiability in ciphertexts simultaneously under the *shared* multi-owner scenario shown in Fig. 1.

Moreover, a practical verifiable SE scheme should support conjunctive keyword search [24–26], as single keyword search [27,28] will return many irrelevant search results and definitely bring down user search experience.

In order to provide all aforementioned functionalities, we present a **Verifiable Conjunctive Keyword Search** scheme over mobile e-health cloud in the *shared* Multi-owner settings (**VCKSM**) by utilizing a third-party auditor [29,30]. Different from the previously proposed SE schemes, our scheme can resist the off-line keyword guessing attacks [31] in standard model. Specifically, our contributions can be summarized as follows:

- (1) Different from the traditional *unshared* multi-owner SE schemes, our scheme can support both results verification and conjunctive keyword search in the *shared* multi-owner settings.
- (2) We evaluate the performance of our scheme by using a real-world dataset and show its efficiency and feasibility in mobile cloud computing. The results verification overhead mainly depends on the number of search results rather than the whole data collection.
- (3) We formally prove that our scheme is secure against the off-line keyword guessing attacks in standard model.

**Table 1**  
Functional comparison.

Schemes	SRV	CKS	SMO
CP-VABKS [16]	✓	✗	✗
VCKS [17]	✓	✓	✗
ABKS-UR [18]	✓	✓	✗
Re-dtPECK [24]	✗	✓	✗
VCKSM	✓	✓	✓

“SRV”: Search Results Verification.

“CKS”: Conjunctive Keyword Search.

“SMO”: *Shared* multi-owner settings.

## 2. Related work

The SE technique enables CSP to provide fundamental information retrieval services for the cloud clients in a privacy-preserving way. Since Boneh et al. [11] proposed the first asymmetric SE scheme, abundant works [27,32] focusing on single keyword search have been proposed. However, these single keyword schemes suffer from a key limitation that too many undifferentiated and useless search results are returned, which also lead to the waste of bandwidth and computing resources. To address this limitation, Yang et al. [24] presented a novel conjunctive keyword search scheme which enabled data owner to delegate the flexible access permissions to data users with time enabled proxy re-encryption function. Xia et al. [33] demonstrated a multi-keyword (conjunctive and disconjunctive keyword) ranked search scheme by utilizing the widely-used vector space model and  $TF \times IDF$  model. Later, a considerable number of SE schemes focusing on enriching the search efficiency [34–36] have been proposed to improve user search experience. However, all the aforementioned schemes cannot verify the authenticity of search results in the *semi-honest-but-curious* cloud environment.

To the best of our knowledge, the most of existing SE schemes assume that the CSP is a honest-but-curious entity which will not deviate from the established protocols or return incorrect search results. However, such an assumption is usually insufficient in practice as the partial trust CSP is provided by the third-party and may intentionally return incorrect search results under various motivations. For instance, in order to save computing and bandwidth resources, the *semi-honest-but-curious* CSP may execute a part of search operations or return a fraction of false search results. To tackle this challenge, Zheng et al. [16] developed a verifiable keyword search scheme through attribute-based encryption (ABE) [37,38] and Bloom Filter, but this scheme will incur high false positive rate due to the inherent defect of Bloom filter. Later, Fu et al. [8] presented a semantic keyword search scheme which not only supported the verifiability of search results but also returned the results that semantically matched the submitted keyword, but one defect of this scheme was the high communication overhead. To boost search efficiency and gain a broad range of applications in practice, Sun et al. [17] constructed an efficient verifiable conjunctive keyword search scheme over large dynamic encrypted cloud data. However, these aforementioned schemes only support the single-contributor scenario where the outsourced data is encrypted and managed by a single data owner.

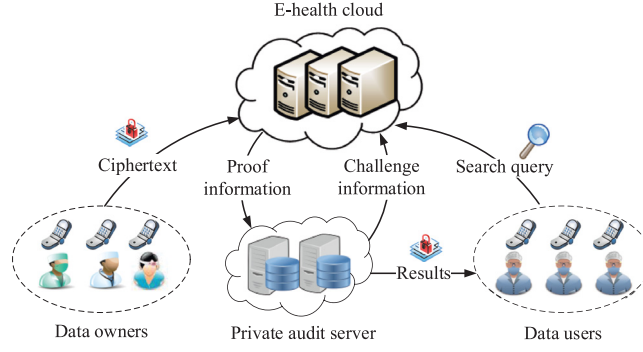
In practice, the CSP serves multiple data owners to share the benefits brought by mobile cloud computing. In the traditional *unshared* multi-owner settings, each data owner encrypts and manages his own data independently. That is, each data record stored in the cloud is owned by a single data owner. To cater for the multi-data source settings, Li et al. [19] proposed a fine-grained data access control in multi-owner settings scheme which can reduce key distribution complexity by leveraging ABE. Sun et al. [18] presented the first attribute-based keyword search scheme with user revocation and fine-grained search authorization. Zhang et al. [39] dealt with the problem of privacy-preserving ranked multi-keyword search in multi-owner model. However, these schemes only focus on the *unshared* multi-owner settings, where each data record is owned by different data owners. When applied in the *shared* multi-owner settings where a data record is kept by a group of data owners, these schemes will incur additional indexes and high computational overhead. Furthermore, these schemes cannot still guarantee the correctness of search results.

Notably, there are no existing schemes that can achieve both results verification and conjunctive keyword search in the *shared* multi-owner settings, as shown in the Table 1. From Table 1, we notice that the CP-VABKS [16] and Re-dtPECK [24] schemes just support Search Results Verification (SRV, for short) or Conjunctive Keyword Search (CKS, for short), respectively. Although the better VCKS [17] and ABKS-UR [18] schemes can simultaneously support SRV and CKS, these schemes will run into multiple indexes in the Shared Multi-Owner (SMO, for short) settings due to the different random elements. However, in our proposed scheme, the index building has nothing to do with the random elements selected by multiple data owners.

## 3. Preliminaries

In this section we introduce a group of cryptography concepts as the basic of VCKSM.

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 map  $G_1 \times G_1 \rightarrow G_2$ . Given a set  $X$ , the symbol  $x \in_R X$  is defined as choosing an element  $x$  uniformly at random from the set  $X$ ,  $[1, n]$  is denoted as a series of integers  $\{1, 2, \dots, n\}$ , where  $n$  is a non-zero integer.



**Fig. 2.** System model of our scheme.

**Definition 1 (DDH Problem).** Let  $G_1$  be a cyclic group of order  $p$ , and  $g$  be a generator of  $G_1$ . Given a tuple  $(g, g^a, g^b)$  and an element  $Z \in_R G_1$ , the **DDH (Decisional Diffie-Hellman)** problem is to decide whether  $Z$  equals to  $g^{ab}$  or to a random element in  $G_1$ .

**Definition 2 (CDH Assumption).** Let  $G_1$  be a cyclic group of order  $p$ ,  $g$  be a generator of  $G_1$ . Given the tuple  $g^a, g^b \in_R G_1$  and randomly selected elements  $a, b \in_R Z_p^*$ , it is computationally infeasible to compute  $g^{ab} \in_R G_1$  for any probabilistic polynomial time (PPT) adversary  $\mathcal{A}$  with a negligible advantage  $\epsilon$ , where the  $\mathcal{A}$ 's advantage in breaking the **CDH (Computational Diffie-Hellman)** problem is defined as  $Pr[\mathcal{A}_{CDH}(g, g^a, g^b) = g^{ab}] < \epsilon$ .

**Definition 3 (DL Assumption).** Let  $G_1$  be a group of order  $p$ , and  $g$  be the generator of  $G_1$ . For any PPT adversary  $\mathcal{A}$ , its advantage in solving the **DL (Discrete Logarithm)** problem in group  $G_1$  is negligible, which is defined as  $Pr[\mathcal{A}(g, g^a) = a] < \epsilon$ , where  $a \in_R Z_p^*$ .

## 4. Problem formulation

### 4.1. System model and threat model

In this paper we consider an mobile e-health cloud system which mainly involves four entities, namely **multiple data owners** (patient and his doctors), Cloud Service Provider (**CSP**), **data user** (other authorized doctors or healthcare center) and Private Audit Server (**PAS**), as demonstrated in Fig. 2, where the unmentioned Trusted Third-Party (TTP) server is in charge of the system initialization and key generation. Once the system is set up, the TTP does not need to stay online. The role of each main entity is shown as follows:

- **Multiple data owners:** For each EHR owned by a specific patient and his doctors, it is extracted with keywords and is built with index, and then each of its data owners generates a signature on this record, finally the indexes and signatures are sent to CSP. Note that the detailed algorithm used to encrypt each EHR is beyond the scope of our discussion, any public key encryption algorithm can be applied.
- **CSP:** The cloud server that has expertise and capabilities can provide data storage and retrieval services for the authorized cloud clients (data owner and data user), while it is curious to deduce the sensitive information available to it and selfish to return false search results under various motivations.
- **Data user:** Once gaining the authorization from the data owners, he can issue search query for his interested EHRs by submitting his trapdoor (search token) to CSP.
- **PAS:** This fully trusted server is responsible for verifying the correctness of search results.

Similar to the literature [14], the CSP is assumed to be *semi-honest-but-curious*. That is, it honestly performs a fraction of search operations but is curious to spy out the sensitive information that is valuable for the CSP. Furthermore, it may return the false search results to save the computing resources. On the other hand, the PAS is fully-trusted and able to guarantee the correctness of search results. Besides, the authorized data user can issue search queries without leaking any valuable information to CSP.

