Received 31 October 2023, accepted 10 November 2023, date of publication 20 November 2023, date of current version 15 December 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3334733

RESEARCH ARTICLE

Multi-Keywords Searchable Attribute-Based Encryption With Verification and Attribute Revocation Over Cloud Data

HUA SHEN[1], (Member, IEEE), JIAN ZHOU[1], GE WU[2], AND MINGWU ZHANG[1]
[1]School of Computer Science, Hubei University of Technology, Wuhan 430068, China
[2]School of Cyber Science and Engineering, Southeast University, Nanjing 211189, China

Corresponding author: Mingwu Zhang (csmwzhang@gmail.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61702168, Grant 62072134, Grant 62002058, and Grant U2001205; in part by the Natural Science Foundation of Guangxi Province under Grant 2019JJD170020; in part by the Natural Science Foundation of Jiangsu Province under Grant BK20200391; and in part by the Green Industry Technology Leading Program of Hubei University of Technology under Grant XJ2021000901.

ABSTRACT The convenience and efficient management of cloud servers have resulted in an increasing number of users opting to store their data in the cloud. Consequently, data-outsourcing services relying on cloud servers have been extensively deployed and utilized. In this case, before outsourcing the data to cloud storage, we should ensure unauthorized users cannot access the data. In this article, we present a scheme termed MKSABE-VaAR (multi-keyword searchable attribute-based encryption with verification and attribute revocation). Aiming at the problems of inefficiency, excessive computation in the search process, and only supporting single-keyword search in most attribute-based searchable encryption schemes, first, we package multiple keywords into a polynomial to realize multi-keyword search. Based on this polynomial, MKSABE-VaAR can reduce the amount of search calculation for keyword ciphertext and improve search efficiency by reducing the number of bilinear pairing operations. At the same time, we have made some special constructs for indexing to add verification of user attributes in the keyword search process, which improves the search accuracy. Moreover, we adopt a linear secret-sharing technique to construct the attribute revocation function of MKSABE-VaAR, making a lower computational cost of attribute revocation. Furthermore, the mechanism of storing data and indexes separately greatly reduces the risk of data leakage of MKSABE-VaAR.

INDEX TERMS Searchable encryption, attribute-based encryption, linear secret sharing, attribute revocation.

I. INTRODUCTION

With internet technology's rapid advancement, users increasingly demand efficient data processing and storage [1], [2]. To cater to these requirements, it has become common practice to outsource individual or enterprise data to cloud servers [3]. However, the need for more control over outsourced data poses significant challenges to the privacy and security of data owners [4], [5]. Uploading encrypted data is a common solution. We can use attribute-based encryption to encrypt data, which can realize fine-grained access control. However, the retrieval of ciphertext still needs to be improved. When the data stored on the cloud becomes more, it becomes difficult to retrieve encrypted data. Therefore, how to effectively retrieve encrypted files is critical in practical applications. The conventional approach involves downloading and decrypting files for querying, which incurs network overhead due to unnecessary file downloads and computational overhead for decryption and querying. Consequently, the optimal solution entails the ability to query ciphertext directly on the server, ensuring the privacy of the data while retrieving it and reducing

The associate editor coordinating the review of this manuscript and approving it for publication was Mahdi Zareei.

the communication overhead, leading to the emergence of searchable encryption technology.

Searchable encryption (SE) is a technique that encrypts data files, stores them in the cloud, and then retrieves the ciphertext. To save their resource costs, users outsource files to cloud servers but do not want the cloud service to know the stored file content while ensuring that only legitimate users can search the corresponding ciphertext data based on keywords. Hence, they need to encrypt and store the files using searchable encryption. Searchable encryption schemes allow one to securely search and selectively retrieve encrypted cloud data of interest based on user-specified keywords.

Generally speaking, in a cloud system, two functions are indispensable: one is to retrieve target data without leaking privacy, and the other is fine-grained access control. The ciphertext-policy attribute-based keyword search encryption scheme (CP-ABKS) can meet the above two requirements at the same time. The currently available CP-ABKS schemes can be divided into two types: single-keyword searchable attribute-based encryption schemes and multi-keyword searchable attribute-based encryption schemes (MKSABE). The issues with single-keyword searchable attribute-based encryption are evident. It frequently yields numerous irrelevant search results, wastes computing resources, and reduces search efficiency, which can degrade the user's search experience. As a result, the MKSABE scheme has emerged. However, most existing MKSABE schemes also have many problems. For example, the search process is too computationally intensive. Some MKSABE schemes require a bilinear pairing operation for each keyword, which can be a huge computational overhead when the number of keywords increases. And search results are not verified. Most MKSABE schemes return search results directly to the user after completing the search, without ensuring that the search results are accessible to the user. And also excessive computation and traffic during the attribute revocation process. The traditional MKSABE schemes directly re-encrypt and upload the data as a function of attribute revocation, and the amount of data is often very large, which requires huge computing and communication volumes. Therefore, our research motivations are as follows:

- The single-keyword search model is not powerful. The single-keyword search model returns many irrelevant search results, resulting in inefficient searches, inaccurate search results, and a bad user experience.
- The search process is too computationally intensive. Certain ABKS schemes search processes require a bilinear pairing operation for each keyword, and the bilinear pairing operation is more computationally intensive, which will cause more computational overhead for the cloud server.
- The search results are not verified. Many ABKS schemes do not verify the search results, rather than directly return the search results to users, which cannot ensure that the users are the ones who can access the search results.
- The amount of calculation and communication required for attribute revocation is substantial. Traditional ABKS schemes typically involve directly replacing all ciphertexts in response to attribute revocation, resulting in significant computational and communication overhead.

A. PAPER CONTRIBUTIONS

Based on our motivations, we wanted the ABKS scheme to be more suitable for cloud systems. To achieve this, we proposed the Multi-Keyword Searchable Attribute-Based Encryption with Verification and Attribute Revocation over Cloud Data (MKSABE-VaAR) scheme. The MKSABE-VaAR scheme encompasses multi-keyword search functionality, verification of search results, and low-cost attribute revocation. The key contributions of our paper can be summarized as follows:

- Efficient multi-keyword search. We construct the keyword polynomial, thereby realizing the multi-keyword search function. Due to the existence of keyword polynomials, we will need to calculate the value of the keyword polynomial and perform one bilinear pairing operation (because of packaging multiple keywords into one polynomial value), reducing the computational overhead during the search process.
- Search results verification. We add a verification step to the search process and then verify the search results to confirm that the retrieved results align with the user's intentions and accessibility. Verification of search results ensures their relevance and availability, leading to a significant improvement in search accuracy.
- Low-cost attribute revocation. We convert a new access policy and the attributes into a linear secret sharing matrix. The recalculated row vector of the matrix is then embedded into each attribute of the ciphertext. As a result, only part of the ciphertext needs to be re-uploaded to the cloud server, significantly reducing the computational and communication overhead associated with attribute revocation.

B. PAPER OUTLINE

In Section II, we discuss the related works. In Section III, review some preliminaries. Next, we describe our system model with relevant security model, as well as the definition of relevant algorithms in Section IV. In Section V, we present our proposed MKSABE-VaAR scheme, followed by correctness discussion, security analysis, and performance analysis in sections VI, VII, and VIII, respectively. Section IX concludes this paper.

II. RELATED WORKS

Due to the increasing usage of data outsourcing services, individuals and enterprises increasingly opt to outsource their data to cloud servers. To safeguard the privacy of their data, users typically choose to encrypt it before outsourcing. Although encrypted data successfully ensures integrity and confidentiality, it conceals the characteristics of the original data. Consequently, calculating and sharing encrypted data becomes challenging, reducing flexibility for data outsourcing. This paper aims to provide keyword-based information retrieval and fine-grained access control over encrypted cloud data. Hence, in this section, we particularly relate to Ciphertext-Policy Attribute-Based Encryption (CP-ABE) and SE schemes.

SE schemes can search for specific data based on the user's specified keywords without leaking user privacy. SE schemes are divided into symmetric [6] and asymmetric [7]. Song [6] first proposed a searchable encryption scheme to solve the ciphertext retrieval problem based on symmetric key encryption data. Still, the symmetric SE scheme usually entails expensive key management and communication overhead. Boneh [7] first proposed Public-key Encryption with Keyword Search (PEKS) scheme. Due to the use of public-key cryptography, this scheme is more suitable for cloud computing. Since then, a large number of versatile SE schemes have been presented. Such as single keyword search [8] and multi-keyword search [9]. Huang proposed a Public-key Authenticated Encryption with Keyword Search (PAEKS). The PAEKS solves the problem that an inside adversary may recover the keyword, but this scheme only supports single-keyword searches. Fan [9] proposed a multi-keyword searchable encryption scheme supporting search result verification. However, this scheme cannot guarantee the correctness of the returned search results under a semi-trusted cloud server. A lot of searchable schemes focus on fuzzy keywords [10], [11], [12] and sorted search keywords [13], [14]. Liu [10] used vector inner product to realize keyword search and the file decryption key required key agreement between users. Ahmed [13] used searchable symmetric encryption to sort documents according to the correlation between documents and queries. Any user can obtain the required documents by uploading the search trapdoor.

