IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 27, 2021, accepted August 4, 2021, date of publication August 16, 2021, date of current version August 23, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3104899

OO-ABMS: Online/Offline-Aided Attribute-Based Multi-Keyword Search

SHAHZAD KHAN[1], MAHDI ZAREEI[2], (Senior Member, IEEE), SHAWAL KHAN[3], FAISAL ALANAZI[4], MASOOM ALAM[3], AND ABDUL WAHEED[5,6]

[1]Department of Information Security, National University of Sciences and Technology, Islamabad 44000, Pakistan
[2]School of Engineering and Sciences, Tecnologico de Monterrey, Zapopan 45201, Mexico
[3]Department of Computer Science, COMSATS Institute of Information Technology, Islamabad 44000, Pakistan
[4]College of Engineering, Prince Sattam Bin Abdulaziz University, Al-Kharj 16278, Saudi Arabia
[5]Department of Information Technology, Hazara University Mansehra, Mansehra 21120, Pakistan
[6]School of Electrical and Computer Engineering, Seoul National University, Seoul 08826, South Korea

Corresponding author: Shahzad Khan (shehzad@sbbu.edu.pk)

ABSTRACT Recently, Attribute-Based Encryption (ABE) is used as a baseline technology for keyword searching mechanisms to cater users' security and privacy-related concerns who outsource their data to the powerful and resource-rich cloud server. Almost all the existing Attribute-Based Keyword Searching (ABKS) schemes either operate on a single-keyword or multiple; conjunctive-keyword search setting. These schemes work on the premise of an ''all-or-nothing'' retrieval mode, which greatly limits users' searching capabilities. The most recent work incorporates multi-keyword ranked search in attribute-based encryption, which improves user search results accuracy and the flexibility of retrieval mode. However, these schemes demand high computation overhead, which ultimately suffers searching efficiency. This dilemma prevents further research and application of ABKS schemes, especially mobile-resource constrained devices such as mobile phones. This paper proposes an Online/Offline-aided Attribute-Based Multi-keyword Search (OOABMS) scheme to delegate most laborious computation operations to the offline phase before acquiring the attribute-based access control policy or keywords. An online phase then quickly assembles the pre-computed index or trapdoor with the required specifics when it becomes known. Theoretical comparison and simulation results show that our scheme is more efficient and practical in the real world scenario for mobile cloud computing in terms of computation overhead and flexibility.

INDEX TERMS Attribute-based keyword search, multi-keyword ranked search, online/offline operations.

I. INTRODUCTION

Nowadays, cloud computing is considered a new enterprise IT service. In cloud computing architecture, online storage services, e.g., Amazon's S3, Azure cloud storage, and Apple iCloud, are regarded as efficient data storage services over the cloud. These services allow the individual users and companies to move their databases and application to remote cloud servers, with several unprecedented advantages such as flexibility and ubiquitous access, on-demand computing allocation, huge capital expenditure saving, etc. Since cloud users' sensitive data are stored on the public cloud servers, which are out of their controlled premises, it causes a significant source of worry for their data privacy. This privacy issue has a prime barrier in the adaptability of cloud services for a broader range of individual users and companies.

The associate editor coordinating the review of this manuscript and approving it for publication was Pedro R. M. Inácio.

Therefore, adequate privacy and security protection mechanism for outsource data is of great concern. Encrypting outsource data have been investigated in the literature as a fundamental mechanism for safeguarding the privacy of outsourced data against the untrusted cloud server [1]–[3]. However, it incurs a new challenge while outsourced encrypted data can be efficiently searched, as searching capabilities are often stripped from the user data after encryption. A simple way to get around this dilemma is to download all the outsourced data from the cloud and search them locally for the desired data. This indiscriminately downloading user data would consume too much computation, storage, and bandwidth of user resource constraint devices especially mobile devices.

Searchable encryption [4], [5] has been recently considered an efficient mechanism to perform searches over outsourced encrypted data. These schemes operate on the client/server model, where the client (i-e the data writers) encrypts the

data along with the associated keywords and upload it to the untrusted cloud server. The clients (i-e the data readers) then generate search tokens for the cloud server, where the cloud server then performs the requested search operations on behalf of the data users.

Searching on encrypted data in symmetric searchable encryption (SSE) [6] setting are called the single owner and single user (S/S) architecture, where the role of data user or reader is limited to the data owner or writer only. Searchable encryption with public-key encryption with keyword search (PEKS) setting is called the multi-owner and single user (M/S) architecture. Several schemes are available in the literature on SSE and PEKS settings. However, minimal work is done in the multi-owner and multi-user (M/M) setting. One potential application in this setting is encrypted data sharing. More than one data owner hosts their encrypted files and searches with multiple data users (e.g., mymedwall.com, 4shared.com, resulting in a more challenging scenario. Attribute-based encryption [7]–[9] was proposed to solve aforementioned problem. Sahai and Waters, in 2005 first time introduced this concept.

The attribute-based encryption schemes [10] can operate in two basic architectures: Ciphertext-Policy (CP-ABE) and Key-Policy (KP-ABE). CP-ABE architecture enables the data owner to associate its own defined access policy to ciphertext while the attribute authority embeds the claimed attribute set of data user in its private key. Any data user whose attributes set satisfies the access policy can perform a decryption operation. Using KP-ABE, the claimed set of attributes in the form of a secret key is attached to the access control policy by the attribute authority. Among the two variations of attribute-based encryption, CP-ABE provides more control to the data owner and has become more popular in the research community. Thus, in our proposed scheme, we deploy CP-ABE architecture.

At the same time, despite its appealing properties, researchers found some of its implementation shortcomings. In CP-ABKS, the computational workload in the keyword encryption (KE) is a direct function of the number of attributes associated with the keyword ciphertext. Similarly, in KP-ABKS, computation costs in token generation scale with the Boolean access formula assigned to the user's private key. An exacerbating factor is the cumbersome workload of pairing operations, which is much costly than other basic mathematical operations in each keyword encryption and token generation hence forcing these schemes for a worst-case scenario implementation. Consequently, these schemes are not appropriate for devices with resource utilization constraints, especially mobile phones.

There are broadly three approaches to enhance the efficiency of attribute-based keyword search (ABKS). Non-paring, as the name suggests, this approach avoids the time-consuming and most expensive bilinear operations with lighters one, i.e., Elliptic Curve Cryptography (ECC). However, this approach fails to avoid the underlying linearity problem of ABE. Outsourcing approach delegates most of the expensive computation operations to a resource-rich cloud server. As a result, data users are left with small or constant numbers of operations. This delegation can be done either for encryption or decryption end or can be performed at both ends. However, this approach needs an edge node at the underlying framework and cannot be deployed for all the ABKS schemes. The Online/Offline approach constructs an index and query token splitting into two phases. Before knowing the exact specifications, the most computation can be done for both index and token generation during the first phase. Then, it performs rapid assembling of an intermediate index or token when the specific becomes known during the second phase. Because of its pre-computation support, our proposed scheme uses this technique for its index and token generation.

A. CONTRIBUTIONS

We proposed a splitting technique to practically and efficiently address costly bilinear pairing operations in an attribute-based multi-keyword ranked search scheme. Based on this splitting, a vast amount of computation, both for keyword encryption and token generation, is shifted to the offline phase. The contribution of this work can be summarized as follows:

1) To our knowledge, we are the first to add on the multi-keyword ranked search scheme with online/offline technology for both indexes as well as token generation.
2) For comparative analysis, we equipped the ABKRS scheme [11] with the online/offline capability. Our construction inherits all the advantages of the basic scheme, including fine-grained data access control, the flexibility of retrieval mod and security in the same strict model, and making them practical in its low efficiency.
3) We provide detailed security proofs and informal performance analysis of our proposed construction.
4) The proposed OO-ABMS scheme is implemented along with the basic one ABKRS [11], and the simulation results demonstrate the practicality of our proposed scheme.

B. PAPER ORGANIZATION

The rest of this paper is organized as follows. We present some related work in Section 2; introduce the notation and preliminaries in Section 3; present system model and security definition in Section 4; present the proposed scheme and security analysis in Section 5; evaluate the performance in Section 6; finally, the paper concludes in Section 7.

II. RELATED WORK

This section presents a brief overview of contemporary techniques in the research field of searching over encrypted data.

A. S/S SEARCHABLE KEYWORD ENCRYPTION

Single-owner/single-user encryption allows a single data owner, and the very same data owner is authorized to perform search queries as a data user. Golle *et al.* [12] were the first to introduce the concept of conjunctive keyword searching on encrypted data in the single-owner/single-user architecture and proposed two relevant schemes. In the first technique, the size of the encrypted search index is the linearly increasing function of the number of underlying encrypted data. In the second technique, using bilinear paring, the size of the search index remains constant at the expense of computational cost. S/S setting has been researched extensively, with different cryptographic primitives such as symmetric-key setting [6], [13]–[15] asymmetric-key architecture [4], [16] supporting dynamic setting [17], or with complicated (conjunctive, boolean, or subset) queries [18], [19].

In the trade-off of security versus efficiency, two provably secure techniques [20] and [21], was recently proposed. Curtmola *et al.* [14] proposed a generic framework that is capable of combining any S/S scheme with broadcast encryption to construct a S/M searchable encryption scheme. The searchable encryption algorithm of Raykova *et al.* [20] is performing deterministic encryption, which exposes the underlying search interest along with the access pattern. Its searching cost is a linearly increasing function of the number of underlying encrypted documents. In [21] scheme of Yang *et al.* at the cost of efficiency; searching cost is a direct function of the number of keywords per document, is secure under the strong security assumption. In literature, the S/M or M/S schemes construct usually incorporate a trusted third party (TTP) for re-encryption of the indexes or user authentication, and no formal security proofs are provided.

B. M/M SEARCHABLE KEYWORD ENCRYPTION

Bellar *et al.* [4] construct, at the expense of a weaker security assumption of deterministic encryption and efficient scheme in the M/M architecture. Dong *et al.* [22] propose two variants of their scheme where each user needs to maintain its unique secret key to encrypt, search, and decrypt data. Both these constructs depend on the TTP key management server for their key generation. Bao *et al.* [23] propose a multi-user framework, where the data encryption and token generation are achieved via interactive algorithms. Hwang and Lee [24] propose a multi-user asymmetric encryption along with conjunctive keyword search. They introduce the concept of multi-receiver asymmetric key encryption to improve computation while reduces communication overhead. Wang *et al.* introduce three schemes [25]–[27] for searchable encryption in the M/M construct. These schemes either require cumbersome key management for the data user's end or depend on a TTP. Li *et al.* [28] propose two schemes for searching over encrypted data for an authorized keyword. In their schemes, data owners depend on trusted third parties for access control policies definition. Recently propose schemes [7]–[9] utilizes attribute-based encryption primitive which enables users to construct the access control policies for authorized data access. However, the required time for both index generation and trapdoor generation is linearly increasing with their associated set of attributes, respectively, which rendered them less suitable for limited-resource devices.

C. ATTRIBUTE-BASED ENCRYPTION

As a generalized derivation of Identity-Based Encryption (IBE), the attribute-based encryption (ABE) scheme maps the identity of a particular user through a set of attributes. ABE is one of the contemporary methods for enforcing access control policies. It incorporates the access control directives in the ciphertext and key, which controls the data user's decryption limits and secures the ciphertext [29]. There are two types of implementation available in the literature, Key-Policy ABE schemes where the decryption key is ascribed to the access tree [10], and Ciphertext-Policy ABE, where the access control policy is assigned to the ciphertext [30]. With the in-depth analysis of ABE, the research community finds some challenges in ABE schemes, specifically for resource constraints devices. There are many pairing-based operations in encryption and decryption, which incur huge computational costs on the end-users. Schemes in [31], [32] introduce other variants of attribute-based encryption to enhance encryption and key generation efficiency.

