Received August 3, 2020, accepted September 2, 2020, date of publication September 16, 2020,
date of current version September 25, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3023893

Attribute-Based Encryption Approach for Storage, Sharing and Retrieval of Encrypted Data in the Cloud

MIGUEL MORALES-SANDOVAL[1], MELISSA HINOJOSA CABELLO[1], HEIDY MARISOL MARIN-CASTRO[2], AND JOSE LUIS GONZALEZ COMPEAN[1]
[1]Center for Research and Advanced Studies of the National Polytechnic Institute (CINVESTAV) Tamaulipas, Ciudad Victoria 87130, Mexico
[2]Cátedras CONACYT, Universidad Autónoma de Tamaulipas, Ciudad Victoria 87000, Mexico

Corresponding author: Heidy Marisol Marin-Castro (heidy.marinc@gmail.com)

This work was supported by the Fondo Sectorial de Investigación Para la Educación, CB SEP-CONACyT, Mexico, under Project 281565.

ABSTRACT One of the most cost-effective services in cloud computing is storage, used by businesses and individuals to outsource their massive data to untrusted servers. Efforts have studied problems around this application scenario in different fronts: efficiency, flexibility, reliability, and security. In this paper we address the security concerns of cloud storage under the scenario where users encrypt-then-outsource data, share their outsourced data with other users, and the service provider can be queried for searching and retrieval of encrypted data. As main distinctive, we propose a security approach for storage, sharing and retrieval of encrypted data in the cloud fully constructed on the basis of attribute-based encryption (ABE) thus enabling access control mechanisms over both the encrypted data and also for the information retrieval task through search access control. Compared to related works, our approach considers efficient encryption at three different levels: *i*) bulk encryption of data outsourced to the cloud, *ii*) keys management for access control over encrypted data by means of digital envelopes from attribute based encryption, and *iii*) novel construction for attribute based searchable encryption (ABSE). Our underlying ABE algorithms are carefully selected from the body of knowledge and novel constructions for ABSE are provided over the asymmetric setting (Type-III pairings) to support security levels of 128-bits or greater. Experimental results on benchmark data sets demonstrate the viability of our approach for practical realizations using Barreto-Naehrig curves.

INDEX TERMS Attribute based encryption, asymmetric pairings, cloud storage, information retrieval, security, searchable encryption.

I. INTRODUCTION

The high availability (access anytime, anywhere) and reliability of data at low cost are the main incentives for organizations and individuals to adopt cloud storage services. These services are in increasing demand due to the high amount of data generated by different sources (Internet of Things) and cloud enabled applications. However, data owners (DOs) outsourcing their data to untrusted servers in the cloud face the security concern that the cloud service provider (CSP) honestly stores the DO's data and follows the agreed protocol but tries to learn as much as possible from the computations and interactions with the users (DU) that access the DO's data. This issue can be solved by providing DOs with a confidentiality security service for DOs to encrypt data before uploading it to the cloud.

A straightforward encryption approach to prevent DO's data disclosure and to keep DO's data private from CSP or from any other entity, causes the provider cannot manipulate data, that is, loss of utility appears as the encrypted data cannot be used by the CSP for retrieval/searching purposes.

Due to that inconvenience, DUs should download large volume of encrypted data, decrypt, and then search over the plaintext data (locally), re-encrypt and upload again its data to the cloud. Of course, so one approach incurs in huge communications and computations overhead and is completely inefficient. Searchable encryption (SE) [1] has been the most known approach to cope with the problem of searching over

The associate editor coordinating the review of this manuscript and approving it for publication was Sun-Yuan Hsieh.

encrypted data stored in untrusted servers. SE is defined as the ability to identify and retrieve a set of objects from an encrypted collection that satisfy a query. In SE, the CSP executes DU's encrypted queries over encrypted data without decryption, so it does not learn anything about the data content, search criteria, nor search patterns.

The most general method to implement SE in any of its existing families [2] is as follows:

1) DO creates cryptographic keys $\{k_1, k_2, k_3\}$ with security strength λ;
2) Given a set of documents D, DO extracts representative keywords W in D;
3) DO encrypts D with a symmetric cipher E using key k_1 ($E_{k_1}(D)$);
4) DO creates a secure index from W, using key k_2 ($I_{k_2}(W)$);
5) DO uploads $E_{k_1}(D)$ and $I_{k_2}(W)$ to the cloud;
6) DO gives authorization to some DUs to search over its data, using k_3.
7) DU generates trapdoors for a given query word w_q in W using k_3, $T_{k_3}(w_q)$;
8) DU queries the CSP using $T_{k_3}(w_q)$, to retrieve D' documents containing w_q.
9) CSP searches over encrypted data using $T_{k_3}(w_q)$ and $I_{k_2}(W)$ (CSP will not learn anything about data, search criteria nor query patterns).
10) The encrypted documents $E_{k_1}(D')$ found by CSP (satisfying the query) are sent back to DU.
11) If DO authorizes DU the access D' in plaintext form, then DO shares k_1 with DU and thus DU decrypts $E_{k_1}(D')$ and get access to the documents set D' in clear.

Generally, $E_{k_1}(D)$ is not detailed in SE schemes nor any approach for key management for k_1, which is crucial for practical SE since E is a symmetric cipher so the key for encryption and decryption must be the same. Keys $\{k_2, k_3\}$ are relevant values in SE because k_2 enables the creation of the secure index and k_3 allows creating the trapdoors (encrypted queries) used for querying. The most studied and popular SE technique is Symmetric Searchable Encryption (SSE) [3], where $k_2 = k_3$ (sometimes $k_1 = k_2$). In public key encryption with Keyword search (PEKS) [4], $\{k_2, k_3\}$ are related keys (based on public key encryption security assumptions), being k_2 public and k_3 private.

A. ATTRIBUTE BASED ENCRYPTION FOR CLOUD DATA SECURITY

Data security and specifically confidentiality in untrusted storage servers has attracted attention from academia and industry and still remains as a challenging problem [5]–[9].

Access control is also a security service required for DO to selectively decide the authorized DUs being able to access its data. Searching capabilities over encrypted data are essential for practical cloud storage. For these three main requirements in cloud storage, attribute based encryption [10] has been pointed out as a potential solution.

Particularly, the digital envelope technique based on ciphertext-policy attribute-based encryption (CP-ABE), known as DET-ABE, has been proposed to provide the confidentiality and access control services in cloud storage applications [11]. Furthermore, attribute-based searchable encryption (ABSE) [12] allows fine-grained search control, which regulates how authorized users acquire trapdoors to query the CSP and enables DUs and DOs to perform secure information retrieval operations. This aspect of security has been referred in the literature as Access Control Aware Search [13]. In ABSE, DOs create a secure index without using a key k_2 but an *access policy*. In this case, k_3 is an ABSE private key created from DU's attributes. Thus, retrieving data of interest is possible only if the attributes used to create the trapdoor satisfy the access policy used during keywords encryption (index creation). There are various ABSE schemes found in the literature [14]–[18] with different properties, suitable for different use cases. The various versions of ABSE are also due to the various forms of ABE cores, for example, some of them considering DU's revocability (DU's search right), small or large attribute universe, constant-ciphertext, policies expressiveness, among others.

B. OUR CONTRIBUTIONS

In this work, we present a security approach for storing, sharing and retrieving of encrypted data in the cloud, fully constructed on the basis of attribute-based encryption (ABE). Our approach is well suited for a known cloud-based storage and sharing model [8], where DO uploads encrypted data to the cloud to ensure confidentiality (by means of symmetric data encryption) and establishes access control mechanisms for data sharing using attribute based encryption; DU can selectively locate specific documents using an index-based structure and retrieve documents of interest in encrypted form, without revealing any information to the CSP and under a fine-grained search control. Our proposed approach aims at meeting the following four requirements to enable practical storage, sharing and retrieval of encrypted data in the cloud:

R1 - DO can execute $E_{k_1}(D)$ efficiently to provide *confidentiality* over outsourced data to the cloud at the same time that enables *fine-grained data access control* and secure distribution of k_1 for DUs, thus enabling secure data sharing.

R2 - DUs can query $I_{k_2}(W)$ (via the CSP) by computing and using $T_{k_3}(w_q)$ at the time that secure *fine-grained search control* is enabled.

R3 - DUs can ask the CSP to return the k-most relevant documents from the retrieval task results, ordered accordingly to their relevance to the query.

R4 - Both R1 and R2 comply with recommended security levels[1] (i.e. $\lambda \geq 128 - bit$).

We called our approach FABECS (Fully Attribute-Based Encryption scheme for Cloud Storage, Sharing and Retrieval) which fulfills requirements R1-R4. FABCS includes a novel

[1]url key lengths

TABLE 1. **Generalities of searchable encryption approaches (non attribute-based) in the literature for cloud-based storage scenarios.**

Ref.	$M(k_1)$	SE	IR_B	S_{type}	*Reqs*			Exp	R4 - λ	
					R1	R2	R3		R1	R2
[5], 2010	✗	SSE	✓	SKS	✗	✗	✗	✓	128	128
[22], 2014	✗	SIPC	✓	MKS	✗	✗	✓	✓	-	-
[6], 2017	✗	SSE	✓	SKS/MKS	✗	✗	✓	✓	-	80
[8] , 2018	✗	SSE	✓	SKS/MKS	✗	✗	✓	✓	-	-
[7], 2011	✓	SSE, PKC	✗	Metadata[4]	✓	✓	✗	✗	-	-
Ours, 2020	✓	ABSE	✓	MKS	✓	✓	✓	✓	≥ 128	≥ 128

Cipher-text policy ABSE (CP-ABSE) construction to achieve R2 and R3 requirements. At the same time, FABECS reuses the settings of ABSE (pairings and curve parameters) for the set up of DET-ABE which provides cryptographically enforced fine-grained access controls needed to meet requirement R1. The novel CP-ABSE construction in FABECS is based on the Type-III pairing. This construction improves the only known previous realization of CP-ABSE [9], which was limited to the symmetric pairing setting (Type-I) and in practice limited to a security level of 80-bits (outdated). Unlike [9], our Type-III CP-ABSE construction uses a more efficient implementation of ABE based on the body of knowledge (see the discussion at the end of Section II) and its construction on the Type-III pairing setting allows realizations for security levels greater than or equal to 128-bits.

The experimental evaluation of our constructions over the benchmark LISA [19] revealed the feasibility of our proposal and confirmed the viability of the novel fully attribute-based searchable approach at the time that demonstrated the advantages of the asymmetric setting for CP-ABSE compared to the proposed solution in [9], in terms of performance and lower memory resources for the main components (keys, ciphertext, access structure).

The rest of this paper is organized as follows: Section II presents a review of related works that frame the contributions of our proposal. Section III presents some preliminaries and requirements. Section IV describes our fully attribute-based approach for cloud-based searchable encryption and secure cloud information retrieval. Section VI describes the details of experiments, results and comparisons. Finally, Section VII concludes this work and points out future research directions.

II. RELATED WORKS

Attribute-based encryption (ABE) has become an enabler technology for secure storage and sharing in the cloud, as described in Section I-A. ABE allows many-to-many encryption, which is not possible to achieve in traditional public key cryptosystems (PKC), i.e., RSA. Thus, potential receivers may not necessarily be known at the time of encryption, a task that is done by using an access policy that enforces all those whose attributes satisfy the policy can decrypt and gain access to plaintext data. Atomically, ABE provides both confidentiality and fine-grained access control over data. Attributes are taken from a universe $\mathcal{U}$. The access policy, expressed as an *access structure* $\mathbb{A}$, restricts the decryption capabilities of the intended destinations depending on the attributes set possessed.

