Multi-client Sub-Linear Boolean Keyword Searching for Encrypted Cloud Storage with Owner-enforced Authorization

Kai Zhang, Mi Wen, *Member, IEEE,* Rongxing Lu *Senior Member, IEEE,* and Kefei Chen

Abstract—To date, cloud computing has emerged as a primary utility for providing remote data storage services for users, since users can thus be relieved from cumbersome document maintenance. Despite of the benefits brought by data outsourcing, the unexpected data breaches raise concerns about data confidentiality and privacy. To deal with this, a straightforward and convincing strategy is to encrypt data before outsourcing them to the cloud. However, securely sharing and searching over outsourced encrypted data has turned into a challenge due to the hindrance led by data encryption. To address the challenge, this paper proposes a new highly-scalable searchable encryption scheme for encrypted cloud storage. The scheme achieves sub-linear Boolean keyword searching, and moreover allows the data owner to authorize which clients could search or access the documents in cloud. Technically, we revisit searchable symmetric encryption primitive by non-trivially combining it with a novel access control technique, and build inverted index data structure for both attribute-based access control and sub-linear search process. Furthermore, we introduce a formalized security definition for the system, and prove its security in the simulation-based security model. Finally, we conduct a couple of experiments over a representative real-world dataset to show practicality.

Index Terms—Cloud Security, Searchable Encryption, Access Control, Keyword Search, Boolean Query.

1 Introduction

Cloud services have taken a promising trend in providing ubiquitous and on-demand access to a shared pool of configurable storage/computing resources [1] for users in recent years, thereby users are increasingly outsource personal documents to cloud server. The personal documents usually involve some sensitive information (such as passwords, identifiable information and health information), there have been many incidents of outsourced data leaking to adversarial attackers or to public [2]. As discovered by academic [3] and insustry [4], data breaches are considered to be primary security risks in cloud computing (untrusted outsiders) over the past decade. Hence, data confidentiality and privacy have been a primary area of focus.

To deal with the security challenge, it has been strongly suggested to employ an "encryption-before-outsourcing" mechanism: a data owner encrypts his/her documents before outsources them to cloud. Nevertheless, simple encryption of outsourced data will certainly hinder the efficiency of data processing compared to plaintext data domain. Hence, searching documents archive and sharing documents in cloud have turned into an arduous challenge. Therefore, the focus of this work, efficient document retrieval/sharing and the privacy preservation in data usage should not be conflicting goals.

A Motivation Case. Considering an example of documents retrieval and sharing in encrypted cloud storage. Assume the data warehouse of a company is hosted in cloud, there are three staffs Bob, Charles and David from different departments collaboratively processing routine work based on cloud., i.e., searching documents archive and sharing documents. To preserve data confidential and privacy, the outsourced data are encrypted before outsourcing to the cloud. For instance, a financial office staff Bob encrypts a set of financial reports where these reports origins from North America region during 2014 to 2018, and uploads them to cloud. At the same time, he determines that only the colleagues in Financial Office of the North America division could access these files. Here, the owner-enforced authorization policy for which staffs could access these files is formulated as *"Financial Office" AND "North America Division"*. To efficiently retrieve target documents from massive documents stored in cloud, a staff carries out documents searching with submitting a search query expression based on documents' keywords.

Hence, a manager Charles of the financial office in Asia-Pacific division is certainly unable to access these reports, since his description attributes set (*"Financial Office"*, *"Asia-Pacific Division"*) fails to satisfy the authorization policy. Nevertheless, a VP David of the finical office in North America Division is accepted by the authorization policy. Moreover, when to check the 2017-year or 2018-year annual financial reports except for advertising expenses, David submits an searching expression: *"North America Division" AND ("2017-Year" OR "2018-Year") NOT "Advertising"*, and

- *Kai Zhang, Mi Wen and Kefei Chen are with College of Computer Science and Technology, Shanghai University of Electric Power, Shanghai, China. (E-mail: kzhang@shiep.edu.cn, wenmi2222@163.com, kfchen@hznu.edu.cn).*
- *Rongxing Lu is with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: rlu1@unb.ca).*

Manuscript received April 19, 2005; revised August 26, 2015. This work was supported by National Natural Science Foundation of China (61802248, 61872230, 61572311, U1705264) and the "Chenguang Program" supported by Shanghai Education Development Foundation and Shanghai Municipal Education Commission (No.18CG62).

generates a corresponding search token to cloud. After a check on whether David satisfies the authorization policy, the cloud uses the search token to search and return financial reports whose keywords set matches David's search query expression.

We can observe that the data owner enforces an attribute-based authorization policy for determining which staffs have documents access/searching rights. Therefore, only staffs whose description attributes satisfy authorization policy could retrieve target documents, where the associated keywords match a queried boolean keyword search formula. Hence, a problem from this example arises naturally: *is there a secure and efficient multi-user documents retrieval and sharing system that supports expressive search patterns and access control sharing policy* ?

Single Keyword Search over Encrypted Data. Song et al. [15] introduced the searchable encryption (SE) primitive that allowed a remote server to search over encrypted data with an authorized search token from clients. This enables efficient searching over encrypted documents and makes the server learn nothing about plaintext data. Nevertheless, [15] with the following work [16], [17], [18] only considered symmetric case (single data owner and single client), which cannot be deployed in real-world. This is because documents are usually shared across a couple of clients rather simple "one-to-one" data sharing scene in public cloud. To accommodate multiple clients supporting fine-grained authorization, the attribute-based searchable encryption (ABSE) primitive was introduced in [5], which combined attribute-based encryption with other cryptographic primitives (e.g., proxy based re-encryption). Recently, the ABSE works have been intensively researched [6], [7], [8].

Generally, a data owner in ABSE systems is able to specify an authorization policy and thus determine which clients could search over encrypted data in cloud. Nevertheless, the query clients can only insert just one keyword (e.g., "w_1") into a query pattern, and unfortunately retrieve all documents associated with the keyword "w_1". As can be seen, single keyword searching SE systems bring about forced search expression for query clients, which still suffer from expensive linear searching costs $\mathcal{O}(\#\mathsf{doc})$ (where "#doc" is the total number of all outsourced documents). When extended to process a conjunctive keyword search (e.g. "$w_1 \wedge w_2 \wedge \cdots \wedge w_q$") across different keywords, the searching costs in ABSE systems [5], [6], [7], [8], [9] are still in relation with the total number of all outsourced documents. Particularly for highly-scalable documents outsourcing, such ABSE systems may not be directly deployed due to unacceptable search costs and/or limited search expressions. Hence, more efficient and expressive searchable encryption systems are highly appreciated.

Boolean Keyword Search over Encrypted Data. Quite recently, Cash et al. [11] designed a searchable symmetric encryption (SSE) system supporting boolean queries, that is, $w_1 \wedge \psi(w_2, \cdots, w_q)$ where ψ is a boolean formula over keywords $(w_2, \cdots, w_q)$. The searching costs are only associated with the least frequent term in the conjunction and thus reduced to a sub-linear complexity (independent of the total number of all stored documents). And the work [11] used inverted-index data structure to organize documents. Given a set of documents invert-indexed with a keywords set (w_1, w_2, w_3, w_4), the boolean keyword search pattern "$w_1 \wedge (\neg w_2 \vee w_3 \wedge w_4)$" supports negations, disjunctions, threshold etc. formulas for keywords searching, which certainly involves conjunctive and single keyword search.

However [11] only considered a symmetric situation, that is, a same secret key was used in both documents encryption and documents searching process. This greatly limits its wide deployments in multi-client collaborative cloud services. Following it, the works [12], [13] considered richer outsourcing scenarios for SSE schemes, supporting single data owner and multiple query clients.

Motivation and Utility. So far, existing SE-based solutions either only considered owner-enforced authorization but suffer from strong search expression and/or expensive search costs; or only achieve sub-linear Boolean keyword search but not support fine-grained authorization towards multiple clients. Hence, *a searchable encryption system that supports both sub-linear boolean query and fine-grained authorization across multiple clients simultaneously* is still an unaddressed problem.

1.1 Our Contributions

This work proposes a novel multi-client searchable encryption scheme, which achieves sub-linear boolean query and supports owner-enforced attribute-based authorization across multiple clients. Formally, the key features of the SE system are summarized as follows, where Table 1 also shows a functionality comparison with related works.

1) **Supporting owner-enforced attribute-based authorization for multiple clients data sharing.** The data owner encrypts the outsourced documents under a specified attribute-based authorization policy, in such a way that, it can non-interactively determine which clients could search or access the stored documents in cloud. This authorization in our SE system is similar to the attribute-based control paradigm in ABE primitive, where all clients are depicted by a description attributes set and the fine-grained authorization policy is an "AND" formula over attributes. We note that the attribute-based authorization paradigm is not employed in a trivial way, since a subtlety combination between a boolean keyword search and an "AND"-gate access policy has to be built in indispensable (c.f. technical details in Section 4.1).
2) **Supporting Boolean keyword search with sub-linear complexity.** For documents searching, a query client submits a boolean expression of keywords and thus generates a search token for cloud. With the search token, the cloud returns target documents under a condition: not only the documents' associated keywords match the submitted boolean expression but also the attributes of a query client satisfy owner-enforced authorization policy. To process a boolean query with sub-linear search costs (independent of the total number of stored documents) for cloud, we introduce a new inverted index method for attribute-based authorization paradigm, to deal with both documents encryption and attribute-based authorization tuples in a two-fold way (c.f. technical details in Section 4.1).

TABLE 1
Feature Comparison with Related Works

Works	Boolean Query	Search Authorization[a]	Multiple clients[b]	Sub-linear
IEEE: INFOCOM'14 [5], TIFS'15 [6], TPDS'16 [7] TSC'17 [8], TPDS'18 [9], TDSC'19 [10]	✗	✓	✓	✗
CRYPTO'13 [11], ACM CCS'13 [12] ESORICS'16 [13], EUROCRYPT'17 [14]	✓	✗	✗	✓
Our work	✓	✓	✓	✓

[a]Search authorization: *A data owner gives attribute-based search authorization on which clients can search its encrypted data.*
[b]Multiple clients: *The scheme considers multiple clients model (multiple data owners* vs. *multiple query clients).*

Moreover, we introduce a formal security definition for the SE system and moreover give a rigorous simulation-based security analysis. To further illustrate practical usability, we conduct a couple of experiments over a real-world dataset [19] (Enron: http://www.cs.cmu.edu/~./enron/) for the proposed system and related work.

Organization. Notations and background knowledge are described in Section 2. Section 3 introduces formal system model, security guarantee model and design goals of this work. We provide a main construction supporting conjunctive keyword search and its security analysis in Section 4 and Section 5, then extend it to process boolean query in Section 6. A comparison between related work and a simulated experiment are given in Section 7. Section 8 concludes this work.

2 Background Knowledge

The notations that used in this paper are shown in Table 2.

