

Singapore Management University Institutional Knowledge at Singapore Management University

Research Collection School Of Information Systems

School of Information Systems

11-2016

m(2)-ABKS: Attribute-based multi-keyword search over encrypted personal health records in multi-owner setting

Yinbin MIAO
Xidian University

Jianfeng MA
Xidian University


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

Fushan WEI
Xidian University

Zhiquan LIU
Xidian University

See next page for additional authors

Follow this and additional works at 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; WEI, Fushan; LIU, Zhiquan; and WANG, Xu An. m(2)-ABKS: Attribute-based multi-keyword search over encrypted personal health records in multi-owner setting. (2016). *Journal of Medical Systems*. 40, (11), 1-12.
Research Collection School Of Information Systems.

Available at: https://ink.library.smu.edu.sg/sis_research/3272

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.

Author

Yinbin MIAO, Jianfeng MA, Ximeng LIU, Fushan WEI, Zhiquan LIU, and Xu An WANG

m^2 -ABKS: Attribute-Based Multi-Keyword Search over Encrypted Personal Health Records in Multi-Owner Setting

Yinbin Miao¹ · Jianfeng Ma^{2,3} · Ximeng Liu⁴ · Fushan Wei¹ · Zhiquan Liu^{2,3} · Xu An Wang¹

Abstract Online personal health record (PHR) is more inclined to shift data storage and search operations to cloud server so as to enjoy the elastic resources and lessen computational burden in cloud storage. As multiple patients' data is always stored in the cloud server simultaneously, it is a challenge to guarantee the confidentiality of PHR data and allow data users to search encrypted data in an efficient and privacy-preserving way. To this end, we design a secure cryptographic primitive called as attribute-based multi-keyword search over encrypted personal health records in multi-owner setting to support both fine-grained access control and multi-keyword search via Ciphertext-Policy Attribute-Based Encryption. Formal security analysis proves our scheme is selectively secure against chosen-keyword attack. As a further contribution, we conduct empirical experiments over real-world dataset to show its feasibility and practicality in a broad range of actual scenarios without incurring additional computational burden.

Keywords Personal health record · Attribute-based encryption · Multi-keyword · Multi-owner · Chosen-keyword attack

Introduction

With the development of cloud storage, considerable amount of individuals and enterprises are motivated to ease heavy computation and management burden in a cost-effective way through outsourcing their records to cloud service providers (CSP). While data security and privacy concerns remain significant barriers to the adoption of cloud storage [1, 2] as CSP is always considered to be a honest-but-curious [3] entity. Encrypting is considered as a simple and efficient solution to guarantee the data confidentiality, but it also makes search over encrypted data extremely difficult, especially for the encrypted personal health record (PHR) system. Therefore, exploring efficient searchable encryption (SE) techniques [4, 5] which enable cloud clients to securely search through keywords and selectively retrieve files of interest over encrypted data has recently drawn a significant amount of interest in both industry and academia fields. As SE technique enables data users to conduct search operation over encrypted data without any loss of data confidentiality, a variety of follow-up work has been proposed to meet various application requirements, such as conjunctive keyword search, fuzzy keyword search and ranked keyword search. However, one desires such protocols which provide secure search yet still enable data owner to impose enforcement of access control over encrypted data in multi-user setting. To tackle this problem, the state-of-the-art attribute-based encryption (ABE) technique [6–8] will be adopted to achieve fine-grained access control.

Jianfeng Ma
jfma@mail.xidian.edu.cn

Yinbin Miao
ybmiao@stu.xidian.edu.cn

- ¹ School of Telecommunication Engineering, Xidian University, Xi'an, China
- ² School of Computer Science and Technology, Xidian University, Xi'an, China
- ³ School of Cyber Engineering, Xidian University, Xi'an, China
- ⁴ School of Information Systems, Singapore Management University, 80 Stamford Road, Singapore, Singapore

Consider a Personal Health Record (PHR) system [9], patients lose physical control over their data through outsourcing records to CSP. Moreover, encryption-before-outsourcing mechanism makes it is challenging to achieve fine-grained access control over encrypted data, especially for the existence of multiple patients (In here, we call the scenario as multi-owner setting). Thus, enforcing the data owner’s access control over outsourced data is of prime importance in multi-owner setting. Aside from enabling to guarantee that only authorized users can browse the encrypted records, this system should also support retrieving corresponding encrypted records that contain given query keywords. And the SE schemes supporting single-keyword search inevitably return many irrelevant results and incur great computational burden when multiple keywords have to be matched simultaneously. To enable cloud clients to quickly search the most relevant records, it is crucial for SE schemes to support multi-keyword search so as to improve search result accuracy as well as user search experience.

To the best of our knowledge, the multi-owner setting mentioned in previous schemes [8, 9] actually discusses about data owners’ updating (i.e., the enrollment or revocation). Along these directions, in this paper we organically integrate SE with Ciphertext-policy ABE (CP-ABE) scheme [7] and devise a new cryptographic primitive, namely **m²-ABKS** (Attribute-Based *m*ulti-*K*eyword Search over encrypted personal health records in *m*ulti-user setting). Our scheme can achieve fine-grained access control over encrypted data and support multi-keyword query in multi-owner setting simultaneously. Formal security analysis proves that m²-ABKS scheme is selectively secure against chosen-keyword attack and achieves the keyword secrecy. We also make theoretical computation complexity analysis and conduct empirical experiments to show its feasibility in practice. The experimental results indicate that the actual performance evaluation is in accord with the theoretical study of asymptotic computation complexity. Thus, our scheme can admit a broad of applications in practice without incurring heavy computational burden. The main contributions of this work can be summarized as follows:

- **Supporting real multi-owner setting.** m²-ABKS scheme allows multiple patients to enhance access control over encrypted records through utilizing CP-ABE scheme.
- **Supporting multi-keyword query.** Our scheme allows data users to perform multi-keyword search query rather than single keyword query so as to save bandwidth and computation resources.
- **Resisting chosen-keyword attack and ensuring keyword secrecy.** Security analysis formally proves that

our scheme is selectively secure against the chosen-keyword attack, and it can also ensure keyword secrecy. Additionally, collusion attack and unauthorized access can be effectively prohibited.

- **Efficiency and feasibility.** The experimental results over real-world dataset show that our scheme is efficient and feasible in practical applications.

The remainder of this paper is organized as follows. We first review some cryptographic preliminaries used in our work in Section “**Preliminaries**”, followed by Section “**Problem formulation**” which introduces the system model, algorithm definition and security models. The specific construction of our scheme is presented in Section “**Concrete construction of m²-ABKS**”. Security and performance analysis of m²-ABKS scheme are presented in Section “**Security and performance analysis**”. Then Section “**Related work**” presents the related work associated with our scheme. Finally, the concluding remark of this whole paper is summarized in Section “**Conclusion**”.

Preliminaries

In this part, we first briefly review some cryptographic background associated with our scheme.

Definition 1 (Bilinear Group and Pairings) Let G_1, G_2 be two (multiplicative) cyclic groups of prime order q , g be a generator of group G_1 and e be the bilinear map $G_1 \times G_1 \rightarrow G_2$ with following properties:

- Bilinearity: Given four random elements $a, b \in G_1, x, y \in \mathbb{Z}_q$, we get $e(a^x, b^y) = e(a^y, b^x) = e(a, b)^{xy}$.
- Non-degeneracy: $e(g, g) \neq 1$.
- Computability: There exists an efficient algorithm to compute $e(g, g)$.

Definition 2 (Access Tree \mathcal{T}) Let \mathcal{T} be a tree describing an access structure, each non-leaf node can be treated as a threshold gate and described by its children and threshold value. Let num_x be the number of children for node x and k_x be the threshold value, where $0 < k_x \leq num_x$. When $k_x = 1$, the threshold gate is an OR gate, and when $k_x = num_x$, it is an AND gate. Each leaf node is described by an attribute and its threshold value $k_x = 1$. The parent of node x is denoted as $parent(x)$, and $att(x)$ represents the attribute associated with each leaf node x . In addition, for each node y , who is a child of x (i.e., $parent(y) = x$), we assign a number from $\{1, 2, \dots, num_x\}$ as its index, denoted by $index(y)$. The index values are uniquely assigned to nodes in the access structure for a given key in an arbitrary manner. That is, $\forall y \neq y',$ if $parent(y) = parent(y'),$ $index(y) \neq index(y')$.