### 4.2. Design goals

Our scheme for mobile e-health cloud is designed to achieve the following goals:

- **Results verification.** As the *semi-honest-but-curious* CSP may return the false search results, which compromises data security and seriously affects the search experience, our scheme should be equipped with a results verification mechanism to ensure the correctness of search results.
- **Conjunctive keyword search.** To search the EHRs quickly and avoid the waste of bandwidth and computing resources, our scheme should enable data users to submit multiple keywords in a single search query.
- **Efficiency and feasibility.** To support a broad range of applications in mobile cloud computing, the performance of our scheme over a real-world dataset should be efficient and feasible without incurring extra computational burden on the resource constrained data users.
- **Security goals.** For the security concerns, our scheme should guarantee both data security without loss of confidentiality. Besides, as the keywords are always selected from a small space and random oracle has its inherent defects, our scheme should resist the keyword guessing attacks in standard model. Furthermore, our proposed scheme can correctly check the correctness of search results for the *semi-honest-but-curious* CSP.

### 4.3. Security model

As the keyword space is always small, the outside attack may utilize this weakness to exhaustively guess some candidate keywords and verify whether his guess is right or not. Moreover, the trapdoor indistinguishability is a sufficient condition for guaranteeing the security against the off-line keyword guessing attacks. That is, the outside attacker who has gained the trapdoors for challenged keywords still cannot deduce the relationship between the trapdoors and any keyword.

**Definition 4 (IND-KGA Security).** If there is no adversary  $\mathcal{A}$  that can break our scheme with a negligible advantage  $Adv_{\mathcal{A}}^{IND-KGA}(1^k)$ , then our scheme is secure against the keyword guessing attacks.

Let  $\mathcal{A}$  be an outside adversary which can issue KGA, and  $k$  be the security parameter, then the **IND-KGA (Indistinguishability against Keyword Guessing Attacks)** [31] game between the challenger  $\mathcal{B}$  and  $\mathcal{A}$  is shown as follows:

- **Setup:**  $\mathcal{B}$  first calls the **Setup**( $1^k$ ) algorithm and **KeyGen**( $pk, \mathcal{U}$ ) algorithm to generate the public/secret key pairs  $\{(pk_u, sk_u), (pk_s, sk_s)\}$  for a specific data user  $u$  and the CSP, respectively. Then,  $\mathcal{B}$  sends  $pk_u, pk_s$  to  $\mathcal{A}$ , while  $sk_u, sk_s$  are unknown to  $\mathcal{A}$ .
- **Query phase 1:**  $\mathcal{A}$  issues the following queries:
  - **TrapGen** oracle:  $\mathcal{A}$  can adaptively ask  $\mathcal{B}$  for the search token  $T_{W'}$  of any keyword set  $W'$  of his interest, and  $\mathcal{B}$  responds to it by performing **TrapGen**( $pk, pk_s, sk_u, W'$ ) algorithm.
- **Challenge:**  $\mathcal{A}$  submits two keyword sets  $W'_0, W'_1$  on which it wishes to be challenged, but the restriction is that  $T_{W'_0}$  and  $T_{W'_1}$  have not been queried by  $\mathcal{A}$  in **Query phase 1**. Then  $\mathcal{B}$  selects a random bit  $b \in \{0, 1\}$  and returns the trapdoor  $T_{W'_b}$ .
- **Query phase 2:**  $\mathcal{A}$  still makes a number of search queries to the **TrapGen** oracle as in **Query phase 1**, but there is a restriction that  $W'_0, W'_1$  are not queried to **TrapGen** oracle.
- **Guess.**  $\mathcal{A}$  outputs his guess bit  $b' \in \{0, 1\}$  and wins the IND-KGA game if  $b' = b$ ; otherwise, it fails.

$\mathcal{A}$ 's advantage in breaking the IND-KGA game is denoted as  $Adv_{\mathcal{A}}^{IND-KGA}(1^k) = |\Pr[b' = b] - \frac{1}{2}|$ , and our scheme is secure against the IND-KGA if the advantage  $Adv_{\mathcal{A}}^{IND-KGA}(1^k)$  is negligible.

## 5. The proposed VCKSM

In this section, we first formally give the definitions of VCKSM scheme as well as some notations used in our scheme, as shown in the [Table 2](#). Afterwards, we describe the concrete construction of our VCKSM scheme in detail.

### 5.1. Definition of VCKSM

Our scheme is a tuple of six algorithms which are shown as follows:

- **Setup**( $1^k$ )  $\rightarrow pk$ : Given the security parameter  $k$ , TTP outputs the public parameters  $pk$ .
- **KeyGen**( $pk, \mathcal{U}, \mathcal{O}$ )  $\rightarrow \{pk_s, sk_s, pk_u, sk_u, pk_t, sk_t\}$ : For the data owner set  $\mathcal{O}$ , data user set  $\mathcal{U}$  and cloud server, TTP generates the public/secret key pairs  $\{pk_t, sk_t\}$  ( $1 \leq t \leq d$ ),  $\{pk_u, sk_u\}$  and  $(pk_s, sk_s)$ , respectively, where  $u \in \mathcal{U}$ .
- **Enc**( $pk, F, W, ID, \{sk_t\}, pk_s, pk_u$ )  $\rightarrow \{I, C, Sig\}$ : Given the files  $F$  and keyword fields  $W$ , the data owners output the ciphertexts  $C$ , indexes  $I$  and signatures  $Sig$ .
- **TrapGen**( $pk, pk_s, sk_u, W'$ )  $\rightarrow T_{W'}$ : Given the queried keyword set  $W'$ , a specific data user generates the search token  $T_{W'}$ .
- **Search**( $pk, T_{W'}, I, C, sk_s$ )  $\rightarrow C'$ : After gaining the search token  $T_{W'}$ , the CSP tries to find a match within the index set  $I$  and returns the relevant ciphertext set  $C'$  to PAS.
- **Verify**( $pk, C', ID', \{pk_t\}$ )  $\rightarrow \{0, 1\}$ : After receiving the search results  $C'$ , the PAS outputs "1" when  $C'$  passes the results verification mechanism; otherwise, it outputs "0".

**Table 2**  
Notation descriptions.

Notations	Descriptions
$\mathcal{O} = \{O_1, \dots, O_d\}$	Multiple data owners for each record
$(pk_s, sk_s)$	Public/secret key pair for CSP
$(pk_t, sk_t)$	Public/secret key pair for data owner
$(pk_u, sk_u)$	Public/secret key pair for data user
$F = \{f_1, \dots, f_n\}$	Record set
$ID = \{id_1, \dots, id_n\}$	Identity set for $F$
$C = \{c_1, \dots, c_n\}$	Ciphertext set for $F$
$W = \{w_1, \dots, w_m\}$	Keyword fields for each record
$sig_{i,t}$	Data owner $O_t$ ' signature for $f_i$
$sig_i$	Data owners' multisignatures for $f_i$
$Sig = \{sig_1, \dots, sig_n\}$	Signature set for $F$
$I_i$	Index for $f_i$
$I = \{I_1, \dots, I_n\}$	Index set for $F$
$W' = \{w'_1, \dots, w'_l\}$	Queried keyword set
$T_{W'}$	Search token for $W'$
$C' = \{c'_1, \dots, c'_q\}$	Search results
$ID' = \{id'_1, \dots, id'_q\}$	Identity set for $C'$
$\{\tau, \pi_\tau\}_{\tau \in [1,q]}$	Challenging information
$(\varphi, \sigma)$	Proof information

## 5.2. Construction of VCKSM

In the EHR system, each EHR can be encrypted by the traditional public key encryption algorithm, which is beyond the scope of our discussion. The algorithms in the following mainly focus on how to build indexes and generate signatures for EHRs so that the conjunctive keyword search query from a specific data user can be processed efficiently in the cloud system and the PAS can check the correctness of the search results, respectively. For ease of the following discussion, we define a function  $\rho(\cdot)$  which maps the subscript in the set  $[1, q]$  to its corresponding subscript in the set  $[1, n]$ , namely  $\rho(\tau)|_{\tau \in [1,q]} \in [1, n]$ .

**Setup**( $1^k$ ): Given the security parameter  $k$ , TTP first outputs the public bilinear map parameters  $(G_1, G_2, e, p, g)$ . Then, it selects two hash functions  $h_1 : \{0, 1\}^* \rightarrow G_1$ ,  $h_2 : \{0, 1\}^* \rightarrow Z_p^*$ . Finally, it sets the public parameters  $pk$  as  $pk = \{G_1, G_2, e, p, g, h_1, h_2\}$ .

**KeyGen**( $pk, \mathcal{U}, \mathcal{O}$ ): According to our system model defined in Section 4, each EHR is owned by a fixed number of data owners, namely  $\mathcal{O} = \{O_1, \dots, O_d\}$ , then TTP generates the public/secret key pairs for CSP, data owner set  $\mathcal{O}$  and data user set  $\mathcal{U}$ , respectively.

For CSP, it first selects two elements  $\alpha \in_R G_1$ ,  $a_s \in_R Z_p^*$  and computes  $\beta = g^{a_s}$ , then the CSP's public/secret key pair is denoted as  $pk_s = (\alpha, \beta)$ ,  $sk_s = a_s$ . For each data owner  $O_t \in \mathcal{O}$  ( $t \in [1, d]$ ), it chooses an element  $x_t \in_R Z_p^*$  and computes  $X_t = g^{x_t}$ , then it sets the public/secret key pair of  $O_t$  as  $pk_t = g^{x_t}$ ,  $sk_t = x_t$ . While for a specific data user  $u \in \mathcal{U}$  who is authorized by  $\mathcal{O}$ , the generation of public/secret key pair is similar to each data owner. This algorithm first selects an element  $y \in_R Z_p^*$  and computes  $Y = g^y$ , then it sets the public/secret key pair of the specific data user  $u$  as  $pk_u = Y$ ,  $sk_u = y$ .