TABLE 2
Notations

Notation	Meaning
κ	A security parameter.
$[n]$	$\{1, 2, \cdots, n\}$.
$\mathbf{A}$	An array.
$\mathbf{A}[i]$	The i-th element of $\mathbf{A}$.
$\lvert\mathbf{A}\rvert$	The length of $\mathbf{A}$.
att	An attribute.
Γ	An attribute set $\Gamma := (\mathsf{att}_1, \mathsf{att}_2, \cdots, \mathsf{att}_n)$.
$\mathcal{N}$	An attribute universe.
ind	The indice of a document.
ω	A keyword.
W	A set of keywords $W := (\omega_1, \omega_2, \cdots, \omega_q)$.
Doc	A document Doc is labeled with (ind, W_{ind}).
$\mathcal{ACC}$	An access policy.
id	A client's identity.
DB	An outsourced database.
$\mathsf{DB}[w]$	The indices of documents that associated with keyword w in DB.
$\mathsf{DB}[w, id]$	The indices of documents that the access policy can be satisfied by client id in $\mathsf{DB}[w]$.
$id \vDash \mathcal{ACC}$	The client id can satisfy the access policy $\mathcal{ACC}$
s-term	The least frequent keyword among the keywords in a query
xterm	Any queried keyword in a query.

2.1 Inverted Keyword Search Index

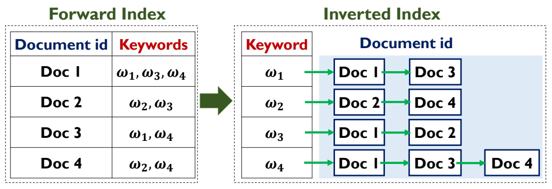


Fig. 1. Keyword Search Index Data Structure

Fig.1 lists two "keyword search index" data structures in search engines for finding target documents where keyword occurs: *forward index* and *inverted index*, in which each document is labeled with a set of keywords and indice pair. The traditional forward index needs to list all documents with all related keywords based on existing keywords list for each document, which brings about expensive time-consuming costs and memory storage for documents searching. Instead, as the most popular data structure in search engines, the inverted index builds a set of pointers to documents for each same keyword. As a result, search engines could search documents with a search token to reduce keywords into core meaning, which efficiently reduce time-consuming costs and memory storage.

To deal with encrypted domain for search engines, the inverted index are usually used in symmetric key cryptosystems and cannot be directly deployed in our multiple clients scenario. Aiming to achieve owner-enforced attribute-based authorization towards multiple clients with boolean keyword search, we introduce a new encrypted inverted index utilizing attribute-based access control technique as [20]. Hence, both fine-grained authorization and sub-linear boolean keyword search with inverted index are achieved simultaneously.

2.2 Keyword Dictionary

A keyword dictionary δ maintains a couple of tuples (w, c), where w is a keyword and c is a counter. This is used to extend a static SE to a dynamic SE with supporting database changes (i.e., documents adding, documents deleting and documents modifying). Generally, the δ has the following two functions:

- $c \leftarrow \mathsf{Get}(\delta, w)$: Outputs the counter of a keyword w. If w does not exist in δ, outputs 0 as the answer.
- $\mathsf{Update}(\delta, w, c)$: Updates the counter of a keyword w to c. If w does not exist in the dictionary, inserts the tuple (w, c) into it.

2.3 Cryptographic Assumptions and Primitives

Definition 1 (PRF [21]). A pseudo-random function (PRF) F is a polynomial time computable function that cannot be distinguished from random functions F' by any probabilistic polynomial time (PPT) adversary $\mathcal{A}$. That is, for any PPT adversary $\mathcal{A}$, the advantage is defined as

$$\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{A},F}(\kappa) = |\Pr[\mathcal{A}^{F(K,\cdot)}(1^\kappa)] - \Pr[\mathcal{A}^{F'(\cdot)}(1^\kappa)]|,$$

where $K \xleftarrow{\$} \{0,1\}^\kappa$. The F is a PRF if $\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{A},F}(\kappa)$ is negligible for any PPT adversary $\mathcal{A}$.

Definition 2 (DDH Assumption). Let $\mathbb{G}$ be a cyclic group with a prime order p, the Decisional Diffie-Hellman problem is to distinguish (g, g^a, g^b, g^{ab}) from (g, g^a, g^b, g^r), where g is an element randomly selected from $\mathbb{G}$ and a, b, r are randomly selected from $\mathbb{Z}_p$. For any PPT distinguisher $\mathcal{D}$, the advantage is defined as

$$\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{D},\mathbb{G}}(\kappa) = \left|\Pr[\mathcal{D}(g, g^a, g^b, g^{ab})] - \Pr[\mathcal{D}(g, g^a, g^b, g^r)]\right|.$$

The DDH assumption says $\mathsf{Adv}^{\mathsf{DDH}}_{\mathcal{D},\mathbb{G}}(\kappa)$ is negligible in κ for any PPT distinguisher $\mathcal{D}$.

Definition 3 (SXDH Assumption). Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be three cyclic groups with a same prime order p, and an efficient asymmetric bilinear pairing e: $\mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ satisfying: (1) *Non-degenerate*: if g_1 is a generator of $\mathbb{G}_1$ and g_2 is a generator of $\mathbb{G}_2$, then $e(g_1, g_2)$ is a generator of $\mathbb{G}_T$. (2) *Bilinear*: $\forall a, b \in \mathbb{Z}_p$, we have $e(g_1^a, g_2^b) = e(g_1^b, g_2^a) = e(g_1, g_2)^{ab}$. The Symmetric eXternal Diffie-Hellman [22] assumption says that the three groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are DDH groups.

Definition 4 (CP-ABE [23]). In general, a ciphertext-policy attribute-based encryption consists of four algorithms:

- ABE.Setup: The setup algorithm inputs a secure parameter and an attribute universe, and outputs a public parameter $\mathsf{mpk}_{\mathsf{ABE}}$ and a master key $\mathsf{msk}_{\mathsf{ABE}}$.
- ABE.Enc: The encryption algorithm takes as input $\mathsf{mpk}_{\mathsf{ABE}}$, a message M and an access structure $\mathcal{ACC}$, it generates a ciphertext CT.
- ABE.Extract: The key generation algorithm takes as input $\mathsf{msk}_{\mathsf{ABE}}$ and a set of attributes set Γ of a client, outputs a private key $\mathsf{pvk}_{\mathsf{ABE}}$ for the client.
- ABE.Dec: The decryption algorithm inputs a CT and a $\mathsf{pvk}_{\mathsf{ABE}}$ associated with an attribute set Γ, it decrypts the ciphertext if and only if Γ satisfies the access policy $\mathcal{ACC}$ that involved in CT.

3 Problem Formulation

3.1 System Model

There are three different entities in our SE system as shown in Fig. 2: *Authority, Cloud Server* and *Clients*. In the system, the authority is a trusted party that maintains system and deals with registration for different parties; the semi-honest cloud server honestly runs algorithms and provides data outsourcing storage/searching services; the (multiple) clients consist of multiple data owners and multiple query clients, where each client can outsource personal documents to cloud and search documents that are contributed from other clients. Once system initialized by the authority, any legal client receives a secret key by registering in the system with submitting his/her attributes set to the authority. Then, a data owner enforces an authorization policy to encrypt his/her documents and outsources the encrypted documents to cloud; to process documents searching, a query client generates a search token for search query and sends it to cloud. Finally, the cloud uses the search token to search over the encrypted database and returns corresponding documents. Note that there is one condition should be satisfied, i.e., not only the document's keywords set matches search query expression but also the owner-enforced authorization policy accepts the attributes set of a query client. Especially, the system running routine can be seen in Fig. 2.

3.2 Function Definition

- $\mathsf{Setup}(1^\kappa, \mathcal{N}) \to (\mathsf{PP}, \mathsf{MK})$: With a security parameter κ and an attribute universe description $\mathcal{N}$, this algorithm outputs a public parameter PP and a master secret key MK for the system.
- $\mathsf{KeyGen}(\Gamma, \mathsf{MK}) \to \mathsf{sk}$: The key generation algorithm inputs an attributes set Γ of a client and MK, and outputs a secret key sk for the client.
- $\mathsf{Encrypt}(\mathsf{PP}, \mathsf{Doc}, \mathsf{sk}, \mathcal{ACC}) \to (\mathsf{EDB}, \mathsf{XSet})$: The encryption algorithm inputs PP, a document Doc, a sk of a client who encrypts a document and an access policy $\mathcal{ACC}$, and outputs $(\mathsf{EDB}, \mathsf{XSet})$ as the searchable ciphertext.
- $\mathsf{TrapGen}(Q, \mathsf{sk}) \to \mathsf{Token}$: The search token generation algorithm accepts a client's secret key sk along with a search query Q as input, and outputs a search token Token for the query.
- $\mathsf{Search}(\mathsf{Token}) \to R$: Input a search token Token, the search algorithm outputs encrypted search results R.
- $\mathsf{Retrieve}(R, \mathsf{sk}) \to \mathsf{Documents}$: This retrieval algorithm inputs an encrypted search results R and a client's secret key sk, and returns target original documents.

3.3 Security Guarantee Model

Different from previous sub-linear boolean keyword search SE schemes [11], [12], [13], security guarantees against two types of honest-but-curious adversaries should be seriously considered: one is an adversary server and one is the colluded clients. Although the cloud honestly runs the designed protocols, it may snoop privacy information involved in stored documents during searching; and a client may collude with other clients to access/search documents beyond permission. Same as in [11], [12], [13], we assume that there is no collusion between cloud and clients to get privacy information of documents and/or during searching.

3.3.1 Security against adversary server

Recall the security definition in [11] under the simulation-based security [24], where the view of adversary can be simulated by given only permitted leakage during an adaptive attack, in the sense that, the adversary cannot learn anything beyond the permitted leakage. The security is parameterized by a leakage function $\mathcal{L}$ that describes some information being known to the adversary during each operation. Hence, it is possible that the outputs of a simulator have the same distribution with $\mathcal{L}$ as in real-world. We

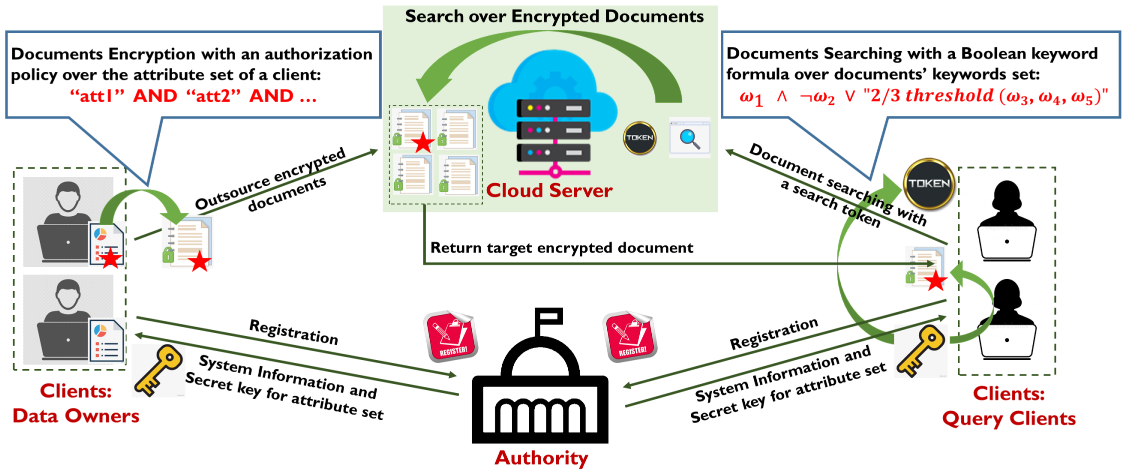

Search over Encrypted Documents
Documents Encryption with an authorization policy over the attribute set of a client: "att1" AND "att2" AND ...
Documents Searching with a Boolean keyword formula over documents' keywords set: $\omega_1 \wedge \neg\omega_2 \vee$ "2/3 threshold $(\omega_3, \omega_4, \omega_5)$"
Outsource encrypted documents
Cloud Server
Document searching with a search token
Return target encrypted document
Registration
System Information and Secret key for attribute set
Registration
System Information and Secret key for attribute set
Clients: Data Owners
Authority
Clients: Query Clients

Fig. 2. System Model of Our Multi-client Sub-Linear Boolean Keyword Searching for Encrypted Cloud Storage with Owner-enforced Authorization

remark that original documents retrieval is not necessarily modeled as that in [11].

