Efficient and Expressive Keyword Search Over Encrypted Data in Cloud

Hui Cui, Zhiguo Wan, Robert H. Deng, Guilin Wang, and Yingjiu Li

Abstract—Searchable encryption allows a cloud server to conduct keyword search over encrypted data on behalf of the data users without learning the underlying plaintexts. However, most existing searchable encryption schemes only support single or conjunctive keyword search, while a few other schemes that are able to perform expressive keyword search are computationally inefficient since they are built from bilinear pairings over the composite-order groups. In this paper, we propose an expressive public-key searchable encryption scheme in the prime-order groups, which allows keyword search policies (i.e., predicates, access structures) to be expressed in conjunctive, disjunctive or any monotonic Boolean formulas and achieves significant performance improvement over existing schemes. We formally define its security, and prove that it is selectively secure in the standard model. Also, we implement the proposed scheme using a rapid prototyping tool called Charm [37], and conduct several experiments to evaluate it performance. The results demonstrate that our scheme is much more efficient than the ones built over the composite-order groups.

Index Terms—Searchable encryption, cloud computing, expressiveness, attribute-based encryption.

1 Introduction

Consider a cloud-based healthcare information system that hosts outsourced personal health records (PHRs) from various healthcare providers. The PHRs are encrypted in order to comply with privacy regulations like HIPAA. In order to facilitate data use and sharing, it is highly desirable to have a searchable encryption (SE) scheme which allows the cloud service provider to search over encrypted PHRs on behalf of the authorized users (such as medical researchers or doctors) without learning information about the underlying plaintext. Note that the context we are considering supports private data sharing among multiple data providers and multiple data users. Therefore, SE schemes in the private-key setting [1], [2], [3], which assume that a single user who searches and retrieves his/her own data, are not suitable. On the other hand, private information retrieval (PIR) protocols [4], [5], [6], which allow users to retrieve a certain data-item from a database which publicly stores data without revealing the data-item to the database administrator, are also not suitable, since they require the data to be publicly available. In order to tackle the keyword search problem in the cloud-based healthcare information system scenario, we resort to public-key encryption with keyword search (PEKS) schemes, which is firstly proposed in [7]. In a PEKS scheme, a ciphertext of the keywords called "PEKS ciphertext" is appended to an encrypted PHR. To retrieve all the encrypted PHRs containing a keyword, say "Diabetes", a user sends a "trapdoor" associated with a search query on the keyword "Diabetes" to the cloud service provider, which selects all the encrypted PHRs containing the keyword "Diabetes" and returns them to the user while without learning the underlying PHRs. However, the solution in [7] as well as other existing PEKS schemes which improve on [7] only support equality queries [8].

Set intersection and meta keywords[1] [9], [10] can be used for conjunctive keyword search. However, the approach based on set intersection leaks extra information to the cloud server beyond the results of the conjunctive query, whilst the approach using meta keywords require 2^m meta keywords to accommodate all the possible conjunctive queries for m keywords. In order to address the above deficiencies in conjunctive keyword search, schemes such as the ones in [11], [12] were put forward in the public-key setting.

Ideally, in the practical applications, search predicates (i.e., policies) should be expressive such that they can be expressed as conjunction, disjunction or any Boolean formulas[2] of keywords. In the above cloud-based healthcare system, to find the relationship between diabetes and age or weight, a medical researcher may issue a search query with an access structure (i.e., predicate) ("Illness = Diabetes" AND ("Age = 30" OR "Weight = 150-200")). SE schemes supporting expressive keyword access structures were presented in [8], [13], [14], [15]. Unfortunately, the scheme in [13] has exponentially increasing complexity [16], while the schemes in [8], [14], [15] are based on the inefficient bilinear pairing over composite-order groups [17]. Though there exist techniques [17] to convert pairing-based schemes from composite-order groups to prime-order groups, there is still a significant performance degradation due to the

- H. Cui (hcui@smu.edu.sg) is with the Secure Mobile Centre, Singapore Management University, Singapore.
- Z. Wan (wanzhiguo@sdu.edu.cn) is with the School of Computer Science and Technology, Shandong University, Jinan, China.
- R. H. Deng (robertdeng@smu.edu.sg) and Y. Li (yjli@smu.edu.sg) are with the School of Information Systems, Singapore Management University, Singapore.
- G. Wang (wang.guilin@huawei.com) is with the Shield Lab, Central Research Institute, Huawei International Pte Ltd, Singapore.

Manuscript received , 2016; revised , 2016.

1. Meta keywords are composed of several keywords. For example, a document that contains the keywords "Bob", "urgent" and "finance" may be augmented with the meta-keyword "Bob: urgent: finance"

2. In this paper, unless otherwise specified, the Boolean formulas we talk about are monotonic. That is, they consist of only AND and OR gates, for example, A AND (B OR C).

required size of the special vectors [18].

In this paper, we propose a public-key based expressive SE scheme in prime-order groups, which is especially suitable for keyword search over encrypted data in scenarios of multiple data owners and multiple data users such as the cloud-based healthcare information system that hosts outsourced PHRs from various healthcare providers.

1.1 Overview of Our Proposed Scheme

Our expressive SE scheme consists of a trusted trapdoor generation center which publishes a public system parameter and keeps a master key in secret, a cloud server which stores and searches encrypted data on behalf of data users, multiple data owners who upload encrypted data to the cloud, and multiple data users who would like to retreive encrypted data containing certain keywords. To outsource an encrypted document to the cloud, a data owner appends the encrypted document with keywords encrypted under the public parameter and uploads the combined encrypted document and encrypted keywords to the cloud. To retrieve all the encrypted documents containing keywords satisfying a certain access structure (i.e., predicate or policy) such as (“Illness = Diabetes” AND (“Age = 30” OR “Weight = 150-200”)), a data user first obtains a trapdoor associated with the access structure from the trapdoor generation center and then sends the trapdoor to the cloud server. The latter will conduct the search and return the corresponding encrypted documents to the data user.

The basic idea of our scheme is to modify a key-policy attributed-based encryption (KP-ABE) scheme constructed from bilinear pairing over prime-order groups. Without loss of generality, we will use the large universe KP-ABE scheme selectively secure in the standard model proposed by Rouselakis and Waters in [18] to illustrate our construction during the rest of the paper. In KP-ABE, a ciphertext is computed with respect to a set of attributes and an access policy is encoded into a user’s private key. A ciphertext can be decrypted by a private key only if the set of attributes associated with the ciphertext satisfies the access policy associated with the private key. Access policies in [18] can be very expressive, supporting any monotonic Boolean formulas. At first sight, a KP-ABE scheme can be transformed to an expressive SE scheme by treating attributes as keywords to be searched, by directly transforming the key generation algorithm on attribute access structures to a trapdoor generation algorithm on keyword search predicates, and by using the decryption algorithm to test whether keywords in a ciphertext satisfy the predicate in a trapdoor. However, KP-ABE schemes (e.g., [18], [19]) are not designed to preserve privacy of attributes (keywords) associated with ciphertexts. Specifically, given the public parameter and a ciphertext, the attributes (keywords) in the ciphertext can be discerned by anyone. In the following, to keep our description compact and consistent, we will use access structure, policy and predicate interchangeably.

In order to hide keywords in a ciphertext, inspired by the “linear splitting” technique in [20], we firstly split ciphertext components corresponding to every keyword into two randomized complementary components. Thus, even though the ciphertext still contains information about the keywords, this information is computationally infeasible to obtain from the public parameter and the ciphertext. We secondly re-randomize trapdoor components corresponding to every keyword associated with an access structure to match the splitted components in the ciphertext.

In addition to hiding keywords in ciphertexts, we also need to preserve keyword privacy in a trapdoor which contains an access structure as a component. First, to preserve keyword privacy in an access structure, we adopt the method in [21] to divide each keyword into a generic name and a keyword value. Since keyword values are much more sensitive than the generic keyword names, the keyword values in an access structure are not disclosed to the cloud server, whereas a partial hidden access structure with only generic keyword names is included in a trapdoor and sent to the cloud server. Take the aforementioned keyword access structure (“Illness = Diabetes” AND (“Age = 30” OR “Weight = 150-200”)) as an instance, “Illness”, “Age” and “Weight” are the generic names whilst “Diabetes”, “30” and “200” are the keyword values. Consequently, the partial hidden access structure (“Illness” AND “Age” OR “Weight”) is included in the trapdoor. Second, as in all the PEKS schemes, trapdoors are subject to the offline keyword dictionary guessing attacks. That is, anyone who knows a trapdoor and the public parameter may discover the keyword values embedded in the trapdoor by launching exhaustive searching attacks on keyword values. As a remedy to such attacks, we assign a designated cloud server as introduced in [22] to perform the searching operations. We equip this designated server with a public and private key pair of which the public key will be used in trapdoor generation such that it is computationally infeasible for anyone without knowledge of the privacy key to derive keywords information from the trapdoor. Thus, trapdoors can be delivered to the cloud server over a public channel.

We define a security model for expressive SE, which takes into account all adversarial capabilities of the standard SE security notion. The adversary is able to learn trapdoors over access structures of its choice, but it should not be able to learn any information about the keyword values in the challenge ciphertext. Note that since the Rouselakis-Waters KP-ABE scheme [18], which the proposed SE scheme is built upon, is selectively secure, our expressive SE scheme can only be proved to be selectively secure where the adversary has to commit the challenge keyword set in advance.

1.2 Contributions

Below we briefly summarize our contributions in this paper.