**Enc**( $pk, F, W, ID, \{sk_t\}, pk_s, pk_u$ ): Given the EHR set  $F = \{f_1, \dots, f_n\}$  with corresponding identities  $ID = \{id_1, \dots, id_n\}$ , it will be encrypted as the ciphertext set  $C = \{c_1, \dots, c_n\}$ . For each EHR  $f_i \in F$  ( $i \in [1, n]$ ) with identity  $id_i$ , each data owner  $O_t \in \mathcal{O}$  generates his signature  $sig_{i,t} = (h_1(id_i)g^{h_2(c_i)})^{x_t}$ , and the multisignatures generated by multiple data owners are set as  $sig_i = \prod_{t=1}^d sig_{i,t}$ , where  $t \in [1, d]$ .

Besides, the index for each EHR  $f_i$  is generated according to the keyword fields  $W = \{w_1, \dots, w_m\}$ . First, a  $m$ -degree polynomial should be constructed by the equation  $\mathcal{F}(x) = b_m x^m + b_{m-1} x^{m-1} + \dots + b_1 x + b_0$  so that  $y h_2(w_1), \dots, y h_2(w_m)$  are the  $m$  roots of the equation  $\mathcal{F}(x) = 1$ . Then, select  $\lambda, \mu \in_R Z_p^*$  and compute  $I_{i,1} = \gamma \cdot e(g, g)^{-\mu}$ ,  $I_{i,2} = g^\lambda$ ,  $v_{i,j} = g^{\mu \cdot b_j}$  ( $0 \leq j \leq m$ ), where  $\gamma = e(\beta, \alpha)^\lambda$ .

Finally, data owners' signature set  $Sig = \{sig_1, \dots, sig_n\}$ , index set  $I = \{I_1, \dots, I_n\}$  and the ciphertext set  $C$  are sent to CSP, where  $I_i = \{I_{i,1}, I_{i,2}, v_{i,0}, v_{i,1}, \dots, v_{i,m}\}$ .

**TrapGen**( $pk, pk_s, sk_u, W'$ ): Given the queried keywords set  $W' = \{w'_r\}$  ( $r \in [1, l]$ ), a specific data user  $u$  first selects two elements  $\theta, \eta \in_R Z_p^*$  and sets  $T_{W',0_1} = \theta$ . Then, he computes  $T_{W',0_2} = g^\eta$ ,  $T_{W',j} = g^{T_{W',0_1} \cdot y^j \cdot \sum_{r=1}^l h_2(w'_r)^j} \cdot \beta^\eta$ , where  $0 \leq j \leq m$ . Finally, he sends the search token (trapdoor)  $T_{W'} = \{T_{W',0_1}, T_{W',0_2}, T_{W',0}, T_{W',1}, \dots, T_{W',m}\}$  to CSP.

**Search**( $pk, T_{W'}, I, C, sk_s$ ): After gaining the search token  $T_{W'}$ , the CSP first computes  $\gamma = e(I_{i,2}, \alpha)^{a_s}$  and verifies whether Eq. (1) holds or not. If Eq. (1) (described in the following) holds, the CSP returns the relevant ciphertext set  $C' = \{c'_1, \dots, c'_q\}$  and its corresponding identity set  $ID' = \{id'_1, \dots, id'_q\}$  to PAS; otherwise, it returns  $\perp$ . The specific search process is shown in the Algorithm 1. The specific search process is shown in the Algorithm 1. At the beginning, given the trapdoor  $T_{W'}$  and the index  $I_i$  for each record  $f_i$  ( $1 \leq i \leq n$ ), the CSP preprocesses the search query with performing  $m$  exponentiation operations (Lines 4-6). Afterwards, the CSP checks whether the submitted trapdoor matches with the index  $I_i$  by checking the Eq. (1).



---

**Algorithm 1** Search over encrypted data

---

**Input:** Trapdoor  $T_{W'}$ , indexes  $I$ , ciphertexts  $C$ , secret key  $sk$ , and public parameters  $pk$ .

**Output:** Search results  $C'$  with corresponding identity set  $ID'$  or  $\perp$ .

- 1:  $T_{W'} = \{T_{W',o_1}, T_{W',o_2}, T_{W',0}, T_{W',1}, \dots, T_{W',m}\}$ ;
  - 2:  $I = \{I_1, \dots, I_n\}, I_i = \{I_{i,1}, I_{i,2}, v_{i,0}, v_{i,1}, \dots, v_{i,m}\}$ ;
  - 3: **for**  $1 \leq i \leq n$  **do**
  - 4:   **for**  $0 \leq j \leq m$  **do**
  - 5:     Compute  $\prod_{j=0}^m e(v_{i,j}, T_{W',j}/T_{W',o_2}^{a_s})$ ;
  - 6:   **end for**
  - 7:   Check  $I_{i,1}^{T_{W',o_1}} \cdot \prod_{j=0}^m e(v_{i,j}, T_{W',j}/T_{W',o_2}^{a_s}) \stackrel{?}{=} \gamma^{T_{W',o_1}}(1)$ ;
  - 8:   If Eq.1 holds, CSP returns the ciphertext  $c_i$ ; otherwise, it returns  $\perp$ ;
  - 9: **end for**
  - 10: CSP returns the relevant results  $C'$  with corresponding identity set  $ID'$  or  $\perp$  to PAS.
- 

**Algorithm 2** Search results verification

---

**Input:** Search results  $C'$  with corresponding identity set  $ID'$ , public keys  $\{pk_t\}$  and public parameters  $pk$ .

**Output:** "Accept" or "Reject".

- 1:  $C' = \{c'_1, \dots, c'_q\}, ID' = \{id'_1, \dots, id'_q\}$ ;
  - 2:  $\{pk_1, \dots, pk_d\}$ ;
  - 3: PAS sends the challenging information  $\{\tau, \pi_\tau\}_{\tau \in [1,q]}$  to CSP;
  - 4: **for**  $1 \leq \tau \leq q$  **do**
  - 5:   Compute  $\varphi = \sum_{\tau=1}^q \pi_\tau h_2(c'_\tau)$ ;
  - 6:   Compute  $\sigma = \prod_{\tau=1}^q (sig'_\tau)^{\pi_\tau}$ ;
  - 7:   Send  $(\varphi, \sigma)$  to PAS;
  - 8:   Compute  $\prod_{\tau=1}^q h_1(id'_\tau)^{\pi_\tau}$ ;
  - 9: **end for**
  - 10: **for**  $1 \leq t \leq d$  **do**
  - 11:   Compute  $\prod_{t=1}^d pk_t$
  - 12: **end for**
  - 13: Check  $e(\sigma, g) \stackrel{?}{=} e(\prod_{\tau=1}^q h_1(id'_\tau)^{\pi_\tau} \cdot g^\varphi, \prod_{t=1}^d pk_t)(2)$ ;
  - 14: If Eq.2 holds, output "Accept" and send  $C'$  to data user; otherwise, output "Reject".
- 

If the Eq. (1) holds, CSP returns the corresponding ciphertext  $c_i$ ; otherwise, he returns  $\perp$  (Lines 7-8). Finally, the CSP returns the whole relevant ciphertext sets  $C' = \{c'_1, \dots, c'_q\}$  and its corresponding identity set  $ID' = \{id'_1, \dots, id'_q\}$  or  $\perp$  (Lines 9-10) to PAS. As the value of  $m$  is very small in practice, the Algorithm 1 will not exert heavy computational burden for CSP. Thus, the search algorithm is feasible and practical in actual scenarios.

$$I_{i,1}^{T_{W',o_1}} \cdot \prod_{j=0}^m e(v_{i,j}, T_{W',j}/T_{W',o_2}^{a_s}) = \gamma^{T_{W',o_1}}. \quad (1)$$

**Verify**( $pk, C', ID', \{pk_t\}$ ): After receiving the search results  $C'$ , the PAS first selects an element  $\pi_\tau \in_R Z_p^*$  for each encrypted EHR  $c'_\tau (\tau \in [1, q])$ . Then, it sends the challenging information  $\{\tau, \pi_\tau\}_{\tau \in [1,q]}$  to CSP. After gaining the challenging information, the CSP first computes  $\varphi = \sum_{\tau=1}^q \pi_\tau h_2(c'_\tau)$  and  $\sigma = \prod_{\tau=1}^q (sig'_\tau)^{\pi_\tau}$ . Then, it sends the proof information  $(\varphi, \sigma)$  to PAS, where  $sig'_\tau = \prod_{t=1}^d sig_{\rho(\tau),t}$ . Finally, the PAS verifies whether Eq. (2) holds.

$$e(\sigma, g) = e\left(\prod_{\tau=1}^q h_1(id'_\tau)^{\pi_\tau} \cdot g^\varphi, \prod_{t=1}^d pk_t\right). \quad (2)$$

In the above equation,  $id'_\tau = id_{\rho(\tau)}$ ,  $c'_\tau = c_{\rho(\tau)}$ . If Eq. (2) holds, the PAS justifies that the search results  $C'$  are correct and sends them to the specific data user  $u$ ; otherwise, it aborts. The detailed process of results verification can be found in Algorithm 2. At the beginning, the PAS interacts with the CSP based on the challenge-response mode (Lines 3-9). Afterwards, the PAS computes  $\prod_{t=1}^d pk_t$  and verifies whether the search results  $C'$  is correct or not (Lines 10-13). Finally, the PAS draws the conclusion and returns the correct search results to the specific data user (Line 14).