Definition 5. Let $\prod$ be a scheme described in Section 3.2, $\mathcal{L}$ be a leakage function, $\mathcal{A}$ be a stateful semi-honest adversary and $\mathcal{S}$ be a simulator, we define the security via the following two experiments $\mathsf{Real}^{\Pi}_{\mathcal{A}}(\kappa)$ and $\mathsf{Ideal}^{\Pi}_{\mathcal{A},\mathcal{S}}(\kappa)$.

$\mathsf{Real}^{\Pi}_{\mathcal{A}}(\kappa)$: $\mathcal{A}(1^\kappa)$ repeatedly chooses an encryption tuple $(\mathsf{Doc}, \mathcal{ACC}, id)$[1] or a searching query tuple (Q, id)[2]. For an encryption tuple $(\mathsf{Doc}, \mathcal{ACC}, id)$ chosen by the adversary, this game runs $\mathsf{Encrypt}(\mathsf{PP}, \mathsf{Doc}, sk_{id}, \mathcal{ACC})$ to generate $(\mathsf{EDB}, \mathsf{XSet})$ and gives it to $\mathcal{A}$; otherwise, this game runs $\mathsf{Token} \leftarrow \mathsf{TrapGen}(Q, sk_{id})$ and $R \leftarrow \mathsf{Search}(\mathsf{Token})$, then gives the transcript to $\mathcal{A}$. In the end $\mathcal{A}$ returns a bit as an output of this game.

$\mathsf{Ideal}^{\Pi}_{\mathcal{A},\mathcal{S}}(\kappa)$: This game initializes two empty lists $\mathbf{d}$ and $\mathbf{q}$, and initializes two counters $i = 1$ and $j = 1$. $\mathcal{A}(1^\kappa)$ repeatedly chooses an encryption tuple $(\mathsf{Doc}, \mathcal{ACC}, id)$ or a searching query tuple (Q, id). If it chooses an encryption tuple, $\mathcal{A}$ records $(\mathsf{Doc}, \mathcal{ACC}, id)$ as $\mathbf{d}[i]$ and increases i, this game runs $\mathcal{S}(\mathcal{L}(\mathbf{d}, \mathbf{q}))$ to get $(\mathsf{EDB}, \mathsf{XSet})$, and gives it to $\mathcal{A}$. Otherwise, this game records (Q, id) as $\mathbf{q}[i]$, increases i and runs $\mathcal{S}(\mathcal{L}(\mathbf{d}, \mathbf{q}))$ to output a transcript to $\mathcal{A}$. In the end $\mathcal{A}$ returns a bit as an output of this game.

We say the Π is $\mathcal{L}$-semantically secure against an adaptive adversary if there exists an algorithm/simulator $\mathcal{S}$ such that

$$\Pr[\mathsf{Real}^{\Pi}_{\mathcal{A}}(\kappa) = 1] - \Pr[\mathsf{Ideal}^{\Pi}_{\mathcal{A},\mathcal{S}}(\kappa) = 1] \leq negl(\kappa)$$

where $negl(\cdot)$ is a negligible function under a security parameter κ. This security guarantee lets the adversary be unable to learn any knowledge beyond the permitted leakage information, otherwise, it successfully distinguishes two games with a non-negligible advantage.

3.3.2 Security against colluded clients

In the system, the clients may collude each other to search/access the documents beyond permission, thus a valid search token is successfully computed for making the merging attribute set satisfies the aiming challenged authorization policy. Therefore, the security against collusion attacks should be considered, where the game is sketched by the following.

- **Init**. The challenger runs Setup to initialize the game and returns public parameter to an adversary $\mathcal{A}$.
- **Key extraction**. Receiving a secret key query for an attribute set Γ from $\mathcal{A}$, the challenger runs KeyGen to return a secret key associated with Γ to $\mathcal{A}$.
- **Output**. $\mathcal{A}$ outputs a search token where the involved attribute set has not been queried before. If the search token is valid, the challenger outputs 1.

We say a scheme Π is secure against colluded clients if the challenger outputs 1 in the above game with a negligible probability. This security guarantee lets colluded clients be unable to search/access documents beyond permission according to the owner-enforced authorization policy.

3.4 Design Goals

The design goals of our system are formalized as follows:

1) Multiple clients (including data owners and query clients). To accommodate multiple clients, each client could search outsourced documents that contributed from multiple data owners.

2) Fine-grained authorization towards document searching/accessing. The data owner can non-interactively determine which clients search over his/her documents based on owner-enforced attribute-based authorization policy across multiple clients who are described by a set of attributes.

3) Boolean keyword search expression. In the system, a query client is able to issue practical boolean queries that have been intensively deployed in searching engines.

4) Sub-linear Search Costs. The search costs are independent of the total number of all stored documents, and only related to the least frequent term in the conjunction.

5) Running securely. During the process of outsourced encrypted documents sharing and searching query, no other knowledge are leaked to the cloud/outsiders.

1. For an encryption tuple $(\mathsf{Doc}, \mathcal{ACC}, id)$, $\mathcal{ACC}$ is a access policy and id is an identity of the client who encrypts a document Doc.
2. For a search tuple (Q, id), id is an identity of the client who issues a query Q.

4 The Proposed SE System Dealing With Conjunctive Keyword Search

In this section, we give a secure and efficient multi-client SE system supporting conjunctive keyword search, and continue to enable boolean queries in Section 6.

4.1 High-level Idea

To realize attribute-based authorization over dynamic documents searching across multiple clients, we should seriously deal with the following technical challenges: dynamic documents searching and attribute-based authorization along with document conjunctive keyword searching.

For one thing, with employing the keyword dictionary structure $\delta := (c, w)$ inspired from [11], [21] for extending a static searchable symmetric encryption construction to dynamic SSE one, we can deal with dynamic databases operations (where documents can be added, deleted and modified). As c is a keyword-specific counter, the cloud server can hence search for a keyword w by first searching in the static case and then re-computing all labels corresponding to w in δ, since a w-specific key is provided by the client and a running counter.

For another thing, the attribute universe for describing clients is assumed to be $\mathcal{N} = \{\mathsf{att}_1, \cdots, \mathsf{att}_n\}$ whose scale is n. Among the attribute universe, each attribute att_i has two values: att_i^+ denotes that a client has this description attribute and $\neg\mathsf{att}_i$ indicates not having this description attribute for a client. Inspired by [7], [20], we separately define and use two sets of variables $\{(x_k, y_k)\}$ or $\{(x_{k+n}, y_{k+n})\}$ to grant corresponding secret keys in the KeyGen algorithm, which is determined by whether Γ (a non-empty subset of $\mathcal{N}$) has the attribute att_i in $\mathcal{N}$. While in the Encrypt algorithm, we let an attribute $\mathsf{att}_i \in I$ be att_i^+ or $\neg\mathsf{att}_i$ since an "AND" gate access structure $\mathcal{ACC}$ is also designed over a set of attributes $I \subseteq \mathcal{N}$. Hence, we can simultaneously build a same conjunctive normal form (CNF) operation over both conjunctive keyword searching "$(w_1 \wedge w_2 \wedge \cdots \wedge w_q)$" towards different keywords $\omega_1, \cdots, \omega_q$, and "AND"-gate attribute-based authorization $\mathcal{ACC} = \bigwedge_{\mathsf{att}_i \in I} \mathsf{att}_i$ towards attribute universe $\mathcal{N} = \{\mathsf{att}_1, \cdots, \mathsf{att}_n\}$.

4.2 Formal Construction

In Fig. 3, we present the formal construction of our document searching system. For a document Doc labeled with a set of $\{(ind, W_{ind})\}$, where ind is an unique indice of document and W_{ind} is the associated keywords set. And assume the employed two functions $c \leftarrow \mathsf{Get}(\delta, w)$ and $\mathsf{Update}(\delta, w, c)$ of δ are either encrypted and stored in cloud server or maintained by a local server in real world. To encrypt a document Doc into an encrypted version, we choose to use a symmetric key encryption algorithm (e.g., AES) with a secret key K_{ind} to realize. Furthermore, let attribute universe be $\mathcal{N} = \{\mathsf{att}_1, \cdots, \mathsf{att}_n\}$ where each attribute has two values: att_i^+ denotes the client has this attribute and $\neg\mathsf{att}_i$ denotes the opposite.

To have a better understanding on our construction, we recall Table 2 for notation descriptions about used and generated variables/parameters, and present the formal procedure details of the system in Fig. 3.

Correctness Guarantee. Provided the owner-enforced authorization policy $\mathcal{ACC}$ can be satisfied, the correctness guarantee for document searching Search holds as follows:

$$
\begin{aligned}
& e(v^{H(w_j)/z_c}, e_2) \cdot e(\sigma_j, e_1) \\
&= e(v^{H(w_j)/z_c}, x_I^{z_c \cdot xind}) \cdot e((\textstyle\prod_{\mathsf{att}_i \in I} \sigma_i)^{H(w_j)/z_c}, g_2^{z_c \cdot xind}) \\
&= e(v^{H(w_j)}, (\prod_{\mathsf{att}_i \in I} x_{\mathsf{att}_i})^{xind}) \cdot e((\prod_{\mathsf{att}_i \in I} t_{i'} v^{r_{i'}})^{H(w_j)}, g_2^{xind}) \\
&= e(v^{H(w_j)}, g_2^{-xind \cdot \sum_{\mathsf{att}_i \in I} r_{i'}}) \cdot e(v^{H(w_j) \cdot \sum_{\mathsf{att}_i \in I} r_{i'}}, g_2^{xind}) \\
&\quad \cdot e((\textstyle\prod_{\mathsf{att}_i \in I} t_{i'})^{H(w_j)}, g_2^{xind}) \\
&= (\textstyle\prod_{\mathsf{att}_i \in I} e(t_{i'}, g_2))^{H(w_j) \cdot xind} = y_I^{H(w_j) \cdot xind}.
\end{aligned}
$$

In which, if $\mathsf{att}_i = \mathsf{att}_i^+$ then $t_{i'} = t_i$ and $r_{i'} = r_i$; else if $\mathsf{att}_i = \neg\mathsf{att}_i$ then $t_{i'} = t_{i+n}$ and $r_{i'} = r_{i+n}$.