Let the root node of \mathcal{T} be r and \mathcal{T}_x be the subtree of \mathcal{T} rooted at the node x . If an attribute set γ satisfies the access tree \mathcal{T}_x , then $\mathcal{T}_x(\gamma) = 1$. In particular, $\mathcal{T}_x(\gamma)$ can be recursively computed as follows. If x is a non-leaf node, compute $\mathcal{T}_{x'}(\gamma)$ for all children x' of node x . $\mathcal{T}_x(\gamma)$ returns 1 if and only if at least k_x children return 1. If x is a leaf node, then $\mathcal{T}_x(\gamma)$ returns 1 if and only if $att(x) \in \gamma$.

Problem formulation

System and threat models

We consider a cryptographic cloud storage system (e.g., PHR) supporting both information retrieval and fine-grained access control over encrypted personal data files (e.g., health records). In such system, there exist multiple data owners (e.g., patients) and multiple data users. Data owners can create, manage and modify their files, data users (e.g., doctor, healthcare provider, researcher) can access these sensitive files with authorizations from certain patient. There are four entities involved in the system framework, namely data owner (DO), data user (DU), cloud server provider (CSP) and third trusted server (TTS) in Fig. 1. To enjoy the elastic resources and reduce the operational cost, it is attractive for DO to outsource data storage and retrieval to CSP, where encryption-before-outsourcing solution makes this a challenging issue to enable fine-grained access control over encrypted data in a scalable and efficient way. Based on that, m^2 -ABKS scheme works as follows:

- (1) DO first extracts keyword set from each health record and builds index with specified access policy by leveraging CP-ABE, then sends ciphertext to CSP.

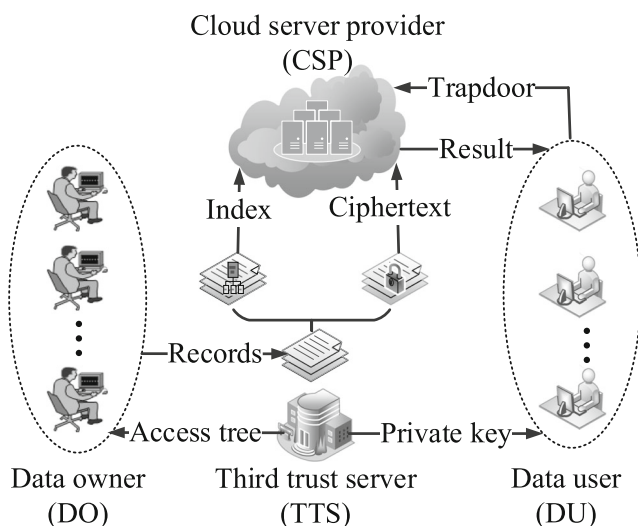


Fig. 1 System model of m^2 -ABKS scheme

- (2) DU gains the secret key generated through submitting his attribute set to TTS, then he generates the trapdoor of interested keyword set and submits it as well as attribute set to CSP.
- (3) After gaining the trapdoor and attribute set, CSP first verifies the legitimacy of DU, then matches the trapdoor with the indexes and returns relevant records to DU.
- (4) Only when the attribute set satisfies the specified access policy, DU can gain the relevant search results.

For the fully-trusted TTS, it is assumed to be uncompromisable by any attack. While for CSP, we assume that it is honest-but-curious like many previous schemes. That is, it genuinely follows the specified protocols to execute search operations, but it may be curious to deduce valuable information through analyzing the statistic of outsourced data, which results in severe security breaches on the confidentiality of encrypted raw data. To be more convenient, DOs intend to erase the heavy burden of large scale data management in a cost-effective manner with the help of CSP. However, they must enforce the access control over encrypted data to avoid data privacy leakage when exploiting the CSP's huge computing power. Notice that the record encryption is beyond the scope of our discussion.

Algorithm definition

Let $U = \{1, 2, \dots, n\}$ be an attribute set for the access policy, $W = \{w_1, \dots, w_m\}$ be keyword dictionary, $D = \{d_1, \dots, d_p\}$ be the record set, the tuple (G_1, G_2, q, g, e) be the bilinear map parameters, and $(\mathcal{W}, \mathcal{G})$ be the keyword space and access structure space, respectively. And our scheme is a tuple $(\text{Setup}, \text{KeyGen}, \text{Enc}, \text{Trap}, \text{Search})$ of five polynomial-time algorithms shown as follows:

- (1) $(mk, pk, s) \leftarrow \text{Setup}(k, U)$: Take the security parameter k , attribute set U as input, this algorithm outputs the public key pk , master key mk and a symmetrical secret key s , where mk is owned by TTS.
- (2) $sk \leftarrow \text{KeyGen}(mk, S, \mathcal{T})$: TTS runs this algorithm to generate the private key sk for DU who has an attribute set S according to the access tree \mathcal{T} .
- (3) $(C, I) \leftarrow \text{Enc}(pk, s, W, f, \mathcal{P})$: DO performs this algorithm to generate ciphertext C and index I according to specified access structure \mathcal{P} for his records, where \mathcal{P} can be used to specify the DUs who are authorized to access the sensitive information.
- (4) $T \leftarrow \text{Trap}(sk, W')$: DU runs this algorithm to output the trapdoor T for keyword set W' , then he sends it to CSP.
- (5) $\{0, 1\} \leftarrow \text{Search}(T, I, C)$: After gaining the trapdoor T , CSP first matches it with the index I , then it returns the relevant search results C to DU, where "1" denotes that the algorithm returns correct results, "0" means \perp .

Security models

The adversary is allowed to submit search tokens except those corresponding to the keyword set in the challenge phase, while he learns nothing beyond search results. We formalize the *security models* through selectively chosen-keyword game and keyword secrecy game, respectively. Specifically, if there exists no probabilistic polynomial-time adversary \mathcal{A} who can deduce any plaintext information, then we consider a semantic security against chosen-keyword attack. The security of our scheme can be defined by two asymptotically equivalent security games (i.e. indistinguishability of ciphertext from ciphertext chosen-keyword attack, indistinguishability of ciphertext from random chosen-keyword attack.) except for *Challenge* phase [10]. Our scheme is selectively secure based on the indistinguishability of ciphertext from ciphertext chosen-keyword attack. We formally introduce the selectively chosen-keyword attack (SCKA) game as follows.

- (1) **Setup:** Given the security parameter k , the challenger \mathcal{C} runs the $\text{KeyGen}(mk, S, \mathcal{T})$ algorithm to output the secret key sk , while the adversary \mathcal{A} just has the public key pk , where the mk is owned by \mathcal{C} .
- (2) **Phase 1:** \mathcal{A} adaptively queries a series of keyword sets W_1, \dots, W_t to Trap oracle as follows:
 - $\text{Trap}(sk, W)$. \mathcal{C} runs $\text{Trap}(sk, W_i)$ algorithm to generate the trapdoor T_{W_i} , then responds it to \mathcal{A} .
- (3) **Challenge:** \mathcal{A} first selects two target keyword set (W_0, W_1) and sends them to \mathcal{C} , then \mathcal{C} randomly chooses a bit $b \in \{0, 1\}$, while it is required that W_0 and W_1 cannot be issued in the Trap oracle. Finally \mathcal{C} sets $I_b = \text{Enc}(pk, W_b, \mathcal{P})$ and sends it to \mathcal{A} .
- (4) **Phase 2:** \mathcal{A} issues keyword sets W_{t+1}, \dots, W_τ to Trap oracle as follows:
 - $\text{Trap}(sk, W_i \neq W_0, W_1)$. \mathcal{C} first runs $\text{Trap}(sk, W_i)$ algorithm to generate the trapdoor T_{W_i} , then \mathcal{C} sends it to \mathcal{A} .
- (5) **Guess:** \mathcal{A} outputs a guess $b' \in \{0, 1\}$. It wins the game if $b = b'$.