Although searchable encryption can search ciphertext data, users who perform search tasks in a cloud storage environment can use any keyword to search for the desired data in all ciphertext databases. Data owners cannot control that specific data can be shared with specific users, making it challenging to achieve fine-grained access control.

Attribute-based encryption can meet the needs of fine-grained access control. Shamir [15] proposed an identity-based encryption and signature scheme. In this scheme, the user can choose a unique user identifier as his public key. Then, the data owner can send a ciphertext that only the user can view and verify to the user. Sahai and Waters [16] first proposed an attribute-based encryption scheme. Data owners need to formulate access policies, and only the user's attributes meet the access policy can decrypt the data. The attribute-based encryption has two types: key policy attribute-based encryption (KP-ABE) [17] and ciphertext policy attribute-based encryption (CP-ABE) [18]. The KP-ABE embeds the policy into the key and the attribute into the ciphertext [17]. The key corresponds to an access control, and the ciphertext corresponds to an attribute set. The CP-ABE embeds the policy into the ciphertext and the attribute into the key. The ciphertext corresponds to an access structure, and the key corresponds to the attribute set [18]. Both schemes can satisfy fine-grained access control, but the KP-ABE is more biased towards static scenarios. Therefore, in the cloud storage environment, CP-ABE is more used. The user obtains the key from the attribute organization according to his conditions or attributes, and then the data owner formulates the access control of the message. Waters [18] used a Linear Secret-Sharing Scheme (LSSS) matrix to represent the access control structure of ciphertext. LSSS has better expression ability under the same performance and function as the tree access control structure. Despite the number of research efforts on this topic, existing CP-ABE schemes still need to solve the problem of keyword-based data retrieval entirely.

Due to these problems, researchers are committed to extending attribute-based encryption to the SE scheme. Michalas [19] combined symmetric searchable encryption with attribute-based encryption to construct a hybrid searchable encryption scheme. This scheme has the characteristics of a symmetric searchable encryption scheme that can retrieve ciphertext and realize access control functions. Yin [20] proposed a ciphertext-policy attribute-based searchable encryption scheme. Because this scheme can only perform single-keyword search, it will cause the search results returned by the server to contain much irrelevant content, waste network bandwidth, and increase communication overhead due to multiple rounds of search. Li [21] proposed an attribute-based searchable encryption scheme supporting multi-keyword search. This scheme allows multiple users to query keyword ciphertext. Aiming at the problem that malicious users and malicious cloud servers illegally search encrypted data files, a trusted attribute-based searchable encryption scheme based on cloud storage is proposed by Zhang [22] to achieve finer-grained keyword search. The schemes of Wang et al. [23], Zhu et al. [24], Zheng et al. [25] introduced attribute-based encryption algorithms based on cloud storage. Zheng [25] proposed a verifiable attribute-based keyword searchable encryption scheme, which is also a single keyword search with low efficiency. The schemes of Liu et al. [26] and Song et al. [27] all adopt the method of directly verifying the access policy in the cloud. If successful, only the ciphertext is returned to the user for decryption. The decryption of the ciphertext is not related to the policy, resulting in an attacker skipping the access policy and directly performing index matching and file decryption.

In practice, we should construct a preferred CP-ABKS scheme without sacrificing efficiency while supporting fine-grained access control, multi-keyword search, search results verification, and attribute revocation. Zhou [28] proposed a multi-keyword searchable encryption scheme. The scheme controls the user's search rights by the data owner. Data owners can also share data, and data sharing computation and communication between data owners is less than data sharing between data owners and users. However, this scheme does not support attribute revocation and search result verification. Chaudhari and Das [29] proposed a multi-keyword searchable attribute encryption scheme with a hidden access policy. The search algorithm of the scheme is also based on keyword search. Hiding access policies helps protect data confidentiality. When the user performs a search query, the user's attributes will be hidden in the search query. The shortcoming of this scheme is that it still needs to support attribute revocation and search result verification. Zhang et al. [30] proposed a verifiable multi-keyword searchable attribute encryption scheme. The scheme adds third-party entities to verify search results. In addition, the data file of the scheme is encrypted by multiple data owners, which enhances access control. Data users can decrypt files only after obtaining the permissions of multiple data owners. The scheme supports search result verification but does not support attribute revocation. Miao et al. [31] proposed a multi-keyword searchable attribute-based encryption scheme that supports attribute updates. The scheme supports attribute revocation.

Compared with the traditional CP-ABE scheme that needs to update the entire ciphertext, the scheme only needs to update a fraction of ciphertexts and indexes to achieve the purpose of attribute revocation. However, the scheme does not support search result verification. Sangeetha et al. [32] combined location-based encryption (LBE) and dynamic location-based re-encryption (DLBRE) techniques with an attribute-based searchable encryption scheme to improve the security of ciphertext. Varri et al. [33] proposed a lattice-based attribute-based searchable encryption scheme. The scheme can resist quantum attacks since there is no quantum attack in lattice-based encryption. Zhang et al. [34] proposed a consistent ABKS system with cryptographic reverse firewalls (CRF) that supports fine-grained owner-enforced search authorization over different data users and considers enhanced security and soundness for ABKS against inside and outside threats to a data user. Yang et al. [35] introduces the new conception of updatable and transferable message-lock encryption (UT-MLE) for block-level dynamic encrypted file update, where the owner does not have to download the whole ciphertext, decrypt, re-encrypt, and upload for minor document modifications.

III. PRELIMINARIES

A. BILINEAR MAPS

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p. There exists a bilinear map e: $\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ satisfying the following three properties:

Bilinear: For all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.

Non-degeneracy: $\exists\, u, v \in \mathbb{G}, e(u, v) \neq 1$.

Computability: $\forall\ u, v \in \mathbb{G}$, $e(u, v)$ can be efficiently computed.

B. ACCESS STRUCTURE

Let $\{P_1, P_1, \cdots, P_n\}$ be the entity set of n participants, set $A \subseteq 2^{\{P_1, P_2, \cdots, P_n\}}$. If $\forall\, B, C, B \in A, B \subseteq C$ have $C \in A$, then A is monotonic. If A is not empty and $A \subseteq 2^{\{P_1, P_2, \cdots, P_n\}}$, then A is an access structure. The set in A is called an authorized set, and a set not in A is called an unauthorized set.

C. LINEAR SECRET SHARING SCHEME

Let U be an attribute set. The plaintext is encrypted into ciphertext using a specific access control policy. Only users whose attributes meet the access policy requirements can decrypt the ciphertext. Using a linear secret sharing scheme, we can convert the entity attribute set U into a linear secret sharing matrix M with dimensions $l \times n$. Each row in the matrix M corresponds to an attribute value, the mapping function ρ: $\{1, 2, \cdots, l\} \rightarrow U$. Randomly select a column vector $\boldsymbol{v} = (s, r_2, \cdots, r_n)^{\top} \in \mathbb{Z}_p$, $s \in \mathbb{Z}_p$, where s is the secret value that needs to be shared, and s is encrypted by column vector $\boldsymbol{v}$. Denotes the l shares of the secret s, $\lambda_i = \boldsymbol{M_i} \cdot \boldsymbol{v}$ is the share of the participant $\rho(i)$, and $\boldsymbol{M_i}$ is the vector corresponding to the row i in the linear secret sharing matrix.

The secret sharing scheme satisfies the linear reconstruction property: Let Π denote a linear secret sharing scheme, S be an authorization set. I: $\{i\ :\ \rho(i) \in S\} \subseteq \{1, 2, \cdots, l\}$. Then there is a set of constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ satisfying $\sum_{i \in I} \omega_i \cdot \boldsymbol{M_i} = (1, 0, \cdots, 0)$ for the effective share λ_i on Π, satisfies $\sum_{i \in I} \omega_i \cdot \lambda_i = s$.

D. DIFFICULTY ASSUMPTION

1) COMPUTATIONAL BILINEAR DIFFIE-HELLMAN (CBDH) ASSUMPTION

Let a, b, c are chosen randomly from Z_p. $g \in \mathbb{G}$ is the generator. The CBDH assumption is that given a tuple ($A = g^a, B = g^b, C = g^c$), then the advantage of computing $e(g, g)^{abc}$ is negligible in security parameter κ.

2) COMPUTATIONAL DIFFIE-HELLMAN (CDH) ASSUMPTION

Let a, b be chosen randomly from Z_p. $g \in \mathbb{G}$ is the generator. The CDH assumption is that given a tuple ($A = g^a$, $B = g^b$), the advantage of computing g^{ab} is negligible in security parameter κ.

IV. FORMAL DEFINITION AND SECURITY MODEL

A. SYSTEM MODEL

The system comprises five entities: Data Owner, Data Users, Cloud Server, Key Generation Center, and InterPlanetary File System. The system model is illustrated in Fig. 1.

Data Owner (DO): The Data Owner (DO) devises the access strategy, uploads the encrypted file F to the

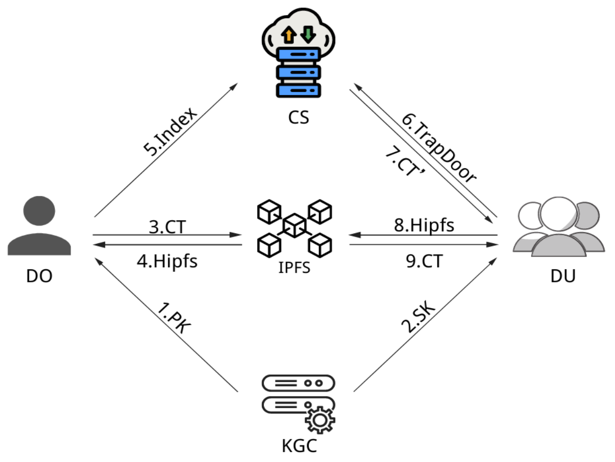


FIGURE 1. System model.