- We propose the first expressive SE scheme in the public-key setting from bilinear pairings in *prime-order* groups. As such, our scheme is not only capable of expressive multi-keyword search, but also significantly more efficient than existing schemes built in composite-order groups.
- Using a randomness splitting technique, our scheme achieves security against offline keyword dictionary guessing attacks to the ciphertexts. Moreover, to preserve the privacy of keywords against offline keyword dictionary guessing attacks to trapdoors, we divide each keyword into keyword name and

keyword value and assign a designated cloud server to conduct search operations in our construction.

- We formalize the security definition of expressive SE, and formally prove that our proposed expressive SE scheme is selectively secure in the standard model.
- We implement our scheme using a rapidly prototyping tool called Charm, and conduct extensive experiments to evaluate its performance. Our results confirm that the proposed scheme is sufficiently efficient to be applied in practice.

1.3 Related Work

Public-Key Encryption with Keyword Search. After Boneh et al. [7] initiated the study of public-key encryption with keyword search (PEKS), several PEKS constructions were put forth using different techniques or considering different situations [8], [11], [12], [13], [14], [15], [22], [23], [24], [25], [26], [27], [28], [29]. They aim to solve two cruces in PEKS: (1) how to make PEKS secure against offline keyword dictionary guessing attacks; and (2) how to achieve expressive searching predicates in PEKS. In terms of the offline keyword dictionary guessing attacks, which requires that no adversary (including the cloud searching server) can learn keywords from a given trapdoor, to the best of our knowledge, such a security notion is very hard to be achieved in the public-key setting [30]. Regarding the expressive search, there are only few works in PEKS [8], [13], [14], [15]. Unfortunately, the construction in [13] is built on the basis of inner-product predicate encryption [16], and the constructions in [8], [14], [15] are built from the pairings in composite-order group. Therefore, they are not sufficiently efficient to be adopted in the practical world [16], [17]. Moreover, the number of keywords allowed in these searchable schemes are predefined in the system setup phase. We compare our scheme to other keyword search schemes in Table 1. It is straightforward to see that compared to the existing ones, our construction make a good balance in that it allows unbounded keywords, supports expressive access structures, and is built in the prime-order groups.

Private-key Searchable Encryption. In a private-key SE setting, a user uploads its private data to a remote database and keeps the data private from the remote database administrator. Private-key SE allows the user to retrieve all the records containing a particular keyword from the remote database [1], [2], [3]. However, as the name suggests, private-key SE solutions only apply to scenarios where data owners and data users totally trusted each other.

Private Information Retrieval. With respect to public database such as stock quotes, where the user is unaware of it and wishes to search for some data-item without revealing to the database administrator which item it is, private information retrieval (PIR) [4], [5], [6] protocols were introduced, which allow a user to retrieve data from a public database with far smaller communication then just downloading the entire database. Nevertheless, in our context, the database is not publicly available, the data is not public, so the PIR solutions cannot be applied.

1.4 Organization

The remainder of this paper is organized as follows. In Section 2, we briefly review some of the notions and definitions to be used in the paper. In Section 3, after depicting the system architecture for our expressive keyword search system, we give a concrete expressive keyword search scheme. In Section 4, we discuss the properties and several extensions of our expressive keyword search scheme. We implement our scheme and compare it with related works in Section 5. We conclude the paper in Section 6.

2 Preliminaries

In this section, we review some basic cryptographic notions and definitions that are to be used later.

2.1 Bilinear Pairings and Complexity Assumptions

Let G be a group of prime order p with a generator g. Let $\hat{e} : G \times G \to G_1$ be an efficiently computable bilinear pairing function satisfying the following properties [31].

- Bilinear: for all $g \in G$, and a, $b \in Z_p^*$, we have $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$.
- Non-degenerate: $\hat{e}(g, g) \neq 1$.

Decisional Bilinear Diffie-Hellman Assumption [31]. The decisional Bilinear Diffie-Hellman (BDH) problem is that for any probabilistic polynomial-time algorithm, given g, g^a, g^b, g^c, it is difficult to distinguish $(g, g^a, g^b, g^c, \hat{e}(g,g)^{abc})$ from (g, g^a, g^b, g^c, Z), where $g \in G$, $Z \in G_1$, a, b, $c \in Z_p^*$ chosen independently and uniformly at random.

Decisional $(q-2)$ Assumption [18]. Let q be an integer. The decisional $(q-2)$ problem is that for any probabilistic polynomial-time algorithm, given $\overrightarrow{A} =$

$$\begin{array}{ll} g, g^x, g^y, g^z, g^{(xz)^2} & \\ g^{b_i}, g^{xzb_i}, g^{xz/b_i}, g^{x^2zb_i}, g^{y/b_i^2}, g^{y^2/b_i^2} & \forall\, i \in [q], \\ g^{xzb_i/b_j}, g^{yb_i/b_j^2}, g^{xyzb_i/b_j}, g^{(xz)^2b_i/b_j} & \forall\, i, j \in [q], i \neq j, \end{array}$$

it is difficult to distinguish $(\overrightarrow{A}, \hat{e}(g,g)^{xyz})$ from $(\overrightarrow{A}, Z)$, where $g \in G$, $Z \in G_1$, x, y, z, $b_1, ..., b_q \in Z_p^*$ chosen independently and uniformly at random.

Decisional Linear Assumption [32]. The decisional linear problem is that for any probabilistic polynomial-time algorithm, given g, g^{x_1}, g^{x_2}, $g^{x_1x_3}$, $g^{x_2x_4}$, it is difficult to distinguish $(g, g^{x_1}, g^{x_2}, g^{x_1x_3}, g^{x_2x_4}, g^{x_3+x_4})$ from $(g, g^{x_1}, g^{x_2}, g^{x_1x_3}, g^{x_2x_4}, Z)$, where g, $Z \in G$, x_1, x_2, x_3, $x_4 \in Z_p^*$ chosen independently and uniformly at random.

2.2 Access Structures and Linear Secret Sharing

Following the definition in [33], [34], we describe the notions of access structures and linear secret sharing schemes.

Definition 1. (Access Structure). Let $\{P_1, ..., P_n\}$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,...,P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$. An (monotone) access structure is a (monotone) collection $\mathbb{A}$ of non-empty subsets of $\{P_1, ..., P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1,...,P_n\}} \setminus \{\emptyset\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.

In our construction, we only consider monotone access structures. Notice that general access structures in large universe ABE can be realized by splitting the attribute universe in half and treating the attributes of one half as the negated

TABLE 1
Comparisons of expressive keyword search schemes.

	Keyword Privacy	Expressiveness	Bilinear Group	Security	Unbounded keywords
BCOP04 [7]	keyword guessing attacks on trapdoors	AND	prime	full random oracle	yes
KSW13 [16]	keyword guessing attacks on trapdoors	AND, OR	composite	full standard model	no
LZDLC13 [8]	keyword guessing attacks on trapdoors	AND, OR	composite	full standard model	no
LHZF14 [14]	no keyword guessing attacks on trapdoors	AND, OR, NOT	composite	full standard model	no
Our scheme	keyword guessing attacks on trapdoors by designated server only	AND, OR	prime	selective standard model	yes

(NOT) versions of the attributes in the other half [35]. Also, it has been presented in [36], [37] how to describe non-monotonic access structures in terms of monotonic access structures with negative (NOT) shares.

Definition 2. (Linear Secret Sharing Schemes). Let P be a set of parties. Let $\mathbb{M}$ be a matrix of size $l \times n$. Let $\rho : \{1, ..., l\} \rightarrow P$ be a function that maps a row to a party for labeling. A secret sharing scheme Π over a set of parties P is a linear secret-sharing scheme (LSSS) over Z_p if

1) The shares for each party form a vector over Z_p.
2) There exists a matrix $\mathbb{M}$ which has l rows and n columns called the share-generating matrix for Π. For $i = 1, ..., l$, the x-th row of matrix $\mathbb{M}$ is labeled by a party $\rho(i)$, where $\rho : \{1, ..., l\} \rightarrow P$ is a function that maps a row to a party for labeling. Considering that the column vector $\overrightarrow{v} = (\mu, r_2, ..., r_n)$, where $\mu \in Z_p$ is the secret to be shared and $r_2, ..., r_n \in Z_p$ are randomly chosen, then $\mathbb{M}\overrightarrow{v}$ is the vector of l shares of the secret μ according to Π. The share $(\mathbb{M}\overrightarrow{v})_i$ belongs to party $\rho(i)$.

It has been noted in [33] that every LSSS also enjoys the linear reconstruction property. Suppose that Π is an LSSS for an access structure $\mathbb{A}$. Let $\mathbf{A}$ be an authorized set, and define $I \subseteq \{1, ..., l\}$ as $I = \{i | \rho(i) \in \mathbf{A}\}$. Then the vector $(1, 0, ..., 0)$ is in the span of rows of $\mathbb{M}$ indexed by I, and there exist constants $\{w_i \in Z_p\}_{i \in I}$ such that, for any valid shares $\{v_i\}$ of a secret μ according to Π, we have $\sum_{i \in I} w_i v_i = \mu$. These constants $\{w_i\}$ can be found in the polynomial time in terms of the size of the share-generating matrix $\mathbb{M}$ [38]. On the other hand, for an unauthorized set $\mathbf{A}'$, no such constants $\{w_i\}$ exist. Also, in this case it is true that if $I' = \{i | \rho(i) \in \mathbf{A}'\}$, there exists a vector $\overrightarrow{w}$ such that its first component w_1 is any non zero element in Z_p and $< \mathbb{M}_i, \overrightarrow{w} > = 0$ for all $i \in I'$, where $\mathbb{M}_i$ is the i-th row of $\mathbb{M}$ [18].