5 Security Analysis

The security analysis of our main construction for conjunctive keyword search in Section 4 is studied.

5.1 Leakage Analysis

We begin with analyzing the leakage information considering a trade-off between security and efficiency. As illustrated in Section 3.3, two lists $\mathbf{d}$ and $\mathbf{q}$ are employed to store respective encryption and searching tuples. Here, let a f-th query be $\mathbf{q}[f] = (\mathbf{s}[f], \mathbf{x}[f, \cdot], \mathbf{id}[f])$, where $\mathbf{s}[f]$ is the s-term, $\mathbf{x}[f, \cdot]$ is the xterms and $\mathbf{id}[f]$ is the identity of the query client. The concrete descriptions on leakage function $\mathcal{L}$ and simulation proccess are given in Section 9: Appendix with Algorithm 1. Additionally, we emphasize that the leakage formation in $\mathbf{id}$, RP, SRP and IP are overstated for designing security proof, and hence not be revealed in actual scheme.

5.2 Security Analysis

Similar to [11], [13], we first prove the security under a non-adaptive attack, which means the adversary will submit the completed lists $\mathbf{d}$ and $\mathbf{q}$ in together. Then, we discuss the security against an adaptive attack.

Theorem 1. Our scheme is $\mathcal{L}$-semantically secure against non-adaptive attacks, if the employed PRFs are secure, the underlying CP-ABE is IND-sCP-CCA secure and the SXDH assumption holds in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$.

Proof 1. By using the outputs of the leakage function $\mathcal{L}$, we construct a simulator who has a same distribution as the real game, which is shown in Section 9: Appendix.

From the construction of simulator, we have

$$
\begin{aligned}
& \Pr[\mathsf{Real}_{\mathcal{A}}^{\Pi}(\kappa) = 1] - \Pr[\mathsf{Ideal}_{\mathcal{A},\mathcal{S}}^{\Pi}(\kappa) = 1] \\
\leq\ & \mathsf{Adv}_{\mathcal{A},\mathbb{G}_1}^{\mathsf{DDH}}(\kappa) + \mathsf{Adv}_{\mathcal{A},\mathbb{G}_2}^{\mathsf{DDH}}(\kappa) + \mathsf{Adv}_{\mathcal{A},\mathbb{G}_T}^{\mathsf{DDH}}(\kappa) + \mathsf{Adv}_{\mathcal{A},F_p}^{\mathsf{PRF}}(\kappa) \\
& + \mathsf{Adv}_{\mathcal{A},\mathsf{ABE}}^{\mathsf{IND-sCP-CPA}}(\kappa).
\end{aligned}
$$

Theorem 2. Our scheme is $\mathcal{L}$-semantically secure against adaptive attacks, if the employed PRFs are secure, the underlying CP-ABE is IND-sCP-CCA secure and the SXDH assumption holds in $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$.

Proof 2. The adaptive attack indicates that the adversary performs attacks as the security definition described in

$\underline{\mathsf{Setup}(1^\kappa, \mathcal{N})}$: Inputs a secure parameter κ and an attribute universe $\mathcal{N} = \{\mathsf{att}_1, \cdots, \mathsf{att}_n\}$ whose size is n, the authority randomly selects $r_1, r_2, \cdots, r_{2n} \in \mathbb{Z}_p$ and $t_1, t_2, \cdots, t_{2n} \in \mathbb{G}_1$. For $k = [2n]$ sets $x_k = g_2^{-r_k}$ and $y_k = e(t_k, g_2)$. Let $H : \{0,1\}^* \leftarrow \mathbb{Z}_p$ be a collision-resistant hash function, F be a PRF with range in $\{0,1\}^*$ and F_p be a PRF with range in $\mathbb{Z}_p$, and ABE be a non-monotonic ABE scheme. The authority runs setup algorithm of ABE to generate key pair

$$(\mathsf{mpk}_{\mathsf{ABE}}, \mathsf{msk}_{\mathsf{ABE}}) \leftarrow \mathsf{ABE.Setup}(1^\kappa, \mathcal{N})$$

and randomly selects two keys K_x and K_z for a PRF F_p, and selects a key K_l for a PRF F. The public parameter is

$$\mathsf{PP} = \{g_1, g_2, e, H, F, F_p, \mathsf{mpk}_{\mathsf{ABE}}, \{(x_k, y_k)\}_{k \in [2n]}\},$$

and the master secret key is

$$\mathsf{MK} = \{\mathsf{msk}_{\mathsf{ABE}}, K_x, K_z, K_l, \{(r_k, t_k)\}_{k \in [2n]}\}.$$

Note that, for $k \in \{1, \cdots, n\}$, the public parameter information (x_k, y_k) in PP corresponds to the positive type of attribute att_k and (x_{k+n}, y_{k+n}) corresponds to the negative type of it; the master secret key information (r_k, t_k) in MK corresponds to the positive type of attribute att_k and (x_{k+n}, y_{k+n}) corresponds to the negative type of it.

$\underline{\mathsf{KeyGen}(\Gamma, \mathsf{MK})}$: Inputs a client's attributes set $\Gamma \subseteq \mathcal{N}$ (a non-empty subset of $\mathcal{N}$) and the master secret key MK. The authority randomly selects an element v from $\mathbb{G}_1$, for each $i \in [n]$: computes $\sigma_i = t_i v^{r_i}$ if $\mathsf{att}_i^+ \in \Gamma$ (i.e., one attribute att_i in $\mathcal{N}$ exists in a client's attribute set Γ); else sets $\sigma_i = t_{i+n} v^{r_{i+n}}$ if $\mathsf{att}_i^+ \notin \Gamma$. Outputs the client's secret key sk as

$$\mathsf{sk} = \{K_x, K_z, K_l, \mathsf{pvk}_{\mathsf{ABE}}, v, \{\sigma_i\}_{i \in [n]}\}$$

where $\mathsf{pvk}_{\mathsf{ABE}} \leftarrow \mathsf{ABE.Extract}(\mathsf{msk}_{\mathsf{ABE}}, \Gamma)$.

$\underline{\mathsf{Encrypt}(\mathsf{PP}, \mathsf{Doc}, \mathsf{sk}, \mathcal{ACC})}$: Inputs public parameter PP, a client's secret key $\mathsf{sk} = \{K_x, K_z, K_l, \mathsf{pvk}_{\mathsf{ABE}}, v, \{\sigma_i\}_{i \in [n]}\}$, a document $\mathsf{Doc} = (ind, W_{ind})$ and an access policy $\mathcal{ACC} = \bigwedge_{\mathsf{att}_i \in I} \mathsf{att}_i$ where $\mathsf{att}_i \in \{\mathsf{att}_i^+, \neg\mathsf{att}_i\}$ (an "AND" operation over some attributes att_i in a non-empty subset I of $\mathcal{N}$). Computes $xind \leftarrow F_p(K_x, ind)$ and (x_i, y_i) as

$$(x_i, y_i) = \begin{cases} (x_i, y_i) = (g_2^{-r_i}, e(t_i, g_2)\rangle & \text{if } \mathsf{att}_i = \mathsf{att}_i^+ \\ (x_{i+n}, y_{i+n}) = (g_2^{-r_{i+n}}, e(t_{i+n}, g_2)) & \text{if } \mathsf{att}_i = \neg\mathsf{att}_i \end{cases}.$$

And sets $(x_I, y_I) = (\prod_{\mathsf{att}_i \in I} x_i, \prod_{\mathsf{att}_i \in I} y_i)$. For each keyword $w \in W_{ind}$, it does the following:

1) Computes an internal counter $c \leftarrow \mathsf{Get}(\delta, w)$, $c \leftarrow c+1$, $l \leftarrow F(K_l, c||w)$, $z \leftarrow F_p(K_z, c||w)$ and runs $\mathsf{Update}(\delta, w, c)$ to updates the counter of each keyword w to c.
2) Computes encrypted tuples $e_0 \leftarrow \mathsf{ABE.Enc}(\mathsf{mpk}_{\mathsf{ABE}}, ind||K_{ind}, \mathcal{ACC})$, $e_1 \leftarrow g_2^{z \cdot xind}$, $e_2 \leftarrow x_I^{z \cdot xind}$, then sets the item in $\mathsf{EDB}[l] = (\mathcal{ACC}, e_0, e_1, e_2)$ and appends a value $\mathsf{xtag} = y_I^{H(w) \cdot xind}$ to s set data structure XSet.

The client outsources (EDB, XSet) and the encrypted original document by using a symmetric key algorithm (e.g., AES) with a secret key is K_{ind} to the cloud.

$\underline{\mathsf{TrapGen}(Q, \mathsf{sk})}$: Given a conjunctive keyword searching query $Q = (w_1 \wedge w_2 \wedge \cdots \wedge w_q)$ and w_1 is assumed to be s-term (the least frequent term among the queried terms/keywords in Q), and a client's secret key $\mathsf{sk} = \{K_x, K_z, K_l, \mathsf{pvk}_{\mathsf{ABE}}, v, \{\sigma_i\}_{i \in [n]}\}$. The clients does the following steps until the cloud sends stop symbol Failure, for an internal counter $c = 1, 2, \cdots$ in keyword dictionary,

1) Computes $l_c \leftarrow F(K_l, c||w_1)$, $z_c \leftarrow F(K_z, c||w_1)$, $\mathsf{Trap}[c][j] = (v^{H(w_j)/z_c}, \{\sigma_i^{H(w_j)/z_c}\}_{i \in [n]})$, for $j = 1, \cdots, q$.
2) Sends generated tokens $\mathsf{Token}[c] = ((l_c, \mathsf{Trap}[c]))$ where $\mathsf{Trap}[c] = \{\mathsf{Trap}[c][j]\}_{j \in [q]}$ to the cloud for $c = 1, 2, \cdots$.

$\underline{\mathsf{Search}(\mathsf{Token})}$: Receiving a search token Token from a client, the cloud proceeds as follows:

1) Initializes an empty set R denoted as searching result, for an internal counter $c = 1, 2, \cdots$, do
   a) Retrieves tuples $(\mathcal{ACC} = \bigwedge_{\mathsf{att}_i \in I} \mathsf{att}_i, e_0, e_1, e_2) \leftarrow \mathsf{EDB}[l_c]$, if this action fails, jump to "Step 2)".
   b) Checks if $e(v^{H(w_j)/z_c}, e_2) \cdot e(\sigma_j, e_1) \in \mathsf{XSet}$ for all $j \in [q]$, where $v^{H(w_j)/z_c}$ and $\sigma_j = (\prod_{\mathsf{att}_i \in I} \sigma_i)^{H(w_j)/z_c}$ comes from $\mathsf{Trap}[c][j] = (v^{H(w_j)/z_c}, \{\sigma_i^{H(w_j)/z_c}\}_{i \in [n]})$. If it exists in XSet for all $j \in [q]$, then add e_0 to the set R.
2) Sends the stop symbol Failure and returns R to clients as the searching result, end off this search process.