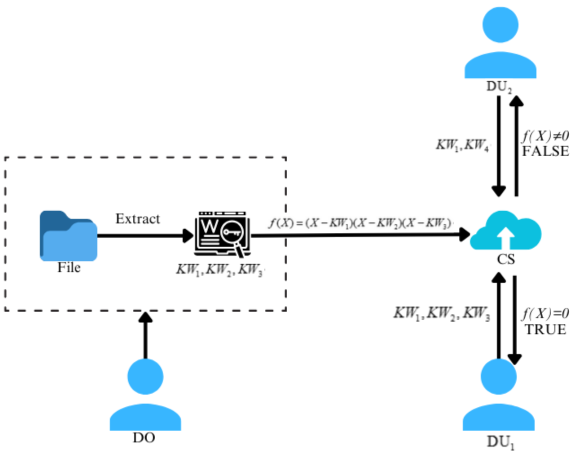


FIGURE 2. Search.

InterPlanetary File System (IPFS), extracts keywords from the files, employs the index generation algorithm to create the index based on these keywords, and finally stores the index on the Cloud Server (CS).

Data Users (DU): The Data User (DU) possesses secret keys that are associated with their attribute set, as well as trapdoors that are associated with their secret keys and query keywords. The DU initiates a search request to the Cloud Server (CS). If the attributes satisfy the access policy and the keywords match the keyword polynomial, the DU can obtain the partial index. Subsequently, the DU can access the InterPlanetary File System (IPFS) storage system to retrieve the ciphertext of the desired file. The DU then proceeds to decrypt the ciphertext.

Cloud Server (CS): The Cloud Server (CS) is responsible for storing encrypted indexes. In addition, it performs keyword searching, verifies whether the attributes of the user satisfy the access policy, and ultimately returns the partial index to the Data User (DU).

Key Generation Center (KGC): The Key Generation Center (KGC) is a fully trusted entity tasked with generating the system master key and public parameters. Furthermore, it generates users' secret key.

The InterPlanetary File System (IPFS): IPFS is a collection of peer-to-peer protocols that are composable and used for addressing, routing, and transferring content-addressed data within a decentralized file system. It is tasked with storing encrypted files and generates a unique hash value for each stored file. Then, it creates a list that contains files' ciphertext, files' hash value and files' address.

The abbreviated search process is shown in Fig. 2. Let's take three keywords and a single file as an example. DO extracts three keywords from the file and performs the following operations on the keywords. Let $f(X) = (X - KW_1)(X - KW_2)(X - KW_3)$ be a polynomial. Then DU_1 and DU_2 submit their queried keywords. We can see from Fig. 2 that the output is TRUE only when the value of the polynomial $f(X) = 0$, that is, continue the subsequent calculation. For ease of understanding, Fig. 2 is just an example of an abbreviated search step. In section V, we give the complete search algorithm.

B. DEFINITION OF ALGORITHM

The proposed MKSABE-VaAR composes of the following algorithms:

- $\underline{Setup(1^{\kappa}, U) \to PK, MSK}$: KGC executes the algorithm, which inputs a security parameter κ and an attribute set U, outputs a system public parameter PK, a master key MSK.
- $\underline{KeyGen(PK, MSK, S) \to SK}$: KGC runs the algorithm, which inputs DU's attribute set S, the system public parameter PK, and the master key MSK, and outputs DU's secret key SK.
- $\underline{Encrypt(PK, F) \to CT, add_i, K_i}$: DO generates symmetric keys K_i $(i = 1, \cdots, |F|)$ for each file of a given file set F, encrypts F using K_i to obtain the ciphertext CT, stores CT into IPFS, and receives the corresponding file address add_i returned by IPFS. IPFS calculates add_i using the public parameter PK.
- $\underline{IndexGen(PK, (M, \rho), F, add_i, K_i) \to Index}$: DO executes the algorithm, which takes the system public parameter PK, an access structure (M, ρ), the data file F, the file address add_i, and the symmetric key K_i $(i = 1, \cdots, |F|)$ as inputs and outputs the index $Index$.
- $\underline{Trapdoor(PK, KW_U, SK) \to T}$: DU executes the algorithm, which inputs the public parameter PK, the query keyword set KW_U, the DU's secret key SK, and outputs the search trapdoor T.
- $\underline{Search(Index, T) \to CT'}$: CS carries out the algorithm, which inputs the index $Index$ and the searches trapdoor T. If the trapdoor submitted by DU matches the index successfully, the algorithm outputs a partial index ciphertext CT'.
- $\underline{Decrypt(CT', SK) \to F_i}$: DU runs the algorithm, which inputs the partial index ciphertext CT' and DU's secret key SK. DU decrypts CT' to get the $H_{3,i}$, making that DU can

accesse IPFS to get CT_i and then decrypts CT_i with K_i to obtain F_i.

We can realize attribute revocation using linear secret sharing scheme. The corresponding processing is as follows:

1) DO calculates a new LSSS matrix M' based on new access policy.
2) DO calculates λ'_i, where λ'_i is the value representing the attribute in the new access policy. Then, DO calculates C'_i based on λ'_i and uploads C'_i to CS.
3) When CS receives C'_i, it replaces C_i in index with C'_i and generates a new $Index'$.

C. SECURITY MODEL

The security analysis of a searchable ABE scheme is formally checked using the IND-CP-CKA (Indistinguishability against ciphertext-policy and chosen-keyword attack) model [29]. The IND-CP-CKA model can ensure that a search operation over encrypted data does not compromise the confidentiality of data or the receiver's anonymity. In this model, the adversary $\mathcal{A}$ is allowed to issue the polynomial number of queries for getting the trapdoors used to conduct the search operation. The challenger $\mathcal{C}$ uses a keyword and the user credentials submitted by the $\mathcal{A}$ to generate the trapdoor. The $\mathcal{A}$ also is given access to an encrypted index. The security model defines that from the available encrypted index and the trapdoor, the $\mathcal{A}$ can not learn the encrypted keyword.

Indistinguishability Against Ciphertext-Policy and Chosen Keyword Attack The following are definitions of indistinguishable games.

- *Initialization*: $\mathcal{A}$ submits a security parameter κ and the access structure (M^*, ρ^*) to $\mathcal{C}$. $\mathcal{C}$ runs the system to create an algorithm that sends the generated system common parameter PK to the $\mathcal{A}$.
- *Phase 1*: $\mathcal{A}$ issues the adaptively generated queries with the input of keyword KW_U and set of attribute S to retrieve the corresponding trapdoor T. The $\mathcal{C}$ responds with T generated with the input KW_U and S.

Challenge: The $\mathcal{A}$ gives two pairs of input (KW^0, S^0) and (KW^1, S^1) to $\mathcal{C}$. The pairs of input offered by $\mathcal{A}$ must fulfill the following criteria.

1) KW^0 and KW^1 are set of keywords with equal length.
2) $\mathcal{A}$ has not gained a trapdoor T which can satisfy any of the challenge ciphertext.

If either of the above mentioned criteria fails, then $\mathcal{C}$ aborts; else, the $\mathcal{C}$ randomly selects $\theta \in \{0, 1\}$ and computes the encrypted index of keywords $Index^\theta$. $\mathcal{C}$ submits $Index^\theta$ to $\mathcal{A}$.

- *Phase 2*: As performed in phase 2, the $\mathcal{A}$ submits the adaptively generated queries as keyword KW_U and a set of attribute S to gain the trapdoor T.The input submitted by $\mathcal{A}$ must follow either criteria.

1) Neither KW^0 nor KW^1 should contain KW_U.
2) S does not match the access policy.

In response, the $\mathcal{C}$ submits to the $\mathcal{A}$ with trapdoor T corresponding to (KW_U, S).

Guess: The $\mathcal{A}$ gives a guess θ' for θ. If $\theta' = \theta$ the $\mathcal{A}$ wins the game. The advantage of $\mathcal{A}$ for winning this game is $\text{Adv}_{\mathcal{A}} = |\Pr[\theta' = \theta] - \frac{1}{2}|$.

Definition 1: The searchable ABE scheme is IND-CP-CKA secure if the advantage of any polynomially bounded adversary to win the above game is non-negligible in the security parameter.