Boolean Formulas [33]. Access structures can also be described in terms of monotonic boolean formulas. LSSS access structures are more general, and can be derived from representations as boolean formulas. There are techniques to convert any monotonic boolean formula into a corresponding LSSS matrix[3]. The boolean formula can be represented as an access tree, where the interior nodes are AND and OR gates, and the leaf nodes denote attributes. The number of the rows in the corresponding LSSS matrix will be the same as the number of the leaf nodes in the access tree.

3 Efficient and Expressive Keyword Search with Unbounded Keywords

In this section, we describe the system model, design goals, threat model and algorithms of our expressive SE scheme.

3.1 System Model and Design Goals

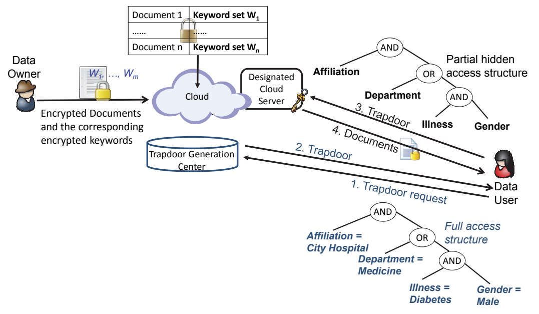


Fig. 1. Architecture of expressive keyword search system.

The architecture of our keyword search system is shown in Fig. 1, which is composed of four entities: a trusted trapdoor generation centre who publishes the system parameter and holds a master private key and is responsible for trapdoor generation for the system, data owners who outsource encrypted data to a public cloud, data users who are privileged to search and access encrypted data, and a designated cloud server who executes the keyword search operations for data users. To enable the cloud server to search over ciphertexts, the data owners append every encrypted document with encrypted keywords[4]. A data user issues a trapdoor request by sending a keyword access structure to the trapdoor generation centre which generates and returns a trapdoor corresponding to the access structure. We assume that the trapdoor generation centre has a separate authentication mechanism to verify each data user and then issue them the corresponding trapdoors. After

3. We give an example on how to convert a boolean formula into an equivalent LSSS matrix in Appendix C.

4. Note that each keyword is composed of a generic name and a keyword value.

obtaining a trapdoor, the data user sends the trapdoor and the corresponding partial hidden access structure (i.e., the access structure without keyword values) to the designated cloud server. The latter performs the testing operations between each ciphertext and the trapdoor using its private key, and forwards the matching ciphertexts to the data user.

As mentioned earlier, a ciphertext created by a data owner consists of two parts: the encrypted document generated using an encryption scheme and the encrypted keywords generated using our SE scheme. From now on, we only consider the latter part of the encrypted document, and ignore the first part since it is out of the scope of this paper.

In summary, the design goals of our expressive SE scheme are fourfold.

- **Expressiveness.** The proposed scheme should support keyword access structures expressed in any Boolean formula with AND and OR gates.
- **Efficiency.** The proposed scheme should be adequately efficient in terms of computation, communication and storage for practical applications.
- **Keyword privacy.** First, a ciphertext without its corresponding trapdoors should not disclose any information about the keyword values it contains to the cloud server and outsiders. Second, a trapdoor should not leak information on keyword values to any outside attackers without the private key of the designated cloud server. We capture this notion of security for the SE scheme in terms of semantic security to ensure that encrypted data does not reveal any information about the keyword values, which we call "selective indistinguishability against chosen keyword-set attack (selective IND-CKA security)" (See Appendix A).
- **Provable security.** The security of the proposed scheme should be formally proved under the standard model rather than the informal analysis.

3.2 Threat Model

We assume that the trapdoor generation centre is a trusted entity. The cloud server is assumed to be "honest-but-curious", i.e., it will honestly follow the protocol but it is also curious to learn any private information from the data stored in the cloud. Data owners are assumed to honestly store their data, while data users are not trusted, and they can even collude with a malignant cloud server in order to discover private information of other parties. We assume that the trusted trapdoor generation centre is equipped with a separate authentication mechanism to verify data users before issuing trapdoors to users. Also, we assume that all adversaries have bounded computational capability, so they cannot break the aforementioned difficult problems.

3.3 Construction

In the system, the trusted trapdoor generation centre is given a public parameter and a master private key generated by the Setup algorithm, and uses the Trapdoor algorithm to generate a trapdoor $T_{\mathbb{M}}$ for some keyword set associated with an access structure ($\mathbb{M}$, ρ, $\{W_{\rho(i)}\}$) at the request of a privileged data user, where $\mathbb{M}$ is an access matrix, ρ is the function that associates the rows of $\mathbb{M}$ to the generic names of keywords, and $\{W_{\rho(i)}\}$ are the corresponding keyword values[5]. The cloud server is given a public and private key pair created by the sKeyGen algorithm, and will input the trapdoor given by a data user and its private key to the Test algorithm to determine whether a document contains the keywords satisfying the keyword access structure ($\mathbb{M}$, ρ, $\{W_{\rho(i)}\}$) specified by the data user.

Let G be a group of prime order p with a generator g, and $\hat{e} : G \times G \to G_1$ be the bilinear map. On the basis of the KP-ABE scheme proposed by Rouselakis and Waters [18], which we will refer to as the Rouselakis-Waters KP-ABE scheme, we describe our expressive and unbounded SE system in the prime-order groups as follows.

- Setup. This algorithm takes the security parameter 1^λ as input. It randomly chooses a group G of prime order p, a generator g and random group elements u, h, $w \in G$. Also, it randomly chooses α, d_1, d_2, d_3, $d_4 \in Z_p^*$, and computes $g_1 = g^{d_1}$, $g_2 = g^{d_2}$, $g_3 = g^{d_3}$, $g_4 = g^{d_4}$. Finally, it publishes the public parameter $pars = (H, g, u, h, w, g_1, g_2, g_3, g_4, \hat{e}(g,g)^\alpha)$, where H is a collision-resistant hash function that maps elements in G_1 to elements in G, and keeps the master private key $msk = (\alpha, d_1, d_2, d_3, d_4)$.
- sKeyGen. This algorithm takes the public parameter $pars$ as input. It randomly chooses $\gamma \in Z_p^*$, and outputs the public and private key pair (pk_s, sk_s) = (g^γ, γ) for the server.
- Trapdoor. This algorithm takes the public parameter $pars$, the server public key pk_s, the master private key msk and an LSSS access structure ($\mathbb{M}$, ρ, $\{W_{\rho(i)}\}$)[6] as input, where $\mathbb{M}$ is an $l \times n$ matrix over Z_p, the function ρ associates the rows of $\mathbb{M}$ to generic keyword names, and $\{W_{\rho(i)}\}$ are the corresponding keyword values. Let $\mathbb{M}_i$ be the i-th row of $\mathbb{M}$ for $i \in \{1, ..., l\}$, and $\rho(i)$ be the keyword name associated with this row by the mapping ρ. It randomly chooses a vector $\overrightarrow{y} = (\alpha, y_2, ..., y_n)^\perp$ where $y_2, ..., y_n \in Z_p$, r, $r' \in Z_p$, $t_{1,1}$, $t_{1,2}$, ..., $t_{l,1}$, $t_{l,2} \in Z_p$, computes $T = g^r$, $T' = g^{r'}$, and outputs the trapdoor $T_{\mathbb{M},\rho} = \big((\mathbb{M}, \rho), T, T', \{T_{i,1}, T_{i,2}, T_{i,3}, T_{i,4}, T_{i,5}, T_{i,6}\}_{i\in[1,l]}\big)$ as
$$T_{i,1} = g^{v_i} w^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}},$$
$$T_{i,2} = H(\hat{e}(pk_s, T')^r) \cdot g^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}},$$
$$T_{i,3} = ((u^{W_{\rho(i)}} h)^{t_{i,1}})^{-d_2},\ T_{i,4} = ((u^{W_{\rho(i)}} h)^{t_{i,1}})^{-d_1},$$
$$T_{i,5} = ((u^{W_{\rho(i)}} h)^{t_{i,2}})^{-d_4},\ T_{i,6} = ((u^{W_{\rho(i)}} h)^{t_{i,2}})^{-d_3},$$
where $v_i = \mathbb{M}_i \cdot \overrightarrow{y}$ is the share associated with the row $\mathbb{M}_i$ of the access matrix $\mathbb{M}$. Note that only ($\mathbb{M}$, ρ) is included in the trapdoor $T_{\mathbb{M},\rho}$.
- Encrypt. This algorithm takes the public parameter $pars$ and a keyword set **W** (each keyword is denoted as $N_i = W_i$, where N_i is the generic keyword name and W_i is the corresponding keyword value) as input. Let m be the size of **W**, and W_1, ..., $W_m \in Z_p$ be

5. To be distinguishable from the keyword value W_i in a ciphertxt, we use $W_{\rho(i)}$ to denote the keyword value in a trapdoor.

6. For the details about how to convert a boolean formula into an equivalent LSSS matrix, please refer to [33].

the values of **W**. It randomly chooses μ, $s_{1,1}$, $s_{1,2}$, ..., $s_{m,1}$, $s_{m,2}$, z_1, ..., $z_m \in Z_p$, and outputs a ciphertext CT $= \big(C, D, \{(D_i, E_{i,1}, E_{i,2}, F_{i,1}, F_{i,2})\}_{i\in[1,m]}\big)$ as

$$C = \hat{e}(g,g)^{\alpha\mu}, \quad D = g^{\mu},$$
$$D_i = w^{-\mu}(u^{W_i}h)^{z_i}, \quad E_{i,1} = {g_1}^{z_i - s_{i,1}},$$
$$E_{i,2} = {g_2}^{s_{i,1}}, \quad F_{i,1} = {g_3}^{z_i - s_{i,2}}, \quad F_{i,2} = {g_4}^{s_{i,2}}.$$

Note that in the implementation, to efficiently conduct keyword search, the ciphertext will be stored along with the generic names $\{N_i\}$ corresponding to keyword values $\{W_i\}$. Thus, before performing the Test algorithm on the encrypted keyword values, the matching on the keyword names will be executed, thereby reducing the searching time.