III. NOTATION AND PRELIMINARIES

Let $a \leftarrow S$ denotes selecting an element a from a set S randomly. Let p is a prime, $\mathbb{Z}_p = \{0, 1, \cdots, p-1\}$, and $\mathbb{Z}_p^* = \{1, \cdots, p-1\}$. U represents the set of universal attributes used to construct access control policies. We utilize preliminaries introduced in paper [10], [16] and [31].

A. BILINEAR MAP

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be three multiplicative cyclic groups of prime order p, having generator g_1, g_2 respectively. We define asymmetric bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ with the conditions:

- Non-degenerate $e(g_1, g_2) \neq 1$.
- Computable for all $u \in \mathbb{G}_1$, $v \in \mathbb{G}_2$, there exists an efficient computable algorithm for computing $e(u, v)$.
- Bilinear for all $v \in \mathbb{G}_1$, $v \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p^*$, $e(u^a, v^b) = e(u, v)^{ab}$.

When $\mathbb{G}_1$ and $\mathbb{G}_2$ represents the same group, the map becomes a symmetric bilinear one.

B. ACCESS TREE

Let T be an access structure tree, representing access control policies. Its leaf node is represented with an attribute, and the inner node denotes a threshold gate. Let num_v represent the number of children of a node v, where each child node is labeled from left to right as $1, \cdots, \text{num}_v$. Let k_v, $1 \leq$

$k_v \leq \text{num}_v$, be the threshold value labeled with node v, where $k_v = \text{num}_v$ represents the AND gate and $k_v = 1$ represents the OR gate. To facilitate depiction of an access tree, each function is described as:

- parent (v) represents the parent node of the given node v in T.
- index (v) denotes the assigned label for node v.
- attr (v) denotes the attached attribute to the leaf nod v.
- lvs (T) denotes the total number of leaf node attached to the given access structure tree T.
- T_v denotes the subtree of T at given node v.
- $f(s,\mathcal{T}) = \begin{cases} 1, & \text{if } S \text{ satisfies the access control policy } \mathcal{T} \\ 0, & \text{otherwise} \end{cases}$

C. SECRET DISTRIBUTION

Given a secret value s, the algorithm which assigned each fragment of the secret value to the non-leaf node is denoted by:

$\{q_v(0)|v \in lvs(T)\} \leftarrow \text{share}(T, S)$. For each node v, the above algorithm constructs a polynomial q_v of degree $k_N - 1$ in a top-down fashion (for leaf node $k_v = 1$):

- for root node v, set $q_v(0) = S$ and randomly choose $k_v - 1$ coefficients for polynomial q_v.
- for inner node v, set $q_v(0) = q_{\text{parent}(v)}(\text{index}(v))$, and randomly choose $k_v - 1$ coefficient for polynomial q_v.
- for leaf node v, set $q_v(0) = q_{\text{parent}(v)}(\text{index}(v))$.

D. SECRET CONSTRUCTION

For a given access control tree T and a set of leaves values $\{E_{v_1}, \cdots, E_{v_m}\}$, where $v_1, \cdots, v_m$ are the leaves of T. g, $h \leftarrow \mathbb{G}$, e a bilinear map, and $E_{u_j} = e(g,h)^{q_{u_j}(0)}$ for $1 \leq j \leq m$, where $q_{u_1}(0), \cdots, q_{u_m}(0)$ are the secret share of s associated to the nodes of T. We represent the algorithm for combining $e(g,h)^s$ by

$e(g,h)^s \leftarrow \text{Combine}(T, \{E_{v_1}, \cdots, E_{v_m}\})$. For a given node v, the following steps are carried out to re-build the secret S in a bottom-up fashion from access tree T:

- if attribute set S do not satisfy the access policy denoted by T_v, then set $E_v = \mathcal{T}$
- if attribute $\text{attr}(u_1), \cdots, attr(u_m)$ satisfies the access policy denoted by T_v, then calculate the following:
  - for leaf node v, set $E_v = E_{u_j}(0) = e(g,h)^{q_{u_j}(0)}$, where $v = u_j$ for some j.
  - for inner node v having children nodes $\{v_1, \cdots, v_{\text{num}}\}$, there exists a set of indexes S such that $|S| = k_v, j \in S$, and attributes set $\text{attr}(u_1), \cdots, attr(u_m)$ satisfy the access policy denoted by tree T_{v_j}. Set $E_v = \Pi_{j \in S} E_{v_j}^{\Delta_{v_j}} = e(g,h)^{q_v(0)}$, where $\Delta_{v_j} = \Pi_{l \in S, I \neq j} \frac{-j}{I-j}$.

when the algorithm stops, reconstructed secrets s, associated with the root of T
$E_{\text{root}} = e(g,h)^{q_{\text{root}}(0)} = e(g,h)^s$.

E. GENERIC BILINEAR GROUP

Let ψ_1,ψ_2, and ψ_T be three random encoding of the additive group $\mathbb{F}_p$ such that they are injective maps:

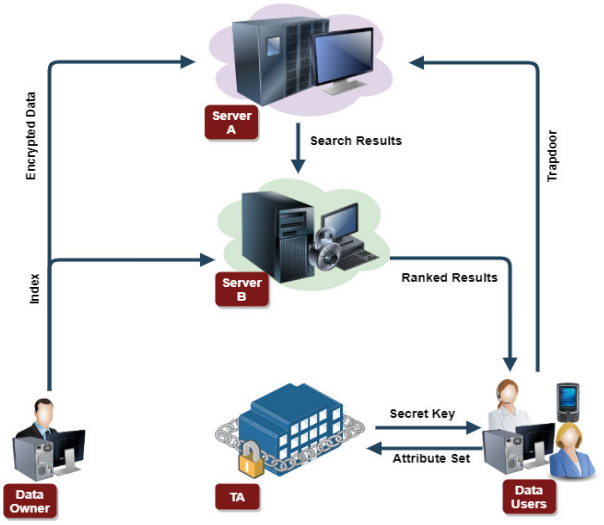


FIGURE 1. System model.

ψ_1,ψ_2,ψ_T;$\mathbb{F}_p \rightarrow \{0,1\}^m, m > 3log(p)$. Let $\mathbb{G}_1 = \{\psi_1(x)|x \in \mathbb{F}_p\}$, $\mathbb{G}_2 = \{\psi_2(x)|x \in \mathbb{Z}_p\}$ and $\mathbb{G}_T = \{\psi_T(x)|x \in \mathbb{F}_p\}$ represent generic bilinear groups. The adversary submits queries to the oracle model in order to simulate the operation in $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$, the hash function and the asymmetric bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$.

IV. SYSTEM MODEL AND SECURITY DEFINITION

A. SYSTEM MODEL

This section presents the system framework of proposed our scheme in Fig. 1, which consists of five entities: Authority, a fully trusted entity, needs to generate a public key and secret key of users and servers; the Data Owner (DO) uploads his/her keyword encryption and encrypted files to servers A and server B respectively; data users submit their generated trapdoors according to some set of keywords to server A; the server A, which receives the ciphertext keyword from the users and secret token from the data owner, conduct their part of designated search and sends the intermediate results to server B; server B, performs the final search operation of encrypted data. Note that both data owners and data users will generate keyword encryption and search tokens in the online/offline way.

B. ALGORITHM IN SYSTEM MODEL

The online/offline ABMS scheme consists of the following algorithms:

- $Setup(1^l) \rightarrow (PK, MK)$. The Authority generates a master key MK and initializes the public key PK.
- $Extract(MK, S) \rightarrow (SK)$. The Authority generates private key SK for a given user according to an attribute set S.
- $Offline.Enc(Pk, T) \rightarrow (IC)$. Data owner runs this algorithm to generate an intermediate ciphertext IC.
- $Online.Enc(Pk, IC, T) \rightarrow (CT_w)$. This algorithm encrypts keyword set w to generate ciphertext CT_w.

- $Offline.TokenGen(Pk, SK) \rightarrow (IT)$. Data user runs this algorithm to obtain an intermediate search token IT.
- $Online.TokenGen(Pk, IT, W') \rightarrow (TK)$. Data user executes this algorithm to obtain a search token TK for a the keyword set W'.
- $Search_A(TK, CT, Q_I, Sk_A) \rightarrow (\bar{R})$. Server A executes this algorithm, it takes the ciphertext CT, data user's interest query Q_I, token TK, and the secret key Sk_A of server A Sk_A. It returns intermediate search result $\bar{R}$ if attributes set satisfies access control structure T; terminates otherwise.
- $Search_B(\bar{R}, Sk_B) \rightarrow (R)$. Server B runs this algorithm, it takes its secret key Sk_B and intermediate search result $\bar{R}$ to return the relevance score R.

C. ADVERSARY MODEL

We let both the servers be semi-trusted which is the most adopted assumption in the literature for other construction as well [8], [9], [33]. Both the cloud servers correctly execute the designed protocol and infer sensitive information based on its data. We assume two types of malicious users; type-1 malicious users try to collude with server A only to get his secret key; type-2 malicious users refer to those users who try to collude with server B only gets his secret key.

D. DESIGN GOALS

The online/offline ABMS scheme should achieve the following security goals.

1) ADAPTIVE SECURITY AGAINST CHOSEN-KEYWORD ATTACK (CKA)

If the attacker is denied to obtain any matching search token, he would not be able to get any extra information about the keyword set only with the index in the adaptive security model. Namely, the adversary is unable to distinguish between the provided challenge encrypted keyword sets.

We demonstrate this security requirement through the following adaptive Chosen Keyword (CK) attack game.

Type-1 Adversary

Setup: After running the setup algorithm, the Challenger gives Sk_A and PK to $\mathcal{A}_1$.

Phase 1: The challenger $\mathcal{C}$ maintains a record of a keyword list L_{kw}, table E, a set D, which are all initially empty and sets an integer $j = 0$. The $\mathcal{A}_1$ can submit multiple queries to the following oracles.

- $O_{Extract}(S)$: The challenger runs Extract, sets $D = D \cup \{S\}$ and return to $\mathcal{A}_1$ the private key SK corresponding to S.
- $O_{TokenGen}(S, W)$: The challenger for a given S generates private key SK, sets $j = j+1$, and it also compute search token $Tk \leftarrow Online.Token$ $(PK, Offline.TokenGen(SK, PK), W)$ to give it to $\mathcal{A}_1$. It stores the entry (j, S, W) in table E.
- $O_{Search}(I, Tk, QI)$: The challenger runs $search_A$ and $search_B$ to compute the relevance score R and gives it to $\mathcal{A}_1$.

Challenge Phase: $\mathcal{A}_1$ select two keywords W_0 and W_1, and an access structure $\mathcal{T}^*$ and delivers to $\mathcal{C}$. In order to prevent $\mathcal{A}_1$ from trivially guessing, it requires that

- for all $S \in D, f(S, \mathcal{T}^*) \neq 1$
- $|W_0| = |W_1|$
- for all elements $\in E$, if $(S, \mathcal{T}^*) = 1$, appends W to L_{kw}; then $W_0 \cap W = W_1 \cap W$, $W \in L_{kw}$.