$\underline{\mathsf{Retrieve}(R)}$: To retrieve the original documents, the client proceeds as follows.

1) Decrypts each e_0 in the returned searching result R as $(ind||K_{ind}) \leftarrow \mathsf{ABE.Dec}(\mathsf{pvk}_{\mathsf{ABE}}, e_0)$.
2) Sends inds in "Step 1)" to the cloud for fetching the encrypted original documents.
3) Decrypts the encrypted original documents with a corresponding secret key K_{ind}, respectively.

Fig. 3. Our main construction for dealing with conjunctive keyword search

Section 3.3, instead of submitting two completed lists $\mathbf{d}$ and $\mathbf{q}$ in together. Generally, the security follows the techniques of [11], where the simulator pads EDB and XSet with random chosen elements to deal with adaptive attacks. To simulate search tokens, it adaptively assigns the random chosen elements by hashing into tables. The main idea is to write EDB and XSet in a random way, and then adaptively initializes the hash tables.

Theorem 3. Our scheme is secure to resist the attacks carried out by the colluded clients, if the underlying CP-ABE is IND-sCP-CPA secure and the SXDH assumption holds in $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$.

Proof 3. The proof of this theorem is trivial, we can see that no colluded clients can generate a valid token beyond permission with the guarantee provided by the SXDH assumption. For the colluded clients, successfully generating a valid token implies that they can generate a new secret key beyond permission. In such a sense, they have to combine secret keys in together to solve for "t_i" and "r_i", but this is impossible since each secret key of a client is related to a random value "v" and the DDH assumption holds in $\mathbb{G}_1$ as well. Therefore, the colluded clients cannot generate a new secret key and a valid search token beyond their permissions.

6 Enhanced Construction: Supporting Boolean Keyword Search

This section extends the main construction supporting conjunctive queries in Section 4 to support boolean queries.

Boolean queries. The boolean query is more expressive for documents searching since it includes negations, disjunctions, threshold queries and more. Concretely, a boolean query formulated as "$w_1 \wedge \psi(w_2, \cdots, w_q)$" enables us to search a document that contains a keyword w_1 and additionally satisfies an arbitrary boolean expression ψ on the remaining keywords $(w_2, \cdots, w_q)$. With employing boolean queries, the search complexity is just proportional to the number of documents that contain the keyword "w_1" rather than the total number of all stored documents in remote server. Without loss of generality, the keyword "w_1" is assumed to be the estimated least frequent keyword.

Technical implementations. The used techniques is similar as [11] for extending conjunctive queries "$w_1 \wedge w_2 \wedge \cdots \wedge w_q$" to boolean queries "$w_1 \wedge \psi(w_2, \cdots, w_q)$" over keywords $(w_1, w_2, \cdots, w_q)$. A client computes l_c and $\mathsf{Trap}[c]$ as same as in processing a conjunctive search but sends them with a boolean expression $\bar{\psi}$ to the cloud, where $\bar{\psi}$ is a copy of ψ except that the keywords are replaced by $(v_2, \cdots, v_q)$. The cloud uses l_c to retrieve tuples $(\mathcal{ACC}, e_0, e_1, e_2)$ that associated with an estimated least frequent keyword w_1, where the difference with conjunctive queries is the way to determine which tuples match $\bar{\psi}$. For each record $(\mathcal{ACC}, e_0, e_1, e_2) \leftarrow \mathsf{EDB}[l_c]$, the cloud computes $(v_2, \cdots, v_q)$ as

$$v_j = \begin{cases} 1 & \text{if } e(v^{H(w_j)/z_c}, e_2) \cdot e(\sigma_j, e_1) \in \mathsf{XSet} \\ 0 & \text{otherwise} \end{cases}$$

where $j = 2, \cdots, q$. If $e(v^{H(w_1)/z_c}, e_2) \cdot e(\sigma_1, e_1) \in \mathsf{XSet}$ and $\bar{\psi}$ holds, this implies that the tuple matches the query, then the cloud appends e_0 to the result set R.

Note that the search complexity for such boolean query is $\mathcal{O}(c_{w_1})$ where w_1 is the s-term of the query. And the leakage profile processing is consistent with processing the leakage incurred in a conjunctive keyword searching over $(w_1, w_2, \cdots, w_q)$. Hence, the leakage information is similar as conjunctive queries except for the cloud server gets $\bar{\psi}$.

7 Performance Evaluation

This section gives a theoretical analysis comparison with boolean query SE work (not support owner-enforced authorization) and an implementation analysis comparison with representative ABSE work (only support conjunctive query).

7.1 Theoretical Analysis

TABLE 3
Numerical Evaluation. The $\mathsf{Mul}_1, \mathsf{Mul}_2$ and Mul_T respectively denotes the multiplication operation in group $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$; $\mathsf{E}_1, \mathsf{E}_2$ denotes the exponentiation operation in group $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively and BP is the bilinear pairing operation; $\mathsf{Time}_{\mathsf{ABE.Setup}}$ and $\mathsf{Time}_{\mathsf{ABE.Extract}}$ represent the running time of the Setup and Extract algorithm in ABE.

Running Process	Computation complexity
System Setup	$n(\mathsf{E}_2 + \mathsf{BP}) + \mathsf{Time}_{\mathsf{ABE.Setup}}$
Key Generation	$n(\mathsf{E}_1 + \mathsf{Mul}_1) + \mathsf{Time}_{\mathsf{ABE.Extract}}$
Encryption	$\lvert\mathcal{ACC}\rvert(\mathsf{Mul}_2 + \mathsf{Mul}_T) + 2\lvert W_{ind}\rvert\mathsf{E}_2$
Token Generation	$qc_{w_1}(n+1)\mathsf{E}_1$
Search	$q\lvert\mathcal{ACC}\rvert'\mathsf{Mul}_1 + qc_{w_1}(2\mathsf{BP} + \mathsf{Mul}_T)$

Table 3 shows a theoretical computation analysis on group operations of the proposed boolean query SE, We assume the size of attribute universe in the system is n and a document (ind, W_{ind}) being in encryption under an access policy $\mathcal{ACC}$, and consider the size of W_{ind} as $|W_{ind}|$ and the number of attributes in an "AND" gate access policy $\mathcal{ACC}$ as $|\mathcal{ACC}|$. For processing a boolean query "$w_1 \wedge \psi(w_2, \cdots, w_q)$", we assume w_1 as its s-term and c_{w_1} as a counter across q keywords, and $|\mathcal{ACC}|'$ denotes the sum of the number of attributes in all access control policies that related to thedocuments in $\mathsf{DB}[w_1]$.

Moreover, we give a feature and efficiency comparison with boolean SE work [11], [12], [13] in Table 4. The efficiency side mainly focuses on generating a search token between a data writer/owner and a data reader/client, where [11], [12], [13] cannot support owner-enforced authorization and only support conjunctive queries $(w_1 \wedge w_2 \cdots \wedge w_q)$. As the expected attribute-based search control is achieved across multiple clients for data sharing, we need to introduce public key encryption techniques. This may bring about additional costs in reader's computation side than those depend on simple symmetric-key operations.

7.2 Conducted Experiment Analysis

We implement both the enhanced SE system supporting boolean query "$w_1 \wedge \psi(w_2, \cdots, w_q)$" where ψ is an arbitrary boolean expression and the *"attribute-based keyword search with fine-grained owner-enforced search authorization"* work [7] that supports conjunctive keyword search "$w_1 \wedge w_2 \wedge \cdots \wedge w_q$", then give an all-around implementation analysis for

TABLE 4
Efficiency Comparison with Boolean Query SE Work. The "Hash" is a hashing operation and "Exp" denotes exponentiation operation in each group.

Work	Multi-writer	Multi-reader	Non-interaction[a]	Search Authorization[b]	Writer's Comp. cost	Reader's Comp. cost
[11]	No	No	-	No	$c_{w_1}(q-1)$Exp	-
[12]	No	Yes	No	No	$(q-1)$Exp	$c_{w_1}(q-1)$Exp
[13]	No	Yes	Yes	No	3Exp	$(c_{w_1}(q-1)+(q+1))$Exp
Ours	Yes	Yes	Yes	Yes	-	$c_{w_1} \cdot n \cdot q$Exp + qHash

a : The interaction needed between a writer and a reader whenever a reader generates a search token for query.
b : The attribute-based search control on encrypted data

comparison. Here, we take the work [7] as a representative ABSE since it inspires brand works in the area of ABSE and achieves better search efficiency among related work. It additionally mainly focuses on fundamental multi-client keyword search with owner-enforced authorization in cloud rather than concerning more security concerns. Moreover only [7] employs the same "AND"-gate search authorization "att$_1 \wedge$ att$_2 \wedge \cdots$" towards multiple clients as this paper.

7.2.1 *Experimental bed-up*

The work [7] and our SE system are conducted on an Ubuntu 16.04 system with an Intel(R) Core(TM) i3-4130 CPU of 3.40GHz and 4.00GB RAM. The codes are implemented with python 3 language under Charm 0.43 library [25] (a widely deployed open-source framework for rapidly prototyping advanced cryptosystems). To optimize the running efficiency of conducted schemes in implementation side, we choose to use an asymmetric elliptic curve "MNT159" to implement them and employ the scheme [26] as the underlying ABE scheme in our SE system.

A well-known representative real-world dataset Enron [19] is taken as a testing dataset for further illustrating convincing practicality, where the MySQL database is used to store encrypted data. We randomly select 1000 different documents from Enron dataset to encrypt under randomly generated access policies $\mathcal{ACC}$. And assume the scale of attribute universe ranges from 10 to 100 (n : $1 \sim 100$), the number of the associated keywords $|W_{ind}|$ with each document ranges from 1 to 50 (#keywords: $1 \sim 50$), and the assumed least frequent term in the conjunction ranges 20 to 60 (s-term: $20 \sim 60$). As other works, the experiment does not consider the original documents encryption process.

To well study and understand the experimental results about [7] and this work, we mainly focus the searching efficiency that includes trapdoor generation phase along with searching phase and documents encryption phase, as well as the system setup phase and key generation phase.

7.2.2 *Results and Comparison*

7.2.2.1 **System Initialization:** The system initialization of both [7]'s conjunctive query SE system and our boolean query SE system respectively includes the Setup and KeyGen algorithm. The setup phase initializes the system by generating public parameters and master secret key, while the key generation phase grants corresponding secret keys for clients. Overall, the system initialization performance comparison are drawn in Fig. 4, whose running time costs are both related to the scale of attribute universe.