The two prominent approaches for attribute-based encryption that have been proposed in the state-of-the-art are KP-ABE and CP-ABE. In KP-ABE [20], policies are associated to decryption keys and attributes to the ciphertext. Contrary, in CP-ABE [21] the ciphertext is created with a policy and decryption keys are associated to user's attributes. CP-ABE is conceptually closer to Role Based Access Control and more natural to apply than is KP-ABE to enforce access control over encrypted data. Therfore, the cryptosystems based on CP-ABE are considered an attractive option for providing confidentiality service and fine-grained access control mechanisms at the same time [11]. Attribute-based searchable encryption (ABSE) is defined using either KP-ABE or CP-ABE as basis.

Table 1 summarizes the main aspects of previous works proposing searchable encryption approaches for cloud-based storage systems. None of them use ABE for both access control over outsourced encrypted data neither for the information retrieval task (ABSE) as we propose in this work.

It is relevant to note from Table 1 is that most of the reported works do not consider any key management mechanism $M(k_1)$ (as it is specified in the R1 requirement), and such a mechanism is crucial in practice, otherwise data could not be decrypted neither accessed by DUs. For the information retrieval task, the most common technique used has been SSE, either for single keyword search (SKS) or multi-keyword search (MKS). Furthermore, it is worth to mention that some solutions consider evaluation based on known benchmarks (IR_B), which include a ranking mechanism ($R3$ requirement) to evaluate the effectiveness of the retrieval task. It is also worth to mention, from Table 1, that some works have carried out experiments (*Exp*) that have been limited only to the retrieval task (if R3 is checked) or only considering a specific security value in R4 for the encryption (R1) and search (R2) tasks.

Authors in [7] proposed to implement $M(k_1)$ by means of public key encryption (PKE) as a kind of digital envelope. Destinations of encrypted data receive a key from their public keys. So, public keys must be distributed previously and digital certificates are mandatory for practical realizations. In addition, PKE only allows coarse-grain access control so fine-grained and many-to-many encryption is not possible. Experimental evaluation was not reported.

TABLE 2. Attribute-based approaches for searchable encryption in cloud-based storage systems, focusing on either access control over encrypted data (ABE) or on searchable encryption (ABSE).

Ref.	Reqs			e	ABE $type$	$\mathbb{A}$	U	M	S_{type}	IR_B	Exp	$R4-\lambda$	
	$R1$	$R2$	$R3$									$R1$	$R2$
[13], 2015	✗	✓	✗	type-1	KP/CP	Tree	L	M	MKS	✗	✓	-	-
[14], 2017	✗	✓	✗	type-1	KP	Matrix	S	M	MKS	✗	✓	-	80
[15], 2018	✓	✓	✗	type-1	CP	Tree	S	M	MKS	✓	✓	-	80
[16], 2018	✗	✓	✗	type-1	Other	Other	S	NM	SKS	✗	✓	-	80
[23], 2018	✓	✗	✗	type-1	CP	Matrix	S	M	MKS	✗	✗	-	80
[24], 2018	✗	✓	✗	type-1	Other	Other	S	NM	SKS	✗	✗	-	80
[17], 2019	✗	✓	✗	type-1	KP	Tree	S	M	MKS	✗	✓	-	80
[18], 2019	✗	✓	✗	type-1	KP	Tree	S	M	SKS	✓	✓	-	80
[9], 2019	✗	✓	✗	type-1	CP	Tree	L	M	SKS	✓	✓	-	80
[25], 2019	✓	✗	✗	type-1	CP	Tree	L	M	SKS	✓	✓	-	80
Ours, 2020	✓	✓	✓	type-3	CP	Matrix	L	M	MKS	✓	✓	≥ 128	≥ 128

Since our approach is fully based on attribute-based encryption (ABE) to fulfill R1-R4, we provide in Table 2 a summary of previous works that have considered ABE or ABSE approaches for enabling a cloud-based searchable encryption system. It is worth noting that in most of the cases, the four requirements (R1 - R4) for a functional cloud-based searchable encryption system have not been met neither both ABE and ABSE have been considered at the same time to achieve R1-R4. Furthermore, provided ABE realizations have been limited to constructions only for outdated security levels (symmetric pairing, 80-bits).

In Table 2, we have highlighted the following relevant aspects of ABE in the provided constructions: the pairing used (e) to deploy ABE; the ABE type used, which is important in terms of access control applicability; the access structure being used, being the most common the tree structure (for ABE constructions based on [21]) or the matrix structure (for LSSS); attribute universe U, small (S) or large (L) (in the former, all the attributes must be known in advance and once ABE is deployed no more attributes can be added); access structure expressiveness M, being better preferred monotonic policies (M) instead of non-monotonic ones (*NM*); S_{type} indicates how the searching process is based (on single word *SKS* or multi-keyword *MKS* search); IR$_B$ indicates if benchmarks were used to evaluate the information retrieval efficiency, very important to fulfill R3 requirement; *Exp* indicates if some experimental evaluation was done.

Authors in [13] proposed to use either KP- or CP- ABE versions only to fulfill R2, however, they do not provide a specific construction and their experiments are vaguely explained (there are no details about datasets, security levels, ABE settings, etc). The approach in [14] also only focuses on the R2 requirement, limited to small universe KP-ABE, and with scarce experimental details to evaluate if R3 was or not met.

In [15], there is a first attempt to fulfill R1 and R2. However, it fails to achieve R3 and R4, besides the construction is for an ABE small attribute universe. In [16], [17], [24], and [18] only R2 is addressed and the construction is limited to a small attribute universe. The usage of non-monotonic access structures in [16] and [24] led to less expressiveness of access policies. In [23], R1 is fulfilled by using ABE, but R2 is achieved by deriving a special key depending on DU's attribute set. However, the secure index is not encrypted with an access structure as required in ABSE. Besides, the construction is limited to a small attribute universe. The main advantage of the construction provided in [9] over [14], [16]–[18] is the usage of CP-ABE in large attribute universe. However, it still fails to fulfill R1. Although in [25] the idea is to use ABE to fulfill R1, R2 is met by using SSE, not with ABSE. So, R2 is not completely fulfilled as the secure index is not encrypted with an access structure as required in ABSE.

For all proposals in Table 2, the provided ABE realizations to fulfill either R1 or R2 requirements are limited to the type-1 pairing, which is a symmetric pairing not practical for realizing ABE for higher security levels as demanded by R4 requirement (the greater the security level, the greater the size of the operands and hence the processing times). ABE construction based on the Type 1 pairing have been only efficiently realized for the outdated 80-bit security level.

In [9], authors proposed a ciphertext-policy attribute-based searchable encryption schema (CP-ABSE), relying on the CP-ABE scheme proposed in [21] (hereafter referred to as BSW07 scheme). Although sufficient for practical use, the BSW07 construction has the main disadvantage that its security analysis is limited to the generic group model.

In this work, we provide a novel CP-ABSE construction over Type-III pairings taking as basis the well known CP-ABE model proposed by Waters [26] (hereafter referred as W11 scheme). The W11 construction was proven to be secure in a more solid model, the standard one. Furthermore, W11 construction was proved to be both foundationally sound and practical, supporting expressive access control structures. Because of that, most of the actual realizations of CP-ABEs are compliant with that construction. The original W11 scheme was proposed for the Type-I pairing, not for searchable encryption and limited to the symmetric pairings. Additionally, we include in our CP-ABSE construction the LSSS scheme reported in [27] in order to reduce the space of the ABE ciphertext, not considered in [9]. Our CP-ABSE construction considers a large attributes universe setting, which allows supporting expressive access structures.

III. PRELIMINARIES

This section introduces definitions relevant for the design of our attribute-based encryption approach for secure documents storage, sharing and retrieval.

A. PAIRINGS AND SECURITY ASSUMPTIONS

Pairings are mathematical objects defined over groups and efficient computable pairings are used to construct several cryptographic algorithms and protocols such as ABE.

1) BILINEAR PAIRINGS

A bilinear pairing [28] is defined as the mapping $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, where $\mathbb{G}_1 = \langle g_1 \rangle$, $\mathbb{G}_2 = \langle g_2 \rangle$, and $\mathbb{G}_T$ are cyclic groups of prime order r, with g_1 and g_2 the generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. The paring e must satisfy the properties of

1) *Bilinearlity*: $\forall g \in \mathbb{G}_1, \forall \hat{g} \in \mathbb{G}_2$, and $a, b \in \mathbb{Z}_r^*$, $e(g^a, \hat{g}^b) = e(g, \hat{g})^{ab}$.
2) *Non-degeneracy*: $e(g_1, g_2) \neq 1$.

$\mathbb{Z}_r^*$ is the set $\{1, 2, \ldots, r-1\}$. In practice, $\mathbb{G}_1$ and $\mathbb{G}_2$ are subgroups of a so-called friendly elliptic curve E defined over a finite field $\mathbb{F}_q$, and $\mathbb{G}_T$ is the multiplicative group of extension field $\mathbb{F}_{q^k}$ with k referred to as the embedding degree of the elliptic curve, the smallest positive integer such that r divides to $q^k - 1$. If $\mathbb{G}_1 = \mathbb{G}_2$ the pairing is symmetric (Type-I), otherwise it is asymmetric. If the pairing is asymmetric and no efficient isomorphism is known between $\mathbb{G}_1$ and $\mathbb{G}_2$, then the pairing is said to be of Type-III.

While a pairing-based cryptographic construction for a Type-III pairing is easy to transform to Type-I (by taking $\mathbb{G}_1 = \mathbb{G}_2$), the opposite is not trivial. The main motivation to use Type-III pairing constructions of pairing-based cryptographic protocols such as ABE is because of performance and security [29]. Type-I pairings has shown serious security issues [30]. Several applications of Type-III paring-based constructions are in practical use, for example for protecting privacy of transactions with the zk-SNARKs algorithm [31] and in several Blockchain projects [32]. A kind of friendly elliptic curve is the Barreto-Naehrig curve (BN curve) [33], which constructs Type-III pairings well suited for practical usage.

2) SECURITY ASSUMPTIONS

Definition 1 (Discrete Logarithm (DL) Assumption): Let $\mathbb{G}$ be a cyclic group with λ-bit prime order r. Let g be a generator of $\mathbb{G}$. Given g and g^a, the DL assumption is defined as: no probabilistic polynomial-time adversary $\mathcal{A}$ can compute $a \in \mathbb{Z}_r^*$ with a non-negligible advantage $Adv_{\mathcal{A}}^{\text{DL}}(\lambda)$ in the security parameter λ.

The security level of pairing-friendly curves is estimated by the computational cost of the most efficient algorithm to solve the discrete logarithm problem (DLP) in the groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$. This estimation is reflected in recommended group size $\lambda = \log_2 r$ given by the size in bits for q and k by several standards and international projects [34]. In this work, all the underlying constructions of ABE are for the Type-III pairing realized with updated groups size λ as recommended in [35].

As other pairing-based cryptographic schemes, the ABE constructions proposed in this work are based on the following security assumptions for the Type-III pairings.

Definition 2 (Decisional Diffie-Hellman (DDH) Assumption): Let $\mathbb{G}$ be a cyclic group with λ-bit prime order r. Let g be a generator of $\mathbb{G}$. Given the tuple $\left(g, g^a, g^b\right) \in \mathbb{G}$ with $a, b, c \in \mathbb{Z}_r^*$ (random), the DDH assumption is defined as: no probabilistic polynomial-time adversary $\mathcal{A}$ can decide whether g^{ab} is equal to g^c with a non-negligible advantage $Adv_{\mathcal{A}}^{\text{DDH}}(\lambda)$. The DDH holds if for any $\mathcal{A}$ with output in $\{0,1\}$

$$Adv_{\mathcal{A}}^{\text{DDH}}(\lambda) = \left| \Pr\left[\mathcal{A}(g, g^a, g^b, g^{ab})\right] - \Pr\left[\mathcal{A}(g, g^a, g^b, g^c)\right] \right|$$

is negligible in the security parameter λ.