- Test. This algorithm takes the public parameter $pars$, the server private key sk_s, a ciphertext $\big(C, D, \{(D_i, E_{i,1}, E_{i,2}, F_{i,1}, F_{i,2})\}\big)$ on a keyword set **W** and a trapdoor $T_{\mathbb{M},\rho}$ associated with an access structure ($\mathbb{M}$, ρ, $\{W_{\rho(i)}\}$) as input. It calculates $I_{\mathbb{M},\rho}$ from ($\mathbb{M}$, ρ), which is a set of minimum subsets satisfying ($\mathbb{M}$, ρ). It then checks whether there is an $\mathcal{I} \in I_{\mathbb{M},\rho}$ satisfying

$$\prod_{i\in\mathcal{I}} \Big(\hat{e}(D, T_{i,1})\hat{e}(D_i, \frac{T_{i,2}}{H(\hat{e}(T,T')^{\gamma})})\hat{e}(E_{i,1}, T_{i,3})$$
$$\hat{e}(E_{i,2}, T_{i,4})\hat{e}(F_{i,1}, T_{i,5})\hat{e}(F_{i,2}, T_{i,6})\Big)^{w_i} = C,$$

where $\sum_{i\in\mathcal{I}} w_i \mathbb{M}_i = (1, 0, ..., 0)$. It outputs 0 if no element in $I_{\mathbb{M},\rho}$ satisfying this equation or 1 otherwise.

Remarks. In our construction, the term g^{z_i} in the original construction [18] is split into ${g_1}^{z_i - s_{i,1}}$ and ${g_2}^{s_{i,1}}$. Thus, W_i is subtly hidden from the ciphertext. To see this, we have

$$\hat{e}(D_i, g_1) = \hat{e}(w, D)^{-1} \cdot \hat{e}(u^{W_i}h, {g_1}^{z_i}),$$

where the term ${g_1}^{z_i}$ is embedded in ${g_1}^{z_i - s_{i,1}}$, which cannot be computed from ${g_1}^{z_i - s_{i,1}}$ without knowing the value of ${g_1}^{s_{i,1}}$. Nevertheless, given ${g_2}^{s_{i,1}}$, it is difficult to compute the value of ${g_1}^{s_{i,1}}$. Similarly, the result works with ${g_3}^{z_i - s_{i,2}}$, ${g_4}^{s_{i,2}}$ as well. Note that some redundant elements as g_3, g_4, $T_{i,5}$, $T_{i,6}$, $F_{i,1}$, $F_{i,2}$ are introduced in our scheme to make the proof go smoothly.

3.4 Correctness

If the keyword set **W** embedded in a ciphertext satisfies the access structure associated with the trapdoor, we will have $\sum_{i\in\mathcal{I}} v_i w_i = \alpha$. Therefore,

$$\begin{aligned}
&\prod_{i\in\mathcal{I}} \Big(\hat{e}(D, T_{i,1})\hat{e}(D_i, \frac{T_{i,2}}{H(\hat{e}(T,T')^{\gamma})})\hat{e}(E_{i,1}, T_{i,3})\Big)^{w_i} \\
&\quad \Big(\hat{e}(E_{i,2}, T_{i,4})\hat{e}(F_{i,1}, T_{i,5})\hat{e}(F_{i,2}, T_{i,6})\Big)^{w_i} \\
= &\prod_{i\in\mathcal{I}} \hat{e}(g^{\mu}, g^{v_i} w^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}})^{w_i} \\
&\cdot \hat{e}(w^{-\mu}(u^{W_i}h)^{z_i}, g^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}})^{w_i} \\
&\cdot \hat{e}({g_1}^{z_i - s_{i,1}}, ((u^{W_{\rho(i)}}h)^{t_{i,1}})^{-d_2})^{w_i} \\
&\cdot \hat{e}({g_2}^{s_{i,1}}, ((u^{W_{\rho(i)}}h)^{t_{i,1}})^{-d_1})^{w_i} \\
&\cdot \hat{e}({g_3}^{z_i - s_{i,2}}, ((u^{W_{\rho(i)}}h)^{t_{i,2}})^{-d_4})^{w_i} \\
&\cdot \hat{e}({g_4}^{s_{i,2}}, ((u^{W_{\rho(i)}}h)^{t_{i,2}})^{-d_3})^{w_i} \\
= &\ \hat{e}(g,g)^{\mu \sum_{i\in\mathcal{I}} v_i w_i} = \hat{e}(g,g)^{\alpha\mu}
\end{aligned}$$

3.5 Security Proof

Theorem 1. Under the decisional BDH assumption, the $(q-2)$ assumption and the decisional linear assumption, our scheme is selectively indistinguishable under chosen keyword-set attacks (selective IND-CKA security).

Proof. The details of the selective IND-CKA security definition and its proof are given in Appendix B. The proof is divided into two parts, depending on the role of the adversary. In the first part, the adversary is assumed to be an outside attacker, and in the second part, the adversary is assumed to be the cloud server who performs search operations.

4 Discussion and Analysis

In this section, we discuss the properties as well as extensions of our expressive SE scheme.

4.1 Keyword Privacy

Keyword Value Guessing Attacks on Ciphertexts. Below we briefly review the encryption algorithm of the KP-ABE scheme in [18], and then show that there exists a keyword value guessing attack if it is directly transformed into a searchable encryption scheme.

Encrypt. Let m denote the size of **W**, and W_1, ..., $W_m \in Z_p$ be the specific values of **W**. It randomly chooses μ, z_1, ..., $z_m \in Z_p$, and outputs a ciphertext CT $= \big(C, D, \{(C_i, D_i)\}_{i\in[1,m]}\big)$.

$$C = \hat{e}(g,g)^{\alpha\mu}, \quad D = g^{\mu},$$
$$\forall i \in [m] \quad C_i = w^{-\mu}(u^{W_i}h)^{z_i}, \; D_i = g^{z_i},$$

where g, u, h, w, $\hat{e}(g,g)^{\alpha}$ are the public parameters.

Given a ciphertext CT $= \big(C, D, \{(C_i, D_i)\}_{i\in[1,m]}\big)$, an adversary can easily determine whether a keyword value W_i' is incorporated in the ciphertext by checking whether the following equation holds.

$$\hat{e}(C_i, g) = \hat{e}(w^{-1}, D) \cdot \hat{e}(u^{W_i'}h, D_i).$$

In order to prevent such attacks, in our construction, we use a "linear splitting" technique [20] on each keyword value related component of the ciphertext, and then rerandomize the components upon each keyword value in the trapdoor. The former step prevents keyword value guessing attacks to the ciphertext while the latter step allows the trapdoor to be used for testing keyword values in the ciphertext.

Keyword Value Guessing Attacks on Trapdoors. Concerning this security requirement, we need to tackle two problems in our construction. First, keywords associated with a trapdoor must be hidden from the access structure. We address this problem by separating each keyword into a generic name and a keyword value, i.e., each keyword is in the form of "generic name = keyword value", and a partial hidden access structure, i.e., the full access structure with keyword values being removed (See Fig. 1) is incorporated in a trapdoor and given to the designated cloud server. Second, the entire trapdoor should be immune to the offline keyword value guessing attacks [25]. In our SE system, we resort to a weaker security notion by requiring that a trapdoor will not disclose information about the keyword values in the ciphertext to an adversary excluding the cloud server who executes the searching operations. We assign a designated cloud server [22] to conduct search and equip it with a public and private key pair. Since the components in a trapdoor are tied with the public key of the server, only the designated cloud server with the corresponding private key is capable to learn the keyword values hidden in the trapdoor by performing offline guessing attacks.

4.2 Unbounded keyword search

In "small universe" KP-ABE constructions [18], the size of the keyword space were polynomially bounded in the security parameter and the keywords were fixed at the setup phase. Moreover, the sizes of the public parameters grow linearly with the number of keywords [8], [14], [15]. On the contrary, in "large universe" constructions, the size of the keyword space can be exponentially large, so it is much more desirable in the real-world applications. Our construction of the expressive SE scheme inherits the advantages of the Rouselakis-Waters scheme [18]. Thus, it is straightforward to see that in our SE scheme, the size of the public parameter is immutable with the number of keywords, and the number of the keywords allowed for the system is unlimited and can be freely set.

4.3 Extensions

Our expressive SE system can be extended in several ways.

- Expressive searchable encryption for the range search. Range search is an important requirement for searchable encryption in many applications. By defining keywords in a hierarchical manner as shown in [27], we can directly expand our SE system to support a class of simple range search [27]. Take a keyword name "Age" with keyword values from 0 to 100 as an example. The path of the leaf node "11-20" is ("0-100", "0-30", "11-20"), and "0-30", "0-10" are simple ranges from level-2 and level-3, respectively.
- Anonymous KP-ABE. Our SE system is built by anonymizing the Rouselakis-Waters KP-ABE scheme [18]. Therefore, our scheme can be easily extended to obtain an unbounded and anonymous KP-ABE scheme in the prime-order group without random oracles, in which an adversary, given a ciphertext, cannot learn any information about the associated attribute set.
- Anonymous hierarchical identity-based encryption (HIBE). The Rouselakis-Waters KP-ABE scheme in [18] can be converted to an HIBE scheme using non-repeating identities, "AND" policies and delegation capabilities [19]. Since our SE scheme can be used to construct an anonymous KP-ABE scheme, it can be further converted to an anonymous HIBE scheme using the same method as in [19].