V. CONCRETE SCHEME

A. MKSABE-VAAR

MKSABE-VaAR includes seven algorithms. The descriptions of them are as follows:

- $\underline{Setup(1^\kappa, U) \rightarrow PK, MSK}$: KGC runs this algorithm. According to the security parameter κ, the algorithm generates two multiplicative cyclic groups ($\mathbb{G}$ and $\mathbb{G}_T$) of prime order p. g is the generator of $\mathbb{G}$. The algorithm creates a bilinear mapping function e: $\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. U is an attribute set provided by DO, $U = \{att_1, att_2, \ldots, att_l\}$. The algorithm sets two anti-collision hash functions: H_1: $\{0, 1\}^* \rightarrow \mathbb{Z}_p$ and H_2: $\{0, 1\}^* \rightarrow \mathbb{G}$. For each attribute $att_i \in U$, the algorithm calculates $H_2(att_i)$ to obtain the corresponding attribute value set $U_v = \{H_2(att_1), H_2(att_2), \cdots, H_2(att_l)\}$. The algorithm randomly selects the parameters $\alpha, \beta, \gamma \in \mathbb{Z}_p$. Finally, the public parameter is $PK = \{g, e(g, g)^\alpha, e(g, g)^\gamma, g^\beta, g^\gamma, U_v, H_1, H_2\}$, the master key is $MSK = \{g^\alpha, \alpha, \beta, \gamma\}$.

Algorithm 1 Algorithm MKSABE-VaAR.Setup

```
Setup(1^κ, U)
KGC:
 1: Initialize a Bilinear Maps function $e$: $\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$
 2: Initialize Hash function $H_1$: $\{0, 1\}^* \rightarrow \mathbb{Z}_p$, $H_2$: $\{0, 1\}^* \rightarrow \mathbb{G}$
 3: $g \leftarrow \mathbb{G}$
 4: for $i$ in length $U$ do
 5:     $H_2(att_i)$
 6: end for
 7: $U_v = \{H_2(att_1), H_2(att_2), \cdots, H_2(att_l)\}$
 8: $\alpha, \beta, \gamma \leftarrow \mathbb{Z}_p$
 9: $e(g, g)^\alpha, e(g, g)^\gamma, g^\beta, g^\gamma$
10: $PK = \{g, e(g, g)^\alpha, e(g, g)^\gamma, g^\beta, g^\gamma, U_v, H_1, H_2\}$
11: $MSK = \{g^\alpha, \alpha, \beta, \gamma\}$
12: Return $PK, MSK$
13: Send $PK$ to DO
```

- $\underline{KeyGen(PK, MSK, S) \rightarrow SK}$: KGC carries out the algorithm. The algorithm generates SK associated with the attribute for DU based on PK, MSK, and the attribute set S submitted by DU. The algorithm randomly selects the parameter $t \in \mathbb{Z}_p$, calculates $uk = g^\alpha g^{\beta t}$, $uk_0 = g^t$, $uk_T = g^\gamma g^{\beta t}$ and $uk_i = H_2(att_i)$. Finally, the algorithm sends $SK = (uk, uk_0, uk_T \{uk_i\}_{i \in S})$ to DU.
- $\underline{Encrypt(PK, F) \rightarrow CT, add_i, K_i}$: DO does the following process for his file set $F = \{F_1, F_2, \cdots, F_m\}$ by calling this algorithm. DO encrypts each file $F_i \in F$ with corresponding symmetric key K_i, where $i \in [1, m]$. The encrypted ciphertext $CT_i = Enc_{K_i}(F_i)$, Enc

Algorithm 2 Algorithm MKSABE-VaAR.KeyGen

```
$KeyGen(PK, MSK, S)$
DU:
 1: Send $S$ to KGC
KGC:
 1: $t \leftarrow \mathbb{Z}_p$
 2: $uk = g^{\alpha} g^{\beta t}$
 3: $uk_0 = g^t$
 4: $uk_T = g^{\gamma} g^{\beta t}$
 5: for $i$ in length $S$ do
 6:     $uk_i = H_2(att_i)^t$
 7: end for
 8: $SK = (uk, uk_0, uk_T, \{uk_i\}_{i \in S})$
 9: Return $SK$
10: Send $SK$ to DU
```

TABLE 1. File storage address and hash value.

Files	Hash value	Address
CT_1	H^1_{ipfs}	add_1
CT_2	H^2_{ipfs}	add_2
$\cdots$	$\cdots$	$\cdots$
CT_m	H^m_{ipfs}	add_m

represents a secure symmetric encryption algorithm. $CT = \{CT_1, CT_2, \cdots, CT_m\}$. DO uploads CT to IPFS. IPFS stores CT and generates the corresponding hash value $H_{ipfs} = \{H^1_{ipfs} = H_1(CT_1), H^2_{ipfs} = H_1(CT_2), \cdots, H^m_{ipfs} = H_1(CT_m)\}$. Since our scheme uses the multiplicative cyclic group, IPFS calculates $add_i = e(g, g)^{\beta H^i_{ipfs}}$ $(i = 1, 2, \cdots, m)$ to represent file address and returns them to DO.

Algorithm 3 Algorithm MKSABE-VaAR.Encrypt

```
$Encrypt(PK, F)$
DO:
 1: for $i \in [1, m]$ do
 2:     Generate $K_i$
 3:     $CT_i = Enc_{K_i}(F_i)$
 4: end for
 5: $CT = \{CT_1, CT_2, \cdots, CT_m\}$
 6: Return $CT$
 7: Send $CT$ to IPFS.
IPFS:
 1: for $i \in [1, m]$ do
 2:     Store $F_i$
 3:     Generate $H^i_{ipfs}$
 4:     $add_i = e(g, g)^{\beta H^i_{ipfs}}$
 5: end for
 6: Return $add_i$
 7: Send $add_i$ to DO
```

- $IndexGen(PK, (M, \rho), F, add_i, K_i) \rightarrow Index$: DO performs the following calculation for each file F_i. DO converts the access policy to the LSSS access matrix $M_{l \times n}$, and maps the ith row in the matrix $M_{l \times n}$ to the participant set through the mapping function $\rho : \{1, 2, \cdots, l\} \rightarrow P_i$. DO randomly selects the column vector $\boldsymbol{v} = (s, y_2, \cdots, y_n)^{\top} \in \mathbb{Z}_p$, where s is the secret value to be shared, and calculates $\lambda_i = \boldsymbol{M_i} \cdot \boldsymbol{v}$, $i \in [1, l]$. DO extracts μ keywords from the file F_i, assumes that the keyword set $KW = \{KW_1, KW_2, \cdots, KW_{\mu}\}$ of a single file, and hashes each keyword. DO calculates the polynomial $f(X) = (X - H_1(KW_1))(X - H_1(KW_2)) \ldots (X - H_1(KW_{\mu})) = f_0 X^{\mu} + f_1 X^{\mu-1} + \ldots + f_{\mu-1} X + f_{\mu}$ where f_i is the coefficient of the $i + 1$ phase of polynomial, then $E_i = g^{f_i}$. DO randomly selects $r_1, r_2, \cdots, r_l \in \mathbb{Z}_p$ and calculates $C = add_i \cdot e(g, g)^{\alpha s}$, $C_K = K_i \cdot e(g, g)^{\alpha s}$, $C' = e(g, g)^{\gamma s}$, $C_0 = g^s$, $C_i = g^{\beta \lambda_i} H_2(att_i)^{-r_i}$, $D_i = g^{r_i}$. The index is $Index = (C, C_K, C', C_0, \{C_i, D_i\}_{i \in [1,l]}, \{E_j\}_{j \in [0,\mu]})$. DO sends the $Index$ to CS.

Algorithm 4 Algorithm MKSABE-VaAR.IndexGen

```
$IndexGen(PK, (M, \rho), F, add_i, K_i)$
DO:
 1: $\boldsymbol{v} = (s, y_2, \cdots, y_n)^{\top} \leftarrow \mathbb{Z}_p$
 2: $\lambda_i = \boldsymbol{M_i} \cdot \boldsymbol{v}$
 3: Extract keywords from $F_i$
 4: $KW = \{KW_1, KW_2, \cdots, KW_{\mu}\}$
 5: $f(X) = (X - H_1(KW_1))(X - H_1(KW_2)) \ldots (X - H_1(KW_{\mu})) = f_0 X^{\mu} + f_1 X^{\mu-1} + \cdots + f_{\mu-1} X + f_{\mu}$
 6: for $i \in [1, l]$ do
 7:     $r_i \leftarrow \mathbb{Z}_p$
 8:     $C_i = g^{\beta \lambda_i} H_2(att_i)^{-r_i}$
 9:     $D_i = g^{r_i}$
10: end for
11: $C = add_i \cdot e(g, g)^{\alpha s}$
12: $C_K = K_i \cdot e(g, g)^{\alpha s}$
13: $C' = e(g, g)^{\gamma s}$
14: $C_0 = g^s$
15: $E_i = g^{f_i}$
16: $Index = (C, C_K, C', C_0, \{C_i, D_i\}_{i \in [1,l]}, \{E_j\}_{j \in [0,\mu]})$
17: Return $Index$
18: Send $Index$ to CS
```