Definition 3 (Symmetric External Diffie-Hellman (SXDH) Assumption): Let P_λ be the setting for a Type-III pairing consisting on the tuple $\{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, r\}$. The SXDH assumption holds if DDH holds in both $\mathbb{G}_1$ and $\mathbb{G}_2$.

Definition 4 (Decisional Bilinear Diffie-Hellman (DBDH) Type-III Assumption [36]): Let P_λ be the setting for a Type-III pairing consisting on the tuple $\{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, r\}$. The DBDH assumption is defined as: no probabilistic polynomial-time adversary $\mathcal{A}$ can distinguish $\left\{g_1, g_2, g_1^a, g_1^b, g_1^c, g_2^b, g_2^c, e(g_1, g_2)^{abc}\right\}$ from $\left\{g_1, g_2, g_1^a, g_1^b, g_1^c, g_2^b, g_2^c, e(g_1, g_2)^z\right\}$ for random elements $a, b, z \in \mathbb{Z}_r^*$ with a non-negligible advantage $Adv_{\mathcal{A}}^{\text{DBDH}}(\lambda)$.

B. CP-ABE AND DET-ABE

A Secret Sharing Scheme (SSS) in attribute-based encryption (e.g. CP-ABE) is crucial. In this context, an access structure $\mathbb{A}$ is defined as a non-empty subset of the power set $\mathcal{P}(U)$, with U a set of attributes. S_A is called an authorized set if $S_A \in \mathbb{A}$. An SSS involves a dealer that shares a secret s with a set of n parties defined by U. In CP-ABE, S_As are authorized sets of attributes to decrypt a ciphertext. $\mathbb{A}$ is said to be monotone, if given $S_A \in \mathbb{A}$ and $S_A \subset S_B$, then $S_B \in \mathbb{A}$. That is, decryption privileges are not lost even if more attributes are acquired by decryptors. In the W11 ABE construction, $\mathbb{A}$ is an $n \times l$ matrix (for an access policy including n attributes) instead of the tree access structure used in the BSW07 construction, without loss of efficiency. In this work, the linear SSS (LSSS) proposed by Liu and Cao in [27] is used, where the concept of Formatted Boolean Formula (FBF) over attributes is proposed to represent the access policy associated to the access structure $\mathbb{A}$.

CP-ABE (as other PKC schemes) relies on mathematical operations over large numbers (hundreds of bits). That is why CP-ABE is not used to encrypt data massively. For this purpose, symmetric ciphers E using the encryption key k_1

(i.e. $E_{k_1}(D)$) are faster and preferred. Digital envelopes [37] have been effective methods to encrypt large data with E and securely distribute k_1 to intended decryptors. In [38], digital envelopes realized from attribute-based encryption (called DET-ABE) were proposed by using the BSW07 construction. Later, in [11] the construction was extended to the ABE construction based on W11. In DET-ABE, data D is encrypted with the Advanced Encryption Standard (AES) using a session key k_1, which is encrypted with CP-ABE with an access policy. Only those authorized entities with a valid set of attributes satisfying the ABE encryption policy could decrypt and recover the data encryption key k_1 to then decrypt and recover D. DET-ABE can be formally defined by the four polynomial-time algorithms [11]:

1) **DET-ABE.setup**$(1^\lambda) \rightarrow \{MK, PK\}$: Creates MK (master private key) and PK (master public key), such that the length of both keys is compliant with the λ security strength. Initializes the attributes universe set U.
2) **DET-ABE.privateKeyGen**$(MK, S_u) \rightarrow SK_u$: Creates the user private key SK_u by using MK, and given a set of attributes $S_u \subset U$, specific for the user u.
3) **DET-ABE.encrypt**$(D, PK, \mathbb{A}) \rightarrow CT_D$: Encrypts data D using the AES cipher with a private session key k_1. Then, encrypts k_1 with CP-ABE using PK and an access structure $\mathbb{A}$, which corresponds to an access policy over attributes in U. The resulting encryption is the package $CT_D = \{AES_{k_1}(D), \text{CP-ABE}(k_1, \mathbb{A})\}$.
4) **DET-ABE.decrypt**$(CT_D, SK_u) \rightarrow D$: Decrypts CP-ABE$(k_1, \mathbb{A})$ using decryptor's private key SK_u to recover k_1. Then, decrypts $AES_{k_1}(D)$ using k_1. Decryption works only if the attribute set S_u used to generate SK_u is in $\mathbb{A}$.

DET-ABE provides effective access control mechanisms and confidentiality over a large documents dataset. It has been successfully tested in other systems [39], [40], however, DET-ABE does not support searchable encryption (DET-ABE only meets requirement R1).

C. ATTRIBUTE-BASED SEARCHABLE ENCRYPTION (ABSE)

In [9], authors proposed CP-ABSE, a realization of ABSE using the CP-ABE BSW07 construction as basis. As it is the case of other searchable encryption techniques, CP-ABSE constructs a secure index, that is, the index is encrypted so it does not support traditional queries over plaintext keywords; the matching is cryptographically tested. The secure index maps encrypted keywords to encrypted documents, so the searching process does not leak information about the data being queried. CP-ABSE consists of the following five polynomial-time algorithms [9].

1) **CP-ABSE.setup**$(1^\lambda) \rightarrow \{dk_1, dk_2\}$: Creates two keys dk_1 and dk_2; dk_1 is used to encrypt an index keyword given an access structure $\mathbb{A}$ and dk_2 is used to generate user's private key given its attribute set S_A in an attribute universe U. With private keys, users can create encrypted queries for data retrieval.
2) **CP-ABSE.privateKeyGen**$(dk_2, S_u) \rightarrow SK_u$: Creates the user private key SK_u using dk_2 from a set of attributes $S_u \subset U$, specific for the user u.
3) **CP-ABSE.encInd**$(dk_1, w, \mathbb{A}) \rightarrow CT_w$: Creates a secure index by encrypting with CP-ABE each keyword w in the index. The result is the set of each encrypted keyword $CT_w = \text{CP-ABE}(dk_1, w, \mathbb{A})$.
4) **CP-ABSE.Trpdr**$(SK_u, w_q) \rightarrow T_u(w_q)$: Generates an encrypted query from keyword w_q, by using the user private key SK_u. The encrypted query is the trapdoor $T_u(w_q)$.
5) **CP-ABSE.search**$\big(CT_w, T_u(w_q)\big) \rightarrow \{0, 1\}$: Given the encrypted keyword CT_w and a trapdoor $T_u(w_q)$, the algorithm returns '1' if $w = w_q$ (cryptographically tested) and if the attribute set S_u that generated SK_u and used to create $T_u(w_q)$ is in the access structure $\mathbb{A}$ used to encrypt CT_w, simultaneously; otherwise it outputs '0'.

The original CP-ABSE construction in [9] was provided for the Type-I pairing (symmetric), based on the BSW07 CP-ABE construction. In this work, we provide a novel CP-ABSE construction over the Type-III pairing, which allows realizations for security levels equal or greater than 128-bits. Furthermore, our construction is not based on the CP-ABE BSW07 construction but on the W11, which has been proved to be both foundationally sound and practical, supporting expressive access control structures using a large attributes universe setting.

IV. SYSTEM MODEL AND THREAT MODEL

Our approach, fully constructed on attribute-based encryption is named FABECS. It is created on the basis of DET-ABE and CP-ABSE, each one relying on CP-ABE.

The system model in this work is for a cloud-based document sharing and retrieval system with enabled searchable encryption, graphically shown in Fig. 1. The main actors in this model and their capabilities are summarized in Table 3. DO possesses a set of documents D. It creates the secure index SI from a keywords set in D by using CP-ABSE (with *search control policies* given by access structure $\mathbb{A}_s$). DO encrypts D using DET-ABE (with *data access control policies* given by access structure $\mathbb{A}_d$). Thus, DO achieves R1 and R2 requirements. By using the available service provided by the CSP, DO uploads to the cloud both the encrypted documents CT_D (as digital envelopes) and the secure (encrypted) index SI. DUs with properly authorized attribute set S_u creates encrypted queries (trapdoors) $T_u(w_q)$ using ABSE private keys for given keywords of interest w_q. DU sends the encrypted query to the CSP. The CSP uses $T_u(w_q)$ to query the encrypted index SI and to locate the documents set $E_{k_1}(D')$ ($D' \subset D$) that satisfies the search criteria. For efficiency and bandwidth savings, the CSP can rank the results and returns back to DU the k-most relevant encrypted documents that satisfy the query. Finally, DU uses its DET-ABE private key (derived from its attributes) to decrypt and obtain D' (documents in clear).

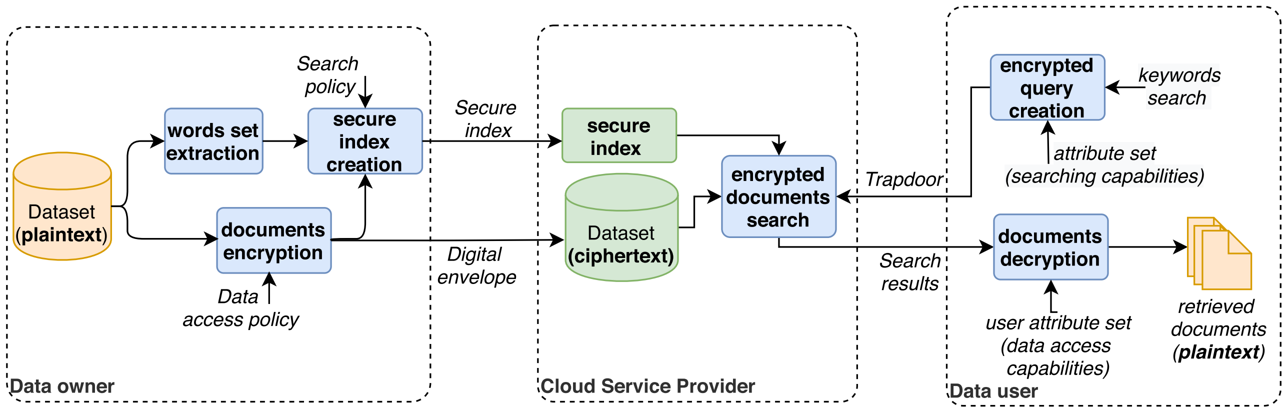


FIGURE 1. System model for document sharing and retrieval in the cloud.

TABLE 3. Actors and their main actions in the system model.

Actor	Actions
Data owner (DO)	i) *Extracts* keywords W from D and *creates* the secure index using a search policy represented by the access structure $\mathbb{A}_s$. ii) *Encrypts* the documents dataset D with session private key k_1 and ensures access control over k_1 using the data access policy represented by the access structure $\mathbb{A}_d$. iii) *Uploads* to the cloud repository both, the encrypted documents and the secure index.
Data user (DU)	iv) *Creates* an encrypted query or trapdoor using its private key SK_u. v) *Sends* the trapdoor to the CSP to retrieve the k-most relevant documents of interest from the cloud. vi) *Decrypts* the retrieved encrypted data.
Cloud service provider (CSP)	vii) *Stores* encrypted documents and its associated secure index. $viii$) *Searches over* the repository to locate the data of interest, given a trapdoor. ix) *Ranks* the results to deliver only the k-most relevant to the requester.