5 Performance Analysis

We implement our construction of expressive SE in the prime-order group in Charm [39], which is a programming environment for rapid prototyping of cryptographic primitives. In this section, we compare the computational cost, communication and storage overhead of our scheme with other existing schemes.

5.1 Comparison

Let $|pars|$, $|msk|$, $|\mathrm{CT}|$, $|T_{\mathbb{M},\rho}|$, $|\mathbb{M}|$ be the sizes of the public parameter, the master private key, the ciphertext, the trapdoor and the access structure, respectively. Let k be the length of the vector corresponding to the ciphertext in [16], l be the number of keywords in an access structure, n be the maximum number of keywords allowed for the system, and m be the size of a keyword set ascribed to a ciphertext. Denote E as an exponentiation operation, P as a pairing operation, χ_1 as the number of elements in $I_{\mathbb{M},\rho} = \{\mathcal{I}_1, ..., \mathcal{I}_{\chi_1}\}$, χ_2 as $|\mathcal{I}_1| + ... + |I_{\chi_1}|$, and χ_3 as the number of primed keywords [14] in a search predicate.

TABLE 2
Comparison of Storage and Communication Overhead

	Public parameter $|pars|$	Master private key $|msk|$	Trap-door $|T_{\mathbb{M},\rho}|$	Cipher-text $|\mathrm{CT}|$
KSW13 [16]	$2k+3$	$2k+4$	$2k+1+|\mathbb{M}|$	$2k+1$
LZDLC13 [8]	$n+5$	$n+4$	$2l+|\mathbb{M}|$	$m+2$
LHZF14 [14]	$n+4$	$n+2$	$3l+|\mathbb{M}|$	$m+2$
Our Scheme	9	5	$6l+|\mathbb{M}|$	$5m+2$

We compare our searchable encryption system with the other three known expressive SE schemes [8], [14], [16] in Table 2 which are all constructed over composite order groups. From Table 2, it is not difficult to see that our construction is the only one that supports unbounded number of keywords in the expressive keyword search systems. Note that our scheme is measured in terms of number of elements in prime order groups while the other three schemes are measured in terms of number of elements in composite order groups. According to the analysis in [40][7], in terms of the pairing-friendly elliptic curves, prime order groups have a clear advantage in the parameter sizes over composite order groups.

TABLE 3
Comparison of Computation Overhead.

	KSW13 [16]	LZDLC13 [8]	LHZF14 [14]	Ours
Trap-door	$6k \cdot \mathrm{E}$	$4l \cdot \mathrm{E}$	$4l \cdot \mathrm{E}$	$16l \cdot \mathrm{E} + \mathrm{E}$
Enc.	$4k \cdot \mathrm{E} + \mathrm{E}$	$2(m+1) \cdot \mathrm{E}$	$(m+2) \cdot \mathrm{E} + \mathrm{P}$	$7m \cdot \mathrm{E} + 2 \cdot \mathrm{E}$
Test	$2k \cdot \mathrm{P} + \mathrm{P}$	$\leq \chi_2 \cdot \mathrm{E} + 2\chi_2 \cdot \mathrm{P}$	$\leq \chi_2 \cdot \mathrm{E} + 2\chi_2 \cdot \mathrm{P} + 2\chi_3 \cdot \mathrm{P}$	$\leq \chi_2 \cdot \mathrm{E} + \mathrm{E} + \mathrm{P} + 6\chi_2 \cdot \mathrm{P}$
Group Order	Composite	Composite	Composite	Prime

In Table 3, we compare the computational costs incurred in the systems from [8], [14], [16] and our system. It is worth noticing that as mentioned in [17], "a Tate pairing on a 1024-bit composite-order elliptic curve is roughly 50 times slower than the same pairing on a comparable prime-order curve,

7. See Table 3 in [40] for the results.

and this performance gap will only get worse at higher security levels". Therefore, although our SE system requires more exponentiation and pairing operations than the other systems, it is far more computationally efficient than the other three schemes.

5.2 Experimental Results

We implement our scheme in Charm [39][8], which is a framework developed to facilitate rapid prototyping of cryptographic schemes and protocols. Based on the Python programming language, Charm enables one to implement a cryptographic scheme with very few lines of code, significantly reducing development time. Meanwhile, computationally intensive mathematical operations are implemented with native modules, so the overhead due to Python in Charm is less than 1%. Since all Charm routines are designed under the asymmetric groups, our construction is transformed to the asymmetric setting before the implementation. That is, three groups G, $\hat{G}$ and G_1 are used and the pairing $\hat{e}$ is a function from $G \times \hat{G}$ to G_1. Notice that it has been stated in [18] that the assumptions and the security proofs can be converted to the asymmetric setting in a generic way.

We use Charm of version charm-0.43 and Python 3.4 in our implementation. Along with charm-0.43, we install the latest PBC library for underlying cryptographic operations. Our experiments run on an all-in-one desktop computer with Intel Core i7-4785T CPU (4 core 2.20GHz) and 8GB RAM running 64-bit Ubuntu 15.10.

The computational costs of the Setup and sKeyGen algorithms are straightforward, and we focus on the computational costs of the Trapdoor, Encrypt and Test algorithms. In our experiments, a set of keywords is generated, of which every keyword contains a generic name such as "Illness", "Position", "Affiliation" and a keyword value such as "Diabetes", "Doctor", and "City Hospital". For the sake of simple implementation, we use integers to denote keyword values, e.g., a keyword as "Illness = 6" is expressed by "Illness = Diabetes". In this way, we generate a random set of keywords containing 10 to 50 keywords, and use them to encrypt 5,000 documents. We then remove the keyword values in the ciphertexts such that they contain only generic names of keywords like "Illness", "Position", as specified in our concrete construction.

Thereafter, we randomly choose 2 to 10 keywords to form a random access structure. The number of keywords in a searching query is normally less than 10, according to the searching query logs of search engines [41]. The policy tree is formed such that for any interior node the difference on the node number of its left branch and that of its right branch is less than 2. We generate 50 different access policy trees, 10 for each different number of keywords, and create a trapdoor for each policy tree. We also remove the keyword value information from the trapdoors. So the policy tree in a trapdoor contains only keyword names, e.g., (("Illness" AND "Position") OR "Affiliation").

8. For the explicit information on Charm, please refer to [39]. Note that since it has been clearly shown in [18], [40] that the efficiency of schemes in composite-order groups is much worse than that of schemes in prime-order groups, we will not implement those schemes in composite-order groups (e.g., [8], [14], [16]). In addition, the current version Charm does not support cryptographic schemes in composite-order groups.

ECs(time in ms)	Exp. G	Exp. $\hat{G}$	Exp. G_1	Pairing
SS512	0.194	0.194	0.027	0.881
MNT159	0.068	0.584	0.160	3.148
MNT201	0.101	0.762	0.207	4.194
MNT224	0.131	0.968	0.252	5.169

Fig. 2. Computational costs for the group operations and pairings over different elliptic curves on a desktop with 2.2GHz 4 core CPU.

Also, we take each trapdoor to conduct search over the ciphertexts. For any combination of the keyword names in the ciphertext that satisfies the access policy of the trapdoor, our keyword search scheme runs the Test algorithm to further confirm whether it is an exact match.

All these experiments are conducted over 4 different elliptic curves: SS512, MNT159, MNT201 and MNT224, of which SS512 is a supersingular elliptic curve with the bilinear pairing on it being symmetric Type 1 pairing, and the pairings on the other three curves are asymmetric Type 3 pairings. These four curves provides security levels of 80-bit, 80-bit, 100-bit and 112-bit, respectively. The computation time for the exponentiation and pairing calculation over the four curves are listed in Fig. 2.

Fig. 3 shows the computational overhead for generating trapdoors containing 2 keywords to 10 keywords, from which we can see that the computation time for the trapdoor generation is almost linear to the number of keywords associated with the access structure in the trapdoor. The MNT curves with higher security levels have longer computation time, so MNT224 has higher computation cost among all curves. The computation time of SS512 is close to that of MNT224 due to its higher exponentiation cost over G. The computation time of generating a trapdoor with 10 keywords is only 0.22s for MNT224, which is quite modest for a powerful trapdoor generation centre.

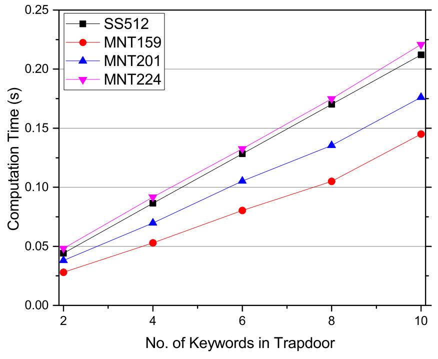


Fig. 3. Computational cost of the Trapdoor algorithm for different curves with respect to number of keywords in trapdoor.

Fig. 4 demonstrates the computation time for the Encrypt algorithm over 10 keywords to 50 keywords. As expected in our analysis, it shows that the computation time is approximately linear to the number of keywords used to generate the ciphertext. The MNT curves with higher security levels are more expensive in computation cost, while the encryption cost of SS512 is much less than that of MNT curves. This is due to the fact that $(4m+1)$ exponentiations are done in $\hat{G}$ for the total $(7m+2)$ exponentiations (see Table 3). To encrypt a document with 50 keywords using MNT224 curve, the computation time is about 1.6s, which is acceptable for most applications.