The \mathcal{A} 's advantage in breaking our scheme is denoted as $Adv_{m^2-ABKS, \mathcal{A}}^{SCKA}(1^k) = |Pr[b = b'] - \frac{1}{2}|$.

To protect the keyword security, our scheme must ensure that the malicious adversaries deduce nothing from keyword ciphertext or search token. Specifically, if there exists no probabilistic polynomial-time adversary \mathcal{A} who can learn the keyword plaintext from keyword ciphertext or search token, the keyword secrecy can be guaranteed. The keyword secrecy game is shown as follows:

- (1) **Setup:** Given a security parameter k , the challenger \mathcal{C} runs $\text{Setup}(k, \mathcal{U})$ algorithm to generate the public key pk and master key mk .
- (2) **Phase 1:** The adversary \mathcal{A} issues the following algorithms for polynomially many times.
 - **KeyGen:** \mathcal{C} sends corresponding secret key sk to \mathcal{A} and adds the queried keyword set in the list I_{KeyGen} .
 - **Trap:** Given the secret key sk and keyword set W , \mathcal{C} generates the trapdoor T and returns it to \mathcal{A} .
- (3) **Challenge:** \mathcal{A} first chooses a challenging secret key sk and gives it to \mathcal{C} , then \mathcal{C} chooses keyword set W' from message space and runs Enc algorithm, finally \mathcal{C} sends the index I to \mathcal{A} .
- (4) **Guess:** After issuing τ distinct keyword set, \mathcal{A} outputs a keyword set W' and wins the keyword secrecy game if $W = W'$.

Our scheme can achieve keyword secrecy if \mathcal{A} 's probability in breaking the keyword secrecy game is at most $\frac{1}{|\mathcal{W}|^{\tau}} + \epsilon$, where τ denotes the number of keyword sets, ϵ is an negligible probability in security parameter k and \mathcal{W} is the keyword space.

Concrete construction of m^2 -ABKS

Our scheme allows certain DO to outsource encrypted indexes and ask CSP to perform keyword query. Specifically, each DO (e.g., patient) generates the index for his data file (e.g., health record) with specified access policy which is expressed by a series of 'AND' and 'OR' gates, such as the access structure shown in Fig. 2. Afterwards, certain DO delegates the keyword search capability to DUs (e.g., healthcare providers, a doctor in certain hospital) according to his specified access policy. Only the authorized DUs whose attributes satisfy the access policy can retrieve encrypted records. Otherwise, the illegal DUs have no corresponding secret keys to generate search token. Next, we give the concrete construction of our scheme and show its correctness, respectively.

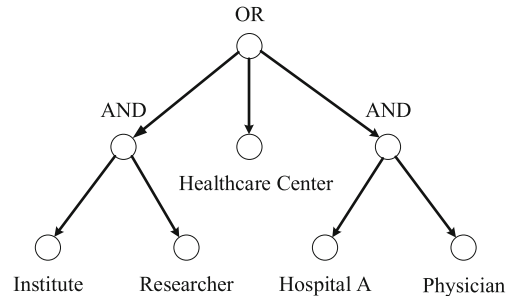


Fig. 2 An example of access policy

Scheme details

In this part, we first denote the Lagrange Coefficient as $\Delta_{i,\omega}(x) = \prod_{j \in \omega, j \neq i} \frac{x-j}{i-j}$, where $i \in \mathcal{Z}_q$, ω is a set of elements in \mathcal{Z}_q . Besides, we introduce some notations (shown in Table 1) used in our scheme before presenting the specific construction of m^2 -ABKS scheme as follows:

System setup Given the secure parameter k and the bilinear map parameters (G_1, G_2, q, g, e) , TTS first calls $(mk, pk, s) \leftarrow \text{Setup}(k, \mathcal{U})$ algorithm to generate the public key pk , master key mk and a symmetrical secret key s , then it selects two collision-resistant hash functions $h_1 : \{0, 1\}^* \rightarrow G_1, h_2 : \{0, 1\}^* \rightarrow \mathcal{Z}_q$. Besides, it randomly selects elements $\alpha, \beta, \gamma \in \mathcal{Z}_q$ and computes $X = g^\alpha, Y = g^\beta, Z = g^\gamma$. Finally, it sets the public key pk and master key msk as follows:

$$pk = (G_1, G_2, q, g, e, h_1, h_2, X, Y, Z),$$

$$mk = (\alpha, \beta, \gamma).$$

Key generation Given a set of attributes S , TTS calls $sk \leftarrow \text{KeyGen}(mk, S, \mathcal{T})$ algorithm to generate the private key sk for authorized DU. It first chooses a random element $r \in \mathcal{Z}_q$, then selects random $r_j \in \mathcal{Z}_q$ for each attribute $j \in S$, finally outputs secret key sk through the following equation.

$$sk = (\pi = g^{(\alpha\gamma-r)/\beta}, \{\lambda_j = g^r h_1(j)^{r_j}, \mu_j = g^{r_j}\}_{j \in S}).$$

Index generation Before outsourcing ciphertext to CSP, DO needs to call $(C, I) \leftarrow \text{Enc}(pk, s, W, D, \mathcal{P})$ algorithm to generate index and encrypt record set. He first extracts keyword set $W_d = \{w_{I_1}, \dots, w_{I_j}, \dots, w_{I_m}\}$ from certain record $d \in D$ and selects random elements $r_1, r_2 \in \mathcal{Z}_q$, where $I_j = 1$ represents that the j -th keyword in W is included in d , otherwise, it is not included in the record.

Table 1 Notation definitions

Symbols	Descriptions
\mathcal{T}	Access structure
$U = \{1, \dots, n\}$	Attribute set for \mathcal{T}
$W = \{w_1, \dots, w_m\}$	Keyword set
$W' = \{w'_1, \dots, w'_t\}$	Submitted keyword set
$D = \{d_1, \dots, d_p\}$	Record set
S	DU's attribute set
$W_d = \{w_{I_1}, \dots, w_{I_m}\}$	Keywords included in d
$I_j j \in \{1, \dots, m\}$	j -th keyword in W
$I'_i i \in \{1, \dots, t\}$	i -th submitted keyword in W'
I	Index for record d
C	Ciphertext for record d
T	Trapdoor for keywords W'

Then he chooses a polynomial q_x for each node x of \mathcal{T} in the following way, which starts from the root node r in a top-down manner. For each node x , set the degree d_x of polynomial q_x as $d_x = k_x - 1$, where k_x is the threshold value of x . Starting with the root node r , this algorithm sets $q_r(0) = r_2$, then it randomly chooses d_r other points of polynomial q_r to define it completely. For other node x , this algorithm sets $q_x(0) = q_{parent(x)}(index(x))$ and randomly chooses d_x other points to completely define polynomial q_x . Mark the leaf node set of \mathcal{T} as ln and compute CT as follows:

$$\{\delta_i = X^{r_1 h_2(w_i)}\}_{i \in \{1, \dots, m\}};$$

$$E_0 = X^{r_2}, E_1 = Y^{r_2}, E_2 = Z^{r_1}, C = \text{Enc}_s(d);$$

$$\{\delta_y = g^{q_y(0)}, \theta_y = h_1(att(y))^{q_y(0)}\}_{y \in ln}.$$

Finally he sends the index and encrypted record $CT = (C, I)$ to CSP, where $I = \{\{\delta_i\}, \{\delta_y, \theta_y\}, E_0, E_1, E_2\}$. While the record encryption $C = \text{Enc}_s(d)$ is beyond the scope of our discussion.