In the system model, it is assumed that DO is fully trusted. The adversary is the CSP, e.g., a system administrator that maliciously obtain remote access to storage infrastructure of CSP but cannot access the volatile memory. The adversary does not modify/destroy the stored data but tries to derive sensitive information from the stored documents, DU's queries as well as search outcomes. The CSP is supposed to only know the DO's encrypted data set and the DO's searchable index I (known ciphertext model). During search operations, the CSP could posse more knowledge as correlation relationship of given search requests (trapdoors), as well as statistical information of data sets (Known background model) [22].

The system model previously described to support query-based retrieval over encrypted data relies on the following assumptions.

1) There are means to authenticate each actor in the system.
2) There is a secure way for DU to generate and obtain the encrypted queries.
3) Key management, encryption and decryption of documents and queries are operations of proven semantically secure ciphers for data encryption and secure index generation.
4) Denial-of-service attacks are excluded. The adversary aims to compromise the confidentiality of data or searching privileges by forging existing access policies generated by the corresponding data owners.

V. OUR APPROACH FULLY BASED ON ATTRIBUTE-BASED ENCRYPTION

FABECS is constructed taking advantage of shared components in DET-ABE and CP-ABSE. Setup in DET-ABE and CP-ABSE are in essence the same, with $MK = dk_2$ and $PK = dk_1$. So, one single setup can serve for both realizations. With DET-ABE and ABSE defined over a large universe scenario, the attribute set U is dynamically updated. The *privateKeyGen* function in DET-ABE and CP-ABSE, with the proper attribute set, are the same. CP-ABE encryption can be displayed as a single module used to realize completely *CP-ABSE.encInd* and the symmetric key encryption in *DET-ABE.encrypt*. In each case, a different access structure can be used if searching and data access capabilities are different. Finally, the core of CP-ABE decryption consisting in recovering a secret from the attributes in the decryption key (for *DET-ABE.decrypt*) or in the trapdoor (for *CP-ABSE.search*) is the same, but the final result is different. This module can also be reused to serve as a common engine for DET-ABE and CP-ABSE.

From a functional point of view, FABECS is defined by the following seven polynomial-time algorithms.

1) **FABESC.setup**$(1^\lambda) \rightarrow \{MK, PK\}$**.** Creates MK (master private key) and PK (master public key), such that the length of both keys is compliant with the λ security strength.

2) **FABESC.encrypt**$(D, PK, \mathbb{A}_d) \rightarrow \{CT_D\}$: Encrypts data D as $CT_D \leftarrow$ *DET-ABE.encrypt*$(D, PK, \mathbb{A}_d)$. The access policy is a Formatted Boolean Formula (FBF) expressed by the access structure $\mathbb{A}_d$.
3) **FABECS.userKeyGen**$(MK, S_u) \rightarrow SK_u$: Creates the user private key SK_u using MK, from a set of attributes $S_u \subset U$, specific for the user u.
4) **FABECS.encInd**$(PK, W, \mathbb{A}_s) \rightarrow SI_W$: Creates a secure index by invoking *CP-ABSE.encInd*$(PK, w, \mathbb{A}_s)$, for each w in the keywords set W obtained from D. The search policy is a Formated Boolean Formula (FBF) expressed by the access structure $\mathbb{A}_s$.
5) **FABECS.Trpdr**$(SK_u, w_q) \rightarrow T_u(w_q)$: Generates an encrypted query from keyword w_q by computing $T_u(w_q) \leftarrow$ *CP-ABSE.Trpdr*(SK_u, w_q).
6) **FABECS.search**$\left(SI_W, T_u(w_q)\right) \rightarrow CT_{D'}$: By using *CP-ABSE.search*, test if there is a match of w_q with a encrypted keyword w in SI_W. If the match exists, all the encrypted documents $CT_{D'}$ associated to the encrypted word w in SI_W are ranked to the k most relevant results and returned to the requester.
7) **FABECS.decrypt**$(CT_{D'}, SK_u) \rightarrow \widehat{D}$: Decrypts and obtains $\widehat{D} \subset D$ (in plaintext form) as $\widehat{D} \leftarrow$ *DET-ABE.decrypt*$(CT_{D'}, SK_u)$.

In FABECS, note that the private key used in *FABECS.Trpdr* and in *FABECS.decrypt* is not necessarily the same; a DU in FABECS can be given a single key with both search and data access capabilities, only one of those capabilities or none. In the same way, $\mathbb{A}_d$ used for the dataset encryption is not necessarily the same than $\mathbb{A}_s$ for the secure index generation.

Correctness: FABECS scheme is correct if

$$Pr\left[\begin{array}{l|l} decrypt(CT_{D'}, & setup(\lambda) \rightarrow \{MK, PK\}; \\ SK_u) \rightarrow \widehat{D} & encrypt(D, PK, \mathbb{A}_d) \rightarrow CT_D; \\ & userKeyGen(MK, S_u) \rightarrow SK_u; \\ & encInd(PK, W, \mathbb{A}_s) \rightarrow SI_W; \\ & Trpdr(SK_u, w_q) \rightarrow T_u(w_q); \\ & search\left(SI_W, T_u(w_q)\right) \rightarrow CT_{D'}; \end{array}\right] = 1.$$

A. FABECS CONSTRUCTION IN THE TYPE-III PAIRING

The *FABESC.setup* function takes as input the security strength λ given in bits. This function is executed by DO. FABECS construction is for the Type-III pairings, so $\lambda \geq$ 128-bit is possible for example by using BN curves to produce the Type-III pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. After the setup execution, the public parameters $P_\lambda = \{e, \mathbb{G}_1 = \langle g_1 \rangle, \mathbb{G}_2 = \langle g_2 \rangle, \mathbb{G}_T, r\}$ are produced and used in all the FABECS algorithms. The elements g_1, g_2 are generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively, while r is the prime order of the three groups defining e. Also, two hash functions are created and initialized: $H_1 : \{0, 1\}^* \rightarrow \mathbb{G}_1$ and $H_2 : \{0, 1\}^* \rightarrow \mathbb{Z}_r^*$. Finally, the *setup* function produces the keys $\{PK, MK\}$ defined as:

$$PK_\lambda = \{g_1, g_2, h = g_1^\beta, e(g_1, g_2)^\alpha\}$$
$$\alpha, \beta \in \mathbb{Z}_r^* (\text{random})$$
$$MK_\lambda = \{g_1^\alpha\}$$

PK_λ is the public master key used to create the secure index and for digital envelopes generation, MK_λ is the private master key safeguarded by DO and used to create DU's private keys for trapdoors generation (searching on encrypted data) or data decryption (data access).

1) DOCUMENTS ENCRYPTION AND KEY MANAGEMENT

The *FABECS.encrypt* function performs symmetric bulk encryption of a dataset D composed of several documents owned by DO. By using the *DET-ABE.encrypt* function, the result is $CT_D = \left\{AES_{k_1}(D), CP\text{-}ABE(k_1, \mathbb{A}_d)\right\}$, where k_1 is a secure session key of strength λ. The ciphertext produced by CP-ABE is the tuple $CT_D = \{C, C', [C_{y_i}, C'_{y_i}]^+\}$, where

$C' = k_1 \times e(g_1, g_2)^{\alpha s}$, $s \in \mathbb{Z}_r^*$ (random)
$C = g_2^s$
`Mat` is an $n \times l$ matrix that represents $\mathbb{A}_d$. $\texttt{Mat}_i$ is the i-th row of `Mat`. ρ associates rows i of `Mat` to attributes y in the access policy.
$\forall\ \rho(i)$, attribute y at row $\texttt{Mat}_i$
$C_{y_i} = g_2^{r_i}$, $r_i \in \mathbb{Z}_r^*$ (random)
$C'_{y_i} = h^{\lambda_i} \times H_1(\rho(i))^{-r_i}$
$\lambda_i = \mathbf{v} \times \texttt{Mat}_i$
$\mathbf{v} = \{s, s_1, s_2, \ldots, s_{n-1}\}$, $s_j \in \mathbb{Z}_r^*$ (random)

2) USER KEY GENERATION

The *FABECS.userKeyGen* function produces the key SK_u for DU to get access to encrypted data CT_D produced by the *FABECS.encrypt* function. That key can be used for querying data by means of function *FABECS.search*, it all depends on the attributes S_u being used to produce the key. This key is created by DO, giving access privileges to the user u from the private master key MK_λ and u's attributes set $S_u \subset U$. SK_u is defined by the tuple $\{D, D', [d_y]^+\}$, where

$D = g_1^{(\alpha+\beta r)}$, $r \in \mathbb{Z}_r^*$ (random)
$D' = g_2^r$
$\forall y \in S_u$: $d_y = H_1(y)^r$

3) SECURE INDEX CREATION

The function *FABECS.encInd* executes a pre-processing over the data set D consisting in two tasks. The first one is for obtaining the keywords set W and the other is to implement the mechanism used later for ranking the results in the searching process. W is obtained in two rounds. In the first one, the keywords are identified from D discarding punctuation marks and stop-words. In the second one, W is enriched by forming compound words of length m, and adding them to W. In this way, it is eliminated the need for knowing the index words offset position to allow the service provider to search for query phrases, which also helps avoiding attacks such as known plaintext or statistical attacks [6].

The ranking mechanism implemented in FABECS is based on the word membership metric introduced in [6]. Words membership M_{w_i} is helpful to determine how well a word w_i represents the contents of a document $D_j \in D$. M_{w_i} is computed by Eq. 1 given w_i's frequency f_{w_i} and the total number of index words $|W|$.

$$M_{w_i} = \frac{f_{w_i}}{|W|} \tag{1}$$

The result of the pre-processing is a set W containing single and compound words and its respective membership values. The secure index SI_W is a key-value map formed by the encrypted keyword w_1 associated with all the encrypted documents containing w_1. The encryption of each $w_i \in W$ is obtained by invoking *CP-ABSE.encInd*(PK_λ, w_i, $\mathbb{A}_s$). The ciphertext for w_i is defined by the tuple $CT_{w_i} = \{C, C', [C_{y_i}, C'_{y_i}]\}$, where

$C' = e(g_1^{H_2(w_i)*s}, g_2) \times e(g_1, g_2)^{\alpha s}$, $s \in \mathbb{Z}_r^*$ (random)
$C = g_2^s$
The access structure $\mathbb{A}_s$ is expressed as the $n \times l$ matrix `Mat` and rows of `Mat` related to attributes in the access policy being used, which is expressed by $\rho(i)$.
$\forall\ \rho(i)$, attribute y at row $\mathtt{Mat}_i$
$C_{y_i} = g_2^{r_i}$, $r_i \in \mathbb{Z}_r^*$ (random)
$C'_{y_i} = h^{\lambda_i} \times H_1(\rho(i))^{-r_i}$
$\lambda_i = \mathbf{v} \times \mathtt{Mat}_i$
$\mathbf{v} = \{s, s_1, s_2, \ldots, s_{n-1}\}$, $s_j \in \mathbb{Z}_r^*$ (random)

In the same way than in [9], encrypted index keywords and encrypted data are organized as inverted index construction by using an array as well as a look-up table data structures, which allows achieving sub-linear search complexity.

4) TRAPDOOR GENERATION

A data user DU with the proper search privileges expressed in its key SK_u can generate the trapdoor $T_u(q)$ given a keywords set $q = \{q_1, q_2, \ldots, q_t\}$. The set q is processed in the same way that DO obtained W during the creation of SI_W, that is, stop words are removed from q resulting in the set W_s. With the single keywords in W_s, compound words of length at most m are created to form the set W_c.