- $Trapdoor(PK, KW_U, SK) \rightarrow T$: The keyword set to be queried by DU is $KW_U = \{KW_{U_1}, KW_{U_2}, \cdots, KW_{U_k}\}$, where k is the number of keywords to be queried. DU performs a hash operation on each keyword and outputs $T_i = H_1(KW_{U_i})$. DU randomly selects $r_a \in Z_p$ and calculates $uk'_T = g^{\gamma} g^{\beta t r_a}$, $uk'_0 = g^{t r_a}$, $uk'_i = H_2(att_i)^{t r_a}$, $T' = g^{\frac{\gamma}{k}}$ and the trapdoor is $T = (uk'_T, uk'_0, \{uk'_i\}_{i \in S}, T', \{T_j\}_{j \in [1,k]})$.
- $Search(Index, T) \rightarrow CT'$: DU submits trapdoors to CS. CS calculates $q = \prod_{j=1}^{k} (C_0 \prod_{i=0}^{\mu} E_i^{{T_j}^{\mu-i}}) = \prod_{j=1}^{k} g^{(H_1(KW_{Uj}) - H_1(KW_1)) \ldots ((H_1(KW_{Uj}) - H_1(KW_{\mu})) + s}$, only the value of DU's query keyword set $KW_U \subseteq KW$, $q = g^{ks}$. CS first performs the keyword search step and calculates $Q_1 = e(q, T')$ for the DU's trapdoor. If $Q_1 = C'$, verify whether DU's attributes satisfy the access policy. Calculate $Q_2 = e(C_0, uk'_T)$, $Q_3 = \prod_{i \in S}(e(C_i, uk'_0) \cdot e(D_i, uk'_i))^{w_i}$. CS verifies whether $\frac{Q_2}{Q_3}$ is equal to C'. If the equation holds, it indicates that the search is successful. If the equation does not hold,

Algorithm 5 Algorithm MKSABE-VaAR.Trapdoor

```
Trapdoor(PK, KW_U, SK)
DU:
1: $KW_U = \{KW_{U_1}, KW_{U_2}, \cdots, KW_{U_k}\}$
2: for $i$ in $[1, k]$ do
3:     $T_i = H_1(KW_{U_i})$
4: end for
5: $r_a \leftarrow \mathbb{Z}_p$
6: $uk'_T = g^{\gamma} g^{\beta t r_a}$
7: $uk'_0 = g^{t r_a}$
8: $uk'_i = H_2(att_i)^{t r_a}$
9: $T' = g^{\frac{\gamma}{k}}$
10: $T = (uk'_T, uk'_0, \{uk'_i\}_{i \in S}, T', \{T_j\}_{j \in [1,k]})$.
11: Return $T$
```

the search fails. DU's attribute set does not satisfy the access policy, or DU's query keyword set is not fully contained in the file's keyword set, which will cause the algorithm to terminate and return $\perp$. When DU successfully searches, CS will return $CT' = (C, C_K, C_0, \{C_i, D_i\}_{i \in [1,l]})$ to DU.

Algorithm 6 Algorithm MKSABE-VaAR.Search

```
Search(Index, T)
DU:
1: Send $T$ to CS
CS:
1: $q = \prod_{j=1}^{k} (C_0 \prod_{i=0}^{\mu} E_i^{T_j^{\mu-i}}) = \prod_{j=1}^{k} g^{(H_1(KW_{Uj})-H_1(KW_1))...((H_1(KW_{Uj})-H_1(KW_\mu))+s}$
2: $Q_1 = e(q, T')$
3: if $Q_1 = C'$ then
4:     $Q_2 = e(C_0, uk'_T)$
5:     $Q_3 = \prod_{i \in S}(e(C_i, uk'_0) \cdot e(D_i, uk'_i))^{w_i}$
6: else
7:     Return $\perp$
8: end if
9: if $\frac{Q_2}{Q_3}$ is equal to $C'$ then
10:     $CT' = (C, C_K, C_0, \{C_i, D_i\}_{i \in [1,l]})$
11:     Return $CT'$
12:     Send $CT'$ to DU
13: else
14:     Return $\perp$
15: end if
```

- $Decrypt(CT', SK) \to F_i$: DU decrypts CT' by SK to ensure that only DU can calculate it and needs to use the secret value s in the access structure to prevent attackers from skipping the access policy. DU calculates the following to obtain the file address:

$$
\begin{aligned}
add_i &= \frac{C \cdot \prod_{i=1}^{w}(e(C_i, uk_0)e(D_i, uk_i))^{\omega_i}}{e(C_0, uk)} \\
&= \frac{C \cdot \prod_{i=1}^{w}(e(g^{\beta\lambda_i}H_2(att_i)^{-r_i}, g^t) \cdot e(g^{r_i}, H_2(att_i)^t))^{\omega_i}}{e(g,g)^{\alpha s} \cdot e(g,g)^{\beta ts}} \\
&= \frac{add_i \cdot e(g,g)^{\alpha s} \cdot e(g,g)^{\beta t \sum_i^{\omega} \lambda_i \omega_i}}{e(g,g)^{\alpha s} \cdot e(g,g)^{\beta ts}} \\
&= \frac{add_i \cdot e(g,g)^{\alpha s} \cdot e(g,g)^{\beta ts}}{e(g,g)^{\alpha s} \cdot e(g,g)^{\beta ts}}
\end{aligned}
$$

The symmetric key K_i can also be decrypted by the above calculation. When DU obtains add_i and K_i, DU sends add_i to IPFS. IPFS queries the list for CT_i corresponding to add_i. IPFS then sends CT_i to DU. DU decrypts $F_i = Dec_{K_i}(CT_i)$ by K_i to obtain the desired file.

Algorithm 7 Algorithm MKSABE-VaAR.Decrypt

```
Decrypt(CT', SK)
DU:
1: $add_i = \frac{C \cdot \prod_{i=1}^{w}(e(C_i, uk_0)e(D_i, uk_i))^{\omega_i}}{e(C_0, uk)}$
2: Send $add_i$ to CS
IPFS:
1: Query the $CT_i$ corresponding to $add_i$ from the list
2: Send $CT_i$ to DU
DU:
1: $K_i = \frac{C_K \cdot \prod_{i=1}^{w}(e(C_i, uk_0)e(D_i, uk_i))^{\omega_i}}{e(C_0, uk)}$
2: $F_i = Dec_{K_i}(CT_i)$
3: Return $F_i$
```

B. ATTRIBUTE REVOCATION

From the linear secret sharing scheme in Section III, we can see that DO maps the attributes to each row of the matrix M. Therefore, if DO wants to revoke some attributes, he only needs to recreate the linear secret sharing matrix and recalculate the C_i. The specific process is as follows:

Step1: DO creates a new access policy, and calculates a new LSSS matrix M' based on this access policy.

Step2: DO randomly selects the column vector $\boldsymbol{v'} = (s, y'_2, \cdots, y'_n)^{\top} \in \mathbb{Z}_p$, where the value of s is constant, and calculates $\lambda'_i = \boldsymbol{M'_i} \cdot \boldsymbol{v'}$, $i \in [1, l]$, where λ'_i is the value representing the attribute in the new access policy. Then DO embeds the λ'_i into the hash value of each attribute to implement the updated attribute binding. DO calculates $C'_i = g^{\beta\lambda'_i}H_2(att_i)^{-r_i}$ and uploads C'_i to CS.

Step3: CS replaces C_i with C'_i.

VI. SCHEME CORRECTNESS

In this section, the proposed MKSABE-VaAR's correctness is discussed from the following two perspectives: If the query keyword set given by DU is all contained in the keywords set created by DO, CS's retrieval activity will continue; otherwise, it will cease. The correct retrieval result will only be returned when DU's attribute set complies with the access policy; otherwise, $\perp$ will be returned.

(1) Keyword search: According to q and Q_1, when DU's keyword set $KW_U \subseteq KW$, $q = g^{ks}$ and $Q_1 = e(g,g)^{\gamma s}$. The value of Q_1 is calculated as follows:

$$Q_1 = e(q, T') = e(g^{ks}, g^{\frac{\gamma}{k}}) = e(g,g)^{\gamma s}$$

CS verifies $Q_1 = C' = e(g,g)^{\gamma s}$ and continues to confirm whether DU's attributes satisfy the access policy.

(2) Results verification: According to the linear reconstruction property of the linear secret sharing scheme, if the DU's attribute set satisfies the access policy of *Index*, the secret value s can be reconstructed by the effective share λ_i, that is, $\sum_{i\in I}\omega_i\lambda_i = s$ holds, otherwise the secret value s cannot be recovered by the share. The value of Q_2, Q_3 is calculated as follows:

$$\begin{aligned} Q_2 &= e(C_0, uk'_T) = e(g^s, g^\gamma \cdot g^{\beta tr_a}) = e(g,g)^{\beta tr_a s}\cdot e(g,g)^{\gamma s} \\ Q_3 &= \prod_{i\in I}(e(C_i, uk'_0)\cdot e(D_i, uk'_i))^{\omega_i} \\ &= \prod_{i\in I}(e(g^{\beta\lambda_i}H_2(att_i)^{-r_i}, g^{tr_a})\cdot e(g^{r_i}, H_2(att_i)^{tr_a}))^{\omega_i} \\ &= \prod_{i\in I}(e(g^{\beta\lambda_i}, g^{tr_a})\cdot e(H_2(att_i)^{-r_i}, g^{tr_a}) \\ &\quad \cdot e(g^{r_i}, H_2(att_i)^{tr_a}))^{\omega_i} \\ &= e(g,g)^{\beta tr_a\sum_{i\in I}\omega_i\lambda_i} = e(g,g)^{\beta tr_a s} \end{aligned}$$

From the above results, we can see that

$$\frac{Q_2}{Q_3} = \frac{e(g,g)^{\beta tr_a s}\cdot e(g,g)^{\gamma s}}{e(g,g)^{\beta tr_a s}} = e(g,g)^{\gamma s}.$$

CS verifies whether $\frac{Q_2}{Q_3} = C' = e(g,g)^{\gamma s}$. If the equation holds, it means that the keywords to be queried by the DU are consistent with the keywords of the required files. And the DU's attributes also satisfy the access policy. At this point, the CS can send CT' to DU.