The challenger selects $\beta \xleftarrow{R} \{0, 1\}$, compute $I^* \leftarrow Online.Encrypt(PK,$ $Offline.Encrypt(PK, \mathcal{T}^*), W_\beta, \mathcal{T}^*)$ and delivers I^* to $\mathcal{A}_1$.

Phase 2: $\mathcal{A}_1$ acts as in phase 1 with the exception that he cannot query $O_{Extract}$ if $f(S, \mathcal{T}^*) = 1$, and also (S, W) cannot be submitted to $O_{TokenGen}$ if $W \cap W_0 \neq W \cap W_1$.

Guess: $\mathcal{A}_1$ guess and outputs a bit β', and if $\beta' = \beta$, the adversary wins the CKA security game.

Type-2 Adversary

The CKA security game for $\mathcal{A}_2$ remains the same as the security game for $\mathcal{A}_1$, except the Setup and Phase 1 which are as follows:

Setup: After running the setup algorithm, the Challenger gives Sk_B and PK to $\mathcal{A}_2$.

Phase 1: $\mathcal{A}_2$ acts the same as that of $\mathcal{A}_1$ security game, except it is given an additional access to the following oracle.

- $O_{Search_A}(I, Tk, QI)$: The challenger runs $Search_A$ to obtain the intermediate search result $\bar{R}$ and delivers to $\mathcal{A}_2$.

2) INDISTINGUISHABLITY AGAINST KEYWORD GUESSING ATTACK (IND-KGA)

The notion is to ensure that the adversary cannot infer any additional information from the available challenged trapdoor. In this experiment, a trapdoor is provided instead of the index to the adversary. We formalize this security requirement through the following security game.

Type-1 Adversary

Setup: After running the setup algorithm, the challenger C gives Sk_A and PK to $\mathcal{A}_1$.

Phase 1: $\mathcal{A}_1$ can submit multiple queries to the following oracles.

- $O_{Extract}(S)$: The challenger runs Extract, and delivers Sk along with S to $\mathcal{A}_1$.
- $O_{TokenGen}(S, W)$: The challenger for a given S, generates private key SK by running $TokenGen$, and compute $TK \leftarrow Online.TokenGen(PK,$ $Offline.TokeGen(PK, SK), W)$ and give it to $\mathcal{A}_1$.
- $O_{Search}(I, Tk, QI)$: The challenger runs $search_A$ and $search_B$ to compute the relevance score R and gives it to $\mathcal{A}_1$.

Challenge Phase: $\mathcal{A}_1$ select an attribute set S^* and two keyword set W_0 and W_1 having an identical number of elements and delivers to $\mathcal{C}$. The challenger selects $\beta \xleftarrow{R} \{0, 1\}$, run $Extract(PK, MK, s^*)$ to gets SK for S^*, and executes $TK^* \leftarrow$ Online.TokenGen (PK,Offline.TokeGen(PK,SK),S^*) and delivers TK^* to $\mathcal{A}_1$.

Phase 2: $\mathcal{A}_1$ acts as in phase 1 with the exception that he can not query O_{search} with the input index I with $\mathcal{T}$ and W, if $f(S^*, \mathcal{T}) = 1$, and $W \cap W_0 \neq W \cap W_1$.

Guess: $\mathcal{A}_1$ outputs a bit β', and wins the IND-KGA security game if $\beta' = \beta$.

Type-2 Adversary The IND-KGA security game for $\mathcal{A}_2$ remains the same as the security game for $\mathcal{A}_1$, except the Setup, phase 1 and Phase 2 which are as follows:

Setup: After running the setup algorithm, the challenger gives Sk_B and PK to $\mathcal{A}_2$.

Phase 1: $\mathcal{A}_2$ acts the same as that of $\mathcal{A}_1$ security game except it is given an additional access to the following oracle.

- $O_{Search_A}(I, Tk, QI)$: The challenger runs $Search_A$ to generate the intermediate search result $\bar{R}$ to $\mathcal{A}_2$.

Phase 2: $\mathcal{A}_1$ acts as in phase 1 with the exception that he can not query query O_{search}, O_{search_A} with the input index I with $\mathcal{T}$ and W, if $f(S^*, \mathcal{T}) = 1$, and $W \cap W_0 \neq W \cap W_1$.

V. PROPOSED SCHEME

Setup(1^k): Select $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, where $\mathbb{G}_1$,$\mathbb{G}_2$ and $\mathbb{G}_T$ are cyclic group of order P, which is a $k - bit$ prime.

Let $H : \{0, 1\} \to \mathbb{G}_1$ is a one-way collision-resistant hash function. Select $a, b, c, \{\alpha_i, \beta_i\} \xleftarrow{R} \mathbb{Z}_p$,$g_1 \xleftarrow{R} \mathbb{G}_1$ and $g_2 \xleftarrow{R} \mathbb{G}_2$. Compute $h = g_1^c$,$h_1 = g_1^{\alpha_1+\beta_1}$ and $h_1 = g_1^{b\alpha_1\beta_2}$.

$Sk_A = (\alpha_1, \alpha_2)$, $Sk_B = (\beta_1, \beta_2)$ are the secret key of server A and B respectively. Set $PK = (g_1, g_2, h, \{h_i\})$ and $MK = (g_1, g_1^{ab}, Sk_A, Sk_B)$.

Extract (MK, PK, S): Select $r, \mathcal{T}_j \xleftarrow{R} \mathbb{Z}_p$, where $1 \leq j \leq M$, represents the limit for user interest keyword set in a single query. The algorithm then computes $D_j = g_1^{ab\mathcal{T}_j} h^r$,$\acute{D}_j = g_1^{a\mathcal{T}_j}$, $\acute{D} = g_2^r$ and $D'' = H(j)^r$. Set $SK = \{(D_j, D_j')|j \in [1, M]\}$, $D\prime$, $(\{D''|j \in S\})$.

Offline.Encrypt (PK, T): Select $s, v_i \leftarrow \mathbb{Z}_p$ and compute $c = g_2^s$, $c' = h_2^s$, $I_{i,1} = g_1^{v_i}$ and $I_{i,2} = h_1^{v_i}$. Now compute the secret share of s for each leaf node of access tree T as $\{q_v(0)|y \in lvs(T)\} \leftarrow$ share(T, S). For each $y \in lvs(T)$, compute $c_y = h^{q_y(0)} H(att(y))^{-s}$. Set the intermediate ciphertext $IC = (c, c\prime, \{(c_y)|y \in lvs(T)\}, \{(I_{i,1}, I_{i,2})|i \in [1, n]\})$.

Online.Encrypt (PK, IC, W): Let $W = \{w_1, w_2, \cdots, w_n\}$ be the keyword set for an encrypted file. Compute $I'_{i,2} = I_{i,2}.H(w_i)^s$ and set the ciphertext $CT_w = (IC.C, IC.C', \{(IC.CY)|y \in lvs(T)\}, \{IC.I_{i,1}, I'_{i,2})|i \in [1, n]\})$.

Offline.TokenGen (SK, PK): Select $u, u_1 \xleftarrow{R} \mathbb{Z}_p$ and compute $tk' = D'^v$. For each attribute $atj \in S$, compute $tk_j = D'^{v}$, also compute $T = h_2^{u_1}$,$T_{j,1} = D_j^v$ and $T_{j,2} = D_j'^v$. Set the intermediate token $IT = (tk\prime, T, \{(tk_j)|j \in S\}, \{(T_{j,1}, Tj, 2)|j \in [1, m]\})$.

Online.TokenGen (PK, IT, W'): Let $W' = \{w'_1, w'_2, \cdots, w'_m\}$ be the user interest keyword set in the token. For each interest keyword compute $T'_{j,2} = IT.T_{J,2}.H(w'_j)^{-u_1}$ and set the token as $Tk = (IT.tk', IT.T, \{(IT.tk_j)|j \in S)\}, \{IT.T_{j,1}, T'_{j,2})|j \in [1, m]\})$.

Search$_A(I, Tk, QI, Sk_A)$: Let users query be $QI = (q_1, q_2, \cdots, q_m)$. Server A runs this algorithm in two phases.

Phase 1: The server first checks whether the given attribute set S, as labeled in Tk, satisfies the access structure T, assigned to the CT_W, if it fails, terminate the search operation and return 0; otherwise it computes:

let $t = att(v)$, where v is a leaf node and $v \in S$. Server A computes $\psi_v = e(C_v, tk').e(tk_t, C) = e(h, g_2)^{urq_v(0)}$. If $v \notin S$, $\psi_v = \perp$. If v is a nonleaf node and $v \in T$, ψ_v can be compute by a recursive algorithm $e(h, g)^{urq_{\text{root}}(0)} \leftarrow$ combine$(T, \{\psi_v|v \in S\})$.

Now, it can finally compute $\psi_R = e(h, g_2)^{urs}$.

Phase 2: For each $j \in [1, m]$, it chooses $k \xleftarrow{R} \mathbb{Z}_p$ and computes $\phi_j = e(T_{j,a}, C^k)/\psi_R^k = e(g1, g2)^{absu\mathcal{T}_jk}$, $T'_j = T_{j,2}$. Server A also need to computes $T' = T^{1/k\alpha_2}$, $C' = C'^{k/\alpha_2}$, $\{\bar{I}_{i,1} = I_{i,1}^{k^2}, \bar{I}_{i,2} = \left(I_{i,2}/I_{i,1}^{\alpha_1}\right)|j \in [1, n]\}$. The search result for server B is set to $\bar{R} = (\bar{C}, \bar{T}, \{(\phi_j, \bar{T}_j)j \in [1, m]\}$, $T' = T'/k\alpha_2$, QI, $\{(\bar{I}_{i,1}, \bar{I}_{i,2})|i \in [1, n]\})$.

Search$_B$ $(\bar{R}, Sk_B)$: Server B sets $R = 0$ and computes the score elements as follows:

$A_j = (e(\bar{T}_j, \bar{C}^{1/\beta_2}$
$= \{e(g_1, g_2)^{absu\mathcal{T}_jk} e(H(w_j\prime, g_2)^{-bsu_1k})|j \in [1, m]\}$.

$$B_i = e(\frac{barI_{1,2}}{barI_{i,1}^{\beta_1}}, \bar{T}^{\frac{1}{\beta_2}}) = e(H(w_i, g_2)^{ksu_1b})|i \in [1, n]\}.$$

For $i \in [1, n]$, server B multiply A_j and B_i and store the j value in table Γ, if it equals to ϕ_j. Finally, the rank score of user interest keyword set is computed as $R = \sum_{j \in \gamma} q_j$.

A. SECURITY OF ABKRS

Theorem 1 [11]: Adversary $\mathcal{A}_1$ can obtain group elements by quering the oracles for $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ and e or by the direct interaction with the challenger during security game. Let q represents the maximum number of received group elements, then the advantage of $\mathcal{A}_1$ in the CKA security game is $O(q^2/p)$.