**Remark:** If the specific data user is authorized and his queried keywords set satisfies  $W' \subseteq W$ , the CSP will return the relevant search results  $C'$  to PAS. And the search results  $C'$  can pass the results verification mechanism only if the CSP does not tamper or forge incorrect search results.

## 6. Analysis of our scheme

In this section, we present the analysis our scheme in terms of correctness, security and performance, respectively.

## 6.1. Correctness

In our VCKSM scheme, the specific data users can gain the desired search results if and only if his submitted keyword set satisfies  $W' \subseteq W$  and its corresponding search token matches with the indexes. In addition, the data users can be assured of the correctness of search results if the search results pass the results verification mechanism performed by PAS.

**Theorem 1.** For any security parameter  $k$ , the proposed VCKSM scheme satisfies

$$\text{Search} \rightarrow \begin{cases} 1, & W' \subseteq W \\ 0, & \text{Otherwise} \end{cases}, \text{Verify} \rightarrow \begin{cases} 1, & C' \subseteq C \\ 0, & \text{Otherwise} \end{cases}.$$

**Proof 1.** For Eq. (1), we first have

$$I_{i,1}^{T_{W',o_1}} = e(\beta, \alpha)^{\lambda\theta} \cdot e(g, g)^{-\mu\theta}.$$

Based on  $\mathcal{F}(x)$ , then

$$\begin{aligned} \prod_{j=0}^m e(v_{i,j}, \frac{T_{W',j}}{T_{W',o_2}^{a_s}}) &= \prod_{j=0}^m e(g^{\mu \cdot b_j}, g^{l^{-1} \cdot \theta \cdot y^j \cdot \sum_{r=1}^l h_2(w_r^j)}) \\ &= e(g^\mu, g^{l^{-1}\theta})^{\sum_{j=0}^m b_j \cdot y^j \cdot \sum_{r=1}^l h_2(w_r^j)} \\ &= e(g, g)^{\mu\theta}. \end{aligned}$$

Thus, we finally check that Eq. (1) holds when  $W' \subseteq W$ .

$$I_{i,1}^{T_{W',o_1}} \cdot \prod_{j=0}^m e(v_{i,j}, T_{W',j}/T_{W',o_2}^{a_s}) = e(\beta, \alpha)^{\lambda\theta} = \gamma^{T_{W',o_1}}.$$

For Eq. (2), we can check that it holds when  $id'_\tau = id_{\rho(\tau)}$ ,  $c'_\tau = c_{\rho(\tau)}$ .

$$\begin{aligned} e(\sigma, g) &= e\left(\prod_{\tau=1}^q \prod_{t=1}^d \text{sig}_{\rho(\tau),t}^{\pi_\tau}, g\right) \\ &= e\left(\prod_{\tau=1}^q h_1(id'_\tau)^{\pi_\tau} \cdot g^{\sum_{\tau=1}^q h_2(c'_\tau)\pi_\tau}, \prod_{t=1}^d pk_t\right) \\ &= e\left(\prod_{\tau=1}^q h_1(id'_\tau)^{\pi_\tau} \cdot g^\varphi, \prod_{t=1}^d pk_t\right). \end{aligned}$$

## 6.2. Security

For security issue, IND-KGA, as known to all, can ensure that the outsider attacker cannot infer the relationship between the target trapdoor and challenging keyword set even though it can gain other trapdoors, and the IND-KGA security can be guaranteed by our VCKSM scheme, which is shown as below.

**Theorem 2.** Our scheme is secure against IND-KGA in standard model on the condition that DDH problem is intractable (Definition 1).

**Proof 2.** If there is a PPT adversary  $\mathcal{A}$  that can break the IND-KGA security of our scheme with the probability  $(\vartheta, \varepsilon)$ , where  $\vartheta, \varepsilon$  are denoted as the running time and advantage of  $\mathcal{A}$  in breaking the IND-KGA security, respectively. Then we construct a PPT adversary  $\mathcal{B}$  to break the IND-KGA security of our scheme with the probability  $(\vartheta', \varepsilon')$ , where  $\varepsilon' \geq \varepsilon, \vartheta' \geq \vartheta + t_e(2m+3)q_t$ ,  $t_e$  is the running time of an exponentiation,  $q_t$  is the number of trapdoor queries.

Assume that  $\mathcal{A}$  has given a tuple  $(g, g^{\delta_1}, g^{\delta_2}, Z)$  as an instance for the DDH problem, where  $Z$  is equal to  $g^{\delta_1 \cdot \delta_2}$  or a random element in  $G_1$ , then the security game between  $\mathcal{A}$  and  $\mathcal{B}$  is shown as follows:

- **Setup:**  $\mathcal{B}$  first selects  $\alpha \in_R Z_p^*$  and computes  $\beta = g^{\delta_2}$ , the CSP's public/secret key pair is denoted as  $pk_s = (\alpha, \beta)$ ,  $sk_s = \delta_2$ , respectively. Then,  $\mathcal{B}$  chooses  $y \in_R Z_p^*$  and computes  $Y = g^y$ , the public/secret key pair of a specific data user  $u$  is set as  $pk_u = Y, sk_u = y$ . Finally,  $\mathcal{B}$  sends the tuple  $(pk_s, pk_u)$  to  $\mathcal{A}$ .
- **Query phase 1:**  $\mathcal{A}$  issues the following queries:

- **TrapGen oracle:**  $\mathcal{A}$  first adaptively issues the search token query for the submitted keyword set  $W'_i = \{w'_{i,1}, \dots, w'_{i,l}\} (l \leq m)$ , then  $\mathcal{B}$  selects  $\theta, \eta \in_R Z_p^*$  and computes  $T_{W'_i,0,1} = \theta, T_{W'_i,0,2} = g^\eta, T_{W'_i,j} = g^{l^{-1} \cdot T_{W'_i,0,1} \cdot \sum_{r=1}^l h_2(w'_{i,r})^j} \cdot \beta^\eta$ , where  $0 \leq j \leq m$ .

- **Challenge:**  $\mathcal{A}$  outputs two keyword sets  $W'_0, W'_1$  on which to be challenged,  $\mathcal{B}$  first selects a random bit  $b \in \{0, 1\}$  and an element  $\theta^* \in_R Z_p^*$ . Then, he sets  $T_{W'_b,0,1} = \theta^*, T_{W'_b,0,2} = g^{\delta_1}, T_{W'_b,j} = g^{l^{-1} \cdot T_{W'_b,0,1} \cdot \sum_{r=1}^l h_2(w'_b)^j} \cdot Z$ . Finally,  $\mathcal{B}$  sends the trapdoor  $T_{W'_b} = (T_{W'_b,0,1}, T_{W'_b,0,2}, \{T_{W'_b,j}\}_{0 \leq j \leq m})$  to  $\mathcal{A}$ . Set  $\eta^* = \delta_1$ , if  $Z = g^{\delta_1 \cdot \delta_2}$ , then  $T_{W'_b,0,2} = g^{\eta^*}, T_{W'_b,j} = g^{l^{-1} \cdot T_{W'_b,0,1} \cdot \sum_{r=1}^l h_2(w'_b)^j} \cdot \beta^{\eta^*}$ . Therefore,  $T_{W'_b}$  is the valid search token for the keyword set  $W'_b$ .
- **Query phase 2:**  $\mathcal{A}$  repeats the process in **Query phase 1**, but there is one restriction that  $W'_0, W'_1$  have not been queried.
- **Guess:**  $\mathcal{A}$  outputs a random bit  $b' \in \{0, 1\}$ . If  $b' = b$ ,  $\mathcal{B}$  outputs "1" meaning  $Z = g^{\delta_1 \cdot \delta_2}$ ; otherwise, it returns "0", which means that  $Z$  is a random element in  $G_1$ .

In the **Guess** phase, if  $\mathcal{A}$  can break our scheme with an advantage  $\varepsilon$ , then the term  $g^{\delta_1 \cdot \delta_2}$  will appear in the tuple  $(g, g^{\delta_1}, g^{\delta_2}, g^{\delta_1 \cdot \delta_2})$  with the probability  $\frac{1}{2} + \varepsilon$  at least. In other words,  $\mathcal{B}$ 's advantage in breaking the DDH problem is at least  $\varepsilon' \geq \varepsilon$ . And  $\mathcal{B}$ 's execution time  $\vartheta' \geq \vartheta + t_e(2m + 3)q_t$  of this simulation is mainly decided by the exponent operations in the query phase. This completes the proof of [Theorem 2](#).

In addition, the malicious CSP cannot forge the valid proof information to pass the results verification mechanism, as shown in [Theorem 3](#).

**Theorem 3.** For CSP, it is computationally infeasible to forge the valid proof information to pass the results verification mechanism under the CDH and DL assumptions ([Definitions 2](#) and [3](#)).

**Proof 3.** To forge the valid proof information, the malicious adversary can achieve it with the following two ways:

- The adversary first forges valid multisignatures on each returned record on behalf of multiple data owners, then it generates the valid proof information on search results according to the generated multisignatures. However, it is computationally infeasible to forge valid multisignatures based on the CDH assumption as it does not have the secret keys of multiple data owners, of which the detailed proof is demonstrated in the literature [[23](#)]. Thus, it is computationally infeasible to forge the valid proof information in this way.
- The adversary can directly generate the valid proof information based on the incorrect search results  $C^*$  without forging multisignatures by winning the following security game.