Trapdoor generation Given the queried keyword set $W' = \{w'_1, \dots, w'_t\}$, the $T \leftarrow \text{Trap}(sk, W', I'_1, \dots, I'_t)$ algorithm is called by DU to generate the trapdoor T . He first selects a random element $s \in \mathcal{Z}_q$ and computes $T_1 = \prod_{i=1}^t g^{s \alpha h_2(w'_i)}, T_2 = g^{s\gamma}, T_3 = \pi^s$. Then he computes $\lambda'_j = \lambda_j^s, \mu'_j = \mu_j^s$ for each attribute $j \in S$ and sets the trapdoor of submitted keyword set as $T = (S, T_1, T_2, T_3, \{\lambda'_j, \mu'_j\}_{j \in S})$. Finally he sends T to CSP. Notably, $\{I'_1, \dots, I'_t\}$ represent the positions of interested keywords in dictionary.

Ciphertext search After verifying that the attribute set S satisfies the specified access policy \mathcal{T} , CSP calls $\{0, 1\} \leftarrow \text{Search}(T, I, C)$ algorithm to return the relevant results with the following steps:

Step 1: If x is a leaf node, it sets $j' = att(x)$. And if $j' \in S$, it computes D_x through the following equations. Otherwise, $D_x = \perp$.

$$D_x = e(\lambda'_{j'}, \delta_x) / e(\mu'_{j'}, \theta_x) = e(g, g)^{rsq_x(0)}.$$

Step 2: If x is a non-leaf node, it calls the recursive algorithm to compute D_x . For all children x' of node x , it computes $D_{x'}$. Let ω_x be an arbitrary k_x -size set of children x' such that $D_{x'} \neq \perp$. If there is no such set, then $D_x = \perp$. Otherwise, it computes $D_x = \prod_{x' \in \omega_x} D_{x'}^{\Delta_{i, \omega_x}^{(0)}} = e(g, g)^{rsq_x(0)}$, where $i = index(x')$, $\omega'_x = \{index(x') : x' \in \omega_x\}$.

Step 3: Then CSP verifies whether (1) holds. If yes, CSP returns relevant results to DU; otherwise, it returns \perp . Notably, $D_r = e(g, g)^{rsqr(0)} = e(g, g)^{rsr_2}$.

$$e(\prod_{i=1}^t \delta_i E_0, T_2) = e(E_2, T_1) D_r e(E_1, T_3). \quad (1)$$

Remark¹ In practical systems (such as PHR), every patient specifies his own access policy. Therefore, even though some unauthorized DUs have the rights to generate search tokens, they cannot gain relevant records as the submitted attributes do not match the access structure. Besides, the search token is obfuscated by a random element $s \in \mathcal{Z}_q$ such that the malicious CSP is unable to distinguish two trapdoors generated from the same keyword set. Therefore, the record confidentiality and trapdoor unlinkability can be achieved in our scheme. Meanwhile, as a large number of computational operations (e.g., the most time-consuming pairing operation) are outsourced to CSP, the computational burden of resource-limited cloud clients is minimized. In conclusion, our scheme is efficient and feasible in actual scenarios due to its low computational cost and data confidentiality protection.

Correctness analysis

To show that Eq. 1 is correct, we suppose the attribute set of DU matches the specified access policy, and the submitted keyword set, $W' \subseteq W$, is included in the record, then the correctness of the scheme can be verified as follows:

First, we have

$$\begin{aligned} e(E_2, T_1) &= e(g, g)^{s\alpha\gamma r_1 \sum_{i=1}^t h_2(w'_i)} \\ e(E_1, T_3) &= e(g, g)^{s\alpha\gamma r_2 - r s r_2} \\ D_r &= e(g, g)^{r s r_2} \end{aligned}$$

Then, we can get

$$e(E_2, T_1) D_r e(E_1, T_3) = e(g, g)^{s\alpha\gamma(r_2 + r_1 \sum_{i=1}^t h_2(w'_i))}$$

And we know

$$\begin{aligned} e(\prod_{i=1}^t \delta_i E_0, T_2) &= e(g^{r_2\alpha + r_1\alpha \sum_{i=1}^t h_2(w'_i)}, g^{s\gamma}) \\ &= e(g, g)^{s\alpha\gamma(r_2 + r_1 \sum_{i=1}^t h_2(w'_i))} \end{aligned}$$

Finally, we verify that Eq. 1 holds:

$$e(\prod_{i=1}^t \delta_i E_0, T_2) = e(E_2, T_1) D_r e(E_1, T_3).$$

Security and performance analysis

In this section, for security, we formally prove the security properties of our scheme and show it coincides with secure requirements. For actual performance evaluation, we first compare it with other analogous schemes in terms of theoretical performance (in here, it means computational

complexity). Furthermore, we conduct empirical study over a real-world dataset to analyze its efficiency and feasibility in practice.

Security analysis

Aiming to prove the security of our scheme, we first analyze its security properties, and then give the formal security proof. Not only the encrypted records are confidential to non-authorized DUs, but also our scheme can resist collusion attack. For example, Alice has the secret key corresponding to attributes ‘A’ and ‘B’, Bob has the secret key associated with attributes ‘B’ and ‘C’, while they cannot gain the secret key related to attributes ‘A’ and ‘C’ through colluding due to the random element $r_j \in \mathcal{Z}_q$. In our scheme, plaintext keywords are encrypted to ciphertext based on access policy before outsourced to vicious CSP so that the sensitive information cannot be gained from ciphertext.

Only the authorized DUs whose attributes match the specified access policy, can receive the relevant encrypted records. Since each DU is distributed a unique secret key sk , the malicious CSP or adversary cannot forge legal search token. Moreover, vicious CSP or adversary cannot deduce keyword set from search token $T = (S, T_1, T_2, T_3, \{\lambda'_j, \mu'_j\}_{j \in S})$. Therefore, our scheme can achieve keyword confidentiality as the search token is obscured by random elements.

To ensure the scheme secure, malicious CSP must not infer any valuable information except the search results. If there exists no probabilistic polynomial-time adversary \mathcal{A} who can deduce any plaintext information, then our scheme is selectively secure against chosen-keyword attack. The security of m^2 -ABKS scheme can be ensured by the following theorems which have the same proofs as the literature [7, 11]. Unfortunately, the proof of Theorem 1 is based on the generic bilinear group model [12] and the random oracle [13]. Therefore, the security of our scheme may be reduced for the sake of performance and functionality when compared with more secure CP-ABE scheme [14].

Theorem 1 *m^2 -ABKS scheme is selectively secure against chosen-keyword attack based on the generic bilinear group model. Where the hash function h_1 is modeled as a random oracle and h_2 is defined as a one-way hash function.*

Proof Let hash function h_1 be modeled as a random oracle and h_2 be one-way hash function, then we prove that scheme is selectively secure against chosen-keyword attack in the random oracle.

In SCKA game, adversary \mathcal{A} attempts to distinguish $X^{r_1 h_2(w_0)}$ from $X^{r_1 h_2(w_1)}$. Given a random element $v \in \mathcal{Z}_q$, the probability of distinguishing $X^{r_1 h_2(w_0)}$ from g^v is the

¹ m^2 -ABKS also supports user revocation (UR), the majority of existing UR schemes can be attached to m^2 -ABKS without any variation. As it is beyond the focus of this paper, we will not discuss the detail of UR in m^2 -ABKS due to space limit.

same as that of distinguishing $X^{r_1 h_2(w_1)}$ from g^ν , where $X = g^\alpha$. The SCKA game is shown as follows:

Setup: The challenger \mathcal{C} first selects $\alpha, \beta, \gamma \in \mathcal{Z}_q$ and generates public parameters $(e, g, q, g^\alpha, g^\beta, g^\gamma)$ for \mathcal{A} . Then \mathcal{A} chooses an access policy \mathcal{T}' and sends it to \mathcal{C} . $h_1(j)$ is simulated as follows: if attribute j has been queried before, \mathcal{C} selects a random element $r'_j \in \mathcal{Z}_q$ and adds (j, r'_j) to \mathcal{O}_{h_1} and returns $g^{r'_j}$; otherwise, \mathcal{C} retrieves r'_j from \mathcal{O}_{h_1} and returns $g^{r'_j}$.