Proof [11]: The challenge index in CKA simulation game consists of randomly either $(\{I_{i,1}, I_{i,2}\}|w_i \in W_0)$ or $(\{I_{i,1}, I_{i,2}\}|w_i \in W_1)$. We construct a modified game, assume $W = W_0 \cup W_1 = \{w_1, w_2, \cdots, w_n\}$, $I_{\xi,2}$ is either $h_1^{u\xi} H(w_\xi)^s$ or g_1^θ, where $\xi \in [1, n]$, $\theta \xleftarrow{R} \mathbb{F}_p$. Providing the challenge index $I^* = (C, C^*, \{C_y|y \in Y\}, \{I_{i,1}, I_{i,2}\}|i \in [1, n]\})$, $\mathcal{A}_1$ must predict whether $I_{\xi,2}$ is $h_1^{u\xi} H(w_\xi)^s$ or g_1^θ. It is evident that $\mathcal{A}_1$ advantage is $\varepsilon/2$ instead of ε in the modified security game.

Setup. The simulator $\mathcal{B}$ select $g_1 \xleftarrow{R} \mathbb{G}_1$, $g_2 \xleftarrow{R} \mathbb{G}_2$,$a, b, c$, $(\{\alpha_i, \beta_i\}|i \in [1, 2]) \xleftarrow{R} \mathbb{F}_p$, and compute $h = g_1^c$, $h_1 = g_1^{\alpha_1+\beta_1}$ and $h_2 = g_1^{b\alpha_1\beta_2}$. Set and give the public key to $PK = (g_1, g_2, h, \{h|i \in [1, 2]\})$ and $Sk_A = (\alpha_i|i \in [1, 2])$ to $\mathcal{A}_1$.

Hash Queries. If $\mathcal{A}_1$, for any attribute j, calls a hash query, B returns $g_1^{\theta_j}$ and stores (j, θ_j) into the list H; if $\mathcal{A}_1$, for any

TABLE 1. Possible terms for querying group oracle.

	$\mathbb{G}_1$			$\mathbb{G}_2$	
PK & Hash	c	$\alpha_1+\beta_1$	$\eta_i \mid \varphi_j$	$b\alpha_2\beta_2$	
Index	ν_i	$c\lambda_j - s\varphi_i$	$\nu_i(\alpha_1+\beta_1)+\eta_i s$	s	$b\alpha_2\beta_2 s$
SK	$\varphi_j r^{(t)}$	$a\mathcal{T}_j^{(t)}$	$ab\mathcal{T}_j^{(t)} + cr^{(t)}$	$r^{(t)}$	
TK	$u^{(t)}\varphi_j r^{(t)}$	$u^{(t)}(ab\mathcal{T}_j^{(t)} + cr^{(t)})$	$au^{(t)}\mathcal{T}_j^{(t)} - u_1^{(t)}\eta_j$	$u^{(t)}r^{(t)}$	$b\alpha_2\beta_2 u_1^{(t)}$

keyword W_i calls a hash query, the simulator return $g_1^{\eta_i}$ and stores (w_i, η_i) into the list H where $\theta_j, \eta_i \xleftarrow{R} \mathbb{F}_p$.

Phase 1. $\mathcal{A}_1$ submits query to the following oracles. $\mathcal{B}$ keeps an empty set D, keyword list L_{kw}, table E and set $j = 0$. we consider its t^{th} query.

- $O_{Extract}(S)$: $\mathcal{B}$ choose $r^{(t)}, \mathcal{T}_j^{(t)} \xleftarrow{R} \mathbb{F}_p$, where $j \in [1, m]$ and computes $SK = (K_j = g_1^{ab\mathcal{T}^{(t)}} h^{r^{(t)}}, K' = g_2^{r^{(t)}}, \forall j \in S$: $K_j'' = g_1^{\varphi_j r(t)}$, $K_j' = g_1^{a\mathcal{T}a_j}, j \in [1, m])$. It computes $D := D \cup S$ and submits S along with SK to $\mathcal{A}_1$.
- $O_{TokenGen}(S, W')$: Let the keyword list is $W' = \{w_1', \cdots, w_m'\}$. To get SK, $\mathcal{B}$ queries the $O_{Extract}(S)$ with $j := j + 1$, and selects $u^{(t)}, u_1^{(t)} \xleftarrow{R} \mathbb{F}_q$. It stores the tuple (W', S, J) in table E and return $TK = (tk_j = K_j''^{u^{(t)}} | j \in S)$, $tk' = K'^{u^{(t)}}$, $T = h_2^{u_1^{(t)}}$, $(T_{j,1} = K_j'^{u^{(t)}}$, $T_{j,2} = K_j'^{u^{(t)}} g_1^{-u_1^{(t)}\eta_j} | j \in [1, m])$ to $\mathcal{A}_1$.
- $O_{Search}(I, TK, QI)$: After getting all the secret keys, $\mathcal{B}$ executes $Search_A$ and $Search_B$ to compute the relevance score R to $\mathcal{A}_1$.

Challenge. $\mathcal{A}_1$ submits an access structure $\mathcal{T}^*$, two keywords sets W_0, W_1 and a keyword w_ξ to $\mathcal{B}$, such that

- $w_\xi \in W = W_0 \cup W_1 = \{w_1, \cdots, w_n\}$ and having the same number of elements in both sets.
- $f(S, \mathcal{T}^*) \notin 1$ for all $S \in D$.
- if $f(S, \mathcal{T}^*) = 1$, for each entry in table E, adds W' to L_{kw}; then $\forall W' \in L_{kw}, w_\xi \notin W'$.

$\mathcal{B}$ select θ, s, $v_i \xleftarrow{R} \mathbb{F}_p$, $i \in [1, n]$ and executes linear secret sharing technique related with $\mathcal{T}^*$ to gets shares λ_j for s of each associated attribute j. It also sets $C = g_2^s$, $C' = g_2^s$, $C_j = h^{\lambda_j} g_1^{-\varphi_j s}$, $(\{I_{i,1} = g_1^{v_i}, I_{i,2} = h_1^{v_i} g_1^{\eta_i s} | i \in [1, n]\})$. Finally $\mathcal{B}$ returns index I^* to $\mathcal{A}_1$.

Phase 2. $\mathcal{A}_1$ acts as in *Phase*1 with the exception that he cannot query $O_{Extract}$ if $f(\mathcal{T}^*, S) = 1$ and (S, w_ξ) be the input to O_{Tarp}. In Table 1, we showed all the terms that can be submitted to group oracle $\mathbb{G}_T$. With the help of the following two lemmas, we prove that $\mathcal{A}_1$ can differentiate between g_1^θ and $h_1^{u\xi} H(w_\xi)^s$ with an advantage of negligible probability.

Lemma 1 Provided W and S, $\mathcal{A}_1$ can construct a TK such that $w_\xi \in W$ and $f(S, \mathcal{T}^*) = 1$ with a negligible probability.

The proof of this lemma is provided in two parts.

1) Provided TK whose W and S satisfy $f(S, \mathcal{T}^*) = 1$, where $w_\xi \notin W$. Given it is formed in t^{th} oracle, we can easily know from Table 1 that $\mathcal{A}_1$ cannot generate the term $au^{(t)}\mathcal{T}_\xi^{(t)} - u_1^{(t)}\eta_\xi$ in $\mathbb{G}_1$.
2) We now prove that with a given S, $\mathcal{A}_1$ cannot form the SK such that $f(S, \mathcal{T}^*) = 1$, which is equivalent to the statement that $\mathcal{A}_1$ cannot generate $abs\mathcal{T}_j^{(t)}, j \in [1, M]$.

From Table 1 we can easily find that there appears only one term that output $abs\mathcal{T}_j^{(t)}$ in $\mathbb{G}_T$. It is $(ab\mathcal{T}_j^{(t)} + cr^{(t)}).s$, for which $\mathcal{A}_1$ needs to cancel the term $cr^{(t)}s$. We can easily find from Table 1 that, $\mathcal{A}_1$ could get the term $csr^{(t)}$ only by the given formula (1) for some T and constant $\gamma_j \neq 0$.

$$\Sigma_{j\in T}\gamma_j((c\lambda_j - \varphi_j s) \cdot r^{(t)} \cdot s) = csr^{(t)}$$

Hence, it is impossible to reconstruct the secret since it requires in the security game for all $S \in D$.

Lemma 2 Given some g_2^δ, $\mathcal{A}_1$ can construct $e(h_1^{v_x i} H(w_\xi)^s, g_2)^\delta$ from the oracles terms with a negligible probability.

Let us considered how to construct the term $\Delta = \delta((\alpha_1 + \beta_1)v_\xi + \eta_\xi s)$ in $\mathbb{G}_T$ for some δ. $\mathcal{A}_1$ has the term α_1, and needs to construct $^\Delta\prime = \delta(\beta_1 v_\xi + \eta_\xi s)$ in $\mathbb{G}_T$. We consider the term $\delta\beta_1 v_\xi$ in $\mathbb{G}_T$ and the terms that contained β_1 appear in:

1) β_1 can be obtain from term $\alpha_1 + \beta_1$ by canceling out α_1. However, both v_ξ and β_1 are in same group $\mathbb{G}_1$, hence cannot be paired to obtain the combine term $v_\xi\beta_1$.
2) Similarly to the above case, the term $v_\xi\beta_1$ cannot be composed from the output $a_1(\alpha_1 + \beta_1)v_i$.

Hence, the $\mathcal{A}_1$ cannot construct the term Δ in $\mathbb{G}_T$.

Therefore, we conclude that $\mathcal{A}_1$ gains a negligible advantage in differentiating between the terms $h_1^{v_x i} H(w_\xi)^s$ and g_1^θ, which means that the advantage of $\mathcal{A}_1$ in the modified security game is $O(q^2/P)$. □

Theorem 2: Let $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ and q be as defined in Theorem 1, then the advantage in the IND-KGA security game for $\mathcal{A}_1$ is $O(q^2/p)$.

Proof [11]: The challenge trapdoor in IND-KGA simulation game, consist of randomly either $(\{T_{i,1}, I_{T,2}\} | w_i \in W_0)$ or $(\{T_{j,1}, T_{j,2}\} | w_j \in W_1)$. We construct a modified game, assume $W^* = W_0 \cup W_1 = \{w_1, w_2, \cdots, w_m\}$, $T_{\xi,2}$ is either $g_1^{au\xi} H(w_\xi)^{-u}$ or g_1^θ, where $\theta \xleftarrow{R} \mathbb{F}_p$, $\xi \in [1, m]$. With challenge trapdoor $TK^* = (tk', \{tk_j | j \in S^*\}, T, \{T_{j,1}, T_{j,2}\} | j \in [1, m])$, $\mathcal{A}_1$ must predict whether $T_{\xi,2}$ is $g_1^{au\xi} H(w_\xi)^{-u}$ or g_1^θ. It is evident that $\mathcal{A}_1$ initial advantage ε in the IND-KGA security game can be mapped into $\mathcal{A}_1$ that has now at least $\varepsilon/2$ advantage in the modified security game.

All the phases except the *Challenge* and *Phase 2* are similar to those in the proof of *Theorem 1*.

Challenge. $\mathcal{A}_1$ submits an access structure $\mathcal{T}^*$, two keywords sets W_0, W_1 and keyword w_ξ to $\mathcal{B}$, such that, $w_\xi \in W^* = W_0 \cup W_1 = \{w_1, \cdots, w_n\}$ and having the same numbers of elements in both sets. $\mathcal{B}$ select $\theta, u, r, \mathcal{T}_j, u_1 \xleftarrow{R} \mathbb{F}'_p, j \in [1, m]$ and computes $th' = g_2^{ru}$, $(\{tk_j = g_1^{ru\varphi_j}\}|j \in S^*)$, $(\{T_{j,1} = g_1^{ab\mathcal{T}_j u} h^{ru}, T_{j,2} = g_1^{au\mathcal{T}_j - u_1\eta_j}\}|j \in [1, m])$. $\mathcal{B}$ submits TK^* to $\mathcal{A}_1$.