In summary, our scheme MKSABE-VaAR is correct.

VII. SECURITY ANALYSIS AND PROOF

Theorem 1: If solving the CBDH and CDH problem in polynomial time is difficult, then the scheme is safe in the IND-CP-CKA security game.

Proof: Assuming that the proposed scheme is insecure in the IND-CP-CKA security game, then there is a polynomial time attacker $\mathcal{A}$ can win the security game with a non-negligible advantage, and the challenger $\mathcal{C}$ can break the CBDH and CDH assumption by $\mathcal{A}$.

• *Initialization*: $\mathcal{A}$ submits a security parameter κ and the access structure (M^*, ρ^*) to $\mathcal{C}$. $\mathcal{C}$ runs the system to create an algorithm that sends the generated system common parameter PK to the $\mathcal{A}$.

• *Phase 1*: $\mathcal{A}$ issues the adaptively generated queries with the input of keyword KW_U and set of attribute S to retrieve the corresponding trapdoor T. The $\mathcal{C}$ responds with T generated with the input KW_U and S.

Challenge: The $\mathcal{A}$ gives two pairs of input (KW^0, S^0) and (KW^1, S^1) to $\mathcal{C}$. Both the pairs of input offered by $\mathcal{A}$ must have to fulfill the following criteria.

1) KW^0 and KW^1 are set of keywords with equal length.
2) $\mathcal{A}$ has not gained a trapdoor T which can satisfy any of the challenge ciphertext.

If either of the above mentioned criteria fails, then $\mathcal{C}$ aborts; else, the $\mathcal{C}$ randomly selects $\theta \in \{0, 1\}$ and calculates the encrypted index of keywords $Index^\theta$. $\mathcal{C}$ submits $Index^\theta$ to $\mathcal{A}$.

• *Phase 2*: As performed in phase 2, the $\mathcal{A}$ submits the adaptively generated queries in form of keyword KW_U and a set of attribute S to gain the trapdoor T.The input submitted by $\mathcal{A}$ must have to follow either of the following criteria.

1) Neither KW^0 nor KW^1 should contain KW_U.
2) S does not match access policy.

In response, the $\mathcal{C}$ submits to the $\mathcal{A}$ with trapdoor T corresponding to (KW_U, S).

Guess: The $\mathcal{A}$ gives a guess θ' for θ. If $\theta' = \theta$ the $\mathcal{A}$ wins the game. The advantage of $\mathcal{A}$ for winning this game is $\mathrm{Adv}_{\mathcal{A}} = |\Pr[\theta' = \theta] - \frac{1}{2}|$.

To uncover the challenge ciphertext, an adversary has to retrieve the value of E_i for any keyword KW_U, which is included either only in KW^0 or only in KW^1. Then, the adversary calculates the value of Q_1 to determine whether $\mathcal{C}$ encrypts KW^0 or KW^1. We can know that when $Q_1 = C'$, it means that $KW_U \subseteq KW$. The $\mathcal{A}$ possesses the challenge ciphertext component T'. So we can think of q and T' as the bilinear Diffie-Hellman components as follows:

$$A = g^a = C_0, B = g^b = E_i^{T_j}, C = g^c = T'$$

The challenge for $\mathcal{A}$ is to calculate $e(g,g)^{abc}$. The advantage of computing Q_1 is equivalent to computing $e(g,g)^{abc}$, which is negligible in security parameter κ, denoted as ξ_{CBDH}.

Besides, if $\mathcal{A}$ is to calculate the value of Q_1, he needs to know the value of q. The $\mathcal{A}$ possesses the challenge ciphertext component C_0, E_i and trapdoor component T_i which resemble the computational Diffie-Hellman components as follows:

$$A = g^a = C_0, B = g^b = E_i^{T_j}$$

The challenge for $\mathcal{A}$ resembles calculating g^{ab}. The advantage of computing q is equivalent to computing g^{ab}, which is negligible in security parameter κ, denoted as ξ_{CDH}. As a result, the adversary's advantage for breaking the challenge ciphertext is $\xi_{CBDH} + \xi_{CDH}$, which is negligible in security parameter κ. In summary, the proposed MKSABE-VaAR scheme is safe in the IND-CP-CKA security game.

VIII. PERFORMANCE ANALYSIS

A. FUNCTION COMPARISON

In the subsection, we compare our scheme with several schemes proposed in recent years within the attribute-based encryption scheme [23], [24], [25], [28], [29], [30], [31]. The comparison results are presented in Table 2, revealing the advantageous functional characteristics of our scheme. Table 2 illustrates that our scheme offers comprehensive functionality, enabling fine-grained access control and multi-keyword search. Additionally, the scheme incorporates result verification, enhancing search efficiency and attribute revocation, further augmenting its functionality.

TABLE 2. Function comparison.

Schemes	F1	F2	F3	F4
Ref [23]	✓	✕	✕	✕
Ref [24]	✓	✕	✕	✕
Ref [25]	✓	✕	✕	✕
Ref [28]	✓	✓	✕	✕
Ref [29]	✓	✓	✕	✕
Ref [30]	✓	✓	✓	✕
Ref [31]	✓	✓	✕	✓
Ours	✓	✓	✓	✓

F1: Attribute-based keyword search.
F2: Multikeyword search.
F3: Search result verification.
F4: Attribute revocation.

B. THEORETICAL ANALYSIS AND COMPARISON

In Table 3 and Table 4, $|U|$ represents the number of rows in the linear access matrix, $|S|$ represents the number of attributes of DU, $|\mu|$ represents the number of all keywords in the system, $|k|$ represents the number of keywords of DU.

1) THEORETICAL COMPUTATION COST COMPARISON

The exponential operation and the pairing operation are more time-consuming compared to the multiplication operation and the hash operation. Therefore, the subsequent analysis and comparison focus on the pairing operation (T_p) and the exponential operation (T_e). As shown in Table 3, the computational overhead of all schemes is observed to have a linear correlation with the number of system attributes or keywords. However, when the number of all keywords ($|\mu|$) surpasses the number of system attributes ($|U|$), our scheme exhibits higher computational efficiency, particularly in the *IndexGen* algorithm. Besides, since the parts we need in the verification process have been calculated in the search process, our verification process also has a low amount of calculation.

2) THEORETICAL STORAGE COST COMPARISON

In Table 4, $|\mathbb{G}|$, $|\mathbb{G}_T|$, $|\mathbb{Z}_p|$ represents the length of elements in $\mathbb{G}$, $\mathbb{G}_T$ and $\mathbb{Z}_p$. Table 4 shows that these schemes exhibit the smallest storage overhead when the number of all keywords is less than the number of system attributes. Our scheme utilizes polynomial storage for keywords, resulting in a fixed value for keyword storage consumption regardless of the number of keywords. This approach effectively reduces the storage burden on the server.

C. EXPERIMENTAL ANALYSIS

This experiment uses the PBC bilinear pairing package and Python language to program under Linux Ubuntu-22.04.1 operating system running in 16GB memory, 6-core processor. The bilinear pairing package used in the experiment is based on the 512 bit elliptic curve group, and the elliptic curve type is Type A. The size of elements in G and GT is 128 byte. Three schemes are compared experimentally. The experimental results are shown in Fig. 3 and Fig. 4.

Fig. 3a shows the performance of the *KeyGen* algorithm with the number of keywords remaining the same and the number of attributes gradually increasing. It can be seen that under the above conditions, the performance of our scheme in *KeyGen* is close to the performance of scheme [29] and is much better than the performance of scheme [28]. Besides, when the number of attributes exceeds 35, our scheme performs better than scheme [30]. And Fig. 3a also conforms to the *KeyGen* algorithm in Table 3, which means that the *KeyGen* algorithm computational consumption of scheme [28], [29], [30] and our scheme increases with the increase of the number of attributes.

Fig. 3b shows the performance of the *IndexGen* algorithm with the number of keywords remaining the same and the number of attributes gradually increasing. It can be seen from Fig. 3b that the performance of scheme [28]is not ideal, but the performance of scheme [29], scheme [30] and our scheme is better. But as the number of attributes increases, the scheme [28] and our scheme curve rise more moderately, which means that our scenario performs better.

Fig. 3c shows the performance of the *Search* algorithm with the number of keywords remaining the same and the number of attributes gradually increasing. It can be seen from Fig. 3c that the performances of scheme [28] and scheme [29] are stable, and the efficiency of the *Search* algorithm has little relationship with the number of attributes. However, scheme [30] and our scheme do not perform very well on the search algorithm, and its efficiency decreases linearly with the increase of the number of attributes. But by observing Table 3, it is not difficult to see this because the *Search* algorithm of our scheme and scheme [30] is related to the number of attributes.

Fig. 4a shows the performance of the *IndexGen* algorithm with the number of attributes remaining the same and the number of keywords gradually increasing. It can be seen from Fig. 4a that under the above condition, the efficiency of scheme [28], scheme [29], and scheme [30] decreases linearly with the increase of the number of keywords. Not only does our efficiency perform well from the start, but it doesn't increase as the number of keywords increases. We can also trace the reason from Table 3. Our scheme is independent of the number of keywords in the *IndexGen* algorithm.