We may conclude that, though the performance of this work is efficient but still a little expensive than [7] due to invoking the setup and key extraction algorithm of an ABE modular. Nevertheless, this is highly acceptable since the expressive boolean query pattern (i.e., "$w_1 \wedge \psi(w_2, \cdots, w_q)$") are achieved, while previous multi-client SEs with owner-enforced authorization only support conjunctive keyword query. Moreover, once an one-time investment for initializing the system completed, the provided highly-scalable documents searching/sharing service are more efficient than conjunctive query SEs, which is studied in the following.

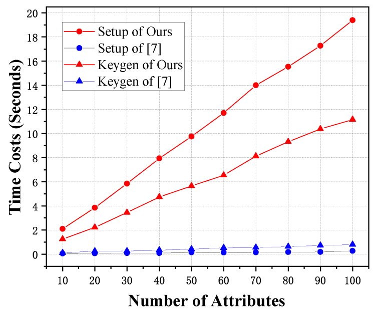
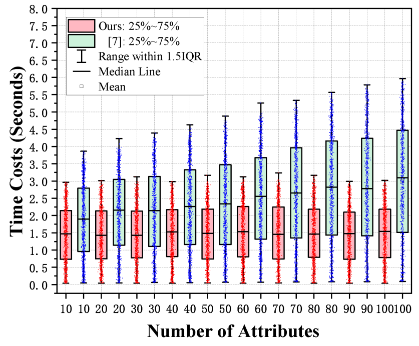


Fig. 4. The time efficiency in setup phase and key generation phase.

7.2.2.2 **Documents Tuples Encryption:** To evaluate the document encryption performance towards randomly chosen 1000 different documents, which are indexed with $1 \sim 50$ keywords and encrypted under randomly specified authorization policies across $10 \sim 100$ attributes. Concretely, from the time-costs distribution in Fig. 5(a) and the storage-costs distribution in Fig. 5(b) that related to the number of attributes, the general efficiency performance in this work is almost same with different #attributes. Nevertheless, the time costs for generating encrypted documents in [7] increases with #attributes and is expensive than ours. Fig. 6(a) and Fig. 6(b) show the time-costs and storage-costs distribution related to the number of keywords involved in documents, in which the consumed costs in [7] and ours are both linear to #keywords. As can be seen, our system yields high time and storage efficiency than [7]. Specifically, the time costs in this work are about $2.5 \sim 3.5$ seconds even the #keywords reaches 50 across different #attributes, while that in [7] are about $3.5 \sim 6$ seconds. A similar result can be observed for the storage costs side, only 23KB $\sim$ 26KB memory space are needed to maintain (EDB, XSet) tuple for a document who contains 50 keywords across different #attributes, while that in [7] are nearly about 100KB $\sim$ 550KB.

Hence, we can conclude from Fig. 5 and Fig. 6, the primary factor that influences the Encrypt algorithm in this work is the number of associated keywords (#keywords), while that in [7] is determined by both the number of keywords and the number of attributes.

7.2.2.3 **Documents Searching:** During documents searching, a client runs TrapGen algorithm to wake up


(a) Time Costs related to the number of Attributes

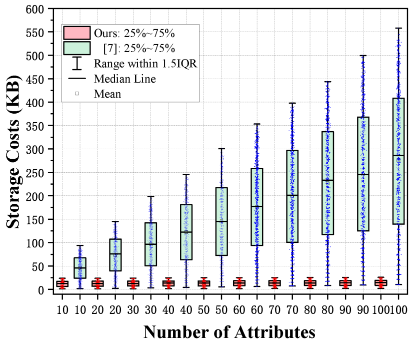
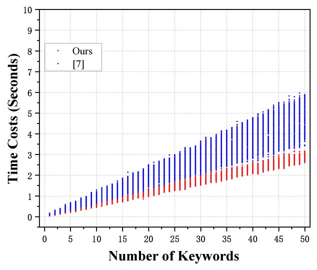

(b) Storage Costs related to the number of Attributes

Fig. 5. The time costs and storage costs of documents encryption phase related to the number of Attributes.


(a) Time Costs related to the number of Keywords

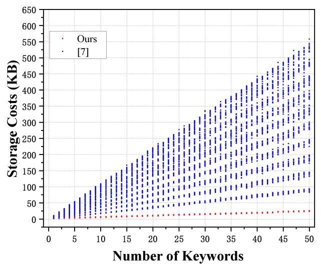
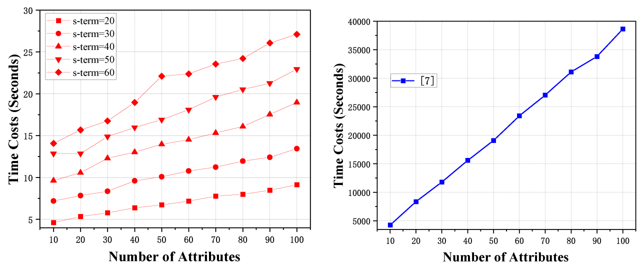
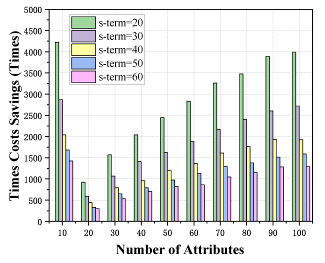

(b) Storage Costs related to the number of Keywords

Fig. 6. The time costs and storage costs of documents encryption of related to the number of Keywords.

the cloud to process Search algorithm, hence the documents searching overhead involve both time-costs and communication-costs of the TrapGen and Search algorithm. Generally, the documents searching performance in our boolean query SE and [7]'s conjunctive SE are both mainly related to #keywords and #attributes, while the least frequent term in the conjunction (i.e., s-term) is still an important factor for this work.

Concretely, by setting #keywords=5 and increasing the value of #attribute from 1 to 100, we can observe from Fig. 7 that: (1) the counter value of s-term (#s-term) included in a query in our scheme is still a very important factor that influence the searching performance in Fig. 7(a); (2) our work enjoys better time efficiency than [7] for a same #keywords (resp. runs 4500 times faster with s-term=20 and 1500 times faster with s-term=60 when #attributes=100) in Fig. 7(b); (3) [7] enjoys better communication efficiency than this work due to no consideration of expected expressive boolean queries in Fig. 7(c). [7] achieves roughly respective 10 times and 125 time savings than ours s-term=20 and s-term=60 realizing expressive boolean query patterns, but it is considered to be acceptable in real-world.


(a) Time costs when #keywords=5


(b) How many times on time costs savings when #keywords=5

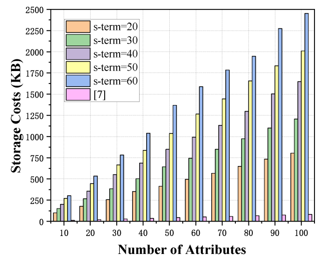
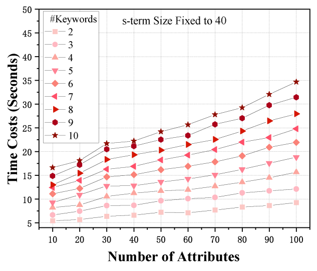

(c) Storage costs when #keywords=5

Fig. 7. The time and storage costs of documents searching with setting #keywords as 5.

To measure the influence of the #attribute and #keywords for documents searching performance, we assume the s-term as 40 in our system and provide a detailed result comparison on time costs between [7] and ours in Table 5. As can be seen, our boolean query SE certainly achieve high time efficiency and slightly increase with #attributes and #keywords, and moreover achieve roughly 300∼2000 times savings than [7]. We remark that if the s-term is set as 20, more time-costs savings are certainly obtained according to the conclusion from Fig. 7. This is because, processing a single/conjunctive-keyword search [5], [6], [7], [8], [9], [10], a client needs to send a keyword to cloud for fetching the documents where this keyword occurs, then finds and downloads the final results by itself. However, this work employs inverted index as the underlying data structure to manage encrypted documents for speeding up performance.


(a) Time costs when s-term=40

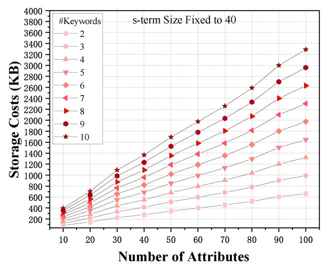

(b) Storage costs when s-term=40

Fig. 8. The time and storage costs of document searching with setting s-term as 40 in our boolean query SE system.

TABLE 5
The time costs comparison between our work (#s-term=40) and [7]'s work in documents searching.

#Att. / Time (s)		10	20	30	40	50	60	70	80	90	100
#Keywords = 2	Ours	5.44	5.69	6.4	6.63	7.24	7.13	7.69	8.32	8.67	9.25
	[7]	1696.65	3351.66	4741.55	6261.72	7655.78	9378.77	10828.88	12464.45	13529.64	15469.14
#Keywords = 3	Ours	6.65	7.47	8.64	8.67	9.64	10.1	10.35	11.31	11.73	12.13
	[7]	2544.95	5041.47	7092.86	9378.20	11466.24	14054.83	16233.52	18686.83	20287.41	23196.26
#Keywords = 4	Ours	8.22	8.76	10.56	11.27	11.74	11.97	12.71	13.55	14.5	15.66
	[7]	3402.97	6710.76	9444.50	12493.76	15277.10	18729.29	21638.19	24906.06	27043.89	30975.06
#Keywords = 5	Ours	9.29	10.89	12.78	12.82	13.61	14.25	15.12	16.26	17.56s	18.82s
	[7]	4269.73	8380.76	11796.85	15610.61	19088.83	23402.77	27041.81	31124.13	33800.24	38644.64
#Keywords = 6	Ours	11.12	12.24	14.68	15.15	16.18	16.91	17.86	19.11	20.92	21.94
	[7]	5119.86	10050.22	14148.86	18726.45	22899.51	28074.97	32445.02	37342.60	40556.84	46366.75
#Keywords = 7	Ours	12.44	13.94	16.29	16.89	18.23	19.27	20.44	21.98	22.94	24.81
	[7]	5967.51	11720.31	16501.53	21843.45	26709.55	32747.16	37845.83	43560.41	47312.82	54089.40
#Keywords = 8	Ours	13.01	15.46	18.34	19.33	20.3	21.49	22.55	24.32	26.46	27.96
	[7]	6816.13	13390.21	18853.62	24958.98	30519.49	37414.62	43247.28	49776.84	54069.59	61811.73
#Keywords = 9	Ours	14.87	17.22	20.5	21.14	22.54	23.4	25.73	27.05	29.78	31.45
	[7]	7664.39	15058.85	21205.96	28074.95	34328.02	42082.35	48648.02	55993.03	60825.03	69533.84
#Keywords = 10	Ours	16.62	18.1	21.71	22.21	24.21	25.65	27.8	29.25	32.06	34.67
	[7]	8512.37	16726.77	23557.90	31190.07	38136.16	46750.85	54049.26	62208.88	67581.42	77255.84

To observe how the #keywords influences the documents searching efficiency for our system, whose searching costs slightly increase with the growth of the amount of attribute in Fig. 8, where the s-term is set 40. Hence, we can conclude that the documents searching costs are related to the least frequent term (s-term) in the conjunction and independent of the total number of stored documents in cloud. This greatly brings about high efficiency for documents searching particularly for highly-scalable documents.