First, the PAS sends the challenging information  $\{\tau, \pi_\tau\}_{\tau \in [1, q]}$  to CSP. The proof information on correct search results  $C'$  should be  $(\varphi, \sigma)$ , while the malicious CSP may forge the proof information  $(\varphi', \sigma)$  on false search results  $C^*$ , where  $\varphi' = \sum_{\tau=1}^q \pi_\tau h_2(c_\tau^*)$ ,  $C^* = \{c_\tau^*\}_{\tau \in [1, q]}$ . Set  $\Delta\varphi = \varphi' - \varphi$ , then  $\Delta\varphi \neq 0$  as  $\varphi' \neq \varphi$  and the tuple  $\{\pi_\tau\}_{\tau \in [1, q]}$  is randomly selected in field  $Z_p^*$ . If the forged proof information  $(\varphi', \sigma)$  can successfully pass the results verification mechanism, the malicious CSP wins this security game; otherwise, it fails.

Assume that the CSP wins the security game, then we can get  $e(\sigma, g) = e(\prod_{\tau=1}^q h_1(id'_\tau)^{\pi_\tau} \cdot g^{\varphi'}, \prod_{t=1}^d pk_t)$ . In addition, we can also have  $e(\sigma, g) = e(\prod_{\tau=1}^q h_1(id'_\tau)^{\pi_\tau} \cdot g^\varphi, \prod_{t=1}^d pk_t)$  based on the proof information on the correct search results  $C'$ . Finally, we can further gain  $g^{\varphi'} = g^\varphi \Leftrightarrow g^{\Delta\varphi} = 1$ .

In the cyclic group, given two elements  $\xi, \varpi \in G_1$ , there is at least an element  $z \in_R Z_p^*$  such that  $\varpi = \xi^z$ . Without loss of generality, the generator  $g$  can be expressed as  $g = \xi^{\varrho_1} \pi^{\varrho_2}$  through  $\xi, \varpi$ , where  $\varrho_1, \varrho_2 \in_R Z_p^*$ . According to the equation  $g^{\Delta\varphi} = 1$  we can get  $(\xi^{\varrho_1} \pi^{\varrho_2})^{\Delta\varphi} = 1 \Leftrightarrow \xi^{\varrho_1 \Delta\varphi} \cdot \pi^{\varrho_2 \Delta\varphi} = 1$ . Therefore, given  $\xi, \xi^z$  we can break the DL problem. This is because we can have  $\pi = \xi^{-\frac{\varrho_1 \Delta\varphi}{\varrho_2 \Delta\varphi}} = \xi^z$  unless  $\varrho_2 \Delta\varphi \neq 0$ , where  $z = -\frac{\varrho_1 \Delta\varphi}{\varrho_2 \Delta\varphi}$ . However, we know that  $\Delta\varphi \neq 0$  and  $\varrho_2$  is a random element. And the equation  $\varrho_2 \Delta\varphi = 0$  holds with only a probability of  $\frac{1}{p}$ . Hence, we can break the DL problem with a probability of  $1 - \frac{1}{p}$ , where  $p$  is a large prime. That is to say, if the malicious CSP can win the security game, we can solve the DL problem, which conflicts with the DL assumption in [Definition 3](#). This completes the proof of [Theorem 3](#).

### 6.3. Performance

In this section, we present the performance evaluation regarding the theoretical and practical analysis of our scheme by comparing with the state-of-the-art ABKS-UR [[18](#)] and CP-VABKS [[16](#)] schemes. As for the theoretical analysis, we mainly analyze the storage costs and computational complexities of the aforementioned three schemes. After that, we conduct a series of experiments using a real-world dataset to show the efficiency and feasibility of our proposed scheme.

Let  $|G_1|, |G_2|, |Z_p|$  be the bit-length of an element in groups  $G_1, G_2$  and field  $Z_p$ , respectively. It is worth noticing that the value of keyword fields ( $m$ ) in EHR is not more than the number of attributes ( $\mathcal{N}$ ) in the practical applications. From [Table 3](#), we notice that our scheme has less storage cost than the other two schemes for the **Trap** and **Search** algorithms. Besides, our scheme also has less storage cost than other two schemes for **KeyGen** algorithm on condition that the number of data

**Table 3**

Storage cost in various schemes.

Schemes	KeyGen	Trap	Search	Verify
ABKS-UR [18]	$(\mathcal{U} + 1) Z_p  + (2\mathcal{N} + 1) G_1  + \mathcal{U} G_2 $	$(2\mathcal{N} + 1) G_1  + 2 Z_p $	$(\mathcal{N} + 2) G_2 $	–
CP-VABKS [16]	$(\mathcal{N} + 1) Z_p  + (2\mathcal{N} + 1) G_1 $	$(2\mathcal{N} + 3) G_1  +  Z_p $	$(\mathcal{N} + 3) G_2 $	–
VCKSM	$(d + \mathcal{U} + 1)( Z_p  +  G_1 ) +  G_1 $	$(m + 2) G_1  + 2 Z_p $	$(m + 3) G_2  +  G_1 $	$(q + 1) Z_p  +  G_1  + 2 G_2 $

“ $\mathcal{N}$ ”: Number of attributes; “ $\mathcal{U}$ ”: Number of data users; “ $d$ ”: Number of data owners.

“ $q$ ”: Number of search results; “ $m$ ”: Number of keyword fields; “–”: Unconsidered.

**Table 4**

Computational complexity in various schemes.

Schemes	KeyGen	Enc	Trap	Search	Verify
ABKS-UR [18]	$(2\mathcal{N} + 1)E_1 + 2\mathcal{U}E_2$	$(\mathcal{N} + 1)E_1 + E_2$	$(2\mathcal{N} + 1)E_1$	$(\mathcal{N} + 1)P + E_1$	–
CP-VABKS [16]	$(2\mathcal{N} + 2)E_1 + \mathcal{N}h_1$	$(2\mathcal{N} + 4)E_1 + \mathcal{N}h_1$	$(2\mathcal{N} + 4)E_1$	$(2\mathcal{N} + 3)P + \mathcal{N}E_1$	–
VCKSM	$(\mathcal{U} + d + 1)E_1$	$(m + 2 + 2d)E_1 + h_1 + 2P + 2E_2$	$(m + 2)E_1$	$(m + 2)P + E_1 + 3E_2$	$(2q + 1)E_1 + qh_1 + 2P$

users is less than 100. As the other two schemes will incur high false positive rate caused by the bloom filter, we just analyze the storage cost for **Verify** algorithm in our scheme. Furthermore, our scheme does not yield much more storage cost for the **Verify** algorithm due to the small value of  $q$ . Thus, our scheme is suitable for the lightweight entities.

For the computational complexity, we mainly consider several time-consuming operations, i.e., exponentiation operation  $E_1$  (or  $E_2$ ) in group  $G_1$  (or  $G_2$ ), pairing operation  $P$ , hash operation  $h_1$  which maps an arbitrary bit string to group  $G_1$ . In Table 4 we give the computational complexity of our VCKSM scheme by comparing with the state-of-the-art ABKS-UR [18] and CP-VABKS [16] schemes which are applied in the *unshared* multi-owner settings.

From Table 4, we notice that our scheme is more efficient than the other two schemes for **KeyGen**, **Trap**, **Search** algorithms except for **Enc** algorithm. As our scheme needs to generate multiple signatures for each record and conduct extra exponentiation operations ( $2dE_1$ ), our scheme has higher computational burden than the ABKS-UR scheme for **Enc** algorithm. For the **KeyGen** algorithm, the ABKS-UR scheme requires 2 exponentiation operations ( $2E_2$ ) for each data user as well as  $2\mathcal{N}$  exponentiation operations ( $2\mathcal{N}E_1$ ) for the whole attribute set, the CP-VABKS scheme needs  $2\mathcal{N}$  exponentiation operations ( $2\mathcal{N}E_1$ ) and  $\mathcal{N}$  hash operations ( $\mathcal{N}h_1$ ) for the system attributes, while our scheme only needs one ( $E_1$ ) for each data owner and data user, respectively. Due to  $m \leq \mathcal{N}$ , the search time of our scheme is less than that of other two schemes for **Search** algorithm. Notice that the ABKS-UR scheme is still superior to CP-VABKS scheme for the **KeyGen**, **Enc**, **Trap** and **Search** algorithms. With the same reason shown in Table 3, the results verification time of our scheme for **Verify** algorithm is still acceptable due to the small value of  $q$ .

To evaluate the actual performance of the aforementioned three schemes, we conduct empirical experiments using a real-world dataset<sup>1</sup> which includes half a million records from 150 users. This public email dataset used in many SE schemes contains half a million records from about 150 users, mostly senior management of Enron, and the Enron corpus contains a total of about 0.5M message. The experiments are performed on an Ubuntu Server 15.04 with Intel Core i5 Processor 2.3 GHz by using C and PBC Library. In PBC Library, the Type A is denoted as  $E(F_\kappa) : y^2 = x^3 + x$ , the groups  $G_1$  and  $G_2$  of order  $p$  are the subgroups of  $E(F_\kappa)$ , where the parameter  $p$  and  $\kappa$  are equivalent to 160 bits and 512 bits respectively. Thus, we can have  $|G_1| = |G_2| = 1024$  bits,  $|Z_p| = 160$  bits. In line with the ABKS-UR scheme [18], we randomly select 10,000 records from this dataset and perform experiments for 100 times. For convenience, we set  $\mathcal{U} \in [1, 50]$ ,  $m \in [1, 100]$ ,  $q \in [1, 50]$ . Note that we also set  $\mathcal{N} = 100$ ,  $d = 10$  throughout this paper.