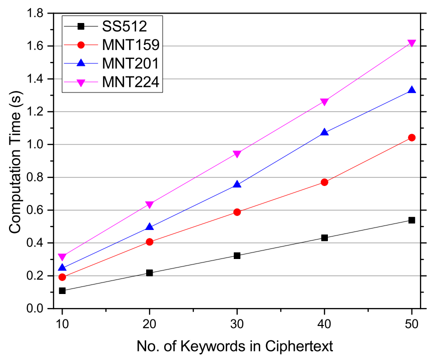


Fig. 4. Computational cost of the Encrypt algorithm for different curves with respect to number of keywords in ciphertext.

The computational cost for the Test algorithm is much more involved. It depends on χ_2, the total number of keywords in all combinations of keywords satisfying the access policy that need to be tried by the cloud server. χ_2 is determined by the access policy and the keywords used to encrypt a document. Fig. 5 shows the relation between the computation time of the Test algorithm and the number of keywords in the access structure of the Trapdoor algorithm. From Fig. 5, it is easy to see that the computation time raises as the number of keywords in the trapdoor and the ciphertext increases. When the trapdoor contains only 2 keywords, the computation time increases quite slowly as keywords in ciphertexts increases. Whilst when the trapdoor has 10 keywords, the computation time grows exponentially as the number of keywords in ciphertexts grows. Among all the curves, SS512 has the best performance, while MNT224 has the highest computational cost. For the 4 curves tested in our experiments, the computation time of searching a document ranges from 50s to 250s for a trapdoor with 10 keywords and a ciphertext with 50 keywords. The computation time can be significantly reduced if keyword search is performed by a powerful cloud server.

6 Conclusions

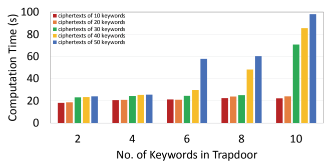
In order to allow a cloud server to search on encrypted data without learning the underlying plaintexts in the public-key setting, Boneh [7] proposed a cryptographic primitive called public-key encryption with keyword search (PEKS). Since then, considering different requirements in practice, e.g., communication overhead, searching criteria and security enhancement, various kinds of searchable encryption systems have been put forth. However, there exist only a few public-key searchable encryption systems that support expressive keyword search policies, and they are all built from the inefficient composite-order groups [17]. In this paper, we focused on the design and analysis of public-key searchable encryption systems in the prime-order groups that can be used to search multiple keywords in expressive searching formulas. Based on a large universe key-policy attribute-based encryption scheme given in [18], we presented an expressive searchable encryption system in the prime-order group which supports expressive access structures expressed in any monotonic Boolean formulas. Also, we proved its security in the standard model, and analyzed its efficiency using computer simulations.

Acknowledgments

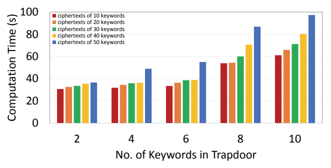
This research work is supported by the Singapore National Research Foundation under the NCR Award Number NRF2014NCR-NCR001-012, and the National Natural Science Foundation of China under the Grant Number 61370027.

References

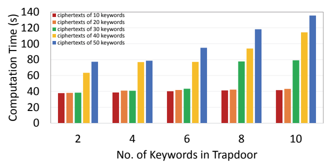
[1] O. Goldreich and R. Ostrovsky, “Software protection and simulation on oblivious rams,” *J. ACM*, vol. 43, no. 3, pp. 431–473, 1996.
[2] D. X. Song, D. Wagner, and A. Perrig, “Practical techniques for searches on encrypted data,” in *2000 IEEE Symposium on Security and Privacy, Berkeley, California, USA, May 14-17, 2000*. IEEE Computer Society, 2000, pp. 44–55.
[3] E. Goh, “Secure indexes,” *IACR Cryptology ePrint Archive*, vol. 2003, p. 216, 2003.
[4] C. Cachin, S. Micali, and M. Stadler, “Computationally private information retrieval with polylogarithmic communication,” in *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, ser. Lecture Notes in Computer Science, vol. 1592. Springer, 1999, pp. 402–414.
[5] G. D. Crescenzo, T. Malkin, and R. Ostrovsky, “Single database private information retrieval implies oblivious transfer,” in *Advances in Cryptology - EUROCRYPT 2000, International Conference on the Theory and Application of Cryptographic Techniques, Bruges, Belgium, May 14-18, 2000, Proceeding*, ser. Lecture Notes in Computer Science, vol. 1807. Springer, 2000, pp. 122–138.
[6] W. Ogata and K. Kurosawa, “Oblivious keyword search,” *J. Complexity*, vol. 20, no. 2-3, pp. 356–371, 2004.
[7] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, “Public key encryption with keyword search,” in *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, ser. Lecture Notes in Computer Science, vol. 3027. Springer, 2004, pp. 506–522.
[8] J. Lai, X. Zhou, R. H. Deng, Y. Li, and K. Chen, “Expressive search on encrypted data,” in *8th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '13, Hangzhou, China - May 08 - 10, 2013*. ACM, 2013, pp. 243–252.
[9] P. Golle, J. Staddon, and B. R. Waters, “Secure conjunctive keyword search over encrypted data,” in *Applied Cryptography and Network Security, Second International Conference, ACNS 2004, Yellow Mountain, China, June 8-11, 2004, Proceedings*, ser. Lecture Notes in Computer Science, vol. 3089. Springer, 2004, pp. 31–45.
[10] D. J. Park, K. Kim, and P. J. Lee, “Public key encryption with conjunctive field keyword search,” in *Information Security Applications, 5th International Workshop, WISA 2004, Jeju Island, Korea, August 23-25, 2004, Revised Selected Papers*, ser. Lecture Notes in Computer Science, vol. 3325. Springer, 2004, pp. 73–86.
[11] Y. H. Hwang and P. J. Lee, “Public key encryption with conjunctive keyword search and its extension to a multi-user system,” in *Pairing-Based Cryptography - Pairing 2007, First International Conference, Tokyo, Japan, July 2-4, 2007, Proceedings*, ser. Lecture Notes in Computer Science, vol. 4575. Springer, 2007, pp. 2–22.
[12] B. Zhang and F. Zhang, “An efficient public key encryption with conjunctive-subset keywords search,” *J. Network and Computer Applications*, vol. 34, no. 1, pp. 262–267, 2011.



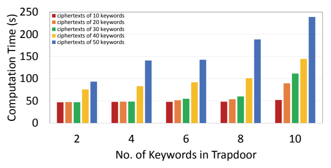
(a) SS512



(b) MNT159



(c) MNT201



(d) MNT224

Fig. 5. Experimental results for the Test algorithm over different elliptic curves.

[13] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings*, ser. Lecture Notes in Computer Science, vol. 4392. Springer, 2007, pp. 535–554.

[14] Z. Lv, C. Hong, M. Zhang, and D. Feng, "Expressive and secure searchable encryption in the public key setting," in *Information Security - 17th International Conference, ISC 2014, Hong Kong, China, October 12-14, 2014. Proceedings*, ser. Lecture Notes in Computer Science, vol. 8783. Springer, 2014, pp. 364–376.

[15] J. Shi, J. Lai, Y. Li, R. H. Deng, and J. Weng, "Authorized keyword search on encrypted data," in *Computer Security - ESORICS 2014 - 19th European Symposium on Research in Computer Security, Wroclaw, Poland, September 7-11, 2014. Proceedings, Part I*, ser. Lecture Notes in Computer Science, vol. 8712. Springer, 2014, pp. 419–435.

[16] J. Katz, A. Sahai, and B. Waters, "Predicate encryption supporting disjunctions, polynomial equations, and inner products," *J. Cryptology*, vol. 26, no. 2, pp. 191–224, 2013.

[17] D. M. Freeman, "Converting pairing-based cryptosystems from composite-order groups to prime-order groups," in *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30 - June 3, 2010. Proceedings*, ser. Lecture Notes in Computer Science, vol. 6110. Springer, 2010, pp. 44–61.

[18] Y. Rouselakis and B. Waters, "Practical constructions and new proof methods for large universe attribute-based encryption," in *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013*. ACM, 2013, pp. 463–474.

[19] A. B. Lewko and B. Waters, "Unbounded HIBE and attribute-based encryption," in *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, ser. Lecture Notes in Computer Science, vol. 6632, 2011, pp. 547–567.

[20] X. Boyen and B. Waters, "Anonymous hierarchical identity-based encryption (without random oracles)," in *Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006, Proceedings*, ser. Lecture Notes in Computer Science, vol. 4117. Springer, 2006, pp. 290–307.

[21] J. Lai, R. H. Deng, and Y. Li, "Expressive CP-ABE with partially hidden access structures," in *7th ACM Symposium on Information, Compuer and Communications Security, ASIACCS '12, Seoul, Korea, May 2-4, 2012*. ACM, 2012, pp. 18–19.

[22] H. S. Rhee, J. H. Park, W. Susilo, and D. H. Lee, "Improved searchable public key encryption with designated tester," in *Proceedings of the 2009 ACM Symposium on Information, Computer and Communications Security, ASIACCS 2009, Sydney, Australia, March 10-12, 2009*. ACM, 2009, pp. 376–379.

[23] M. Bellare, A. Boldyreva, and A. O'Neill, "Deterministic and efficiently searchable encryption," in *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*, ser. Lecture Notes in Computer Science, vol. 4622. Springer, 2007, pp. 535–552.

[24] C. Gu, Y. Zhu, and H. Pan, "Efficient public key encryption with keyword search schemes from pairings," in *Information Security and Cryptology, Third SKLOIS Conference, Inscrypt 2007, Xining, China, August 31 - September 5, 2007, Revised Selected Papers*, ser. Lecture Notes in Computer Science, vol. 4990. Springer, 2007, pp. 372–383.