7.3 Related Work

Encrypted Boolean Query. Boolean query is an indispensable kind of keyword search and has been widely deployed in applications. With a boolean keyword search expression (i.e. $w_1 \wedge \neg w_2 \vee w_3$), document searching is to retrieve documents whose keywords set matches the boolean expression. Considering keywords as a vector, Moataz et al. [27] designed a SE scheme that supported generic boolean queries over encrypted data, but consumed-time costs were linear to the total number of documents in database. Later on, the milestone work by Cash et al. [11] introduced the first SE scheme with the support of sub-linear boolean keyword search, and then widely researched in [12], [13], [14].

Attribute-Based Searchable Encryption. With employing proxy re-encryption technique, Liang et al. [6] designed an ABSE scheme while still preserved encrypted data functionality of a proxy. A number of ABSE works focused on preserving the privacy of the access policy [10], [28], outsourced ABE works with keyword search and enhanced access control [8], [29], and as well as a trade-off between appreciated functionalities [9], [30], [31] and adversarial attack models [17], [18], [32], [33]. Nevertheless, all existing ABSE schemes can only work with simple single-keyword search or conjunctive-keyword search, but lose highly-efficient search costs (i.e., $\mathcal{O}(\#\text{doc})$, where $\#\text{doc}$ is the total number of the stored documents in cloud).

8 Conclusion

This work presents a privacy-preserving documents searching/sharing system for encrypted cloud storage. In the system, a data owner specifies an authorization on which clients could search/access outsourced data, and the cloud can process a client's boolean keyword search with only sub-linear costs. Note that the system does not consider attribute dynamic operations (e.g., attribute adding) at present. Since the size of the attribute universe is required to be fixed as n in the setup algorithm, where the associated random variables should be also assigned to corresponding attributes. As a result, we cannot let the attribute universe vary with the attribute dynamic changes. It seems not a easy job to realize dynamic attribute operations for the system, since once the attribute universe is not fixed when initializes the system, the CNF expression over both keywords and attributes could not be achieved. We leave this as an interesting future work to study.

9 Appendix

Remarks on leakage function and formal simulation process on non-adaptive attacks are described by Algorithm 1.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[2] "7 most infamous cloud security breaches," https://blog.storagecraft.com/7-infamous-cloud-security-breaches/.

[3] L. M. Kaufman, "Data security in the world of cloud computing," *IEEE Security & Privacy*, vol. 7, no. 4, pp. 61–64, 2009.

[4] "Top 10 cybersecurity risks for 2019. united states cybersecurity magazine," https://www.uscybersecurity.net/risks-2019/.

[5] Q. Zheng, S. Xu, and G. Ateniese, "VABKS: verifiable attribute-based keyword search over outsourced encrypted data," in *2014 IEEE Conference on Computer Communications, INFOCOM*, 2014, pp. 522–530.

[6] K. Liang and W. Susilo, "Searchable attribute-based mechanism with efficient data sharing for secure cloud storage," *IEEE Trans. Information Forensics and Security*, vol. 10, no. 9, pp. 1981–1992, 2015.

[7] W. Sun, S. Yu, W. Lou, Y. T. Hou, and H. Li, "Protecting your right: Verifiable attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 4, pp. 1187–1198, 2016.

[8] J. Li, X. Lin, Y. Zhang, and J. Han, "KSF-OABE: outsourced attribute-based encryption with keyword search function for cloud storage," *IEEE Trans. Services Computing*, vol. 10, no. 5, pp. 715–725, 2017.

[9] J. Zhu, Q. Li, C. Wang, X. Yuan, Q. Wang, and K. Ren, "Enabling generic, verifiable, and secure data search in cloud services," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 8, pp. 1721–1735, 2018.

[10] H. Wang, J. Ning, X. Huang, G. Wei, G. S. Poh, and X. Liu, "Secure fine-grained encrypted keyword search for e-healthcare cloud," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–1, 2019.

[11] D. Cash, S. Jarecki, C. S. Jutla, H. Krawczyk, M. Rosu, and M. Steiner, "Highly-scalable searchable symmetric encryption with support for boolean queries," in *Advances in Cryptology - CRYPTO 2013, Part I*, 2013, pp. 353–373.

[12] S. Jarecki, C. Jutla, H. Krawczyk, M. Rosu, and M. Steiner, "Outsourced symmetric private information retrieval," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 875–888.

[13] S. Sun, J. K. Liu, A. Sakzad, R. Steinfeld, and T. H. Yuen, "An efficient non-interactive multi-client searchable encryption with support for boolean queries," in *Computer Security - ESORICS 2016 - 21st European Symposium on Research in Computer Security, Proceedings, Part I*, 2016, pp. 154–172.

[14] S. Kamara and T. Moataz, "Boolean searchable symmetric encryption with worst-case sub-linear complexity," in *Advances in Cryptology - EUROCRYPT 2017, Proceedings, Part III*, 2017, pp. 94–124.

[15] D. X. Song, D. A. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *2000 IEEE Symposium on Security and Privacy*, 2000, pp. 44–55.

[16] P. Xu, Q. Wu, W. Wang, W. Susilo, J. Domingo-Ferrer, and H. Jin, "Generating searchable public-key ciphertexts with hidden structures for fast keyword search," *IEEE Trans. Information Forensics and Security*, vol. 10, no. 9, pp. 1993–2006, 2015.

[17] K. S. Kim, M. Kim, D. Lee, J. H. Park, and W. Kim, "Forward secure dynamic searchable symmetric encryption with efficient updates," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS*, 2017, pp. 1449–1463.

[18] R. Bost, B. Minaud, and O. Ohrimenko, "Forward and backward private searchable encryption from constrained cryptographic primitives," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS*, 2017, pp. 1465–1482.

[19] "Enron dataset," http://www.cs.cmu.edu/~./enron/.

[20] C. Chen, Z. Zhang, and D. Feng, "Efficient ciphertext policy attribute-based encryption with constant-size ciphertext and constant computation-cost," in *Provable Security - 5th International Conference, ProvSec 2011. Proceedings*, 2011, pp. 84–101.

[21] S. Kamara, C. Papamanthou, and T. Roeder, "Dynamic searchable symmetric encryption," in *Proceedings of the 2012 ACM conference on Computer and communications security*. ACM, 2012, pp. 965–976.

[22] J. Chen, H. W. Lim, S. Ling, H. Wang, and H. Wee, "Shorter IBE and signatures via asymmetric pairings," in *Pairing-Based Cryptography - Pairing 2012, Revised Selected Papers*, 2012, pp. 122–140.

[23] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," *ieee symposium on security and privacy*, pp. 321–334, 2007.

[24] S. Goldwasser and S. Micali, "Probabilistic encryption," *J. Comput. Syst. Sci.*, vol. 28, no. 2, pp. 270–299, 1984.

[25] J. A. Akinyele, M. Green, and A. D. Rubin, "Charm: A framework for rapidly prototyping cryptosystems," in *19th Annual Network and Distributed System Security Symposium, NDSS 2012*, 2012.

[26] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Public Key Cryptography - PKC 2011. Proceedings*, 2011, pp. 53–70.

[27] T. Moataz and A. Shikfa, "Boolean symmetric searchable encryption," in *8th ACM Symposium on Information, Computer and Communications Security, ASIA CCS*, 2013, pp. 265–276.

[28] S. Qiu, J. Liu, Y. Shi, and R. Zhang, "Hidden policy ciphertext-policy attribute-based encryption with keyword search against keyword guessing attack," *SCIENCE CHINA Information Sciences*, vol. 60, no. 5, pp. 052 105:1–052 105:12, 2017.

[29] J. Ning, Z. Cao, X. Dong, K. Liang, H. Ma, and L. Wei, "Auditable σ-time outsourced attribute-based encryption for access control in cloud computing," *IEEE Trans. Information Forensics and Security*, vol. 13, no. 1, pp. 94–105, 2018.

[30] Q. Wang, M. He, M. Du, S. S. M. Chow, R. W. F. Lai, and Q. Zou, "Searchable encryption over feature-rich data," *IEEE Trans. Dependable Sec. Comput.*, vol. 15, no. 3, pp. 496–510, 2018.

[31] Z. Wan and R. H. Deng, "Vpsearch: Achieving verifiability for privacy-preserving multi-keyword search over encrypted cloud data," *IEEE Trans. Dependable Sec. Comput.*, vol. 15, no. 6, pp. 1083–1095, 2018.

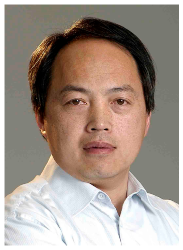
[32] J. Ning, J. Xu, K. Liang, F. Zhang, and E. Chang, "Passive attacks against searchable encryption," *IEEE Trans. Information Forensics and Security*, vol. 14, no. 3, pp. 789–802, 2019.

[33] Y. Zhang, J. Katz, and C. Papamanthou, "All your queries are belong to us: The power of file-injection attacks on searchable encryption," in *25th USENIX Security Symposium, USENIX Security.*, 2016, pp. 707–720.

Kai Zhang received the Bachelors degree with Computer Science and Technology from Shandong Normal University, China, in 2012, and the Ph.D. degree with Computer Science and Technology from East China Normal University, China, in 2017. He visits Nanyang Technological University in 2017. He is currently an Assistant Professor with Shanghai University of Electric Power, China. His research interest includes applied cryptography and information security.

Mi Wen received the M.S. degree in Computer Science from University of Electronic Science and Technology of China in 2005 and the Ph.D. degree in computer science from Shanghai Jiao Tong University, Shanghai, China in 2008. She is currently an Associate Professor of the College of Computer Science and Technology, Shanghai University of Electric Power. From May 2012 to May 2013, she was a visiting scholar at University of Waterloo, Canada. She serves Associate Editor of Peer-to Peer Networking and Applications (Springer). She keeps acting as the TPC member of some flagship conferences such as IEEE INFOCOM, IEEE ICC, IEEE GLOEBECOM, etc from 2012. Her research interests include privacy preserving in wireless sensor network, smart grid etc

Rongxing Lu received the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, in 2012. He was an Assistant Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, from 2013 to 2016. He has been an Associate Professor with the Faculty of Computer Science (FCS), University of New Brunswick (UNB), Fredericton, NB, Canada, since 2016. He was a Post-Doctoral Fellow with the University of Waterloo, from 2012 to 2013. Dr. Lu was a recipient of the Governor Generals Gold Medal for his Ph.D. degree from the Department of Electrical and Computer Engineering, University of Waterloo, the 8th IEEE Communications Society (ComSoc) AsiaPacific Outstanding Young Researcher Award in 2013, and the 2016 to 2017 Excellence in Teaching Award from FCS, UNB. He is currently serves as the Vice-Chair (Publication) of IEEE ComSoc CIS-TC.