In Fig. 3(a), we notice that our scheme has less storage cost than other two schemes with regard to key generation due to  $\mathcal{U}$ ,  $d < \mathcal{N}$ . The storage costs of our scheme and the ABKS-UR scheme increase with the value of  $\mathcal{U}$ , while that of the CP-VABKS scheme almost remains unchanged. Besides, the CP-VABKS scheme has less storage cost than the ABKS-UR scheme when the variable  $\mathcal{U}$  increases to a certain value. In Fig. 3(b) we evaluate the key generation time by varying the number of data users. Obviously, in aforementioned schemes, the computational overhead for **KeyGen** algorithm increases almost linearly with the value of  $\mathcal{U} \in [1, 50]$ . Notice that our scheme outperforms the ABKS-UR and CP-VABKS schemes for **KeyGen** algorithm regarding the key generation time. As our scheme just needs  $(\mathcal{U} + d + 1)E_1$ , while the ABKS-UR scheme and CP-VABKS scheme need  $(2\mathcal{N} + 1)E_1 + 2\mathcal{U}E_2$  and  $(2\mathcal{N} + 2)E_1 + \mathcal{N}h_1$ , respectively. However, the CP-VABKS scheme is less efficient than ABKS-UR scheme as the exponentiation operation  $E_2$  is much more efficient than hash operation  $h_1$ .

In Fig. 4, we evaluate the computational burden of **Enc** algorithm in these schemes by varying the number of encrypted records from 1 to 10,000 (e.g.,  $n \in [1, 10,000]$ ). Obviously, the running time increases with the value of  $n$ . The encryption time of the ABKS-UR and CP-VABKS schemes is also affected by the variable  $\mathcal{N}$ , while that of our scheme is still influenced by other two factors  $m$ ,  $d$ , namely the number of keyword fields and the fixed number of data owners for each EHR. For comparison, we set  $m = 100$  for this algorithm. Notably, the CP-VABKS scheme has much more computational overhead than the ABKS-UR scheme due to the extra operations  $(\mathcal{N} + 3)E_1 + \mathcal{N}h_1$ . Although our scheme has slightly more running time than the ABKS-UR scheme for the **Enc** algorithm, it does not affect the user search experience as the **Enc** algorithm is performed only in the initialization of the system and is a one-time cost. Thus, our scheme is still acceptable in actual applications without deteriorating the user search experience.

<sup>1</sup> <http://www.cs.cmu.edu/~enron/>

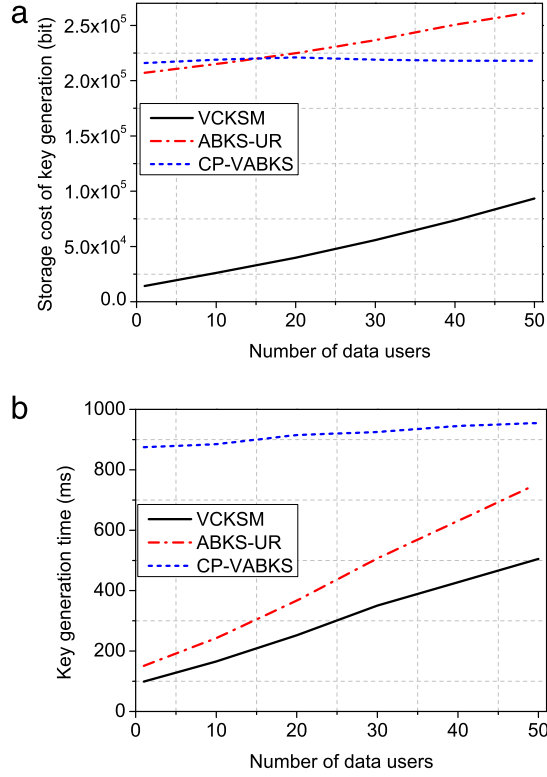


Fig. 3. Storage and computational costs in **KeyGen** algorithm: (a) Storage costs of key generation; (b) Computational costs of key generation.

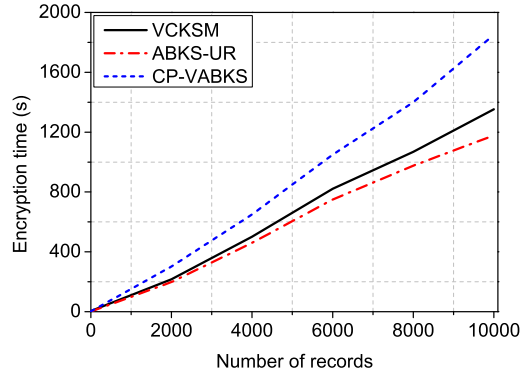


Fig. 4. Computational costs in **Enc** algorithm.

In Fig. 5(a), the storage cost of our scheme is  $(m+2)|G_1| + 2|Z_p|$ , while that of the ABKS-UR scheme is  $(2\mathcal{N}+1)|G_1| + 2|Z_p|$ , and that of the CP-VABKS scheme is  $(2\mathcal{N}+3)|G_1| + |Z_p|$ . However, in practice, the fields of each record are far less than 100, namely  $m \leq \mathcal{N}$ . Thus, our scheme will be inferior to the other two schemes in terms of storage cost for **Trap** algorithm, and the storage cost of ABKS-UR scheme is similar to that of CP-VABKS scheme. We show the computational costs for **Trap** algorithm in the aforementioned three schemes by varying the value of  $m$  from 1 to 100 in Fig. 5(b). As the hash operation  $h_2$  is more efficient than other operations, the computational costs of the aforementioned three schemes mainly depend on values of  $\mathcal{N}$  and  $m$ , respectively, rather than the number of submitted keywords. For **Trap** algorithm, our scheme needs  $(m+2)E_1$ , while the ABKS-UR and CP-VABKS schemes need  $(2\mathcal{N}+1)E_1$  and  $(2\mathcal{N}+4)E_1$ , respectively. Notice that the computational cost of our scheme increases almost linearly with  $m$ , while that of the other two schemes slightly increase with  $m$ . Due to  $m \leq \mathcal{N}$ , our scheme is more efficient than the ABKS-UR and CP-VABKS schemes in terms of trapdoor generation time.

In Fig. 6(a), we notice that our scheme has less storage cost than the other two schemes for **Search** algorithm because of  $m \leq \mathcal{N}$ , but the storage costs of the ABKS-UR  $((\mathcal{N}+2)|G_2|)$  and CP-VABKS  $((\mathcal{N}+3)|G_2|)$  schemes are almost approximately

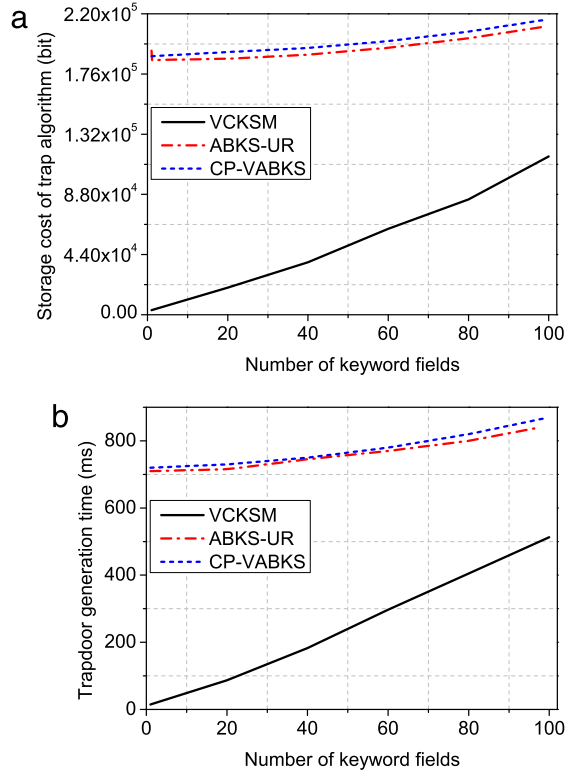


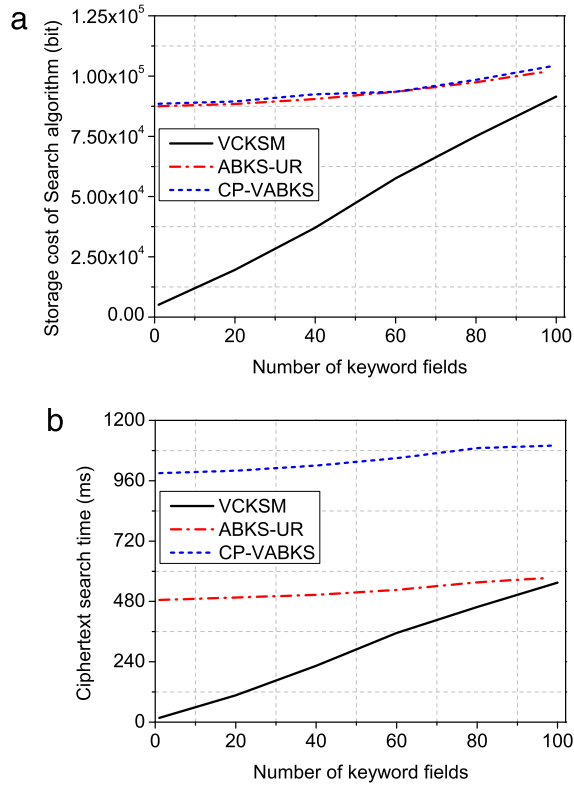
Fig. 5. Storage and computational costs in **Trap** algorithm: (a) Storage costs of trapdoor generation; (b) Computational costs of trapdoor generation.