Phase 1: \mathcal{A} queries \mathcal{O}_{KeyGen} and \mathcal{O}_{Trap} as follows:

- $\mathcal{O}_{KeyGen}(S, mk, \mathcal{T})$. \mathcal{C} first chooses $r^* \in \mathcal{Z}_q$ and computes $\pi = g^{\alpha\gamma - r^*/\beta}$, then \mathcal{C} selects random element $r_j^* \in \mathcal{Z}_q$ and computes $\lambda_j = g^{r^*} h_1(j)^{r_j^*}$, $\mu_j = g^{r_j^*}$ for each attribute $j \in S$. Finally, \mathcal{C} returns the tuple $(S, \pi, \{\lambda_j, \mu_j\}_{j \in S})$.
- $\mathcal{O}_{Trap}(sk, W^*)$. \mathcal{C} first issues $\mathcal{O}_{KeyGen}(mk, S)$ oracle in order to gain secret key $sk = (S, \pi, \{\lambda_j, \mu_j\}_{j \in S})$, then \mathcal{C} selects random element $s \in \mathcal{Z}_q$ and computes $T_1 = \prod_{i=1}^t g^{s\alpha h_2(w_i)}$, $T_2 = g^{s\gamma}$, $T_3 = \pi^s$. If the attribute set S satisfies the access policy, \mathcal{C} adds W^* to the keyword set List L_W .

Challenge Phase: Given keyword sets W_0, W_1 which do not belong to keyword set List L_W , \mathcal{C} selects random elements $r_1, r_2 \in \mathcal{Z}_q$ and computes secret shares of r_2 for each leaf node in tree \mathcal{T}' . Then \mathcal{C} outputs a random bit $b^* \in \{0, 1\}$, if $b^* = 0$, it outputs $\{\delta_i = g^{\nu h_2(w_i)}\}_{i \in \{1, \dots, m\}}$, $E_0 = X^{r_2}$, $E_1 = Y^{r_2}$, $E_2 = Z^{r_1}$, $\{\delta_y = g^{q_y(0)}, \theta_y = h_1(att(y))^{q_y(0)}\}_{y \in ln}$. Otherwise, \mathcal{C} returns $\{\delta_i = X^{r_1 h_2(w_i)}\}_{i \in \{1, \dots, m\}}$, $E_0 = X^{r_2}$, $E_1 = Y^{r_2}$, $E_2 = Z^{r_1}$, $\{\delta_y = g^{q_y(0)}, \theta_y = h_1(att(y))^{q_y(0)}\}_{y \in ln}$, where $att(y) = j \in S$.

Phase 2: This phase has the same procedures as in SCKA game.

We notice that if \mathcal{A} can construct $X^{\xi r_1 h_2(w_i)}$ for some g^ξ included in the oracle outputs he has already queried, then \mathcal{A} is able to distinguish $X^{r_1 h_2(w_i)}$ from g^ν , where $X = g^\alpha$. Next we need to prove that \mathcal{A} can construct $e(g, g)^{\xi \alpha r_1 h_2(w_i)}$ for some g^ξ with a negligible probability, in other words, \mathcal{A} do not have an non-negligible advantage in the SCKA game. We notice that term r_1 can only be found in the term γr_1 , so it requires that ξ should contain γ so as to construct $e(g, g)^{\xi \alpha r_1 h_2(w_i)}$. Let $\xi = \xi' \gamma$ for some ξ' , then \mathcal{A} just needs to construct $e(g, g)^{\xi' \alpha r_1}$ through using term γr_1 . As $\beta r_2(\alpha\gamma - r^*)/\beta = r_2(\alpha\gamma - r^*)$, \mathcal{A} needs to cancel $r_2 r^*$ through using terms r^* and $q_r(0)$. However, it is difficult to construct $r_2 r^*$ as above terms can be reconstructed only when the submitted attribute set matches the specified access policy \mathcal{T}' . In conclusion, \mathcal{A} can break the SCKA game with a negligible advantage. Therefore, our scheme

is selectively secure against chosen-keyword attack in the random oracle. \square

To the best of our knowledge, completely protecting the security of search token is impossible as \mathcal{A} can encrypt a keyword with his choice. As a result, \mathcal{A} can infer whether keyword ciphertext and search token correspond to the same keyword set. Assume \mathcal{A} has an non-negligible probability of learning the keyword set from keyword ciphertext and search token, then the keyword secrecy can be guaranteed.

Theorem 2 *Our scheme can gain keyword secrecy in the random oracle when given the one-way hash functions h_2 .*

Proof We first construct a challenger \mathcal{C} who conducts the following keyword secrecy game.

Setup: \mathcal{C} first chooses random elements $\alpha, \beta, \gamma \in \mathcal{Z}_q, \nu \in G_1$, then selects one hash function $h_2 : \{0, 1\}^* \rightarrow \mathcal{Z}_q$, finally sets public key $pk = (e, g, g^\alpha, g^\beta, g^\gamma, \nu)$, $mk = (\alpha, \beta, \gamma)$.

\mathcal{C} simulates the random oracle $\mathcal{O}_{h_1(j)}$ as follows. If the attribute j has not been queried before, \mathcal{C} randomly selects element $r_j \in \mathcal{Z}_q$, then adds (j, r_j) to \mathcal{O}_{h_1} and outputs g^{r_j} . Otherwise, \mathcal{C} retrieves r_j from \mathcal{O}_{h_1} and outputs g^{r_j} .

Phase 1: The adversary \mathcal{A} adaptively issues the following two oracles for polynomial-time times.

- \mathcal{O}_{KeyGen} : \mathcal{C} runs algorithm $sk \leftarrow \text{KeyGen}(mk, S, \mathcal{T})$ and sends sk to \mathcal{A} , then it adds \mathcal{T} to the list l_{KeyGen} .
- \mathcal{O}_{Trap} : \mathcal{C} first runs the oracle $\mathcal{O}_{KeyGen}(\mathcal{T})$ to gain secret key $sk = (\mathcal{T}, \{\delta_y, \theta_y | y \in ln(\mathcal{T})\})$, then calls $T \leftarrow \text{Trap}(sk, W)$ algorithm and returns T to \mathcal{A} .

Challenge Phase: \mathcal{A} first chooses an attribute set S' , then \mathcal{C} selects an access control policy \mathcal{T}' and computes $sk' \leftarrow \text{KeyGen}(mk, S', \mathcal{T}')$. Given attribute set S' and private key sk' , \mathcal{A} randomly chooses a keyword set W' and computes ciphertext C' and T' , where S' should satisfy the requirements specified in the keyword secrecy game.

Guess: \mathcal{A} first outputs a keyword W' and sends it to \mathcal{C} , then \mathcal{C} computes $C' \leftarrow \text{Enc}(pk, W, \mathcal{P})$. If $\text{Search}(T', I') = 1$, then \mathcal{A} wins the game.