Each $w_s \in W_s$ has a ranking score weight $S_{w_s} = 1$, and for each $w_c \in W_c$ its ranking score weight S_{w_c} is computed as in Eq. 2. Also, the membership entropy E_{w_c} is computed for each query compound word as in Eq. 3. E_{w_c} measures the membership approximation of a long compound word and its partial words. That is, it allows evaluating how much information is embedded on every w_c. Both the ranking weight and the membership entropy will later help the *CSP* to set a score for each found result for $T_u(q)$.

$$S_{w_c} = \sum_{w_s \in w_c} S_{w_s} \times e^{\frac{n_1}{|w_c|}-1} \tag{2}$$

$$E_{w_c} = -\sum_{w_s \in w_c} log_2 \left(\frac{S_{w_s}}{\sum_{w_s \in w_c} S_{w_s}} \right) \tag{3}$$

Using his user secret key SK_u, DU computes the query trapdoors $T_u(q)$ for each query word w_q (in W_s or in W_c) by using the function *CP-ABSE.Trpdr*(SK_u, w_q). The trapdoor is defined by the tuple $T_u(w_q) = \{T, T', [t_y]^+\}$ given $SK_u = \{D, D', [d_y]^+\}$, where

$$\begin{aligned} T &= g_1^{H_2(w_q)} \times D \\ T' &= D' \in SK_u \\ ty &= d_y \in SK_u \end{aligned}$$

5) SEARCH

DU can ask the *CSP* for searching over encrypted data, given an encrypted query $T_u(q)$ with encrypted query words w_q by using the *FABECS.search* function. An encrypted entry CT_w in SI_W will return '1' if and only if *CP-ABSE.search*(CT_w, $T_u(w_q)$) returns '1', that is, if and only if $w = w_q$ (cryptographically tested) and the attribute set in SK_u used to create $T_u(q)$ satisfies the access structure $\mathbb{A}_s$. Given an encrypted word $CT_w = \{C, C', [C_{yi}, C'_{yi}]\}$ in the secure index and the trapdoor $\{T, T', [t_y]^+\}$ for the user with key SK_u, the test ($w = w_q$) is done as follows:

- Compute $C_w = \dfrac{e(T, C)}{C_T}$
- $\prod \left[e(C'_{y_i}, T') e(t_{yi}, C_{y_i}) \right]^{\omega_i} = C_T$
  ω_i is a constant in $\mathbb{Z}_r^*$ given $\rho(i)$ associated to an attribute yi in `Mat` (yi relates the attribute in the ciphertext and in the trapdoor). If $\{\lambda_i\}$ are valid shares of any secret s according to $\mathbb{A}_s$, then $\sum \omega_i \lambda_i = s$.
- if $C_w \equiv C'$, return '1'. Otherwise return '0'.

If the search is successful for a given CT_w in SI_W, then all the encrypted files associated to that entry form the temporary documents (encrypted) set D_T. Next, the *CSP* computes the ranking score S_{D_j} for each document in D_T by using Eq. 4 based on the pre-calculated M_w and S_w, for all w that matched successfully during the search process, and E_{w_q} for all w_q in $T_u(q)$. When all the ranking scores are computed, the documents in D_T are ordered and the k-most relevant form the final result set of encrypted documents sent back to the user that sent the query to the CSP.

$$S_{D_j} = \alpha \times \sum_{\forall w} (S_w \times M_w) - \beta \times \sum_{\forall w_q \in q} E_{w_q}$$
$$(\alpha + \beta = 1; \alpha, \beta \geq 0) \tag{4}$$

6) DATA DECRYPTION

DU receives a result set from the encrypted query $T_u(q)$, with the k-most relevant encrypted documents. To obtain the documents $\widehat{D}$ in plaintext form, DU executes *DET-ABE.decrypt* given $SK_u = \{D, D', [t_y]^+\}$.

Each encrypted document is of the form $CT_{D_j} = \{AES_{k_1}(D_j)$, CP-ABE($k_1$, $\mathbb{A}_d$) $= (C, C', [C_{yi}, C'_{yi}])\}$. First, the symmetric key is recovered as

$k_1 = \dfrac{C'}{\left(\frac{e(D,C)}{C_T}\right)}$, where

$\prod \left[e(C'_{yi}, D') e(t_{yi}, C_{yi}) \right]^{\omega_i} = C_T$
ω_i is a constant in $\mathbb{Z}_r^*$ given $\rho(i)$ associated to an attribute yi in `Mat` (yi relates the attribute in the ciphertext and in

TABLE 4. Main components of FABECS construction (Type-III pairing) and comparison against construction in [9] (Type-I pairing).

	CP-ABSE Type-I [9]	CP-ABSE Type-III	DET-ABE Type-III
$\mathbb{A}$	Tree, with Shamir's secret sharing scheme and policies expressed with threshold gates	Matrix, with Linear secret sharing scheme and FBF access policy	
P_λ	$\{\mathbb{G}, \mathbb{G}_T, g, r\}$, $H_1 : \{0,1\}^* \rightarrow \mathbb{G}$, $H_2 : \{0,1\}^* \rightarrow \mathbb{Z}_r$	$\{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, r\}$, $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1$, $H_2 : \{0,1\}^* \rightarrow \mathbb{Z}_r$	
PK_λ	$\{h = g^\beta, e(g,g)^\alpha\}$, $\{\alpha, \beta\} \in \mathbb{Z}_r^*$ (random)	$\{h = g_1^\beta, e(g_1, g_2)^\alpha\}$, $\{\alpha, \beta\} \in \mathbb{Z}_r^*$ (random)	
MK_λ	$\{\beta, g^\alpha\}$	$\{g_1^\alpha\}$	
SK	$SK_u = \{D, D', [d_y, d'_y]^+\}$, where $D = g^{(\alpha+r)/\beta}$ $D' = g^{1/\beta}$ $\forall y \in S_u$: $d_y = g^r \times H_1(y)^{r_j}$ $d'_y = g^{r_j}$ $r, r_j \in \mathbb{Z}_r^*$ (random)	$SK_u = \{D, D', [d_y]^+\}$, where $D = g_1^{(\alpha+\beta r)}$ $D' = g_2^r$ $\forall j \in S_u$: $d_j = H_1(y)^r, r \in \mathbb{Z}_r^*$ (random)	
CT	$CT_w = \{C', C, [C_y, C'_y]^+\}$, where w is a keyword and $C' = e(g^{H_2(w)s}, g) \times e(g,g)^{\alpha s}$, $s \in \mathbb{Z}_r^*$ (random)	$CT_w = \{C', C, [C_{y_i}, C'_{y_i}]^+\}$, where w is a keyword and $C' = e(g_1^{H_2(w)s}, g_2) \times e(g_1, g_2)^{\alpha s}$, $s \in \mathbb{Z}_r^*$ (random)	$CT_k = \{C', C, [C_{y_i}, C'_{y_i}]^+\}$, where k is a symmetric key and $C' = k \times e(g_1, g_2)^{\alpha s}$, $s \in \mathbb{Z}_r^*$ (random)
	$C = h^s$ $\forall$ leaf node $y \in \mathbb{A}$ containing attribute j, $C_y = g^{q_y(0)}$ $C'_y = H_1(j)^{q_y(0)}$	$C = g_2^s$ $\forall\ \rho(i)$, attribute y at row $\mathtt{Mat}_i$ $C_{y_i} = g_2^{r_i}$ $C'_{y_i} = h^{\lambda_i} \times H_1(\rho(i))^{-r_i}$ $r_i \in \mathbb{Z}_r^*$ (random) $\lambda_i = \mathbf{v} \times \mathtt{Mat}_i$ $\mathbf{v} = \{s, s_1, s_2, \ldots, s_{n-1}\}$, $s_j \in \mathbb{Z}_r^*$ (random)	
T_u	$T_u(w_q) = \{T, [t_y, t'_y]^+\}$, given a query keyword w_q and SK_u, where $T = D'^{H_2(w_q)} \times D$ $t_y = d_y \in SK_u$ $t'_y = d'_y \in SK_u$	$T_u(w_q) = \{T, T', [t_y]^+\}$, given a query keyword w_q and SK_u, where $T = g_1^{H_2(w_q)} \times D$ $T' = D' \in SK_u$ $t_y = d_y \in SK_u$	Not required
Decrypt or Search	Search, given $CT_w = \{C', C, [C_y, C'_y]^+\}$ and $T_u(w_q) = \{T, [t_y, t'_y]^+\}$ $C_w = \frac{e(C,T)}{\prod \left(\frac{e(t_y, C_y)^{\mathcal{L}_y s'}}{e(t'_y, C'_y)} \right)}$ if and only if $C_w \equiv C'$ then w matches w_q	Search, given $CT_w = \{C', C, [C_{y_i}, C'_{y_i}]^+\}$ and $T_u(w_q) = \{T, T', [t_y]^+\}$ $C_w = \frac{e(T,C)}{C_T}$ $\prod \left(\left(e(C'_{y_i}, T') e(t_{yi}, C_{y_i}) \right)^{\omega_i} \right) = C_T$ if and only if $C_w \equiv C'$ then w matches w_q	Decrypt the symmetric key k, given $CT_k = \{C', C, [C_{y_i}, C'_{y_i}]^+\}$ and $SK_u = \{D, D', [d_y]^+\}$ $C_k = \frac{e(D,C)}{C_T}$ $\prod \left(\left(e(C'_{y_i}, D') e(d_{yi}, C_{yi}) \right)^{\omega_i} \right) = C_T$ $k_1 = \frac{C'}{C_k}$

the user private key). If $\{\lambda_i\}$ are valid shares of any secret s according to $\mathbb{A}_d$, then $\sum \omega_i \lambda_i = s$.

With k_1, the plaintext $\widehat{D}$ is obtained by decrypting each $AES_{k_1}(D_j)$ encrypted document.

As a summary, Table 4 shows the main components in FABECS. The table also compares the CP-ABSE construction provided in this work in the Type-III pairing against the CP-ABSE construction in [9], which is limited to the Type-I pairing and does not support data access control nor key management for documents decryption. From the table it can be observed how FABECS takes advantage of the pairing setting and main components and modules shared by both CP-ABSE and DET-ABE.

B. SECURITY ANALYSIS

It is assumed that all the involved cryptographic schemes in FABECS are semantically secure. This includes AES, CP-ABE and the hash functions used as building blocks of DET-ABE and CP-ABSE with a security strength λ sufficient to achieve at least a 128-bit security level [34], [35]. As described in Section IV, CSP is the main adversary in the FABECS system model. Data privacy of DO's data D is achieved in the system by DET-ABE, which prevents the CSP can snoop in the outsourced data while at the same time provides a fine grained access control over D for authorized DUs. Index privacy is achieved through CP-ABSE, by preventing the CSP can perform association attacks (associate keywords with encrypted data). The trapdoor used for querying encrypted data does not reveal to the CSP any information about the content of the retrieved documents. Also, the CSP is not able associate trapdoors with search-keywords (Keyword privacy); that is, the CSP will not learn that two trapdoors are intended for the same query (trapdoor unlinkability), which is achieved by CP-ABSE.

The security analysis of FABECS is presented in this section assuming the presence of an adversary $\mathcal{A}$ under the *known ciphertext attack model*, where $\mathcal{A}$ could be the attacker who get access to the cloud storage system or a malicious cloud service administrator.

1) DATA AND INDEX KEYWORD CONFIDENTIALITY

$\mathcal{A}$ tries to guess the word in SI_W and the information of encrypted documents CT_D.