Kefei Chen was awarded a Ph.d. degree from Justus-Liebig-University Giessen, Germany, in 1994. From 1996 to 2014, he was a professor and doctorate supervisor in the Department of Computer Science and Engineering at Shanghai JiaoTong University. Now he is a distinguished professor at the Shanghai University of Electric Power and a professor at the University of the Chinese Academy of Sciences and Hangzhou Normal University, and director of Westone Cryptologic Research Center. His main research interests include cryptographic algorithms, information hiding, digital watermarking, wireless security technology,and so on.

Algorithm 1 Simulator

Remarks on Leakage Function $\mathcal{L}$:

With taking as input $\mathbf{d}$ and $\mathbf{q}$, the leakage function $\mathcal{L}$ outputs the following items:

- op is an array that records the type of each operation type. We let op$[f]$ denote the type of f-th operation and it is either "encrypt" or "search".
- $\mathcal{ACC}$ is an array records the access policy in each encryption operation. Let $\mathcal{ACC}_{ind}$ be an access policy of the document labeled with ind.
- $\mathbf{id}$ is an array records the identity of client in each search operation. Moreover, we assume the attributes set of a client id is also leaked.
- N is an array that records the size of each EDB and XSet, namely, the number of keywords in each document, and $|N|$ equals with $|\mathbf{d}|$.
- $\mathbf{s}'$ is the equality pattern of $\mathbf{s}$, it reveals whether the queries have the same s-term. If $\mathbf{s} = \{a, b, a, c, b\}$, then we have $\mathbf{s}' = \{1, 2, 1, 3, 2\}$.
- $\mathbf{x}'$ is the equality pattern of $\mathbf{x}$, it reveals whether the queries have the same $x$term.
- SP is the size of each query pattern, it reveals the number of indices that matchs the s-term in each query, i.e., $\mathsf{SP}[f] = |\mathsf{DB}[\mathbf{s}[f]]|$.
- RP reveals the indices that the access policy can be satisfied by a client $\mathbf{id}[i]$ in the intersection of $\mathsf{DB}[\mathbf{s}[f]]$ and $\mathsf{DB}[\mathbf{x}[f,\alpha]]$, i.e., $\mathsf{RP}[f,\alpha] = \mathsf{DB}[\mathbf{s}[f], \mathbf{id}[f]] \bigcap \mathsf{DB}[\mathbf{x}[f,\alpha]]$. Denote $\mathsf{RP}[f,\alpha,d]$ as the element in $\mathsf{RP}[f,\alpha]$ that produced by $\mathbf{d}[d]$.
- SRP reveals the matching results of the s-term of f-th query, i.e, $\mathsf{SRP}[f] = \mathsf{DB}[\mathbf{s}[f]]$.
- $\mathsf{dRP}[f][h] == 1$ implies the indice of $\mathbf{d}[f]$ is contained in $\mathsf{DB}[\mathbf{s}[h]]$, otherwise, the value is 0.
- $\mathsf{IP}[f_1, f_2, \alpha, \beta] = \begin{cases} \mathsf{DB}[\mathbf{s}[f_1]] \cap \mathsf{DB}[\mathbf{s}[f_2]], & \text{if } \mathbf{id}[f_1] = \mathbf{id}[f_2] \text{ and } \exists \mathbf{x}[f_1,\alpha] = \mathbf{x}[f_2,\beta] \\ \mathsf{DB}[\mathbf{s}[f_1], \mathbf{id}[f_1]] \cap \mathsf{DB}[\mathbf{s}[f_2], \mathbf{id}[f_2]], & \text{if } \mathbf{id}[f_1] \neq \mathbf{id}[f_2] \text{ and } \exists \mathbf{x}[f_1,\alpha] = \mathbf{x}[f_2,\beta] \\ \emptyset, & \text{Otherwise.} \end{cases}$
- $\mathsf{xt}[i]$ records the number of xterms in i-th query.

Simulation Process:

```
function Initialize($\mathcal{L}(\mathbf{d},\mathbf{q})$)
    for each $h \in \mathbf{s}'$ do
        $c_h$=0
    end for
    for each $w \in \mathbf{x}'$, $ind \in \mathsf{RP} \cup \mathsf{IP}$ do
        $H_2[w, ind] = y \xleftarrow{\$} \mathbb{Z}_p$
        $H_3[w, ind] = e(g_1, g_2)^{H_2[w,ind]}$
    end for
    for each $w \in \mathbf{x}'$, $ind \in \mathsf{RP} \cup \mathsf{IP}$, $id \in \mathbf{id}$ do
        if $id \vDash (\mathcal{ACC}_{ind} = \bigwedge_{\mathsf{att}_i \in I} \mathsf{att}_i)$ then
            $\{y_0, \cdots, y_n\} \xleftarrow{\$} \mathbb{Z}_p^{n+1}$
            s.t. $H_2[w, ind] == y_0 + \sum_{\mathsf{att}_i \in I} y_i$
            $H_5[w, ind, id] = \{y_0, \cdots, y_n\}$
        else
            $\{y_0, \cdots, y_n\} \xleftarrow{\$} \mathbb{Z}_p^{n+1}$
            $H_5[w, ind, id] = \{y_0, \cdots, y_n\}$
        end if
    end for
    $d = q = 1$
    for $h = 1$ to $|\mathsf{op}|$ do
        if op$[h]$==encrypt then
            $\mathbf{t}[h] \leftarrow$ Encrypt($\mathcal{L}(\mathbf{d},\mathbf{q})$)
            $d{+}{+}$
        end if
        if op$[h]$==search then
            $\mathbf{t}[h] =$ TranGen($\mathcal{L}(\mathbf{d},\mathbf{q})$)
            $q{+}{+}$
        end if
    end for
    return $\mathbf{t}$
end function
function Encrypt($\mathcal{L}(\mathbf{d},\mathbf{q})$)
    $h = 0$, Dup $\leftarrow \{\}$
    for $\mathbf{s}'[q'] \in \{\mathbf{s}'[q] \cdots \mathbf{s}'[|\bar{\mathbf{s}}|]\}$, $\mathbf{s}'[q'] \notin$ Dup and $\mathsf{dRP}[d][q'] == 1$ do
        $c_{\mathbf{s}'[q']}{+}{+}$
        $l \xleftarrow{\$} \{0,1\}^*$
        $l[\mathbf{s}[q'], c_{\mathbf{s}'[q']}] = l$
        $e_0 \leftarrow \mathsf{ABE.Enc}(\mathsf{mpk}_{\mathsf{ABE}}, 0^k, \mathcal{ACC}[d])$
        $y \xleftarrow{\$} \mathbb{Z}_p, H_1[\mathbf{s}'[q'], c_{\mathbf{s}'[q']}] = y$
        $y \xleftarrow{\$} \mathbb{Z}_p, H_4[\mathbf{s}'[q'], c_{\mathbf{s}'[q']}] = y$
        $e_1 \leftarrow g_2^{H_1\left[\mathbf{s}'[q'], c_{\mathbf{s}'[q']}\right]}, e_2 \leftarrow g_2^{H_4\left[\mathbf{s}'[q'], c_{\mathbf{s}'[q']}\right]}$
        $\mathsf{EDB}[l] = (\mathcal{ACC}[d], e_0, e_1, e_2)$
        Dup $\leftarrow$ Dup $\cup\, \mathbf{s}'[q']$, $h{+}{+}$
    end for
    for $h$ to $N[d]$ do
        $l \xleftarrow{\$} \{0,1\}^*$
        $e_0 \leftarrow \mathsf{ABE.Enc}(\mathsf{mpk}_{\mathsf{ABE}}, 0^k, \mathcal{ACC}[d])$
        $e_1 \xleftarrow{\$} \mathbb{G}_2,\ e_2 \xleftarrow{\$} \mathbb{G}_2$
        $\mathsf{EDB}[l] = (\mathcal{ACC}[d], e_0, e_1, e_2)$
    end for
    XSet $\leftarrow$ XSetSetup($\mathcal{L}(\mathbf{d},\mathbf{q})$)
    return EDB and XSet
end function
function XSetSetup($\mathcal{L}(\mathbf{d},\mathbf{q})$)
    XSet $\leftarrow \{\}, h = 0$
    for $w = \mathbf{x}'[t \geq q, \alpha]$ and $\mathsf{RP}[t,\alpha,d] \neq \emptyset$ do
        $ind \leftarrow \mathsf{RP}[t,\alpha,d]$
        xtag $\leftarrow H_3[w, ind]$
        XSet $\leftarrow$ XSet $\cup$ xtag, $h{+}{+}$
    end for
    for $j$ to $N[d]$ do
        xtag $\xleftarrow{\$} \mathbb{G}_T$
        XSet $\leftarrow$ XSet $\cup$ xtag
    end for
    return XSet
end function
function TranGen($\mathcal{L}(\mathbf{d},\mathbf{q})$)
    $\mathbf{l} = \{l[\mathbf{s}'[q], h]\}_{h=1}^{c_{\mathbf{s}'[q]}}, (ind_1, \cdots, ind_{c_{\mathbf{s}'[q]}}) \leftarrow \mathsf{SRP}[\mathsf{q}]$
    for $\alpha \in [\mathsf{xt}[q]]$ do
        $R \leftarrow \mathsf{RP}[q,\alpha] \cup_{q' \in [|\mathbf{s}'|], \beta \in [\mathsf{xt}[q']]} \mathsf{IP}[q, q', \alpha, \beta]$
        for $c \in \left[c_{\mathbf{s}'[q]}\right]$ do
            if $ind_c \in R$ then
                $\{y_0, \cdots, y_n\} \leftarrow H_5[\mathbf{x}'[q,\alpha], ind_c, \mathbf{id}[q]]$
                $\mathsf{Trap}[c][\alpha] = (g_1^{\frac{y_0}{H_4[\mathbf{s}'[q],c]}}, \{g_1^{\frac{y_k}{H_1[\mathbf{s}'[q],c]}} | k \in [n]\})$
            else
                if $\exists\, H_6[\mathbf{s}'[q], \mathbf{x}'[q,\alpha], c, \mathbf{id}[q]]$ then
                    $\mathsf{Trap}[c][\alpha] = H_6[\mathbf{s}'[q], \mathbf{x}'[q,\alpha], c, \mathbf{id}[q]]$
                else
                    $\mathsf{Trap}[c][\alpha] = (R_k \xleftarrow{\$} \mathbb{G}_1^{n+1} | k \in [n])$,
                    $H_6[\mathbf{s}'[q], \mathbf{x}'[q,\alpha], c, \mathbf{id}[q]] = \mathsf{Trap}[c][\alpha]$
                end if
            end if
        end for
    end for
    Token $\leftarrow (\mathbf{l}, \mathsf{Trap})$
    Res $\leftarrow$ Search(Token)
    ResInds $\leftarrow \cap\, \mathsf{RP}[q,\alpha]$ for $\alpha \in [\mathsf{xt}[q]]$
    return (Token, Res, ResInds)
end function
```