Suppose \mathcal{A} has issued τ different keyword sets before returning W' , and the probability for \mathcal{A} winning the keyword secrecy game is at most $\frac{1}{|\mathcal{W}'| - \tau} + \epsilon$. As the size of remaining keyword set space is $|\mathcal{W}'| - \tau$ and h_2 is one way hash function, \mathcal{A} has a negligible probability ϵ to learn W' from $h_2(W')$. Therefore, the probability of winning the keyword secrecy game is at most $\frac{1}{|\mathcal{W}'| - \tau} + \epsilon$ if it has queried τ distinct keyword sets, thus m^2 -ABKS scheme can achieve keyword secrecy. \square

Table 2 Theoretical performance analysis

Algorithms	m ² -ABKS	CP-ABE [7]	ABKS-UR [8]	CP-ABKS [11]
Setup	$3e_1$	$2e_1 + p + e_2$	$3ne_1 + p + e_2$	$3e_1$
KeyGen	$(2n + 2)e_1 + nh_1$	$(2n + 2)e_1 + nh_1$	$3ne_1 + e_2$	$(2n + 2)e_1 + nh_1$
Enc	$(2l + 3)e_1 + mh_2 + lh_1$	$(2l + 1)e_1 + p + lh_1$	$(3l + 2)e_1 + e_2 + mh_2$	$(2l + 4)e_1 + lh_1 + h_2$
Trap	$(2n + t + 2)e_1 + th_2$	--	$(3n + 1)e_1 + h_2$	$(2n + 4)e_1 + h_2$
Search	$(2n + 2 + t)p + e_2$	$(2n + 1)p + e_2$	$(3n + 1)p + e_1$	$(2n + 3)p + e_2$

• n : Number of attributes submitted by DU; l : Number of attributes in policy; m : Number of keyword fields; t : Number of interested keyword submitted by DU

• --: No this operation

Security of PHR with m²-ABKS Here, we analyze security of our scheme in PHR. With regard to encryption for personal health records, we employ the traditional symmetric encryption algorithm to guarantee records confidentiality as long as the secret key s is not leaked to unauthorized DUs. As for the index, its security can be achieved through established access policy \mathcal{P} and random elements r_1, r_2 . Thus, the adversary cannot deduce corresponding keyword set even he gains the index, as it is difficult to construct r_2r^* . While for the search token, \mathcal{A} cannot forge legal search token through collusion attack. In other words, \mathcal{A} can compromise the confidentiality of PHR data only if he has legal attribute set, corresponding secret key and symmetric key.

Performance analysis

In this part, we analyze the performance evaluation of our scheme in terms of asymptotic computational complexity (or theoretical performance analysis) and actual performance analysis through exploiting the Type A curves within the Paring Based Cryptography (PBC) library. The experiments are implemented on an Ubuntu 15.04 Server with Intel Core i5 Processor 2.3 GHz through using C language. In PBC Library, the Type A is denoted as $E(F_q) : y^2 = x^3 + x$, 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. As for asymptotic computational complexity, we consider several computational operations, namely, exponentiation operation e_1 in G_1 , exponentiation operation e_2 in G_2 , hash operation h_1 which maps a bit-string to an element of G_1 , pairing operation p , and hash operation h_2 which maps a bit-string to an element of \mathcal{Z}_q . Notice that h_2 is much more efficient than other operations. In Table 2 we demonstrate that the computational complexities of m²-ABKS, CP-ABE [7], CP-ABKS [11] and ABKS-UR [8] schemes.

In Table 2, we take into account several algorithms, such as Setup, KeyGen, Enc, Trap and Search. In KeyGen phase,

the computational burden of m²-ABKS, CP-ABE and CP-ABKS schemes is just slightly less than that of ABKS-UR due to the additional hash operations nh_1 and varies with increasing n . In Enc phase, the computational overhead of our scheme is less than that of CP-ABE scheme as it needs to encrypt multiple keywords in records and attributes in policy simultaneously, and CP-ABKS scheme is slightly more efficient than m²-ABKS scheme as it just supports single keyword search. While our scheme is still slightly more efficient than ABKS-UR scheme in spite of additional lh_1 hash operations. With respect to Trap algorithm, as the hash operation h_2 is much more efficient than other operations and the value of t is small, the computational overhead of m²-ABKS scheme is similar to that of CP-ABKS scheme. However, our scheme is much more efficient than ABKS-UR scheme. For the Search algorithm, the computational burden of m²-ABKS and CP-ABKS schemes are much less than that of ABKS-UR scheme due to less pairing operations. Therefore, our scheme is efficient and scalable in actual applications without bringing in much computational burden to some extent.

Moreover, we also conduct empirical study over real-world dataset, namely Enron dataset², to evaluate the actual performance of the aforementioned schemes. The dataset contains half million records from around 150 users and has been also used in many other works. In line with ABKS-UR scheme [8], we randomly select 10,000 files from this dataset and run experiments for 100 times. Let the number of keyword fields be 1000, we vary the number of attributes in private key and policy and the number of interested keywords submitted by DU to test the actual performance evaluation in Figs. 3 and 4.

Obviously, the computational burden of ABKS-UR scheme is much heavier than other schemes in Setup algorithm and deteriorates with increasing the value of l . For convenience, we first set the value of l as 2, then show the computational overhead occupied by Setup algorithm among various schemes in Fig. 3. When the value of l

²<http://www.cs.cmu.edu/~enron/>

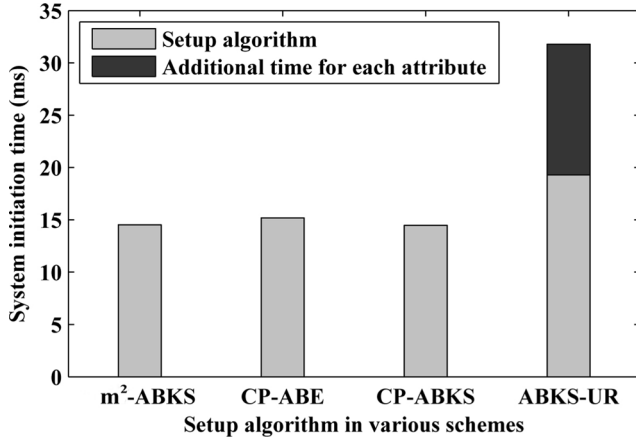


Fig. 3 Performance comparison in setup algorithm

increases, the computational cost of Setup algorithm in ABKS-UR scheme dramatically becomes larger yet that of other schemes remains unchanged as the computational overhead of m²-ABKS, CP-ABE scheme and CP-ABKS scheme is not affected by the value of l .

In Fig. 4a we show that the computational cost of private key generation in KeyGen algorithm varies with the variable n . We notice that CP-ABE, CP-ABKS and m²-ABKS schemes have the same computational cost $(2n + 2)e_1 + nh_1$, while the computational cost of ABKS-UR scheme is $(3n + 1)e_1 + e_2$. Therefore, ABKS-UR scheme has slightly larger computational burden than other three schemes with increasing n .

In Fig. 4b we demonstrate the ciphertext generation time (encrypting files or building indexes) for 10,000 files with fixed 1000 keyword fields ($m = 1000$) in Enc algorithm. Through varying the value of l from 1 to 100, we notice that the computational burden of our scheme is similar to that of CP-ABE and CP-ABKS schemes. Although ABKS-UR scheme needs $(3l + 2)$ exponentiation operations e_1 , its computational burden is slightly larger than that of other three schemes owing to lack of lh_1 hash operations. Notice that mh_2 can be ignored as the hash operation h_2 is much more efficient than hash operation h_1 . However, as Enc algorithm

is one-time cost and does not affect user search experience, its computational cost of is completely acceptable in practice.

Figure 4c shows that the computational cost of Trap algorithm in all schemes (except for CP-ABE scheme) changes with the value of m , while the computational cost of our scheme is affected by two variables n, t . In CP-ABE scheme, there exists no trapdoor operation as it cannot support search queries based on keywords. For comparison, we set $t = 10$, and vary n from 1 to 50 to test the actual performance of Trap algorithm. We notice that the computational cost of Trap algorithm in m²-ABKS scheme is less efficient than CP-ABKS scheme. However, both m²-ABKS and CP-ABKS schemes are superior to ABKS-UR scheme. Obviously, our scheme brings no additional computational burden in spite of supporting multi-keyword search. In addition, the advantages of m²-ABKS and CP-ABKS schemes are even highlighted as n increases, which justifies with our theoretical study in Table 2.

The computational cost of Search algorithm is presented in Fig. 4d. As the computational cost of ciphertext search is affected by two variables n and t , we set $t = 10$ and vary n from 1 to 50 in order to facilitate comparison. As m²-ABKS scheme mainly needs $(2n + 12)p$ operation, and CP-ABE scheme (or CP-ABKS scheme) just needs $(2n + 1)p$ (or $(2n + 3)p$) operations, CP-ABE and CP-ABKS schemes outperform m²-ABKS scheme. While our scheme is still superior to ABKS-UR scheme when the value of n is larger as ABKS-UR scheme needs $(3n + 1)p + e_1$ operations. When the value of n is set as 50, our scheme just needs 0.9106s to search encrypted records when $t = 10$.