Phase 2: $\mathcal{A}_1$ acts as in *Phase*1 except that he cannot query O_{Search} if $f(S^*, \mathcal{T}) = 1$ and (w_ξ), with the input index I with $\mathcal{T}$ and W.

Due to separate selection and combine representation of the generic bilinear group model elements, the only possible way for adv_1 to differentiate among g_1^θ and $g_1^{au\xi} H(w_\xi)^{-u}$ is to composed $e(g_1^{au\xi} H(w_\xi)^{-u}, g_1^\theta)^\delta$ for any g_2^δ which can be constructed from the oracle output terms. let us considered how $\mathcal{A}_1$ cannot construct the term $\Delta = \delta(au\mathcal{T}_\xi - u_1\eta_\xi$ in $\mathbb{G}_2$ for some δ.

Now first considered the required term $\delta u_1 \eta_\xi$, for which we need the term u_1 that appear in:

1) The term $b\alpha_2\beta_2 u_1$ in Table 1 is possible to reduce to $b\beta_2 u_1$ in $\mathbb{G}_2$. Also, the term η_ξ can be paired with $b\beta_2 u_1$ to construct $b\beta_2 u_1 \eta_\xi$. However, we can analyze from Table 1 that it is impossible to compose the term $\delta u_1 \eta_\xi$=$abu\beta_2\mathcal{T}_\xi$ by the $\mathcal{A}_1$ in $\mathbb{G}_T$.
2) The term $au\mathcal{T}_j$-$u_1\eta_j$ such that $j \notin \xi$, from Table 1 we can find that $\mathcal{A}_1$ cannot construct the term $\delta au_1\eta_\xi$ by pairing the terms $au\mathcal{T}_j - u_1\eta_\xi$ and η_ξ which are both in $\mathbb{G}_1$.

□

TABLE 2. Complexity analysis notations.

Notations	Description
$E_{\mathbb{G}_1}$	Exponentiation in $\mathbb{G}_1$
$E_{\mathbb{G}_2}$	Exponentiation in $\mathbb{G}_2$
$M_{\mathbb{G}1}$	Multiplication in $\mathbb{G}_1$
$M_{\mathbb{G}_2}$	Multiplication in $\mathbb{G}_2$
P	Bilinear pairing computation
$M_{\mathbb{G}_q^*}$	Multiplication in $\mathbb{Z}_q^*$
$\parallel V \parallel$	Length of element V in bits
$\mid R \mid$	Elements in set R
$V_{\mathbb{Z}_q^*}$	Inverse element in $\mathbb{Z}_q^*$
$\mathbb{Z}_q^*$	Element in $\mathbb{Z}_q^*$
H	Compute hash $H : \{0,1\}^* \longrightarrow \mathbb{Z}_q^*$
H^*	Compute hash $H : \{0,1\}^* \longrightarrow \mathbb{G}_1$
$\mathcal{I}_{max}$	Maximum keywords for encryption
$\mathcal{T}_{max}$	Maximum query keywords
X	Attributes attached to the access policy
S	Attribute set of a DU.
N	Minimum attribute set for access structure.

TABLE 3. Comparison for computations phase.

Computation phase	[34]	[35]	[36]	[37]	Ours
Offline.Encrypt	✓	✓	✓	✓	✓
Online.Encrypt	✓	✓	✓	✓	✓
Offline.TokGen	×	×	✓	✓	✓
Online.TokGen	×	×	✓	✓	✓

B. SECURITY OF OO-AMBS

The security proof of the online/offline ABMS scheme is based on the security proof of the ABKRS scheme, the proof of which is given in the following two theorems as

Theorem 3: Assuming that the basic ABKRS scheme is adaptively secure against CKA, then the OO-ABMS scheme is also adaptively secure against CKA in the random oracle model.

Proof: Assume an adversary $\mathcal{B}$ wins the *CKA* security game of the OO-ABMS scheme with non-negligible probability. We can easily construct an adversary $\mathcal{A}$ to succeeds in compromising the ABKRS scheme. We let $\mathcal{A}$ impersonate the OO-ABMS challenger for adversary $\mathcal{B}$ by querying the ABKRS challenger and utilizing B' capability in attacking the OO-ABMS scheme to attack the ABKRS scheme.

- During the setup phase, adversary $\mathcal{B}$ selects T^* and submits it to adversary $\mathcal{A}$, who then submits it to the ABKRS challenger. The challenger executes the setup to get the master key MK and public key PK. Adversary $\mathcal{A}$ submits the PK to $\mathcal{B}$.
- In Phase 1, $\mathcal{A}$ forwards $\mathcal{B}$ queries to the challenger in the algorithm of the ABKRS scheme, which generates the keys and indexes. It is trivial to verify that the keys and Index generated in the ABKRS scheme are equivalent to those in the OO-ABMS scheme. Hence, after receiving the query results from $\mathcal{A}$, $\mathcal{B}$ will consider them as constructed by the OO-ABMS scheme challenger.
- During the Challenge phase, $\mathcal{B}$ chooses two keyword sets W_0 and W_1 and submits them to $\mathcal{A}$. These keyword sets are further sent to the challenger, who one of them w_λ ($\lambda = 1$ *or*0), and submits back the ciphertext $CT_{w_\lambda} \leftarrow (Pk, \mathcal{T}^*, w^\lambda)$ to $\mathcal{A}$.
- $\mathcal{B}$ receives the challenge ciphertext from $\mathcal{A}$ and proceeds to submit queries to $\mathcal{A}$. With more constraints, as mentioned in Phase 2, *adv* replies to $\mathcal{B}'s$ queries in a similar way as stated in Phase 1. When $\mathcal{B}$ stops querying, it outputs a bit λ^* as its guess for λ. The adversary $\mathcal{A}$ outputs the same bit as its guess for λ.

□

The security proof of the online/offline ABMS scheme is based on the security proof of the ABKRS scheme, the proof of which is given in the following two theorems as.

Theorem 4: Assuming that the ABKRS scheme is adaptively secure against KGA, then the OO-ABMS scheme is also adaptively secure against KGA in the random oracle model.

Proof: Assume an adversary $\mathcal{B}$ wins the *KGA* security game of the OO-ABMS scheme with non-negligible probability. We can easily construct an adversary $\mathcal{A}$ to

TABLE 4. Features comparison.

Scheme	Expressiveness of access structure	Retrieval mode	Resist KGA	Cloud system	Ranked result
[34]	Monotone span program (MSP)	Multi-keyword	×	✓	×
[35]	LSSS	Multi-keyword	×	✓	×
[36]	LSSS	Multi-keyword	×	✓	×
[37]	LSSS	Multi-keyword	✓	✓	×
Ours	Access tree	Multi-keyword	✓	✓	✓

succeed in compromising the ABKRS scheme. We let $\mathcal{A}$ impersonate the OO-ABMS challenger for adversary $\mathcal{B}$ by querying the ABKRS challenger and utilizing $B\prime$ capability in attacking the OO-ABMS scheme to attack the ABKRS scheme.

- During the setup phase, adversary $\mathcal{B}$ selects S^* and submits it to adversary $\mathcal{A}$, who then submits it to the ABKRS challenger. The challenger executes the setup to get the master key MK and public key PK. Adversary $\mathcal{A}$ submits the PK to $\mathcal{B}$.
- In Phase 1, $\mathcal{A}$ forwards $\mathcal{B}$ queries to the challenger in the algorithm of the ABKRS scheme, which generates the keys and trapdoors. It is trivial to verify that the keys and trapdoor generated in the ABKRS scheme are equivalent to those in the OO-ABMS scheme. Hence, after receiving the query results from $\mathcal{A}$, $\mathcal{B}$ will consider them as constructed by the OO-ABMS scheme challenger.
- During the Challenge phase, $\mathcal{B}$ chooses two keyword sets W_0 and W_1 and submits them to $\mathcal{A}$. These keyword sets are further sent to the challenger, who encrypted one of them w_λ ($\lambda = 1\ or 0$), and submits back the ciphertext $CT_{w^\lambda} \leftarrow (Pk, S^*, w^\lambda)$ to $\mathcal{A}$.
- $\mathcal{B}$ receives the challenge ciphertext from $\mathcal{A}$ and proceeds to submit queries to $\mathcal{A}$. With more constraints, as mentioned in Phase 2, *adv* replies to $\mathcal{B}'s$ queries in a similar way as stated in Phase 1. When $\mathcal{B}$ stops querying, it outputs a bit λ^* as its guess for λ. The adversary $\mathcal{A}$ outputs the same bit as its guess for λ.

Now, if $\mathcal{B}$ wins the CKA security game defined in the OO-ABMS scheme with a non-negligible advantage, the same will $\mathcal{A}$ in the ABKRS scheme, hence, clearly contradicts Theorem 2. Therefore, Theorem 2 holds, and the proposed OO-ABMS scheme is adaptively secure against KGA. □

VI. PERFORMANCE ANALYSIS

This section compares the efficiency enhancement through theoretical and experimental analysis of our online/offline ABMS scheme with that of [34]–[37]. Table 2 presents the notations used for this comparison.

A. THEORETICAL ANALYSIS

Table 3 presents the comparison for computation phases for keyword encryption and token generation algorithms. From this table, we can see that OO-ABMS scheme like [36], [37] not only implements the online/offline computation stage but also implements the online/offline computation stage for token generation. By splitting the token generation into two stages, we shift most computation work to the offline phase in our scheme. While during the online phase, the rest of the work is done through few operations. This can boost the overall system performance, especially for those data users who usually perform the searching operation through resource-scarce devices such as mobile phones.

Table 4 presents a comparison of our scheme with the other related schemes for five aspects, access control structure expressibility, retrieval mode, search result, and security. Many access control structures are available for attribute-based encryption, but the access tree provides better scalability and flexibility in terms of its new user's registration in the system and expressiveness of access policy. Moreover, the search result of all these schemes except ours is a ranked result. The problem with the conjunctive keyword search result is that it operates on the premise of ''all-or-nothing''. More technically, the data user query keywords need to be the same as the index keyword set of the stored file for retrieval of the file. Secondly, the retrieved file is not ranked. Hence the data user has no option to select the most suitable file for its query keyword. This wrongly positioning of user query incur high computation and communication overhead at the resource-constrained data user side. Besides, our scheme, except for [37] can resist both inside and outside keyword guessing attacks (KGA). Last but not least, the servers in our system model can get query keywords from any user. Hence a broader range of applications is possible for our scheme.

For clear understanding, the complexity analysis of aforementioned schemes for different algorithms, *Offline.Encrypt*, *Online.Encrypt*, *Offline.TokenGen*, *Online.TokenGen* and *Search* are tabulated in Tables 5-9, respectively. Also note that we ignore the communication overhead for a valid search query, hence the output size is set to zero for search algorithm in each scheme. We can see for *Offline.Encrypt* algorithm from their respective tables, the computation overhead of our scheme ($| X | + | \mathcal{I}_{max} | +2)E_{\mathbb{G}_1} + | S | H$ is less than that of all other schemes. In [34]–[37], the cost of exponentiation in $\mathbb{G}_1$ is two times of $|X|$, while in [35] it is four times and thus incurs high computation overhead. Similarly, our scheme exhibit low storage cost ($| \mathcal{I}_{max} | + | X | +2)\|\mathbb{G}_1\|$ than others. As depicted in the relevant sections of Table 5-9, the *Online.Encrypt* algorithm of our

TABLE 5. Computation overhead and output size of [34].