In FABECS, D's encryption key and the index are encrypted by DO (fully-trusted). The submitted trapdoors to CSP are generated by an authorized DU. The collusion attacks between DUs can be avoided by the CP-ABE engine. In this way, D and W privacy can be well protected in the known ciphertext model.

FABECS should achieve the chosen plaintext attack (CPA) security, that is, it is required that malicious attackers cannot adaptively ask for the ciphertexts of arbitrary plaintext messages. In this case, messages can be either query words or decryption keys. CPA security in FABECS can be reduced to the DBDH assumption with the following theorem.

Theorem 1: Assume that a polynomial time adversary $\mathcal{A}$ can break the CPA security of FABECS with a non-negligible advantage ϵ, then there must exist a PPT simulator $\mathcal{B}$ which can break the DBDH problem with an advantage $\frac{\epsilon}{2}$.

Proof: Assume that a probabilistic polynomial time adversary $\mathcal{A}$ can recovery the index keyword information from SI_W or it can access the symmetric encryption key of documents D, with a non-negligible advantage ϵ.

Given the public bilinear parameters $P_\lambda = \{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, r\}$ defining the pairing e, the challenger $\mathcal{C}$ chooses $a', b', c', z \in \mathbb{Z}_r^*$, and a random bit $v \in \{0, 1\}$. If $v = 0$, $\mathcal{C}$ sets $Z = e(g_1, g_2)^{a'b'c'}$; otherwise, $Z = e(g_1, g_2)^z$. Assume that $\mathcal{C}$ gives the tuple $(g_1, g_2, g_1^{a'}, g_1^{b'}, g_1^{c'}, g_2^{b'}, g_2^{c'}, Z)$ to $\mathcal{B}$, then $\mathcal{B}$ plays the role of $\mathcal{C}$ in the following security game.

(1) $\mathcal{B}$ computes a public parameter $Y = e(g_1, g_2)^{bc}$ and sends it to $\mathcal{A}$. Also, it selects and sends to $\mathcal{A}$ a challenging access structure $\mathbb{A}^*$.

(2) $\mathcal{A}$ adaptively asks $\mathcal{B}$ for private keys $SK_{\mathcal{A}_1}$, $SK_{\mathcal{A}_2}$, …, $SK_{\mathcal{A}_n}$ of attribute sets $S_{\mathcal{A}_1}, S_{\mathcal{A}_2}, \ldots S_{\mathcal{A}_n}$. $\mathcal{A}$ adaptively asks $\mathcal{B}$ the ciphertexts CT_{m_1}, CT_{m_2}, …, CT_{m_n} where m_i is an index word or a symmetric AES keys. Each $S_{\mathcal{A}_i}$ set does not satisfy $\mathbb{A}^*$ and each $SK_{\mathcal{A}_1}$ can generate legal $T_{\mathcal{A}_i}(w_q)$ trapdoors. For a CT_{m_j}, *CP-ABSE.search*$\big(CT_{m_j}, T_{\mathcal{A}_i}(w_q)\big) = 1$ or *DET-ABE.decrypt*$(D', SK_{\mathcal{A}_i})$ decrypts D'.

(3) $\mathcal{A}$ submits two messages m_0, m_1 on which to be cha-llenged and $\mathbb{A}^*$ to $\mathcal{B}$. $\mathcal{B}$ chooses a bit $b \in \{0, 1\}$ and generates the ciphertext $CT_{m_b} = \{C', C, [C_{y_i}, C'_{y_i}]^+\}$, where

Case m_b is an index keyword: $C' = e(g_1^{aH_2(m_b)}, g_2) \times Z$:
Case m_b is a data decryption key: $C' = m_b \times Z$,
$C = g_2^a$
$\forall\ \rho(i)$, attribute y at row $\mathtt{Mat}_i$ associated to $\mathbb{A}^*$.
$C_{y_i} = g_2^{r_j}$, r_j (random)
$C'_{y_i} = h^{\lambda_i} \times H_1(\rho(i))^{-r_j}$
$\mathcal{B}$ sends CT_{m_b} to $\mathcal{A}$.

(4) $\mathcal{A}$ outputs the guess b' of b. The adversary cannot correctly decide whether $b = 0$ or $b = 1$ by letting *CP-ABSE.search*$\big(CT_{m_b}, T_{\mathcal{A}_i}(m_b)\big)$ to output '1' or by trying to obtain the plaintext from *DET-ABE.decrypt*$(D'$, $SK_{\mathcal{A}_i})$, since none of the attribute sets queried by $\mathcal{A}$ satisfy the access structure $\mathbb{A}^*$.
Let's suppose that $v = 0$. By letting $a = s$ and $bc = \alpha$,
$C' = e(g_1^{H_2(m_b)s}, g_2)e(g_1, g_2)^{a\alpha}$, or
$C' = m_b \times e(g_1, g_2)^{a\alpha}$
$C = g_2^s$
$\forall\ \rho(i)$, attribute y at row $\mathtt{Mat}_i$ associated to $\mathbb{A}^*$.
$C_{y_i} = g_2^{r_j}$, r_j (random)
$C'_{y_i} = h^{\lambda_i} \times H_1(\rho(i))^{-r_j}$
which means that CT_{m_b} is the correct ciphertext of m_b. Otherwise, if $v = 1$ then $Z = e(g_1, g_2)^z$ which means Z is a random value in $\mathbb{G}_T$ from the view of the adversary $\mathcal{A}$ and contains no information about w_b. $\mathcal{A}$ outputs the guess $b' \in \{0, 1\}$.

(5) $\mathcal{B}$ outputs v's guess $v' = 0$ indicating that the Challenger sent the correct encryption parameters to $\mathcal{B}$ and $Z = e(g_1, g_2)^{abc}$ on the condition that $b' = b$. In case that $b' \neq b$, $\mathcal{B}$ outputs $v' = 1$ which means the Challenger sent random encryption parameters to $\mathcal{B}$.
If $Z = e(g_1, g_2)^{abc}$, then $\mathcal{A}$ can break the security game with an advantage ϵ and the probability that $\mathcal{A}$ outputs $b' = b$ is $\dfrac{1}{2} + \epsilon$. Otherwise (Z is random), the probability that $\mathcal{A}$ outputs $b' = b$ is $\dfrac{1}{2}$.
Thus, the overall advantage that $\mathcal{B}$ in breaking the security game previously outlined is defined as

$$Adv_{\mathcal{B}}^{\text{DBDH}}(\lambda) = \left| \frac{1}{2}\Pr\left[v = v' | v = 0\right] + \frac{1}{2}\Pr\left[v = v' | v = 1\right] - \frac{1}{2} \right| = \epsilon/2$$

Since ϵ is non-negligible, $\mathcal{B}$ can solve the DBDH problem with non-negligible advantage, which contradicts the DBDH assumption.

2) QUERY CONFIDENTIALITY

FABECS achieves privacy of a query which is protected in terms of DU. Thus, under the eavesdropper attack model security of query trapdoor in FABECS can be reduced to the DL assumption with the following theorem.

Theorem 2: If the discrete logarithm problem is intractable, FABECS achieves query trapdoor security against the eavesdropper attack model.

Proof: The above theorem is proven by the following game between the polynomial time adversary $\mathcal{A}$ and challenger $\mathcal{C}$.

(1) $\mathcal{A}$ submits queries $w_{q_1}, w_{q_2, \ldots, w_{q_n}}$ to $\mathcal{C}$. For each query w_{q_i}, $\mathcal{C}$ returns the encrypted query $T_{\mathcal{A}}(w_{q_i}) = \{T, T', [t_y]^+\}$, given $SK_{\mathcal{A}}$ a valid private key

TABLE 5. Type-3 pairing setting for P_λ using Barreto-Naehrig curves (BN).

λ	AES key size	BN curve groups size (bits)				BN curve name
		$\mathbb{G}_1$	$\mathbb{G}_2$	$\mathbb{G}_T$	$\mathbb{Z}_r$	
100-bit	128-bit	512-bit	1024-bit	3072-bit	256-bit	BN256
128a-bit	128-bit	768-bit	1536-bit	4608-bit	384-bit	BN384
128b-bit	128-bit	928-bit	1856-bit	5568-bit	464-bit	BN464
>128-bit	192-bit	1280-bit	2560-bit	7680-bit	640-bit	BN640

of adversary $\mathcal{A}$, where

$$T = g_1^{H_2(w_{q_i})+\alpha+\beta r}$$
$$T' = D' \in SK_{\mathcal{A}}$$
$$t_y = d_y \in SK_{\mathcal{A}}$$

(2) $\mathcal{A}$ sends two challenge query words wq_1, wq_2 to $\mathcal{C}$. The restriction is that wq_1, wq_2 have not been queried before.

(3) $\mathcal{C}$ randomly chooses a bit $b \in \{0, 1\}$ and encypts wq_b as $T_{\mathcal{A}}(wq_b) = \{T, T', [t_y]^+\}$, where

$$T = g_1^{H_2(wq_b)+\alpha+\beta r}$$
$$T' = D' \in SK_{\mathcal{A}}$$
$$t_y = d_y \in SK_{\mathcal{A}}$$

and sends $T_{\mathcal{A}}(wq_b)$ to $\mathcal{A}$.

(4) $\mathcal{A}$ can continue asking $\mathcal{C}$ more $T_{\mathcal{A}}(wq)$ trapdoors of any two keywords other than wq_1, wq_2.

(5) $\mathcal{A}$ outputs the guess b' of b. Since $\mathcal{A}$ cannot get access to the encryption oracle, it cannot effectively compute the trapdoors $T_{\mathcal{A}}(wq_1)$, $T_{\mathcal{A}}(wq_2)$ without the security parameters $\{\alpha, \beta, r\}$. Hence, the probability that $\mathcal{A}$ outputs $b' = b$ is at most $\frac{1}{2}$ as long as the DL assumption remains. If DL is computationally feasible in polynomial time, $\mathcal{A}$ can output $b' = b$ with probability 1.

VI. EXPERIMENTS AND RESULTS

FABECS was deployed as a standalone application for validation and evaluation. The Java jPBC library [41] was used to implement all modules in the proposed construction and to run a set of experiments using pairings parameters P_λ shown in Table 5.

Type-III pairings were created using the available tools in jPBC for Barreto-Naehrig curves (BN). Groups size are given in bits and compliant with the ones recommended for a 128-bit security level for the symmetric cipher [34]. With the recent advances for computing discrete logarithms the key sizes were updated. Before, a BN256 curve was believed to provide a 128-bit security level. After the update, that curve was reduced to a 100-bit security level. For 128 bits of security, in [42] it was estimated the minimum groups size of BN curves after exTNFS attack as 383-bit (BN384 curve). That curve is also recommended in the ISO/IEC 15946-5 [43]. However, in [35] the recommendation is to use a BN curve with groups size of 462-bit to achieve a 128-bit security level (BN464 curve). We considered both the BN384 and BN464 curves for evaluating FABECS at a 128-bit security level and additionally, we included a BN curve with greater order, the BN640 curve, which is expected to provide a security level greater than 128-bit but less than 192-bit.