From above figures we notice that the actual performance evaluation is in complete accord with theoretical study of computation complexity shown in Table 2. Hence, our scheme is feasible and scalable in practical applications.

Related work

For simplicity, we roughly divide existing ciphertext retrieval work into two categories, i.e., key-based access

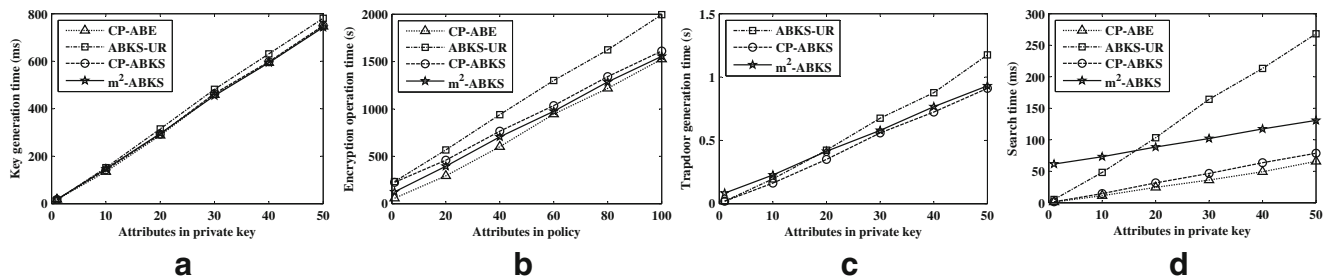


Fig. 4 Actual performance in different algorithms: (a) KeyGen algorithm; (b) Enc algorithm; (c) Trap algorithm; (d) Search algorithm

control and attribute-based access control. Next, we introduce two typical techniques which enable secure search over encrypted data, respectively.

Searchable encryption

Due to the heavy data storage and management burden, more and more individuals and companies are motivated to outsource storage and management to CSP. To protect data confidentiality, encryption has emerged as a fundamental solution to ensuring data security and privacy. As encryption seriously obsoletes the traditional private information retrieval over plaintext, developing an efficient and secure information retrieval technique over encrypted data is of paramount importance. Among the existing SE schemes [15–22, 24] which provide various securities and efficiencies, the typical scenario considered is the outsourced cloud storage in which there are three entities, namely data owner (DO), data user (DU) and CSP. DO first encrypts sensitive data and creates indexes, then outsources ciphertext and indexes to CSP. DU needs to generate a search token for interested keyword and submit it to CSP. Finally, CSP performs search over ciphertext and returns the relevant results.

Since Boneh et al. [4] first proposed the public key encryption with keyword search scheme, many subsequent SE schemes focusing on single-keyword search [15, 16, 19, 24] or multi-keyword search [17, 18, 20–23] have been proposed. However, these schemes are only applicable in single-user scenarios, which are impractical and unscalable in practice. To tackle above problem, advanced schemes [10, 25] are developed to support multi-user search through broadcast encryption and proxy re-encryption technologies. However, traditional SE schemes supporting multi-user setting may complicate the key management as the user revocation requires the updating of keys and ciphertext, which results in heavy computational burden. Therefore, exploring a fine-grained and effective SE scheme has drawn more attention in the industry and academic fields.

Attribute-based encryption

Over time, SE scheme has evolved into generalized concept predicate encryption, including various cryptographic primitives like IBE and ABE. ABE technique is considered as one of most suitable solutions for data access control in cloud computing, and various schemes [14, 26–28] enriched with different features have been proposed in the past few years. In ABE, the private key and ciphertext are associated with attribute or access policy, respectively. Only when the attributes match the specified access policy, the ciphertext can be decrypted by the DU. Depending on the policy definition, ABE can be classified into two groups,

Table 3 Functionality comparison in various schemes

Schemes	Multi-owner	Multi-keyword
CP-ABKS [11]	×	×
ABKS-UR [8]	✓	✓
sPHR [9]	✓	×
m ² -ABKS	✓	✓

*Although the ABKS-UR scheme mentions the multi-keyword functionality, no specific algorithm construction was given

namely key-policy ABE (KP-ABE) and ciphertext-policy (CP-ABE). The latter is considered to be more suitable for access control in cloud computing than KP-ABE as it can specify attributes that DUs need to possess.

The main limitation of SE scheme is that the DO has to know the identity of DU so as to encrypt data with the corresponding encryption key, which can be a grand challenge in multi-user setting. Although ABE can address this problem through distributing attributes to DU, it cannot effectively support expressive keyword search due to its complex attribute and access policy management. To the best of our knowledge, although the state-of-the-art ABE with keyword search schemes [8, 11, 29–31] allow DO to grant keyword search capability to authorized DUs, these schemes still cannot support multi-keyword search. Therefore, in this paper we focus on the multi-owner scenario to support multi-keyword search over encrypted data with CP-ABE scheme.

Through comparing with other analogous schemes [8, 9, 11], we show the functional advantages of m²-ABKS scheme in Table 3. Obviously, our scheme can support multi-keyword search in multi-owner settings. Although the ABKS-UR [8] has the same functionalities, it does not give the specific algorithms for generating the trapdoor and index.

Conclusion

In this paper we propose a multi-keyword search scheme in a more challenging multi-owner setting. With m²-ABKS scheme, multiple DOs are allowed to share with their records with flexible access policy and authorized DUs are permitted to issue search queries according their corresponding attributes. Additionally, the collusion attacks and unauthorized accesses can be effectively avoided. Different from existing schemes, our scheme supports fine-grained access control and multi-keyword search query in cloud storage. In particular, formal security analysis proves that our proposed scheme is selectively secure against chosen-keyword attack in random oracle. As a further contribution,

experimental results over a real-world dataset show its practice and efficiency in practice. As part of our future work, we need to further enhance the security on a firmer theoretical foundation as well as support more expressive search queries.

Acknowledgments This work was supported by the National High Technology Research and Development Program (863 Program) (No. 2015AA016007, No. 2015AA017203), the Key Program of NSFC (No. U1405255, No. U1135002), the Changjiang Scholars and Innovation Research Team in University (No. IRT1078), the Fundamental Research Funds for the Center Universities (No. JY10000903001) and the Major Nature Science Foundation of China (No. 61370078, No. 61309016).