[25] J. Baek, R. Safavi-Naini, and W. Susilo, "Public key encryption with keyword search revisited," in *Computational Science and Its Applications - ICCSA 2008, International Conference, Perugia, Italy, June 30 - July 3, 2008, Proceedings, Part I*, ser. Lecture Notes in Computer Science, vol. 5072. Springer, 2008, pp. 1249–1259.

[26] Q. Tang and L. Chen, "Public-key encryption with registered keyword search," in *Public Key Infrastructures, Services and Applications - 6th European Workshop, EuroPKI 2009, Pisa, Italy, September 10-11, 2009, Revised Selected Papers*, ser. Lecture Notes in Computer Science, vol. 6391. Springer, 2009, pp. 163–178.

[27] M. Li, S. Yu, N. Cao, and W. Lou, "Authorized private keyword search over encrypted data in cloud computing," in *2011 International Conference on Distributed Computing Systems, ICDCS 2011, Minneapolis, Minnesota, USA, June 20-24, 2011*. IEEE Computer Society, 2011, pp. 383–392.

[28] H. S. Rhee, J. H. Park, and D. H. Lee, "Generic construction of designated tester public-key encryption with keyword search," *Inf. Sci.*, vol. 205, pp. 93–109, 2012.

[29] W. Yau, R. C. Phan, S. Heng, and B. Goi, "Keyword guessing attacks on secure searchable public key encryption schemes with a designated tester," *Int. J. Comput. Math.*, vol. 90, no. 12, pp. 2581–2587, 2013.

[30] E. Shen, E. Shi, and B. Waters, "Predicate privacy in encryption systems," in *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, ser. Lecture Notes in Computer Science, vol. 5444. Springer, 2009, pp. 457–473.

[31] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *CRYPTO*, ser. Lecture Notes in Computer Science, vol. 2139. Springer-Verlag, 2001, pp. 213–219.

[32] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," in *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, ser. Lecture Notes in Computer Science, vol. 3152. Springer, 2004, pp. 41–55.

[33] A. B. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, ser. Lecture Notes in Computer Science, vol. 6632. Springer, 2011, pp. 568–588.

[34] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Public Key Cryptography - PKC 2011 - 14th International Conference on Practice and Theory in Public Key Cryptography, Taormina, Italy, March 6-9, 2011. Proceedings*, ser. Lecture Notes in Computer Science, vol. 6571. Springer, 2011, pp. 53–70.

[35] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, Ioctober 30 - November 3, 2006*. ACM, 2006, pp. 89–98.

[36] R. Ostrovsky, A. Sahai, and B. Waters, "Attribute-based encryption with non-monotonic access structures," in *Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, October 28-31, 2007*. ACM, 2007, pp. 195–203.

[37] A. B. Lewko, A. Sahai, and B. Waters, "Revocation systems with very small private keys," in *31st IEEE Symposium on Security and Privacy, S&P 2010, 16-19 May 2010, Berleley/Oakland, California, USA*. IEEE Computer Society, 2010, pp. 273–285.

[38] A. Beimel, "Secure schemes for secret sharing and key distribution," Ph.D. dissertation, Israel Institute of Technology, Israel Institute of Technology, June 1996.

[39] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin, "Charm: a framework for rapidly prototyping cryptosystems," *J. Cryptographic Engineering*, vol. 3, no. 2, pp. 111–128, 2013.
[40] A. Guillevic, "Comparing the pairing efficiency over composite-order and prime-order elliptic curves," in *Applied Cryptography and Network Security - 11th International Conference, ACNS 2013, Banff, AB, Canada, June 25-28, 2013. Proceedings*, ser. Lecture Notes in Computer Science, vol. 7954. Springer, 2013, pp. 357–372.
[41] L. Yang, Q. Mei, K. Zheng, and D. A. Hanauer, "Query log analysis of an electronic health record search engine," in *Proc. of AMIA Annual Symposium*, 2011, p. 915C924.

Appendix A System Framework and Security Definition

Our expressive keyword search system consists of setup algorithm Setup, server key generation algorithm sKeyGen, trapdoor generation algorithm Trapdoor, encryption algorithm Encrypt and testing algorithm Test.

- Setup$(1^\lambda) \rightarrow (pars,\ msk)$. Taking the security parameter λ as the input, this setup algorithm outputs the public parameter $pars$ and the master private key msk for the system. This algorithm is run by the trapdoor centre.
- sKeyGen$(pars) \rightarrow (sk_s,\ pk_s)$. Taking the public parameter $pars$ as the input, the server key generation algorithm outputs a public and private key pair for the designated searching server. This algorithm is run by the trapdoor centre.
- Trapdoor$(pars,\ pk_s,\ msk,\ (\mathbb{M},\ \rho,\ \{W_{\rho(i)}\})) \rightarrow T_{\mathbb{M}}$. Taking the public parameter $pars$, the server public key pk_s and an access structure $(\mathbb{M},\ \rho,\ \{W_{\rho(i)}\})$ over the universe of keywords as the input, this trapdoor generation algorithm generates a trapdoor $T_{\mathbb{M}}$. This algorithm is run by the trapdoor centre.
- Encrypt$(pars,\ \mathbf{W}) \rightarrow$ CT. Taking the public parameter $pars$ and a set of keywords $\mathbf{W}$ as the input, this encryption algorithm outputs a ciphertext CT. This algorithm is run by the data owner.
- Test$(pars,\ sk_s,\ \text{CT},\ T_{\mathbb{M}}) \rightarrow 1/0$. Taking the public parameter $pars$, the server private key sk_s, a ciphertext CT associated with a keywords set $\mathbf{W}$ and a trapdoor $T_{\mathbb{M}}$ for an access structure $(\mathbb{M},\ \rho,\ \{W_{\rho(i)}\})$ as the input, this testing algorithm outputs either 1 when the ciphertext satisfies the access structure of the trapdoor $T_{\mathbb{M}}$ or 0 otherwise. This algorithm is run by the designated server.

We require that a expressive keyword search scheme Π is correct, meaning that for all the sets of keywords $\mathbf{W}$ and access structures $\mathbb{M}$ such that $\mathbb{M}(\mathbf{W}) = 1$, if $(pars,\ msk) \leftarrow$ Setup(1^λ), $(pk_s,\ sk_s) \leftarrow$ sKeyGen$(pars)$, $T_{\mathbb{M}} \leftarrow$ Trapdoor$(pars,\ pk_s,\ msk,\ (\mathbb{M},\ \rho,\ \{W_{\rho(i)}\}))$, CT $\leftarrow$ Encrypt$(pars,\ \mathbf{W})$, then Test$(pars,\ sk_s,\ \text{CT},\ T_{\mathbb{M}}) = 1$.

Following the security model introduced in [7], [25], we give the security definition for an expressive keyword search scheme over encrypted data in terms of the semantic security to ensure that such a scheme does not reveal any information about the keyword values in the ciphertext, which we call "indistinguishability against chosen keyword-set attack (IND-CKA)". Formally, we describe the IND-CKA security in the following game between a challenger algorithm $\mathcal{B}$ and an adversary algorithm $\mathcal{A}$, where algorithm $\mathcal{A}$ is divided into algorithm $\mathcal{A}_1$ (which is assumed to be a designated cloud (searching) server) and algorithm $\mathcal{A}_2$ (which is assumed to be an outside attacker).

1) The security game between algorithm $\mathcal{B}$ and algorithm $\mathcal{A}_1$ is to guarantee that the searching server cannot tell which ciphertext encrypts which set of keywords without obtaining the trapdoors for the access structures that can be satisfied by the keywords associated with the ciphertexts. This is because once the server is given a trapdoor that the keyword set in a ciphertext can satisfy, the server will ascertain that this ciphertext contains at least the keywords associated with the access structure in the given trapdoor.

    - Setup. Algorithm $\mathcal{B}$ runs the Setup algorithm to obtain the public parameter $pars$ and the master private key msk. It gives the public parameter $pars$ to algorithm $\mathcal{A}_1$ and keeps msk to itself. In addition, algorithm $\mathcal{B}$ runs the sKeyGen algorithm to obtain a public and private key pair $(pk_s,\ sk_s)$ for the server. It then gives $(pk_s,\ sk_s)$ to algorithm $\mathcal{A}_1$.
    - Phase 1. Algorithm $\mathcal{A}_1$ adaptively issues queries to algorithm $\mathcal{B}$ for the trapdoors corresponding to the access structures $(\mathbb{M}_1,\ \rho_1\ \{W_{\rho_1(i)}\})$, ..., $(\mathbb{M}_{q_1},\ \rho_{q_1},\ \{W_{\rho_{q_1}(i)}\})$. For each $(\mathbb{M}_j,\ \rho_j,\ \{W_{\rho(i)}\}_j)$ with $j \in [1, q_1]$, algorithm $\mathcal{B}$ runs the Trapdoor algorithm, and sends $T_{\mathbb{M}_j}$ to algorithm $\mathcal{A}_1$.
    - Challenge. Algorithm $\mathcal{A}_1$ outputs two sets of keyword $\mathbf{W}_0^*$, $\mathbf{W}_1^*$ of the same size with the restriction that $\mathbf{W}_0^*$ and $\mathbf{W}_1^*$ satisfy none of the queried trapdoors. Algorithm $\mathcal{B}$ selects a random bit $\beta \in \{0, 1\}$, runs the Encrypt algorithm on $\mathbf{W}_\beta^*$ to obtain the challenge ciphertext CT*, and then forwards CT* to algorithm $\mathcal{A}_1$.
    - Phase 2. Algorithm $\mathcal{A}_1$ continues issuing queries to algorithm $\mathcal{B}$ for the trapdoors corresponding to the access structures $(\mathbb{M}_{q_1+1},\ \rho_{q_1+1},\ \{W_{\rho_{q_1+1}(i)}\})$, ..., $(\mathbb{M}_q,\ \rho_q,\ \{W_{\rho_q(i)}\})$ with the restriction that any $(\mathbb{M}_j,\ \rho_j,\ \{W_{\rho_j(i)}\})$ for $j \in [q_1 + 1, q]$ can be satisfied by neither $\mathbf{W}_0^*$ nor $\mathbf{W}_1^*$.
    - Guess. Algorithm $\mathcal{A}_1$ outputs its guess $\beta' \in \{0, 1\}$ and wins the game if $\beta' = \beta$.