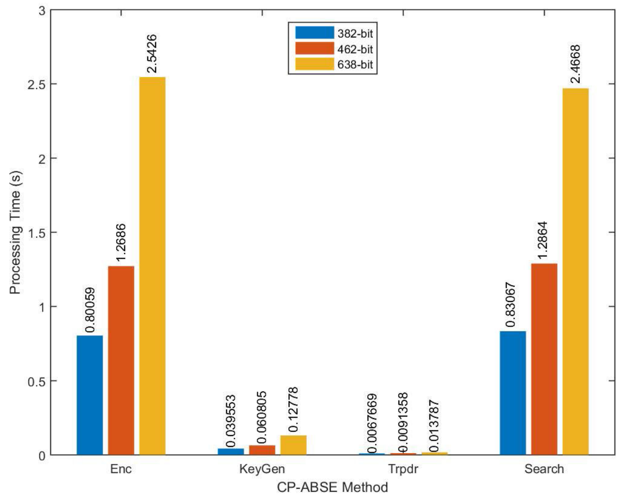
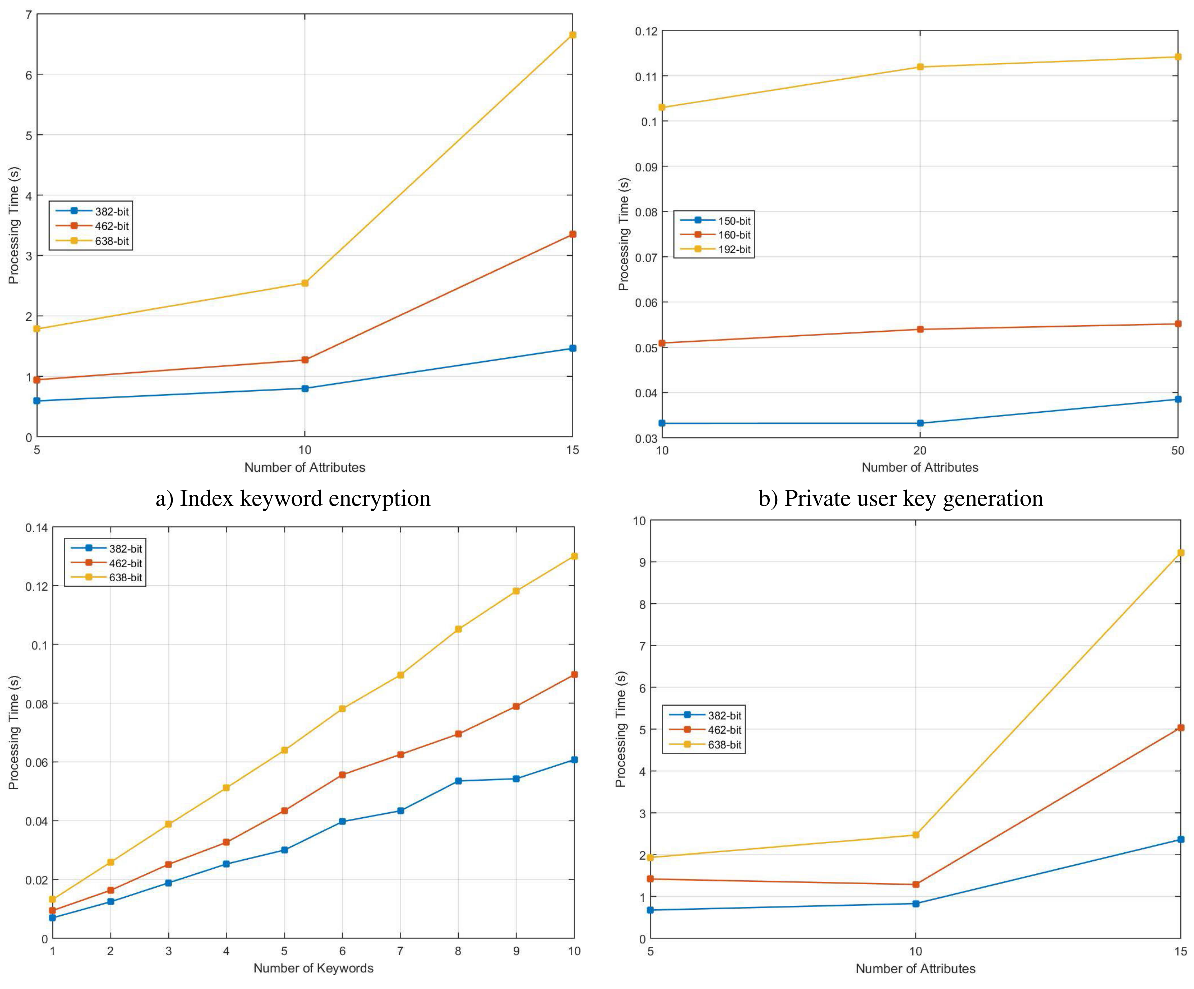


FIGURE 2. Running times for main operations in FABECS.

A. EVALUATION OF MAIN OPERATIONS IN FABECS

Fig. 2 shows the running times for the main operations in FABECS for the ABSE task, for the three BN curves. These results are given as a reference, since no parallelism approach is used in the development of FABECS (e.g. containers, parallelism patterns, threads, etc). *ENC* refers to the time needed to encrypt a single keyword when the function *CP-ABSE.encInd*$(PK, w, \mathbb{A}_s)$ is invoked. The word can be any single or compound word in a data set D. In this case, the search policy is the Formatted Boolean Formula using 10 bit-string attributes from which $\mathbb{A}_s$ is built. *KeyGen* refers to the time required for generating a user key by calling the function *CP-ABSE.userKeyGen*(MK, S_u) that creates the user private key SK_u from an attributes set S_u of 10 attributes, each being a bit-string. This function is very fast and slightly affected by the number of attributes but by the security level. *Trpdr* is the time required for trapdoor generation by means of the function *CP-ABSE.Trpdr*(SK_u, w_q), with a SK_u previously generated (10 attributes) and a single query word. Finally, *Search* shows the time required for executing the test for verifying keywords matching during the search process, by using the function *CP-ABSE.search*$\big(CT_w, T_u(w_q)\big)$, using encrypted words and trapdoors previously generated. As shown, words encryption and search are the most time consuming processes. Both involve the processing of $\mathbb{A}_s$, that is, the creation and processing of the LSSS and the computation of several operations in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ (Table 4).



a) Index keyword encryption

b) Private user key generation

c) Encrypted query generation

d) Searching over encrypted keywords

FIGURE 3. **Running times for main operations in FABECS.**

Users' key and trapdoors generation are much less expensive in running time in comparison with encryption and search. As expected, the running time increases proportionally to the groups size (security level).

Fig. 3 shows the behaviour of FABECS's functions for keyword encryption, user key generation and searching over encrypted data for different attributes in the policies (Fig. 3a, Fig. 3b and Fig. 3 d, respectively). In Fig. 3c is is shown the behaviour of trapdoor generation for different number of query keywords. In case of encryption and search, complexity increases considerably when the number of attributes increases, as the matrix associated to the LSSS also increases the complexity for its manipulation. This can be observed by going from 10 to 15 attributes in the access policy in both cases. The time for user keys generation really does not depend on the number of attributes but on the security level. In the case of encrypted query generation (trapdoors), the time grows linearly as the number of query keywords increases and proportionally to the security level.

We stress at this point that the time for DET-ABE in FABECS for access control over encrypted data follows a behaviour very close to keyword encryption (Fig. 3a), as the complexity in the number of operations is almost the same (see Table 4).

B. FABECS EVALUATION ON THE DATA RETRIEVAL TASK

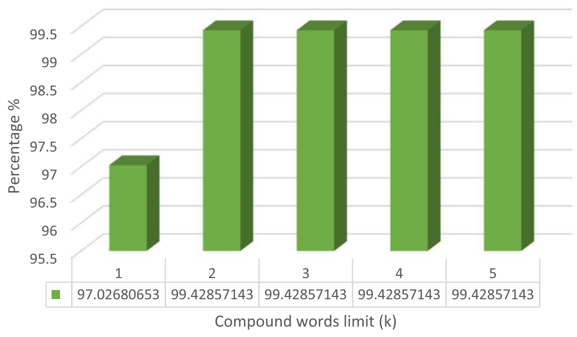
For evaluating the FABECS's performance in the retrieval task, we perform encryption and retrieval capabilities over the real-world dataset Lisa Test Collection [19], containing 6004 documents and 35 queries with relevance judgements.

C. METRICS

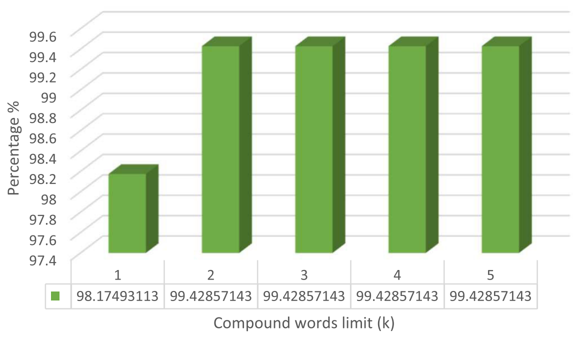
During a retrieval process, the index is consulted for each word or phrase in the query. Also, a ranking is done to select the k-most relevant documents to the query as the result of the retrieval process. Let RL be the set of documents relevant to a given query. Let RT the set of documents retrieved. The documents of interest for users that are both relevant and retrieved are $I = RL \cap RT$. The metrics that assess the quality of text retrieval are:

$$\text{Precision} = \frac{RL \cap RT}{RT} \tag{5}$$

$$\text{Recall} = \frac{RL \cap RT}{RL} \tag{6}$$



a) With words synonyms



b) Without words synonyms

FIGURE 4. Mean average precision for the retrieval task of FABECS.

TABLE 6. Queries set for the retrieval test of FABECS.

Query set	Query keywords	RL expected
Set 1	associative file, associative parallel, machine architecture	LISA3392, LISA3396
Set 2	chemical patents, chemical structures	LISA1407, LISA1431, LISA3794, LISA3795, LISA3796
Set 3	united press, news wire	LISA3401
Set 4	circulating library	LISA1046
Set 5	oral history, oral testimonies, recordings collection	LISA358, LISA1141, LISA5101
Set 6	Christian librarian, recordings collection	LISA5000, LISA5001
Set 7	user behavior, information-seeking behavior, survey of users, user survey, sdi systems, fear of talking, library user studies, users information habits	LISA576, LISA579, LISA580, LISA585, LISA1082, LISA2319, LISA2627, LISA3628, LISA4589, LISA4591, LISA5386
Set 8	consumer information, consumer advice, epies services	LISA743, LISA2151, LISA5201
Set 9	partial clustering, clusters of documents, document clustering, clusters of terms, hierarchical cluster analysis, medical clustering,	LISA2410, LISA2949, LISA3398, LISA3452, LISA3972, LISA4391
Set 10	nearest neighbor, similarity matrix	LISA2949, LISA3448

We performed ten encrypted queries sets based on the provided benchmark in the LISA dataset. The details of these queries are given in Table 6.

We consider two cases in the experimentation for the retrieval task, one on which synonyms (SYN case) are considered during the index creation and the other case when synonyms are not considered (NO-SYN case). The former case causes an increase in the time for building the secure index, the size of the index grows and as a consequence, it also increases the searching time. However, considering synonyms increases the success for retrieving RL documents but it also decreases the precision. Recall in both cases (synonyms or no synonyms usage) remains the same. Fig. 4 shows the mean average precision, which is very similar for both SYN and NO-SYN cases when using compound words of size 2-5. These results are better than using a single keyword when constructing the secure index ($k = 1$).

The results presented in Fig. 4 reveal the efficacy of FABECS to store, share and retrieve encrypted documents ensuring data confidentiality, access control over data, and access control for retrieval capabilities, according with recommended security levels.