References

1. Wang, C., Wang, Q., Ren, K., and et al: Privacy-preserving public auditing for data storage security in cloud computing. In: *Proceedings of 29th IEEE International Conference on Computer Communications (INFOCOM'10)*, pp. 525–533. doi:10.1109/INFCOM.2010.5462173, 2010.
2. Ren, Y. J., Shen, J., Wang, J., and et al, Mutual verifiable provable data auditing in public cloud storage. *J. Intern. Technol.* 16(2):68–81, 2015.
3. Yu, S. C., Wang, C., Ren, K., and et al: Achieving secure, scalable, and fine-grained data access control in cloud computing. In: *Proceedings of 29th IEEE International Conference on Computer Communications (INFOCOM'10)*, pp. 534–542. doi:10.1109/INFCOM.2010.5462174, 2010.
4. Boneh, D., Crescenzo, G. D., Ostrovsky, R., and et al: Public key encryption with keyword search. In: *Proceedings of 23th International Conference on Theory and Applications of Cryptographic Techniques (EUROCRYPT'04)*, pp. 506–522. doi:10.1007/b97182, 2004.
5. Fu, Z. J., Ren, K., Shu, J. G., and et al: Enabling personalized search over encrypted outsourced data with efficiency improvement. *IEEE Transactions on Parallel and Distributed Systems*. doi:10.1109/TPDS.2015.2506573, 2015.
6. Goyal, V., Pandey, O., Sahai, A., and et al: Attribute-based encryption for fine-grained access control of encrypted data. In: *Proceedings of 13th ACM Conference on Computer and Communications Security (CCS'06)*, pp. 89–98. doi:10.1145/1180405.1180418, 2006.
7. Bethencourt, J., Sahai, A., and Water, B.: Ciphertext-policy attribute-based encryption. In: *Proceedings of 28th IEEE Symposium on Security and Privacy (S&P'07)*, pp. 321–334. doi:10.1109/SP.2007.11, 2007.
8. Sun, W. H., Yu, S. C., Lou, W. J., and et al: Protecting your right: attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud. In: *Proceedings of 33th IEEE International Conference on Computer Communications (INFOCOM'14)*, pp. 226–234. doi:10.1109/INFCOM.2014.6847943, 2014.
9. Li, M., Yu, S. C., Ren, K., and et al: Securing personal health records in cloud computing: patient-centric and fine-grained data access control in multi-owner settings. In: *Proceedings of 6th International ICST Conference on Security and Privacy in Communication Networks (SecureComm'10)*, pp. 89–106. doi:10.1007/978-3-642-16161-2, 2010.
10. Hwang, Y. H., and Lee, P. J.: Public key encryption with conjunctive keyword search and its extension to a multi-user system. In: *Proceedings of first International Conference on Pairing-Based Cryptography (Pairing'07)*, pp. 2–22. doi:10.1007/978-3-540-73489-5, 2007.
11. Zheng, Q. J., Xu, S. H., and Ateniese, G.: VABKS: Verifiable attribute-based keyword search over outsourced encrypted data. In: *Proceedings of 33th IEEE International Conference on Computer Communications (INFOCOM'14)*, pp. 522–530. doi:10.1109/INFCOM.2014.6847976, 2014.
12. Boneh, D., Boyen, X., and Goh, E. J.: Hierarchical identity based encryption with constant size ciphertext. In: *Proceedings of 24th Annual International Conference on Theory and Applications of Cryptographic Techniques (EUROCRYPT'05)*, pp. 440–456. doi:10.1007/b136415, 2005.
13. Bellare, M., and Rogaway, P.: Random oracles are practical: a paradigm for designing efficient protocols. In: *Proceedings of first ACM Conference on Computer and Communications Security (CCS'93)*, pp. 62–73. doi:10.1145/168588.168596, 1993.
14. Waters, B.: Ciphertext-policy attribute-based encryption: an expressive, efficient, and provably secure realization. In: *Proceedings of 14th International Conference on Practice and Theory in Public Key Cryptography (PKC'11)*, pp. 53–70. doi:10.1007/978-3-642-19379-8.
15. Curtmola, R., Garay, J. A., Kamara, S., and et al: Searchable symmetric Encryption: improved definitions and efficient constructions. In: *Proceedings of 13th ACM Conference on Computer and Communications Security (CCS'06)*, pp. 79–88. doi:10.1145/1180405.1180417, 2006.
16. Bellare, M., Boldyreva, A., and O'Neill, A.: Deterministic and efficiently searchable encryption. In: *Proceedings of 27th Annual International Conference on Advances in Cryptology (CRYPTO'07)*, pp. 535–552. doi:10.1007/978-3-540-74143-5, 2007.
17. Boneh, D., and Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: *Proceedings of 4th Conference on Theory of Cryptography (TCC'07)*, pp. 535–55. doi:10.1007/978-3-540-70936-7, 2007.
18. Katz, J., Sahai, A., and Waters, B.: Predicate encryption supporting disjunctions, polynomial equations, and inner products. In: *Proceedings of 27th Annual International Conference on Theory and Applications of Cryptographic Techniques (EUROCRYPT'08)*, pp. 146–162. doi:10.1007/978-3-540-78967-3, 2008.
19. Li, J., Wang, Q., Cao, C., and et al: Fuzzy keyword search over encrypted data in cloud computing. In: *Proceedings of 29th IEEE International Conference on Computer Communications (INFOCOM'10)*, pp. 1–5. doi:10.1109/INFCOM.2010.5462196, 2010.
20. Cao, N., Wang, C., Li, M., and et al, Privacy-preserving multi-keyword ranked search over encrypted cloud data. *IEEE Trans. Parallel Distrib. Syst.* 25(1):222–233, 2014. doi:10.1109/TPDS.2013.45.
21. Miao, Y. B., Liu, J., and Ma, J. F.: Fine-grained searchable encryption over encrypted data in multi-clouds. In: *Proceeding of 10th International Conference on Wireless Algorithms, Systems, and Applications (WASA 2015)*, pp. 407–416. doi:10.1007/978-3-319-21837-3-40, 2015.
22. Li, H. W., Liu, D. X., Dai, Y. S., and et al, Enabling efficient multi-keyword ranked search over encrypted mobile cloud data through blind storage. *IEEE Trans. Emerging Topics Comput.* 3(1):127–138, 2015. doi:10.1109/TETC.2014.2371239.
23. Fu, Z. J., Sun, X. M., Liu, Q., and et al, Achieving efficient cloud search services: multi-keyword ranked search over encrypted cloud data supporting parallel computing. *IEICE Trans. Commun.* E98-B(1):190–200, 2015.
24. Xia, Z. H., Wang, X. H., Sun, X. M., and et al, A secure and dynamic multi-keyword ranked search scheme over encrypted

- cloud data. *IEEE Trans. Parallel Distrib. Syst.* 27(2):340–352, 2016. doi:[10.1109/TPDS.2015.2401003](https://doi.org/10.1109/TPDS.2015.2401003).
25. Bao, F., Deng, R. H., Ding, X. H., and et al: Private query on encrypted data in multi-user settings. In: *Proceedings of 4th International Conference on Information Security Practice and Experience (ISPEC'08)*, pp. 71–85. doi:[10.1007/978-3-540-79104-1](https://doi.org/10.1007/978-3-540-79104-1), 2008.
 26. Ostrovsky, R., Sahai, A., and Waters, B.: Attribute-based encryption with non-monotonic access structures. In: *Proceedings of 14th ACM Conference on Computer and Communications Security (CCS'07)*, pp. 195–203. doi:[10.1145/1315245.1315270](https://doi.org/10.1145/1315245.1315270), 2007.
 27. Chase, M., and Chow, S.M.S.: Improving privacy and security in multi-authority attribute-based encryption. In: *Proceedings of 16th ACM Conference on Computer and Communications Security (CCS'09)*, pp. 121–130. doi:[10.1145/1653662.1653678](https://doi.org/10.1145/1653662.1653678), 2009.
 28. Lewko, B.A., Okamoto, T., Sahai, A., and et al, Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In: *Proceedings of 29th Annual International Conference on Theory and Applications of Cryptographic Techniques (EUROCRYPT'10)*, pp. 62–91. doi:[10.1007/978-3-642-13190-5](https://doi.org/10.1007/978-3-642-13190-5), 2010.
 29. Wang, C. J., Li, W. T., Li, Y., and et al: A ciphertext-policy attribute-based encryption scheme supporting keyword search function. In: *Proceeding of 5th International Symposium on Cyberspace Safety and Security (CSS 2013)*, pp. 377–386. doi:[10.1007/978-3-319-03584-0-28](https://doi.org/10.1007/978-3-319-03584-0-28), 2013.
 30. Khader, D.: Introduction to attribute based searchable encryption. In: *Proceeding of 11th International Conference on Communications and Multimedia Security (Communications and Multimedia Security 2014)*, pp. 131–135. doi:[10.1007/978-3-662-44885-4-11](https://doi.org/10.1007/978-3-662-44885-4-11), 2014.
 31. Liang, K. T., and Susilo, W., Searchable attribute-based mechanism with efficient data sharing for secure cloud storage. *IEEE Trans. Inf. Forens. Secur.* 10(9):1981–1992, 2015. doi:[10.1109/TIFS.2015.2442215](https://doi.org/10.1109/TIFS.2015.2442215).