Fig. 4b shows the performance of the *Search* algorithm with the number of attributes remaining the same and the number of keywords gradually increasing. Fig. 4b shows that while the performance of scheme [29], scheme [30], and our scheme will decline as the number of keywords increases the efficiency of scheme [29] drops sharply as the number of keywords increases. Besides, our scheme also performs closer to the scheme [29] and scheme [30]. As can be seen from Table 3, scheme [28], scheme [30] and our scheme are not affected by the number of keywords on this algorithm.

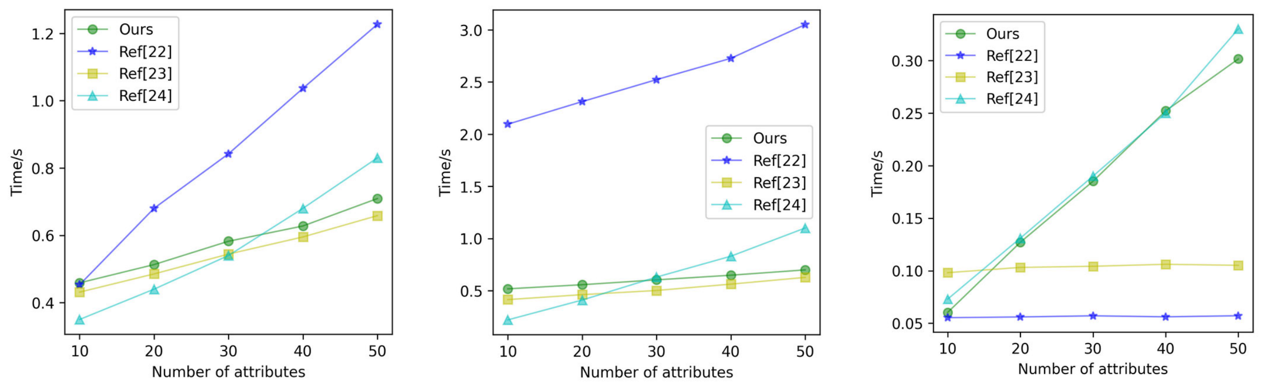
In summary, the number of system attributes generally does not change at initialization, and the number of keywords

TABLE 3. Theoretical computation cost comparison.

Algorithms	Ref [28]	Ref [29]	Ref [30]	Ours
Setup	$T_p + 2T_e$	$2T_e$	$T_p + 3T_e$	$2T_p + 3T_e$
KeyGen	$(\|S\| + 3)T_e$	$(\|S\| + 2)T_e$	$(2\|U\| + 5)T_e$	$(\|S\| + 2)T_e$
IndexGen	$\|\mu\|T_p + (\|U\| + \|\mu\| + 3)T_e$	$\|\mu\|T_p + (\|U\| + 4)T_e$	$(2\|U\| + \|k\| + 6)T_e$	$(3\|U\| + 3)T_e$
TrapDoor	T_e	$(\|k\| + 3)T_e$	$(2\|U\| + 1)T_e$	$(\|S\| + 3)T_e$
Search	$3T_p$	$(\|k\| + 3)T_p$	$(2\|U\| + 1)T_p + T_e$	$(2\|S\| + 2)T_p$
Verify	$-$	$-$	$2T_p + 3T_e$	$2T_p$

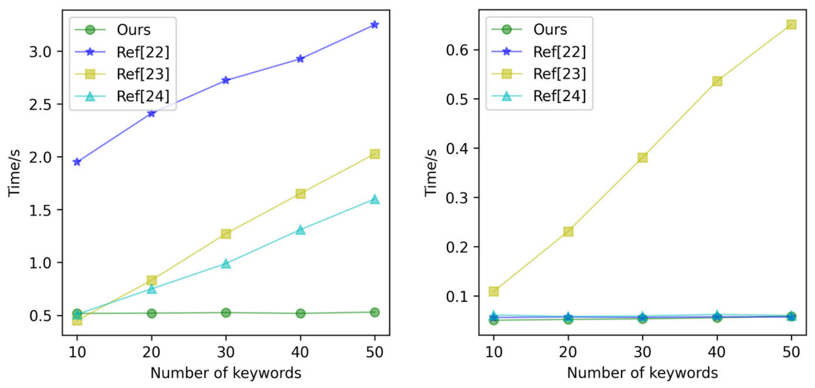
TABLE 4. Theoretical storage cost comparison.

Algorithms	Ref [28]	Ref [29]	Ref [30]	Ours
KeyGen	$(\|S\| + 3)\|\mathbb{G}\|$	$(\|S\| + 1)\|\mathbb{G}\|$	$(2\|U\| + 3)\|\mathbb{G}\| + \|\mathbb{G}_T\| + (\|U\| + 2)\|\mathbb{Z}_p\|$	$(\|S\| + 2)\|\mathbb{G}\|$
IndexGen	$(2\|U\|+2)\|\mathbb{G}\|+\|\mu\|+2)\|\mathbb{G}_T\|$	$(\|U\|+2)\|\mathbb{G}\|+\|\mu\|\|\mathbb{G}_T\|$	$(2\|U\| + \|k\| + 2)\|\mathbb{G}\| + 3\|\mathbb{G}_T\| + (\|U\| + 2)\|\mathbb{Z}_p\|$	$(2\|U\|+1)\|\mathbb{G}\|+3\|\mathbb{G}_T\|$
TrapDoor	$(\|S\| + \|k\|)\|\mathbb{G}\|$	$(\|S\| + \|k\| + 2)\|\mathbb{G}\|$	$(2\|U\| + 1)\|\mathbb{G}\| + 2\|\mathbb{Z}_p\|$	$(\|S\| + 2)\|\mathbb{G}\| + \|k\|\|\mathbb{Z}_p\|$



(a) KeyGen (b) IndexGen (c) Search

FIGURE 3. Performance evaluation for 10 keywords.



(a) IndexGen (b) Search

FIGURE 4. Performance evaluation for 10 attributes.

increases with the increase of stored files, in which case our scheme performs well.

IX. CONCLUSION

This paper presents a multi-keyword searchable attribute-based encryption scheme. Different from previous schemes, our scheme MKABSE-VaAR utilizes keyword polynomials to achieve a multi-keyword search function. In other schemes, n keywords need to perform n bilinear pairing operations. MKABSE-VaAR reduces this step to a single bilinear pairing operation. In addition, other schemes either do not have the function of search result verification or search result verification requires the assistance of third-party entities, which increases the risk of data leakage. In MKABSE-VaAR, we add a value to the index to assist in the verification of search results, so that the process is executed on the cloud server. As for the function of attribute revocation, we use the linear reconstruction function of the linear secret sharing scheme to re-formulate the access policy, calculate part of the index, and then upload it, which greatly reduces the computational and communication overhead in the attribute revocation process. In addition, comparisons with existing schemes are made based on computational overhead and storage overhead. Theoretical analysis and experimental results demonstrate that the scheme offers security, efficiency, and flexibility. Of course, our scheme also has shortcomings such as data sharing between data owners, data traceability, and attribute hiding. Therefore, these will also be our main research directions in the future.

REFERENCES

[1] H. Shen, M. Zhang, H. Wang, F. Guo, and W. Susilo, "Efficient and privacy-preserving massive data processing for smart grids," *IEEE Access*, vol. 9, pp. 70616–70627, 2021.

[2] H. Shen, J. Li, G. Wu, and M. Zhang, "Data release for machine learning via correlated differential privacy," *Inf. Process. Manage.*, vol. 60, no. 3, May 2023, Art. no. 103349, doi: 10.1016/j.ipm.2023.103349.

[3] M. Zhang, S. Chen, J. Shen, and W. Susilo, "PrivacyEAFL: Privacy-enhanced aggregation for federated learning in mobile crowdsensing," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 5804–5816, 2023, doi: 10.1109/TIFS.2023.3315526.

[4] M. Zhang, Y. Chen, and W. Susilo, "Decision tree evaluation on sensitive datasets for secure E-healthcare systems," *IEEE Trans. Depend. Secure Comput.*, vol. 20, no. 5, pp. 3988–4001, Nov. 2023.

[5] H. Shen, G. Wu, Z. Xia, W. Susilo, and M. Zhang, "A privacy-preserving and verifiable statistical analysis scheme for an E-commerce platform," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 2637–2652, 2023, doi: 10.1109/TIFS.2023.3269669.

[6] D. Xiaoding Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proc. IEEE Symp. Secur. Privacy. (S&P)*, Berkeley, CA, USA, May 2000, pp. 44–55, doi: 10.1109/SECPRI.2000.848445.

[7] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, vol. 3027. Interlaken, Switzerland, Berlin, Germany: Springer, May 2004, pp. 506–522, doi: 10.1007/978-3-540-24676-3_30.

[8] Q. Huang and H. Li, "An efficient public-key searchable encryption scheme secure against inside keyword guessing attacks," *Inf. Sci.*, vols. 403–404, pp. 1–14, 2017, doi: 10.1016/j.ins.2017.03.038.