VII. CONCLUSION

We presented for the first time a secure scheme fully based on attribute-based encryption to ensure both, the confidentiality and access control over data outsourced (in encrypted form) by data owners to the cloud and the fine-grained search control for data users when retrieving encrypted data from the cloud; we called this scheme FABECS. Through a formal analysis and experimentation, we proved the correctness and efficacy of FABECS to be used for storing, sharing and retrieval of documents in a cloud based environment. Furthermore, we provided for the first time Type-III constructions for CP-ABSE and DET-ABE as main building blocks of FABECS. This setting allows using more efficient pairing-friendly curves to achieve recommended security levels, as minimum of 128-bits. These constructions where detailed and their efficacy proved by means of experimentation, over the LISA benchmark for the retrieval task. Further work is focused in the efficiency aspect, as the results presented in this paper did not considered acceleration strategies. For example, parallelization at several levels is possible, besides the scheme is friendly enough to be deployed using parallel patterns such as the manager-worker (for processing a group of attributes at a time) or data encryption (AES on GPUs). Also, as FABECS can be realized with other efficient pairing friendly curves, experimental evaluation could consider the Barreto-Lynn-Scott Curve (BLS) that is also being promoted to be used in practical applications.

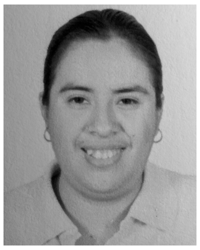
REFERENCES

[1] A. Bagherzandi, B. Hore, and S. Mehrotra, *Search over Encrypted Data*. Boston, MA, USA: Springer, 2011, pp. 1088–1093.
[2] H. Pham, J. Woodworth, and M. A. Salehi, "Survey on secure search over encrypted data on the cloud," *Concurrency Comput. Pract. Exper.*, vol. 31, p. 1– 15, Apr. 2019.
[3] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: Improved definitions and efficient constructions," in *Proc. 13th ACM Conf. Comput. Commun. Secur.*, New York, NY, USA, 2011, pp. 79–88, 2006.

[4] M. Zeng, H.-F. Qian, J. Chen, and K. Zhang, "Forward secure public key encryption with keyword search for outsourced cloud storage," *IEEE Trans. Cloud Comput.*, early access, Sep. 27, 2019, doi: 10.1109/TCC.2019.2944367.
[5] S. Kamara, C. Papamanthou, and T. Roeder, "Cs2: A searchable cryptographic cloud storage system," Microsoft Res., Redmond, WA, USA, Tech. Rep. MSR-TR-2011-58, May 2011.
[6] W. Song, B. Wang, Q. Wang, Z. Peng, W. Lou, and Y. Cui, "A privacy-preserved full-text retrieval algorithm over encrypted data for cloud storage applications," *J. Parallel Distrib. Comput.*, vol. 99, pp. 14–27, Jan. 2017.
[7] A. G. Kumbhare, Y. Simmhan, and V. Prasanna, "Designing a secure storage repository for sharing scientific datasets using public clouds," in *Proc. 2nd Int. workshop Data Intensive Comput. Clouds*, 2011, pp. 31–40.
[8] Z. Yang, J. Tang, and H. Liu, "Cloud information retrieval: Model description and scheme design," *IEEE Access*, vol. 6, pp. 15420–15430, 2018.
[9] 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.
[10] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology* (Lecture Notes in Computer Science), vol. 3494, R. Cramer, Ed. Berlin, Germany: Springer-Verlag, 2005, pp. 457–473.
[11] M. Morales-Sandoval, J. L. Gonzalez-Compean, A. Diaz-Perez, and V. J. Sosa-Sosa, "A pairing-based cryptographic approach for data security in the cloud," *Int. J. Inf. Secur.*, vol. 17, no. 4, pp. 441–461, Aug. 2018.
[12] D. Khader, "Introduction to attribute based searchable encryption," in *Communication Multimedia Security*. Berlin, Germany: Springer, pp. 131–135, 2014.
[13] T. Bouabana-Tebibel and A. Kaci, "Parallel search over encrypted data under attribute based encryption on the cloud computing," *Comput. Secur.*, vol. 54, pp. 77–91, Oct. 2015.
[14] S. Wang, D. Zhao, and Y. Zhang, "Searchable attribute-based encryption scheme with attribute revocation in cloud storage," *PLoS ONE*, vol. 12, no. 8, pp. 1–20, Aug. 2017.
[15] Y. Miao, J. Ma, X. Liu, J. Weng, H. Li, and H. Li, "Lightweight fine-grained search over encrypted data in fog computing," *IEEE Trans. Services Comput.*, vol. 12, no. 5, pp. 772–785, Sep. 2019.
[16] H. Wang, X. Dong, Z. Cao, and D. Li, "Secure and efficient attribute-based encryption with keyword search," *Comput. J.*, vol. 61, no. 8, pp. 1133–1142, Aug. 2018.
[17] Mamta and B. B. Gupta, "An efficient KP design framework of attribute–based searchable encryption for user level revocation in cloud," *Concurrency Comput., Pract. Exper.*, vol. 32, no. 18, p. 5291, Sep. 2020.
[18] H. Yin, Y. Xiong, J. Zhang, L. Ou, S. Liao, and Z. Qin, "A key-policy searchable attribute-based encryption scheme for efficient keyword search and fine-grained access control over encrypted data," *Electronics*, vol. 8, no. 3, p. 265, Feb. 2019.
[19] H. Z. Rad, "Lisa test collection," Sheffield Univ., Sheffield, U.K., Tech. Rep., 2019.
[20] 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.*, 2006, pp. 89–98.
[21] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Secur. Privacy*, Jul. 2007, pp. 321–334, 2007.
[22] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 1, pp. 222–233, Jan. 2014.
[23] S. Wang, J. Ye, and Y. Zhang, "A keyword searchable attribute-based encryption scheme with attribute update for cloud storage," *PLoS ONE*, vol. 13, no. 5, pp. 1–19, May 2018.
[24] W. Guo, X. Dong, Z. Cao, and J. Shen, "Efficient attribute-based searchable encryption on cloud storage," *J. Phys., Conf. Ser.*, vol. 1087, Sep. 2018, Art. no. 052001.
[25] 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.*, Apr. 2019, pp. 146–155.
[26] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Public Key Cryptography*. Berlin, Germany: Springer, 2011, pp. 53–70.
[27] Z. Liu, Z. Cao, and S. D. Wong. (2010). *Efficient Generation of Linear Secret Sharing Scheme Matrices From Threshold Access Trees*. https://eprint.iacr.org/2010/374
[28] D. Boneh, "Pairing-based cryptography: Past, present, and future," in *Advances Cryptoligy*. Berlin, Germany: Springer, 2012, p. 1.
[29] S. D. Galbraith, K. G. Paterson, and N. P. Smart, "Pairings for cryptographers," *Discrete Appl. Math.*, vol. 156, no. 16, pp. 3113–3121, Sep. 2008.
[30] S. Galbraith, (2014). *New Discrete Logarithm Records, and the Death of Type 1 Pairings*. [Online]. Available: https://ellipticnews.wordpress.com/2014/02/01/new-discrete-logarithm-records-and-the-death-of-type-1-pairings/
[31] L. Wang, X. Shen, J. Li, J. Shao, and Y. Yang, "Cryptographic primitives in blockchains," *J. Netw. Comput. Appl.*, vol. 127, pp. 43–58, Feb. 2019.
[32] R. Blum and T. Bocek, "Superlight – a permissionless, light-client only blockchain with self-contained proofs and bls signatures," in *2019 IFIP/IEEE Symp. Integr. Netw. Service Manage. (IM)*, pp. 36–41, 2019.
[33] S. L. M. Paulo Barreto and M. Naehrig, "Pairing-friendly elliptic curves of prime order," in *Selected Areas Cryptography*. Berlin, Germany: Springer, 2006, pp. 319–331.
[34] Bluekrypt. (2020). *Cryptographic Key Length Recommendation*. [Online]. Available: https://www.keylength.com/en/
[35] R. Barbulescu and S. Duquesne, "Updating key size estimations for pairings," *J. Cryptol.*, vol. 32, no. 4, pp. 1298–1336, Oct. 2019.
[36] S. Chatterjee and A. Menezes, "On cryptographic protocols employing asymmetric pairings the role of ψ revisited," *Discrete Appl. Math.*, vol. 159, no. 13, pp. 1311–1322, 2011.
[37] B. Rosenberg, *Handbook Financial Cryptography Security*. Boca Raton, FL, USA: CRC Press, 2010.
[38] M. Morales-Sandoval and A. Diaz-Perez, "DET-ABE: A Java API for data confidentiality and fine-grained access control from attribute based encryption," in *Proc. Int. Conf. Inf. Secur. Theory Pract.*, Heraklion, Greece, Aug. 2015, pp. 104–119.
[39] G. A. Vazquez-Martinez and J. L. Gonzalez-Compean, "Cloudchain: A novel distribution model for digital products based on supply chain principles," *Int. J. Inf. Manage.*, vol. 39, p. 90–103, Apr. 2018.
[40] J. L. Gonzalez-Compean, O. Telles, I. Lopez-Arevalo, M. Morales-Sandoval, V. J. Sosa-Sosa, and J. Carretero, "A policy-based containerized filter for secure information sharing in organizational environments," *Future Gener. Comput. Syst.*, vol. 95, pp. 430–444, Jun. 2019.
[41] A. De Caro and V. Iovino, "JPBC: Java pairing based cryptography," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun. 2011, pp. 850–855.
[42] A. Menezes, P. Sarkar, and S. Singh, "Challenges with assessing the impact of nfs advances on the security of pairing-based cryptography," in *Paradigms Cryptology*. Cham, Switzerland: Springer, 2017, pp. 83–108.
[43] *Cryptographic Techniques Based on Elliptic Curves—Part 5: Elliptic Curve Generation*, Standard ISO/IEC 15946-5:2009, 2009.

MIGUEL MORALES-SANDOVAL received the B.Sc. degree from the Autonomous University of Puebla, in 2002, and the M.Sc. and Ph.D. degrees from the National Institute for Astrophysics, Optics, and Electronics, Mexico, in 2004 and 2008, respectively.

He is currently a Computer Science Researcher at CINVESTAV Tamaulipas, with research lines focused on information systems, data security, cryptography, and embedded systems. He has actively participates with graduate programs in computer science and information technology and the development of research projects. His current research interests include development of software engineering, encrypted data retrieval mechanisms, and security in the Internet of Things domain and cloud. He served as a Reviewer for several journals and conferences and an Associate Editor.



MELISSA HINOJOSA CABELLO received the B.S. degree in information technologies engineering from the Polytechnic University of Victoria, in 2018. She is currently pursuing the M.Sc. degree in engineering and computer technologies with the Center for Research and Advanced Studies (CINVESTAV), Mexico. Her research interests include information security, applied cryptography, information retrieval, data mining, cloud storage, and network security.

HEIDY MARISOL MARIN-CASTRO received the B.Sc. degree from the Autonomous University of Puebla, in 2004, the M.Sc. degree in computer science from the National Institute for Astrophysics, Optics and Electronics (INAOE), in 2008, and the Ph.D. degree in computer science from the Center for Research and Advanced Studies of the National Polytechnic Institute (CINVESTAV), Mexico, in 2014. She is currently a Conacyt Researcher with the Information Technology Department, Autonomous University of Tamaulipas, Mexico. Her research interests include web data management, databases, data mining, and information retrieval.

JOSE LUIS GONZALEZ COMPEAN received the Ph.D. degree in computer architecture from the Universitat Politecnica de Catalunya (UPC), Barcelona, in 2009. He was a Visiting Professor with the Universidad Carlos III de Madrid, Spain. He was also a Researcher with CINVESTAV, Mexico. His research interests include cloud-based storage systems, linguistic archival systems, federated storage networks, design of fault-tolerant, adaptability and availability strategies, task scheduling, and storage virtualization.