Algorithm	Computation overhead	Output size
Setup	$P \mid S \mid + \mid S \mid E_{\mathbb{G}_1} + \mid S \mid M_{\mathbb{Z}_q^*}$	$\mid S \mid \|\mathbb{G}_2\| + \mid S \mid \mid\mid \mathbb{Z}_q^* \mid$
KeyGen	$H \mid S \mid +2 \mid S \mid E_{\mathbb{G}_1}$	$(2 \mid S \mid +1)\|\mathbb{G}_1\|$
Offline.Encrypt	$2 \mid X \mid M_{\mathbb{G}_1} + \mid X \mid (5M_{\mathbb{Z}_q^*} + A_{\mathbb{Z}_q^*})$	$(3 \mid X \mid +2)\|\mathbb{G}_1\| + (\mid X \mid +2)\|\mathbb{G}_2\|$
Online.Encrypt	$E_{\mathbb{G}_2} + M_{\mathbb{G}_2} + 2 \mid X \mid (M_{\mathbb{Z}_q^*} + A_{\mathbb{Z}_q^*})$	$(4+ \mid X \mid)\|\mathbb{G}_1\|$
Offline.TokGen	-	-
Online.TokGen	$(\|\mathcal{T}_{max}\| + 2) \mid S \mid +2E_{\mathbb{G}_1} + \mid S \mid H$	$(4\|\mathcal{T}_{max}\| + \mid S \mid)\|\mathbb{G}_1\| \ + \|\mathbb{G}_2\|$
Search	$(\|\mathcal{I}_{max}\| + \mid S \mid +2)P + (1+ \mid S \mid)E_{\mathbb{G}_2} + (2\|\mathcal{T}_{max}\| + \mid S \mid)M_{\mathbb{G}_2}$	0

TABLE 6. Computation overhead and output size of [35].

Algorithm	Computation overhead	Output size
Setup	$(\mid S \mid +1)E_{\mathbb{G}_1} + \mid S \mid A_{\mathbb{Z}_q^*} + E_{\mathbb{G}_2}$	$(2 \mid S \mid)\|\mathbb{G}_1\| + \|\mathbb{G}_2\|$
KeyGen	$(\mid S \mid +7)E_{\mathbb{G}_1} + \mid S \mid H$	$(\mid S \mid +5)\|\mathbb{G}_1\| + \mid \mathbb{Z}_q^* \mid$
Offline.Encrypt	$(4 \mid X \mid +4)E_{\mathbb{G}_1} + 2E_{\mathbb{G}_2} + (2 \mid X \mid)M_{\mathbb{G}_1} + H$	$(4 \mid X \mid +1)\|\mathbb{G}_1\| + \|\mathbb{G}_2\| + (\|\mathcal{I}_{max}\| + 3)\|\mathbb{G}_1\|$
Online.Encrypt	$(3+ \mid X \mid +1)E_{\mathbb{G}_1} + \mid X \mid (M_q^* + A_{\mathbb{Z}_q^*}) + E_{\mathbb{G}_2}$	$(3 \mid X \mid +1)\|\mathbb{G}_1\| + \|\mathcal{I}_{max}\|\|\mathbb{G}_1\| + \|\mathbb{G}_2\|$
Offline.TokGen	-	-
Online.TokGen	$(2 \mid \mathcal{T}_{max} \mid)E_{\mathbb{G}_1} + \mid \mathcal{T}_{max} \mid M_{\mathbb{G}_1} + H$	$(2 \mid \mathcal{T}_{max} \mid)\|\mathbb{G}_1\|$
Search	$(2 \mid X \mid +4)P + (\mid X \mid +1)E_{\mathbb{G}_1} + E_{\mathbb{G}_2}$	0

TABLE 7. Computation overhead and output size of [36].

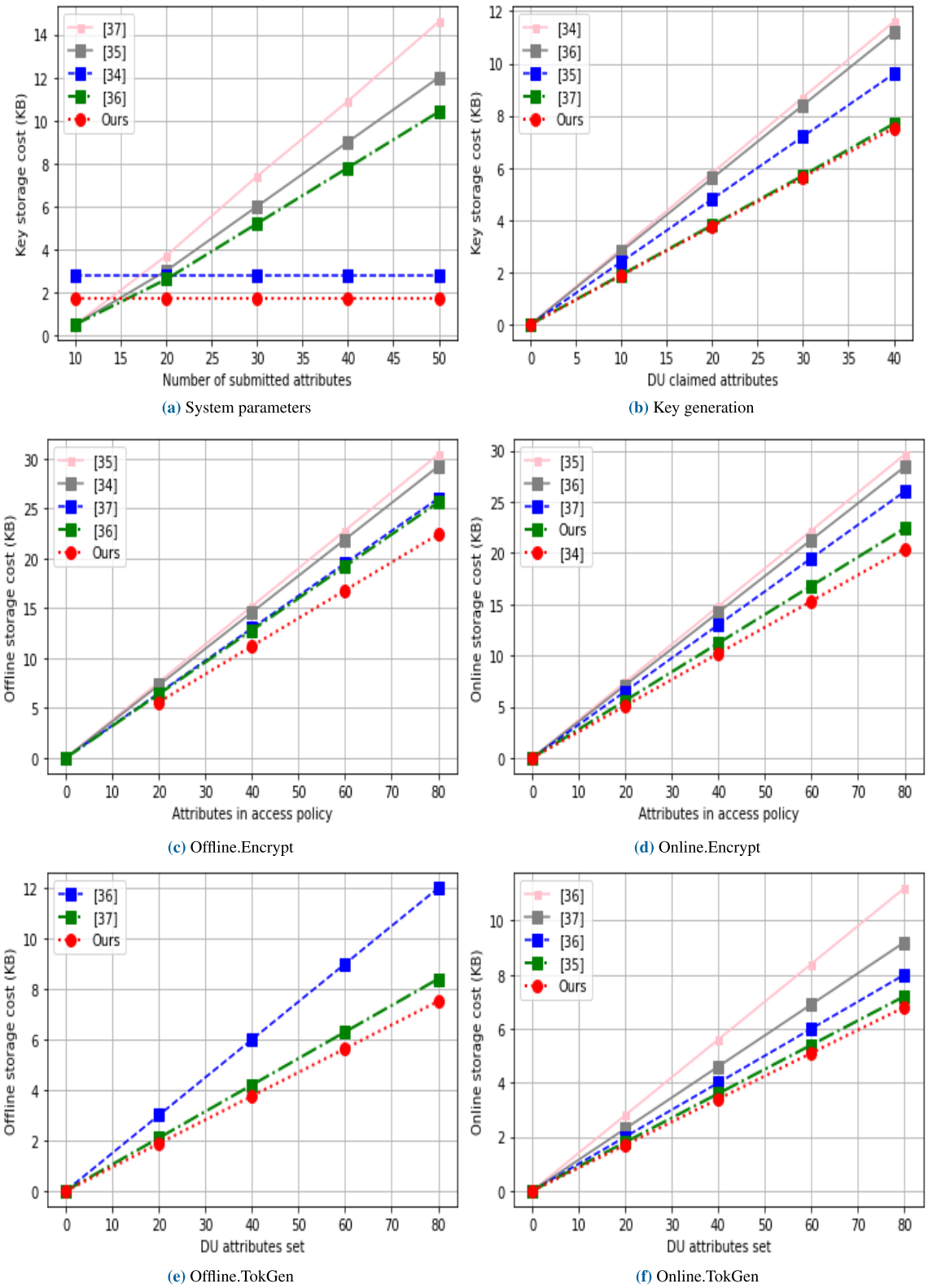
Algorithm	Computation overhead	Output size
Setup	$E_{\mathbb{G}_1} + P$	$2\|\mathbb{G}_1\| + \|\mathbb{G}_2\|$
KeyGen	$(\mid S \mid +3)E_{\mathbb{G}_1} + \mid S \mid H + M_{\mathbb{G}_1}$	$(\mid S \mid +2)\|\mathbb{G}_1 + \mid S \mid \|\mathbb{G}_1\| + \|\mathbb{G}_2\|$
Offline.Encrypt	$(2 \mid X \mid +3)E\mathbb{G}_1 + E_{\mathbb{G}_2} + \mid X \mid M_{\mathbb{G}_1}$	$(2 \mid X \mid +1)\|\mathbb{G}_1\| + \|\mathbb{G}_2\|$
Online.Encrypt	$\mid \mathcal{I}_{max} \mid P+ \mid \mathcal{I}_{max} \mid H+ \mid \mathcal{I}_{max} \mid M_{\mathbb{G}_2} + M_{\mathbb{G}_1}$	$(2 \mid X \mid)\|\mathbb{G}_1\| + \mid \mathcal{I}_{max} \mid \|\mathbb{G}_1\| + \|\mathbb{G}_2\|$
Offline.TokGen	$(2 \mid S \mid + \mid \mathcal{T}_{max} + 1 \mid)E_{\mathbb{G}_1}$	$\mid \mathcal{T}_{max} \mid (2 \mid S \mid +1)\|\mathbb{G}_1\|$
Online.TokGen	$(\mid S \mid +1)E_{\mathbb{G}_1} + \mid \mathcal{T}_{max} \mid H+ \mid \mathcal{T}_{max} \mid M_{\mathbb{G}_1}$	$2 \mid \mathcal{T}_{max} \mid \|\mathbb{G}_1\| + \|\mathbb{G}_2\|$
Search	$2 \mid S \mid P+ \mid \mathcal{T}_{max} \mid P+ \mid X \mid M_{\mathbb{G}_2}$	0

TABLE 8. Computation overhead and output size of [37].

Algorithm	Computation overhead	Output size
Setup	$(2 \mid U \mid +2)E_{\mathbb{G}_1} + P$	$(2 \mid U \mid +5)\|\mathbb{G}_1\| + \|\mathbb{G}_2\|$
KeyGen	$(\mid S \mid +4)E_{\mathbb{G}_1} + M_{\mathbb{G}_1}$	$(\mid S \mid +3)\|\mathbb{G}_1\|$
Offline.Encrypt	$(2 \mid X \mid +2)E_{\mathbb{G}_1} + E_{\mathbb{G}_2}$	$(2 \mid X \mid +3)\|\mathbb{G}_1\|$
Online.Encrypt	$(2 \mid X \mid)E_{\mathbb{G}_1} + \mid \mathcal{I}_{max} \mid H + 2 \mid X \mid M_{\mathbb{G}_1}$	$(2 \mid X \mid) + 3)\|\mathbb{G}_1\|$
Offline.TokGen	$(2 \mid S \mid +2)E_{\mathbb{G}_1}$	$(\mid S \mid +2)\|\mathbb{G}_1\|$
Online.TokGen	$2 \mid \mathcal{T}_{max} \mid E_{\mathbb{G}_1} + \mid \mathcal{T}_{max} \mid H$	$(2 \mid \mathcal{T}_{max} \mid + \mid S \mid)\|\mathbb{G}_1\|$
Search	$(3+ \mid S \mid +2)P + \mid S \mid E_{\mathbb{G}_2} + 2M_{\mathbb{G}_2}$	0

TABLE 9. Computation overhead and output size of OO-ABMS.