equal. The actual execution time for **Search** algorithm in aforementioned three schemes is presented in Fig. 6(b). With the same reason shown in **Trap** algorithm, we present the computational overhead for **Search** algorithm by varying the value of  $m \in [1, 100]$ . Obviously, our scheme is much efficient than other two schemes in terms of ciphertexts retrieval time when  $m < 100$ . When the value of  $m$  is close to 100, the ABKS-UR scheme and our scheme have the similar computational complexities, but these two schemes have much less computational burden than the CP-VABKS scheme for **Search** algorithm. To conduct the ciphertexts search operations, our scheme needs  $(m+2)P + E_1 + 3E_2$ , and the ABKS-UR and CP-VABKS schemes need  $(\mathcal{N} + 1)P + E_1$  and  $(2\mathcal{N} + 3)P + \mathcal{N}E_1$ , respectively. For example, when setting  $m = 100$ , our scheme needs 554 ms, the ABKS-UR scheme needs 576 ms, while the CP-VABKS scheme needs 1100 ms. Thus, our scheme can be applied in a broad range of practical applications.

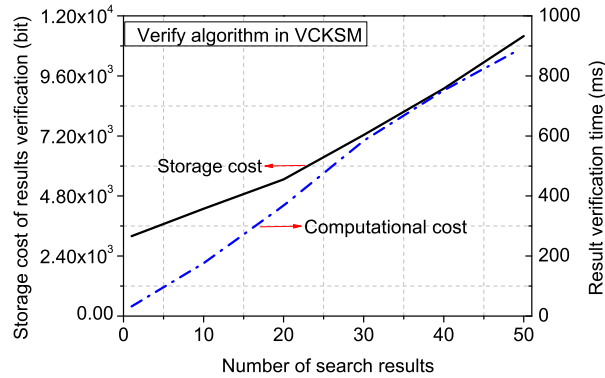
As the ABKS-UR scheme cannot accurately check the correctness of search results due to high false positive rate caused by Bloom Filter, we just show the storage cost and verification time of our scheme for **Verify** algorithm in Fig. 7, respectively. Then, we notice that the storage and computational costs of **Verify** algorithm increase with the number of search results ( $q \in [1, 50]$ ), which are in accord with our discussion in Table 3 and Table 4 respectively. Notably, even if  $q = 50$ , the storage cost of this algorithm almost needs 1.1 KB and the verification process only takes 896 ms. In addition, this algorithm is mainly performed by the fully-trusted PAS. In other words, this algorithm will not impose high storage and computational burden on the data user, especially for resource-limited entities, such as sensor nodes and mobile terminals.

From above figures, we deduce that the actual performance evaluation is completely in accord with the computational complexity study shown in Tables 3 and 4. Comparing with the state-of-the-art ABKS-UR and CP-VABKS schemes, our scheme is still efficient and feasible in mobile cloud computing without exerting high computational and storage burden on the mobile devices.

Besides, the experiments in our scheme are just performed for 100 times. Next, we will evaluate the creditability of experimental results by using the confidence interval [40]. Table 5 shows the storage and computational costs of our scheme for different algorithms, where the value of data users is set as  $\mathcal{U} = 50$  for **KeyGen** algorithm, the number of records is set as 10,000 for **Enc** algorithm, the number of keyword fields is set as  $m = 100$  for **Trap** and **Search** algorithms, and the number of search results is set as  $q = 50$ . In conclusion, we can accurately assess the actual performance of our scheme by conducting the experiments for 100 times.



**Fig. 6.** Storage and computational costs in **Search** algorithm: (a) Storage costs of ciphertexts retrieval; (b) Computational costs of ciphertexts retrieval.



**Fig. 7.** Storage and computational costs in **Verify** algorithm of VCKSM scheme.

## 7. Conclusion

In this paper, we propose the first verifiable conjunctive keyword search scheme in a more challenging model, which can guarantee the correctness of search results and support a broad range of practical applications, especially for the mobile e-health cloud systems. Different from the previously proposed SE schemes, our scheme enables verifiable conjunctive keyword search over encrypted EHRs, where a single record is kept by a group of data owners. In addition, our scheme can perfectly fit with the *shared* multi-owner settings in semi-trusted cloud environment by utilizing multisignatures technique, and meanwhile a specific data user can be assured with the correctness of search results. The theoretical security analysis proves that our scheme is secure against the IND-KGA in standard model. For the malicious CSP, it is computationally infeasible to forge the valid proof information to pass the results verification mechanism. Moreover, empirical study using a real-world dataset shows that our scheme is efficient and provides good user search experience in mobile cloud computing.

**Table 5**

Analysis for experimental results.

Algorithms	Results	Confidence interval ↓	Confidence interval ↑
<i>KeyGen<sub>sc</sub></i> (bit)	93408	93245.75	93475.05
<i>KeyGen<sub>cc</sub></i> (ms)	504.8	497.07	511.93
<i>Enc<sub>cc</sub></i> (s)	1353	1272.13	1354.47
<i>Trap<sub>sc</sub></i> (bit)	115792	115490.34	115804.78
<i>Trap<sub>cc</sub></i> (ms)	513	510.52	527.28
<i>Search<sub>sc</sub></i> (bit)	91520	91506.76	91666.44
<i>Search<sub>cc</sub></i> (ms)	554.7	540.01	555.59
<i>Verify<sub>sc</sub></i> (bit)	11192	11130.49	11263.71
<i>Verify<sub>cc</sub></i> (ms)	896	860.54	916.86

"Results": Average value of experimental tests for 100 times.

"sc": Storage cost; "cc": Computational cost.

"↓": Lower limit of confidence interval.

"↑": Upper limit of confidence interval.

As part of our future work, we will try to solve the problem of dynamic user updating (i.e., user enrollment and user revocation) so that our scheme can adaptively deal with the dynamic access control.

## Acknowledgments

This study was supported by the National High Technology Research and Development Program (863 Program) (No. 2015AA016007), the National Nature Science Foundation of China (No. 61472310, No. 61672413, No. 61370078 and No. 61309016), the Key Program of NSFC (No. U1405255 and No. U1135002), and the Shaanxi Science & Technology Coordination & Innovation Project (No. 2016TZC-G-6-3).