[9] Y. Fan and Z. Liu, "Verifiable attribute-based multi-keyword search over encrypted cloud data in multi-owner setting," in *Proc. IEEE 2nd Int. Conf. Data Sci. Cyberspace (DSC)*, Shenzhen, China, Jun. 2017, pp. 441–449, doi: 10.1109/DSC.2017.36.

[10] Z. Liu, C. Xu, and Z. Yao, "Forward secure searchable encryption with conjunctive-keyword supporting multi-user," in *Proc. 3rd EAI Int. Conf. (SPNCE)*, vol. 344. Lyngby, Denmark, Cham, Switzerland: Springer, Aug. 2020, pp. 423–440, doi: 10.1007/978-3-030-66922-5_29.

[11] X. Ge, J. Yu, C. Hu, H. Zhang, and R. Hao, "Enabling efficient verifiable fuzzy keyword search over encrypted data in cloud computing," *IEEE Access*, vol. 6, pp. 45725–45739, 2018.

[12] W. Lin, H. Cui, B. Li, and C. Wang, "Privacy-preserving similarity search with efficient updates in distributed key-value stores," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 5, pp. 1072–1084, May 2021.

[13] M. A. Ahmed, S. Ramachandram, and K. U. R. Khan, "Conjunctive keyword forward secure ranked dynamic searchable encryption over outsourced encrypted data," in *Proc. Int. Symp. Secur. Comput. Commun.*, vol. 1364. Chennai, India, Singapore: Springer, Oct. 2020, pp. 197–212, doi: 10.1007/978-981-16-0422-5_14.

[14] Y. Zhang, Y. Li, and Y. Wang, "Efficient searchable symmetric encryption supporting dynamic multikeyword ranked search," *Secur. Commun. Netw.*, vol. 2020, pp. 1–16, Jul. 2020, doi: 10.1155/2020/7298518.

[15] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in Cryptology*, vol. 196, Santa Barbara, CA, USA, Berlin, Germany: Springer, Aug. 1985, pp. 47–53, Heidelberg, doi: 10.1007/3-540-39568-7_5.

[16] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proc. Advances in Cryptology—EUROCRYPT*, vol. 3494. Heidelberg, Germany: Springer-Verlag, 2005, pp. 457–473, doi: 10.1007/11426639_27.

[17] 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.*, Oct. 2006, pp. 89–98, doi: 10.1145/1180405.1180418.

[18] U. C. Yadav and S. T. Ali, "Ciphertext policy-hiding attribute-based encryption," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Berkeley, CA, USA, Aug. 2015, pp. 2067–2071, doi: 10.1109/ICACCI.2015.7275921.

[19] A. Michalas, "The lord of the shares: Combining attribute-based encryption and searchable encryption for flexible data sharing," in *Proc. 34th ACM/SIGAPP Symp. Appl. Comput.*, Limassol, Cyprus, Apr. 2019, pp. 146–155, doi: 10.1145/3297280.3297297.

[20] H. Yin, J. Zhang, Y. Xiong, L. Ou, F. Li, S. Liao, and K. Li, "CP-ABSE: A ciphertext-policy attribute-based searchable encryption scheme," *IEEE Access*, vol. 7, pp. 5682–5694, 2019.

[21] R. Li, D. Zheng, Y. Zhang, H. Su, M. Yang, and P. Lang, "Attribute-based encryption with multi-keyword search," in *Proc. IEEE 2nd Int. Conf. Data Sci. Cyberspace (DSC)*, Shenzhen, China, Jun. 2017, pp. 172–177, doi: 10.1109/DSC.2017.97.

[22] Y. Zhang, R. H. Deng, J. Shu, K. Yang, and D. Zheng, "TKSE: Trustworthy keyword search over encrypted data with two-side verifiability via blockchain," *IEEE Access*, vol. 6, pp. 31077–31087, 2018.

[23] S. Wang, D. Zhang, Y. Zhang, and L. Liu, "Efficiently revocable and searchable attribute-based encryption scheme for mobile cloud storage," *IEEE Access*, vol. 6, pp. 30444–30457, 2018.

[24] H. Zhu, L. Wang, H. Ahmad, and X. Niu, "Key-policy attribute-based encryption with equality test in cloud computing," *IEEE Access*, vol. 5, pp. 20428–20439, 2017.

[25] Q. Zheng, S. Xu, and G. Ateniese, "VABKS: Verifiable attribute-based keyword search over outsourced encrypted data," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Toronto, ON, Canada, Apr. 2014, pp. 522–530, doi: 10.1109/INFOCOM.2014.6847976.

[26] X. Liu, T. Lu, X. He, X. Yang, and S. Niu, "Verifiable attribute-based keyword search over encrypted cloud data supporting data deduplication," *IEEE Access*, vol. 8, pp. 52062–52074, 2020.

[27] X. Song, Z. Zhou, W. Duan, Z. Liu, and C. Xu, "Attribute-based searchable encryption scheme with fuzzy keywords," in *Proc. Int. Conf. Secur., Privacy Anonymity Comput., Commun. Storage (SpaCCS)*, vol. 12382. Nanjing, China, Berlin, Germany: Springer, Dec. 2020, pp. 44–61, doi: 10.1007/978-3-030-68851-6_3.

[28] Y. Zhou, J. Nan, and L. Wang, "Fine-grained attribute-based multikeyword search for shared multiowner in Internet of Things," *Secur. Commun. Netw.*, vol. 2021, pp. 1–14, May 2021, doi: 10.1155/2021/6649119.

[29] P. Chaudhari and M. L. Das, "KeySea: Keyword-based search with receiver anonymity in attribute-based searchable encryption," *IEEE Trans. Services Comput.*, vol. 15, no. 2, pp. 1036–1044, Mar. 2022.

[30] Y. Zhang, T. Zhu, R. Guo, S. Xu, H. Cui, and J. Cao, "Multi-keyword searchable and verifiable attribute-based encryption over cloud data," *IEEE Trans. Cloud Comput.*, vol. 11, no. 1, pp. 971–983, Jan. 2023.

[31] Y. Miao, R. H. Deng, X. Liu, K. R. Choo, H. Wu, and H. Li, "Multi-authority attribute-based keyword search over encrypted cloud data," *IEEE Trans. Depend. Secure Comput.*, vol. 18, no. 4, pp. 1667–1680, Jul. 2021.

[32] D. Sangeetha, S. S. Chakkaravarthy, S. C. Satapathy, V. Vaidehi, and M. V. Cruz, "Multi keyword searchable attribute based encryption for efficient retrieval of health records in cloud," *Multimedia Tools Appl.*, vol. 81, no. 16, pp. 22065–22085, Jul. 2022.

[33] U. S. Varri, S. K. Pasupuleti, and K. V. Kadambari, "CP-ABSEL: Ciphertext-policy attribute-based searchable encryption from lattice in cloud storage," *Peer Peer Netw. Appl.*, vol. 14, no. 3, pp. 1290–1302, May 2021.

[34] K. Zhang, Z. Jiang, J. Ning, and X. Huang, "Subversion-resistant and consistent attribute-based keyword search for secure cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 17, pp. 1771–1784, 2022.

[35] Y. Yang, R. H. Deng, W. Guo, H. Cheng, X. Luo, X. Zheng, and C. Rong, "Dual traceable distributed attribute-based searchable encryption and ownership transfer," *IEEE Trans. Cloud Comput.*, vol. 11, no. 1, pp. 247–262, Jan. 2023.

HUA SHEN (Member, IEEE) received the Ph.D. degree in computer science from the School of Computer Science, Wuhan University, China, in 2014. From 2019 to 2020, she was a Visiting Research Fellow with the University of Wollongong, Australia, supervised by Prof. Willy Susilo. She is currently a Professor with the School of Computer Science, Hubei University of Technology, China. Her research interests include privacy computing and information security.

JIAN ZHOU received the B.S. degree from the Hubei University of Technology, Wuhan, China, in 2020, where he is currently pursuing the M.S. degree with the School of Computer Science. His research interests include searchable encryption and privacy computing.

GE WU received the M.S. degree from Nanjing Normal University, Nanjing, China, in 2015, and the Ph.D. degree from the University of Wollongong, Australia, in 2019. He is currently a Lecturer with the School of Cyber Science and Engineering, Southeast University, Nanjing. His research interests include cryptography and information security, in particular the design and security proof of public-key cryptographic schemes.

MINGWU ZHANG received the Ph.D. degree in computer science from the College of Mathematics and Informatics, South China Agricultural University, Guangzhou, China, in 2009. From 2010 to 2012, he was a JSPS Postdoctoral Fellow of Japan Society of Promotion Sciences with the Institute of Mathematics for Industry, Kyushu University, Japan. From 2015 to 2016, he was a Senior Visiting Scholar with the School of Computing and Information Technology, University of Wollongong, Australia. He is currently a Professor with the School of Computer Science, Hubei University of Technology, China. He is also the Director of the Hubei Engineering Research Centre for Industrial Big Data. He has served as a program committee member for several international conferences and published more than 100 papers in international conferences and journals, such as ASIACRYPT, ACISP, ProvSec, ISPEC, Inscrypt, the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, *Theoretical Computer Science*, and IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING. His research interests include cryptography technology for networks and data security, secure computation, and privacy preservation in big data and clouds. He was a recipient of five best paper awards at international conferences, such as ACISP'18 and Inscrypt'18.