Algorithm	Computation overhead	Output size
Setup	$3E_{\mathbb{G}_1} + A_{\mathbb{Z}_q^*}$	$3\|\mathbb{G}_1\|$
KeyGen	$(4+ \mid S \mid)E_{\mathbb{G}_1} + \mid S \mid H$	$(2+ \mid S \mid)\|\mathbb{G}_1\|$
Offline.Encrypt	$(\mid X \mid + \mid \mathcal{I}_{max} \mid +2)E_{\mathbb{G}_1} + \mid S \mid H$	$(\mid \mathcal{I}_{max} \mid + \mid X \mid +2)\|\mathbb{G}_1\|$
Online.Encrypt	$(\mid X \mid + \mid \mathcal{I}_{max} \mid +3)M_{\mathbb{G}_1}$	$(\mid X \mid) + \mid \mathcal{I}_{max} \mid +2)\|\mathbb{G}_1\|$
Offline.TokGen	$(\mid S \mid + \mid \mathcal{T}_{max} \mid +2)E_{\mathbb{G}_1}$	$(\mid S \mid + \mid \mathcal{T}_{max} \mid +2)\|\mathbb{G}_1\|$
Online.TokGen	$(\mid S \mid + \mid \mathcal{T}_{max} \mid)E_{\mathbb{G}_1}$	$(\mid S \mid + \mid \mathcal{T}_{max} \mid)\|\mathbb{G}_1\|$
Search	$(2 \mid S \mid +2 \mid \mathcal{I}_{max} \mid + \mid \mathcal{T}_{max} \mid)P + 3 \mid \mathcal{I}_{max} \mid E_{\mathbb{G}_1} + (\mid \mathcal{T} \mid +1)E_{\mathbb{G}_2}$	0



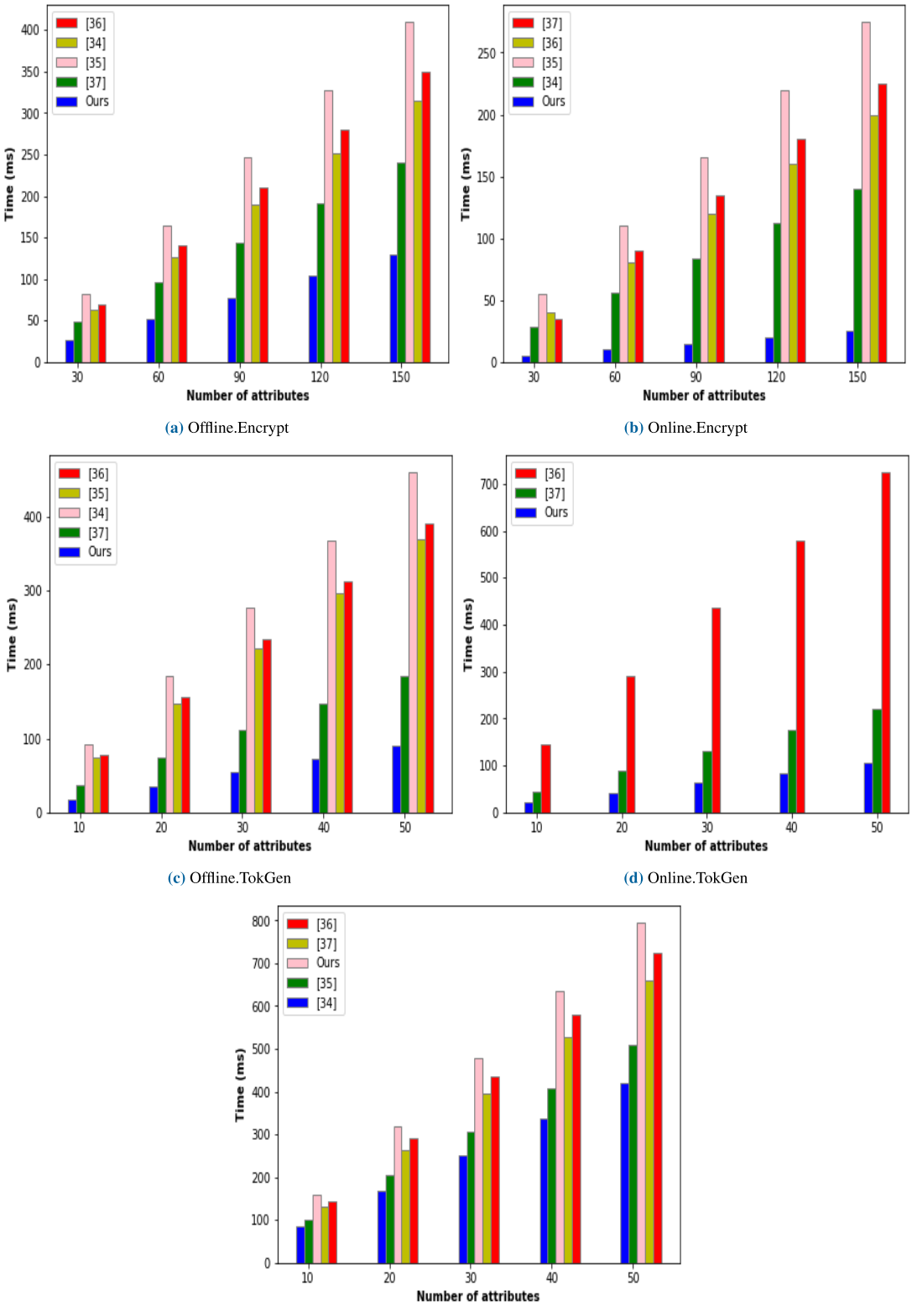
(a) System parameters (b) Key generation

(c) Offline.Encrypt (d) Online.Encrypt

(e) Offline.TokGen (f) Online.TokGen

FIGURE 2. Practical storage analysis in various algorithms.

scheme incur the cost of $(\mid X \mid + \mid \mathcal{I}_{max} \mid +3)M_{\mathbb{G}_1}$ on the user side, while the rest of the schemes suffer from linearity problem, i.e. $|X|$ times exponentiation in group $\mathbb{G}_1$. Moreover, the storage requirements of this algorithm in our scheme $(\mid X \mid)+ \mid \mathcal{I}_{max} \mid +2)\|\mathbb{G}_1\|$ is less than others, except for that in [34]. In comparison for *Offline.TokenGen*



(a) Offline.Encrypt

(b) Online.Encrypt

(c) Offline.TokGen

(d) Online.TokGen

(e) Search

FIGURE 3. **Practical computation analysis in various algorithms.**

algorithm, OO-ABMS computation cost ($|S| + |\mathcal{T}_{max}| + 2)E_{\mathbb{G}_1}$ is less than schemes [36], [37], while schemes [34], [35] does not support offline token generation at the data user end. For storage cost, only scheme [37] has less requirement than ours, however the reason for which is the ranked search result for our scheme and thus brings in an

additional $|\mathcal{T}_{max}|$ to its cost, which is not considered in [37]. Furthermore, as illustrated for *Online.tokenGen* algorithm comparison, our scheme computation cost is significantly better than that of all the other preliminary schemes. It is illustrated for its storage cost, that our scheme incurs $(|\ S\ |\ +\ |\ \mathcal{T}_{max}\ |)\|\mathbb{G}_1\|$, which is less than the cost of others schemes. For searching algorithm, the concerned table's sections demonstrate relatively less efficiency than the works in [34]–[37] of our scheme. The reason is that the OO-ABMS scheme uses a robust security model to resist KGA and an efficient multi-keyword ranked search simultaneously. Moreover, this comparison analysis does not consider the extra computation overhead caused by wrongly putting data users' interests in their retrieval model, which ultimately suffers end-users experience and network bandwidth. Besides, the search algorithm doesn't affect the data users' computation overhead since this is the one-time process performed by a resource-rich cloud server. However, we can increase the search efficiency if we adopt the weaker security assumptions of schemes in [33] and [38].

B. EXPERIMENTAL ANALYSIS

Fig. 2 and 3 illustrate the experimental results for comparative evaluation of the algorithms. To evaluate the proposed scheme's performance, all the algorithms are implemented in Python 2.7, and the bilinear group operation is performed with the Pairing-Based Cryptography (PCB) library [8] framework. The specification of the computer used for experiments is Intel (R) i5-2430M CPU at 3.3 GHz and 6 GB of RAM. Each experiment is run ten times to get the average results.

For storage comparison, the number of attributes $|X|$, associated with access control policy and set of data user attribute $|S|$ is set to 10 and 5, respectively. Also, for uniform comparison we set $||\mathbb{G}_T|| > ||\mathbb{G}_1|| > ||\mathbb{Z}_q||$. From Fig. 2(a)-(f), we can see that OO-ABMS consumes lesser space for the setup phase, public keys for DO, index generation for the server, and token generation for DU. In the conjunctive search query, the search algorithm either outputs 1 or 0. Hence its storage space is not considered in this comparison.

For computation overhead comparison we set $|\mathcal{I}_{max}|$ and $|\mathcal{T}_{max}|$ to 5 attributes. Fig. 3(a) shows the *Offline.Encrypt* algorithm computation cost with $|X|$ varying from 30 to 150 with a step length of 30. OO-ABMS offers better performance as it is affected with a single instance of $|X|$ for its linearity problem, while the rest of the schemes are at least two times. Note that H is a lightweight operation than exponentiation and multiplication in $\mathbb{G}$. Similarly, for *Online.Encrypt* algorithm, Fig. 3(b) shows that our scheme needs multiplication in group $\mathbb{G}_1$, while the schemes in [34], [37] and [35] are affected with a factor of 2 and 3 times of $|X|$ respectively. Besides, a major portion of computation is shifted to the offline phase of encryption. Fig. 3(c) shows the computation done during the offline phase of token generation with $|S|$ varying from 10 to 50 with a step length of 10. The linearity property in terms of $|S|$ affects our scheme on a single occurrence, while the rest of the constructions twice and thus, perform efficiently. Similarly in Fig. 3(d), the *Online.TokenGen* time cost increases single fold of $|S|$, while the rest of the schemes either increases two-fold of $|S|$ or $|\mathcal{T}_{max}|$. Finally, from Fig. 3(e), we can notice that our search algorithm performs less efficiently than that of others schemes [34]–[37]. However, it is worth noticing that our scheme output ranked search results for a better end-user experience. Moreover, our experiment setting does not consider the additional liability in terms of computation and communication overhead by inaccurately pointing the user's interest.

In summary, considering the computation and storage cost, the experiment results are consistent with our theoretical performance analysis. The OO-ABMS is efficient in key, index, and token generation while performing inferior in the search phase. Since the search operation algorithm is run on two resource-rich servers, the OO-ABMS scheme does not bring extra storage and computation cost for the resource-scarce data user devices.

VII. CONCLUSION

This paper proposed a splitting technique to efficiently and practically address costly bilinear pairing operations in ABKS in the mobile cloud computing environment. Based on the online/offline technique, a vast amount of computational work, both for keyword encryption and token generation, is shifted into the offline phase. When the attribute-based access control policy or keyword becomes known during the online phase, the data consumer needs to rapidly construct the token or keyword ciphertext. We are first to introduce the online/offline technique for multiple-keyword ranked search. With the help of this splitting, our experiment results demonstrate the effectiveness of the proposed scheme. Overall, it brings the attribute-based multiple keywords searching technique to an acceptable level in the real-word for mobile resource constraint devices. Moreover, this technique best suits any ABKS schemes where the computation cost is linear with increased attributes. We deploy two servers in our proposed system model in the search phase to resist the keyword guessing attack. Namely, server A performs most of the access control and partial computation for index and trapdoor using its secret key for generating temporal search results. After which, server B gets the final search result using its secret key from the temporal search result. We want to perform all these operations by deploying a single search server and resists the keyword guessing attack as future work.