## References

- [1] Z.H. Xia, X.H. Wang, L.G. Zhang, Z. Qin, X.M. Sun, K. Ren, A privacy-preserving and copy-deterrence content-based image retrieval scheme in cloud computing, *IEEE Trans. Inf. Forensics Secur.* 11 (11) (2016) 2594–2608. <http://dx.doi.org/10.1109/TIFS.2016.2590944>.
- [2] Q. Jiang, J.F. Ma, F.S. Wei, On the security of a privacy-aware authentication scheme for distributed mobile cloud computing services, *IEEE Systems Journal* PP (99) (2016) 1–4. <http://dx.doi.org/10.1109/JSYST.2016.2574719>.
- [3] L.F. Wei, H.J. Zhu, Z.F. Cao, X.L. Dong, W.W. Jia, Y.L. Chen, A.V. Vasilakos, Security and privacy for storage and computation in cloud computing, *Inform. Sci.* 258 (2014) 371–384. <http://dx.doi.org/10.1016/j.ins.2013.04.028>.
- [4] J. Shen, D. Liu, J. Shen, Q. Liu, X. Sun, A secure cloud-assisted urban data sharing framework for ubiquitous-cities, *Pervasive and Mob. Comput.* (2017). <http://dx.doi.org/10.1016/j.pmcj.2017.03.013>.
- [5] Z.J. Fu, K. Ren, J.G. Shu, X.M. Sun, F.X. Huang, Enabling personalized search over encrypted outsourced data with efficiency improvement, *IEEE Trans. Parallel Distrib. Syst.* 27 (9) (2016) 2546–2559. <http://dx.doi.org/10.1109/TPDS.2015.2506573>.
- [6] H.W. Li, D.X. Liu, Y.S. Dai, T.H. Luan, S. Yu, Personalized search over encrypted data with efficient and secure updates in mobile clouds, *IEEE Trans. Emerging Top. Comput. PP* (99) (2015) 1. <http://dx.doi.org/10.1109/TETC.2015.2511457>.
- [7] Z.J. Fu, F.X. Huang, X.M. Sun, A.V. Vasilakos, C.N. Yang, Enabling semantic search based on conceptual graphs over encrypted outsourced data, *IEEE Trans. Services Computing* PP (99) (2016) 1. <http://dx.doi.org/10.1109/TSC.2016.2622697>.
- [8] Z. Fu, J. Shu, J. Wang, Y. Liu, S. Lee, Privacy-preserving smart similarity search based on simhash over encrypted data in cloud computing, *J. Internet Tech.* 16 (3) (2015) 453–460.
- [9] J. Shen, D. Liu, Q. Liu, S. Xingming, Y. Zhang, Secure authentication in cloud big data with hierarchical attribute authorization structure, *IEEE Trans. Big Data* (2017). <http://dx.doi.org/10.1109/TBDATA.2017.2705048>.
- [10] Q. Jiang, M.K. Khan, X. Lu, J.F. Ma, D.B. He, A privacy preserving three-factor authentication protocol for e-health clouds, *J. Supercomput.* 72 (10) (2016) 3826–3849. <http://dx.doi.org/10.1007/s11227-015-1610-x>.
- [11] D. Boneh, G.D. Crescenzo, R. Ostrovsky, G. Persiano, Public key encryption with keyword search, *Proc. Int. Conf. Theory Appl. Cryptographic Tech.* 3027 (2004) 506–522. [http://dx.doi.org/10.1007/978-3-540-24676-3\\_30](http://dx.doi.org/10.1007/978-3-540-24676-3_30).
- [12] J. Li, R. Ma, H. Guan, TEES: An efficient search scheme over encrypted data on mobile cloud, *IEEE Trans. Cloud Comput.* 5 (1) (2017) 126–139. <http://dx.doi.org/10.1109/TCC.2015.2398426>.
- [13] Y.H. Zhang, X.F. Chen, J. Lin, D.S. Wong, H. Li, I. You, Ensuring attribute privacy protection and fast decryption for outsourced data security in mobile cloud computing, *Inform. Sci.* 379 (2017) 42–61. <http://dx.doi.org/10.1016/j.ins.2016.04.015>.
- [14] Q. Chai, G. Gong, Verifiable symmetric searchable encryption for semi-honest-but-curious cloud servers, In: *Proc. IEEE Int. Conf. Commun. PP* (2012) 917–922. <http://dx.doi.org/10.1109/ICC.2012.6364125>.
- [15] W.H. Sun, B. Wang, N. Cao, M. Li, W.J. Lou, Y.T. Hou, H. Li, Verifiable privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking, *IEEE Trans. Parallel Distrib. Syst.* 25 (11) (2014) 3025–3035. <http://dx.doi.org/10.1109/TPDS.2013.282>.
- [16] Q.J. Zheng, S.H. Xu, G. Ateniese, Vabks: Verifiable attribute-based keyword search over outsourced encrypted data, *Proc. IEEE Conf. Comput. Commun. PP* (2014) 522–530. <http://dx.doi.org/10.1109/INFOCOM.2014.6847976>.
- [17] W.H. Sun, X.F. Liu, W.J. Lou, Y.T. Hou, H. Li, Catch you if you lie to me: Efficient verifiable conjunctive keyword search over large dynamic encrypted cloud data, *Proc. IEEE Conf. Computer Communications* PP (2015) 2110–2118. <http://dx.doi.org/10.1109/INFOCOM.2015.7218596>.
- [18] W.H. Sun, S.C. Yu, W.J. Lou, Y.T. Hou, 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.* 27 (4) (2016) 1187–1198. <http://dx.doi.org/10.1109/TPDS.2014.2355202>.
- [19] M. Li, S. Yu, K. Ren, W. Lou, Securing personal health records in cloud computing: Patient-centric and fine-grained data access control in multi-owner settings, *Proc. ICST Conf. Security and Privacy in Communication Networks* PP (2010) 89–106. [http://dx.doi.org/10.1007/978-3-642-16161-2\\_6](http://dx.doi.org/10.1007/978-3-642-16161-2_6).
- [20] X.F. Liu, Y.Q. Zhang, B.Y. Wang, J.B. Yan, Mona: Secure multi-owner data sharing for dynamic groups in the cloud, *IEEE Trans. Parallel Distrib. Syst.* 24 (5) (2013) 1182–1191. <http://dx.doi.org/10.1109/TPDS.2012.331>.



- [21] B.Y. Wang, H. Li, X.F. Liu, X.Q. Li, F.H. Li, Preserving identity privacy on multi-owner cloud data during public verification, *Secur. Commun. Netw.* 7 (11) (2014) 2104–2113. <http://dx.doi.org/10.1002/sec.922>.
- [22] M. Layouni, M. Yoshida, S. Okamura, Efficient multi-authorizer accredited symmetrically private information retrieval, *Proc. Int. Conf. Inf. Commun. Secur. PP* (2008) 387–402. [http://dx.doi.org/10.1007/978-3-540-88625-9\\_26](http://dx.doi.org/10.1007/978-3-540-88625-9_26).
- [23] B.Y. Wang, H. Li, X.F. Liu, F.H. Li, X.Q. Li, Efficient public verification on the integrity of multi-owner data in the cloud, *J. Commun. Netw.* 16 (6) (2014) 592–599. <http://dx.doi.org/10.1109/JCN.2014.000105>.
- [24] Y. Yang, M.D. Ma, Conjunctive keyword search with designated tester and timing enabled proxy re-encryption function for e-health clouds, *IEEE Trans. Inf. Forensics Secur.* 11 (4) (2016) 746–759. <http://dx.doi.org/10.1109/TIFS.2015.2509912>.
- [25] H.W. Li, Y. Yang, T.H. Luan, X.H. Liang, L. Zhou, X.M. Shen, Enabling fine-grained multi-keyword search supporting classified sub-dictionaries over encrypted cloud data, *IEEE Trans. Dependable Secur. Comput.* 13 (3) (2016) 312–325. <http://dx.doi.org/10.1109/TDSC.2015.2406704>.
- [26] Z.J. Fu, X.L. Wu, C.W. Guan, X.M. Sun, K. Ren, Towards efficient multi-keyword fuzzy search over encrypted outsourced data with accuracy improvement, *IEEE Trans. Inf. Forensics Secur.* 11 (12) (2016) 2706–2716. <http://dx.doi.org/10.1109/TIFS.2016.2596138>.
- [27] Y.B. Miao, J.F. Ma, Z.Q. Liu, Revocable and anonymous searchable encryption in multi-user setting, *Concurrency and Computation: Practice and Experience* 28 (4) (2016) 1204–1218. <http://dx.doi.org/10.1002/cpe.3608>.
- [28] H.W. Li, D.X. Liu, Y.S. Dai, T.H. Luan, Engineering searchable encryption of mobile cloud networks: when qoe meets qop, *IEEE Wirel. Commun.* 22 (4) (2015) 74–80. <http://dx.doi.org/10.1109/MWC.2015.7224730>.
- [29] Y.J. Ren, J. Shen, J. Wang, J. Han, S.Y. Lee, Mutual verifiable provable data auditing in public cloud storage, *J. Internet Technol.* 16 (2) (2015) 317–323. <http://dx.doi.org/10.6138/JIT.2015.16.2.20140918>.
- [30] J. Shen, J. Shen, X. Chen, X. Huang, W. Susilo, An efficient public auditing protocol with novel dynamic structure for cloud data, *IEEE Trans. Inf. Forensics Secur.* (2017). <http://dx.doi.org/10.1109/TIFS.2017.2705620>.
- [31] L.M. Fang, W. Susilo, C.P. Ge, J.D. Wang, Public key encryption with keyword search secure against keyword guessing attacks without random oracle, *Inform. Sci.* 238 (2013) 221–241. <http://dx.doi.org/10.1016/j.ins.2013.03.008>.
- [32] R.M. Chen, Y. Mu, G.M. Yang, F.C. Guo, X.F. Wang, Dual-server public-key encryption with keyword search for secure cloud storage, *IEEE Trans. Inf. Forensics Secur.* 11 (4) (2016) 789–798. <http://dx.doi.org/10.1109/TIFS.2015.2510822>.
- [33] Z.H. Xia, X.H. Wang, X.M. Sun, Q. Wang, A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data, *IEEE Trans. Parallel Distrib. Syst.* 27 (2) (2016) 340–352. <http://dx.doi.org/10.1109/TPDS.2015.2401003>.
- [34] H.W. Li, D.X. Liu, Y.S. Dai, T.H. Luan, X.M. Shen, Enabling efficient multi-keyword ranked search over encrypted mobile cloud data through blind storage, *IEEE Trans. Emerging Top. Comput.* 3 (1) (2015) 127–138. <http://dx.doi.org/10.1109/TETC.2014.2371239>.
- [35] Z.J. Fu, X.M. Sun, Q. Liu, L. Zhou, J.G. Shu, Privacy-preserving smart semantic search based on conceptual graphs over encrypted outsourced data, *IEEE Trans. Inf. Forensics Secur.* 12 (8) (2017) 1874–1884.
- [36] Y.B. Miao, J. Liu, J.F. Ma, Efficient keyword search over encrypted data in multi-cloud setting, *Secur. Commun. Netw.* 9 (16) (2016) 3808–3820. <http://dx.doi.org/10.1002/sec.1559>.
- [37] V. Goyal, O. Pandey, A. Sahai, B. Waters, Attribute-based encryption for fine-grained access control of encrypted data, *Proc. ACM Conf. Comput. Commun. Secur. PP* (2006) 89–98. <http://dx.doi.org/10.1145/1180405.1180418>.
- [38] J. Bethencourt, A. Sahai, B. Waters, Ciphertext-Policy attribute-based encryption, *Proc. IEEE Symp. Secur. Priv. PP* (2007) 321–334. <http://dx.doi.org/10.1109/SP.2007.11>.
- [39] W. Zhang, Y.P. Lin, S. Xiao, J. Wu, S.W. Zhou, Privacy preserving ranked multi-keyword search for multiple data owners in cloud computing, *IEEE Trans. Comput.* 65 (5) (2016) 1566–1577. <http://dx.doi.org/10.1109/TC.2015.2448099>.
- [40] T.J. Witt, Experimental sampling distributions and confidence intervals of the allan variance in some dc electrical measurements, *IEEE Trans. Instrum. Meas.* 52 (2) (2003) 487–490. <http://dx.doi.org/10.1109/TIM.2003.811653>.