REFERENCES

[1] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.
[2] M. Li, S. Yu, Y. Zheng, K. Ren, and W. Lou, "Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 1, pp. 131–143, Jan. 2013.
[3] S. Kamara and K. Lauter, "Cryptographic cloud storage," in *Proc. Int. Conf. Financial Cryptogr. Data Secur.* Berlin, Germany: Springer, 2010, pp. 136–149.

[4] M. Bellare, A. Boldyreva, and A. O'Neill, "Deterministic and efficiently searchable encryption," in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 2007, pp. 535–552.
[5] Y. Yang, H. Li, W. Liu, H. Yao, and M. Wen, "Secure dynamic searchable symmetric encryption with constant document update cost," in *Proc. IEEE Global Commun. Conf.*, Dec. 2014, pp. 775–780.
[6] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proc. IEEE Symp. Secur. Privacy (S&P)*, May 2000, pp. 44–55.
[7] J. Shi, J. Lai, Y. Li, R. H. Deng, and J. Weng, "Authorized keyword search on encrypted data," in *Proc. Eur. Symp. Res. Comput. Secur.* Cham, Switzerland: Springer, 2014, pp. 419–435.
[8] W. Sun, S. Yu, W. Lou, Y. T. Hou, and H. Li, "Protecting your right: Attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2014, pp. 226–234.
[9] Q. Zheng, S. Xu, and G. Ateniese, "VABKS: Verifiable attribute-based keyword search over outsourced encrypted data," in *Proc. IEEE Conf. Comput. Commun (INFOCOM).*, Apr. 2014, pp. 522–530.
[10] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. 13th ACM Conf. Comput. Commun. Secur. (CCS)*, 2006, pp. 89–98.
[11] Y. Chen, W. Li, F. Gao, Q. Wen, H. Zhang, and H. Wang, "Practical attribute-based multi-keyword ranked search scheme in cloud computing," *IEEE Trans. Services Comput.*, early access, Dec. 18, 2020, doi: 10.1109/TSC.2019.2959306.
[12] P. Golle, J. N. Staddon, and B. Waters, "System and method for performing a conjunctive keyword search over encrypted data," U.S. Patent 7 783 899, Aug. 24, 2010.
[13] Y.-C. Chang and M. Mitzenmacher, "Privacy preserving keyword searches on remote encrypted data," in *Proc. Int. Conf. Appl. Cryptography Netw. Secur.* Berlin, Germany: Springer, 2005, pp. 442–455.
[14] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: Improved definitions and efficient constructions," *J. Comput. Secur.*, vol. 19, no. 5, pp. 895–934, Jan. 2011.
[15] E.-J. Goh, "Secure indexes for efficient searching on encrypted compressed data," Cryptol. ePrint Arch., Tech. Rep. 2003/216, 2003. [Online]. Available: http://eprint.iacr.org/2003/216
[16] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.* Berlin, Germany: Springer, 2004, pp. 506–522.
[17] S. Kamara, C. Papamanthou, and T. Roeder, "Dynamic searchable symmetric encryption," in *Proc. ACM Conf. Comput. Commun. Secur.*, 2012, pp. 965–976.
[18] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Proc. Theory Cryptogr. Conf.* Berlin, Germany: Springer, 2007, pp. 535–554.
[19] D. Cash, S. Jarecki, C. Jutla, H. Krawczyk, M.-C. Roşu, and M. Steiner, "Highly-scalable searchable symmetric encryption with support for Boolean queries," in *Proc. Annu. Cryptol. Conf.* Berlin, Germany: Springer, 2013, pp. 353–373.
[20] M. Raykova, B. Vo, S. M. Bellovin, and T. Malkin, "Secure anonymous database search," in *Proc. ACM workshop Cloud Comput. Secur.*, 2009, pp. 115–126.
[21] Y. Yang, H. Lu, and J. Weng, "Multi-user private keyword search for cloud computing," in *Proc. IEEE 3rd Int. Conf. Cloud Comput. Technol. Sci.*, Nov. 2011, pp. 264–271.
[22] C. Dong, G. Russello, and N. Dulay, "Shared and searchable encrypted data for untrusted servers," *J. Comput. Secur.*, vol. 19, no. 3, pp. 367–397, 2011.
[23] F. Bao, R. H. Deng, X. Ding, and Y. Yang, "Private query on encrypted data in multi-user settings," in *Proc. Int. Conf. Inf. Secur. Pract. Exper.* Berlin, Germany: Springer, 2008, pp. 71–85.
[24] Y. H. Hwang and P. J. Lee, "Public key encryption with conjunctive keyword search and its extension to a multi-user system," in *Proc. Int. Conf. Pairing-Based Cryptogr.* Berlin, Germany: Springer, 2007, pp. 2–22.
[25] P. Wang, H. Wang, and J. Pieprzyk, "Common secure index for conjunctive keyword-based retrieval over encrypted data," in *Proc. Workshop Secure Data Manage.* Berlin, Germany: Springer, 2007, pp. 108–123.
[26] P. Wang, H. Wang, and J. Pieprzyk, "An efficient scheme of common secure indices for conjunctive keyword-based retrieval on encrypted data," in *Proc. Int. Workshop Inf. Secur. Appl.* Berlin, Germany: Springer, 2008, pp. 145–159.
[27] P. Wang, H. Wang, and J. Pieprzyk, "Keyword field-free conjunctive keyword searches on encrypted data and extension for dynamic groups," in *Proc. Int. Conf. Cryptol. Netw. Secur.* Berlin, Germany: Springer, 2008, pp. 178–195.
[28] M. Li, S. Yu, N. Cao, and W. Lou, "Authorized private keyword search over encrypted data in cloud computing," in *Proc. 31st Int. Conf. Distrib. Comput. Syst.*, Jun. 2011, pp. 383–392.
[29] J. Li, X. Huang, J. Li, X. Chen, and Y. Xiang, "Securely outsourcing attribute-based encryption with checkability," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 8, pp. 2201–2210, Aug. 2014.
[30] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Proc. Int. Workshop Public Key Cryptogr.* Berlin, Germany: Springer, 2011, pp. 53–70.
[31] S. Hohenberger and B. Waters, "Online/offline attribute-based encryption," in *Proc. Int. Workshop Public Key Cryptogr.* Berlin, Germany: Springer, 2014, pp. 293–310.
[32] S. Khan, S. Khan, M. Zareei, F. Alanazi, N. Kama, M. Alam, and A. Anjum, "ABKS-PBM: Attribute-based keyword search with partial bilinear map," *IEEE Access*, vol. 9, pp. 46313–46324, 2021.
[33] Y. Miao, J. Ma, X. Liu, X. Li, Z. Liu, and H. Li, "Practical attribute-based multi-keyword search scheme in mobile crowdsourcing," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 3008–3018, Dec. 2017.
[34] Q. Xu, C. Tan, W. Zhu, Y. Xiao, Z. Fan, and F. Cheng, "Decentralized attribute-based conjunctive keyword search scheme with online/offline encryption and outsource decryption for cloud computing," *Future Gener. Comput. Syst.*, vol. 97, pp. 306–326, Aug. 2019.
[35] Y. Miao, Q. Tong, K.-K.-R. Choo, X. Liu, R. H. Deng, and H. Li, "Secure online/offline data sharing framework for cloud-assisted industrial Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8681–8691, Oct. 2019.
[36] N. Eltayieb, R. Elhabob, A. Hassan, and F. Li, "An efficient attribute-based online/offline searchable encryption and its application in cloud-based reliable smart grid," *J. Syst. Archit.*, vol. 98, pp. 165–172, Sep. 2019.
[37] H. Su, Z. Zhu, and L. Sun, "Online/offline attribute-based encryption with keyword search against keyword guessing attack," in *Proc. 3rd IEEE Int. Conf. Comput. Commun. (ICCC)*, Dec. 2017, pp. 1487–1492.
[38] Y. Miao, J. Ma, X. Liu, X. Li, Q. Jiang, and J. Zhang, "Attribute-based keyword search over hierarchical data in cloud computing," *IEEE Trans. Services Comput.*, vol. 13, no. 6, pp. 985–998, Nov./Dec. 2017.

SHAHZAD KHAN received the B.S. degree in computer science from Malakand University and the M.S. degree in computer and communication security from the School of Electrical Engineering and Computer Science (SEECS), National University of Science and Technology (NUST), Islamabad, Pakistan, where he is currently pursuing the Ph.D. degree with the Military College of Signals (MCS). He is currently an Assistant Professor with Shaheed Benazir Bhutto University. His research interests include applied cryptography and information security.

MAHDI ZAREEI (Senior Member, IEEE) received the M.Sc. degree in computer network from the University of Science, Malaysia, in 2011, and the Ph.D. degree from the Communication Systems and Networks Research Group, Malaysia-Japan International Institute of Technology, University of Technology Malaysia, in 2016. In 2017, he joined the School of Engineering and Sciences, Tecnológico de Monterrey, as a Postdoctoral Fellow, where he has been working as a Research Professor, since 2019. His research interests include wireless sensor and *ad hoc* networks, energy harvesting sensors, information security, and machine learning. He is a member of Mexican National Researchers System (level I). He also serves as an Associate Editor for IEEE Access and *Ad Hoc & Sensor Wireless Networks* journals.

SHAWAL KHAN received the bachelor's degree in computer science from Shaheed Benazir Bhutto University, Upper Dir, Khyber Pakhtunkhwa, Pakistan. He is currently pursuing the master's degree in information security with the COMSATS Institute of Information Technology, Islamabad, Pakistan. His research interests include access control, cryptography, and network security.

FAISAL ALANAZI received the B.Sc. degree in electrical engineering (electronics and communication) from Kennesaw State University (KSU), and the M.Sc. and Ph.D. degrees in electrical and computer engineering from The Ohio State University, in 2013 and 2018, respectively. He is currently working as an Assistant Professor with Prince Sattam Bin Abdulaziz University (PSAU). His research interests include cryptography, vehicular *ad-hoc* networks, and delay tolerant networks. He is a member of the IEEE Communication Society.

MASOOM ALAM received the Ph.D. degree in computer sciences from the University of Innsbruck, Austria. He is currently an Associate Professor with the Department of Computer Science, COMSATS Institute of Information Technology, Islamabad, Pakistan. His interests include access control systems, model-driven architecture, and workflow management systems.

ABDUL WAHEED received the master's and Ph.D. degrees in computer science from the Department of Information Technology, Hazara University Mansehra, in 2014 and 2021, respectively. He was a Ph.D. Researcher with the NetLab-INMC, School of Electrical and Computer Engineering (ECE), Seoul National University (SNU), South Korea, in 2019, under the HEC research program. He is currently a member of the Crypto-Net Research Group, Hazara University. He also serves as a Lecturer for the Department of Computer Science, IQRA National University, Peshawar. He has numerous publications in journals and international conferences. His research interests include information security, secure and smart cryptography, heterogeneous communications within the IoT, mobile *ad-hoc* networks (MANETs), wireless sensor networks (WSNs) security, and fuzzy logic-based decision-making theory.