2) The security game between algorithm $\mathcal{B}$ and algorithm $\mathcal{A}_2$ is to ensure that the outsider attacker who has not obtained the searching server's private key cannot determine the set of keyword values associated with the ciphertext even though the attacker gets the trapdoors over the access structures satisfied by the keywords associated with the ciphertexts. This is because the server's public key is embedded in the trapdoors such that no one can determine whether a trapdoor matches the keyword set of a ciphertext without the server's private key.

- Setup. Algorithm $\mathcal{B}$ runs the Setup algorithm to obtain the public parameter $pars$ and the master private key msk. It gives the public parameter $pars$ to algorithm $\mathcal{A}_2$ and keeps msk to itself. Also, algorithm $\mathcal{B}$ runs the sKeyGen algorithm to obtain a public and private key pair (pk_s, sk_s) for the server. It then gives pk_s to algorithm $\mathcal{A}_2$ and keeps sk_s to itself.
- Phase 1. Algorithm $\mathcal{A}_2$ adaptively issues queries to algorithm $\mathcal{B}$ for the trapdoors corresponding to the access structures $(\mathbb{M}_1, \rho_1 \{W_{\rho_1(i)}\})$, ..., $(\mathbb{M}_{q_1}, \rho_{q_1}, \{W_{\rho_{q_1}(i)}\})$. For each $\mathbb{M}_j$ with $j \in [1, q_1]$, algorithm $\mathcal{B}$ runs the trapdoor generation algorithm Trapdoor, and sends $T_{\mathbb{M}_j}$ to algorithm $\mathcal{A}_2$.
- Challenge. Algorithm $\mathcal{A}_2$ outputs two sets of keyword $\mathbf{W}_0^*$, $\mathbf{W}_1^*$ of the same size. Algorithm $\mathcal{B}$ selects a random bit $\beta \in \{0,1\}$, runs the Encrypt algorithm on $\mathbf{W}_\beta^*$ to obtain the challenge ciphertext CT*, and then gives CT* to algorithm $\mathcal{A}_2$.
- Phase 2. Algorithm $\mathcal{A}_2$ continues issuing queries to algorithm $\mathcal{B}$ for the trapdoors corresponding to the access structures $(\mathbb{M}_{q_1+1}, \rho_{q_1+1}, \{W_{\rho_{q_1+1}(i)}\})$, ..., $(\mathbb{M}_q, \rho_q, \{W_{\rho_q(i)}\})$.
- Guess. Algorithm $\mathcal{A}_2$ outputs its guess $\beta' \in \{0,1\}$ and wins the game if $\beta' = \beta$.

For $\mathcal{A} \in \{\mathcal{A}_1, \mathcal{A}_2\}$, an expressive keyword search system Π is IND-CKA secure if the advantage function referring to the security game Game$^{\text{IND}}_{\Pi,\mathcal{A}}$

$$\mathbf{Adv}^{\text{IND}}_{\Pi,\mathcal{A}}(\lambda) \overset{\text{def}}{=} \Pr[\beta = \beta']$$

is negligible in the security parameter λ for any probabilistic polynomial-time (PPT) adversary algorithm $\mathcal{A}$.

In addition, an expressive keyword search system is said to be selectively IND-CKA secure[9] if an Init stage is added before the Setup phase where algorithm $\mathcal{A}$ commits to the challenge keyword sets $\mathbf{W}_0^*$, $\mathbf{W}_1^*$ which it aims to attack.

Appendix B Security Proof of Theorem 1

Proof. The proof is divided into two parts, depending on the role of the adversary. In the first part, the adversary is assumed to be an outside attacker, and in the second part, the adversary is assumed to be the server.

In terms of the first part of the proof, we prove it via a sequence of games, where game Game$_0$ is the same as the original game, and game Game$_1$ is the same as Game$_0$ except that the trapdoors might be generated in a different way. We finish the proof by showing that if there exists an outside adversary algorithm $\mathcal{A}$ that can distinguish game Game$_1$ from game Game$_0$, then we can build a challenger algorithm $\mathcal{B}$ that solves the decisional BDH assumption.

9. Note that selective IND-CKA security is weaker than IND-CKA security, but it is a useful tool in security reduction and is widely used in the cryptographic systems.

- Algorithm $\mathcal{A}$ gives algorithm $\mathcal{B}$ two challenge keyword sets $\mathbf{W}_0^* = \{W_{0,1}^*, ..., W_{0,m}^*\}$ and $\mathbf{W}_1^* = \{W_{1,1}^*, ..., W_{1,m}^*\}$.
- Setup. Algorithm $\mathcal{B}$ runs the Setup algorithm to generate the public parameter and the master private key as required, and sets the public and private key pair for the server as (g^a, a). Also, algorithm $\mathcal{B}$ selects a random bit $\beta \in \{0,1\}$.
- Phase 1. Since algorithm $\mathcal{B}$ knows the master private key, it easily outputs the trapdoor on any access structures as required. If algorithm $\mathcal{A}$ issues a trapdoor generation query on an access structure that can be satisfied by $\mathbf{W}_\beta^*$, algorithm $\mathcal{B}$ computes $T = g^c$, $T' = g^b$, $T_{i,2} = H(\hat{e}(g,g)^{abc})g^{d_1 d_2 t_{i,1} + d_3 d_4 t_{i,2}}$, and generates the other elements of the trapdoor as in the Trapdoor algorithm.
- Challenge. Algorithm $\mathcal{B}$ runs the Encrypt algorithm on $\mathbf{W}_\beta^*$ to obtain the challenge ciphertext CT*, and gives CT* to algorithm $\mathcal{A}$.
- Phase 2. The same as that in Phase 1.
- Guess. Algorithm $\mathcal{A}$ output a guess β' for β.

On the one hand, if $Z = \hat{e}(g,g)^{abc}$, then algorithm $\mathcal{A}$'s view of this simulation is identical to the original game. On the other hand, if Z is randomly chosen from G_1, then algorithm $\mathcal{A}$'s advantage is nil. Therefore, if algorithm $\mathcal{A}$ can discern game Game$_1$ from game Game$_0$ with a non-negligible probability, algorithm $\mathcal{B}$ has a non-negligible probability in breaking the decisional BDH problem.

Concerning the second part of the proof, we prove the security using a sequence of games. For simplicity, we remove the access structure from the ciphertext, and denote $\big(C^*, D^*, \{(D_i^*, E_{i,1}^*, E_{i,2}^*, F_{i,1}^*, F_{i,2}^*)\}\big)$ as the challenge ciphertext given to the adversary during an attack in the real world. Let Z be a random element of G_1, and $\{Z_{i,1}\}$, $\{Z'_{i,1}\}$ be random elements of G. We define the following games which differ on the type of the challenge ciphertext is given by the challenger to the adversary.

- Game$_0$: The challenge ciphertext is CT$_0^* = \big(C^*, D^*, \{(D_i^*, E_{i,1}^*, E_{i,2}^*, F_{i,1}^*, F_{i,2}^*)\}_{i\in[1,m]}\big)$.
- Game$_1$: The challenge ciphertext is CT$_1^* = \big(Z, D^*, \{(D_i^*, E_{i,1}^*, E_{i,2}^*, F_{i,1}^*, F_{i,2}^*)\}_{i\in[1,m]}\big)$.
- Game$_2$: The challenge ciphertext is CT$_2^* = \big(Z, D^*, (D_1^*, Z_{1,1}, E_{1,2}^*, F_{1,1}^*, F_{1,2}^*), \{(D_i^*, E_{i,1}^*, E_{i,2}^*, F_{i,1}^*, F_{i,2}^*)\}_{i\in[2,m]}\big)$.
- ......
- Game$_{m+1}$: The challenge ciphertext is CT$_{m+1}^* = \big(Z, D^*, \{(D_i^*, Z_{i,1}^*, E_{i,2}^*, F_{i,1}^*, F_{i,2}^*)\}_{i\in[1,m]}\big)$.
- Game$_{m+2}$: The challenge ciphertext is CT$_{m+2} = \big(Z, D^*, (D_1^*, Z_{1,1}, E_{1,2}^*, Z'_{1,1}, F_{1,2}^*), \{(D_i^*, Z_{i,1}, E_{i,2}^*, F_{i,1}^*, F_{i,2}^*)\}_{i\in[2,m]}\big)$.
- ......
- Game$_{2m+1}$: The challenge ciphertext is CT$_{2m+1}^* = \big(Z, D^*, \{(D_i^*, Z_{i,1}, E_{i,2}^*, Z'_{i,1}, F_{i,2}^*)\}_{i\in[1,m]}\big)$.

To complete the proof, we will show that the games Game$_0$, Game$_1$, ..., Game$_{2m+1}$ are computationally indistinguishable from each other.

Lemma 1. Under the $(q-2)$ assumption, the advantage for a polynomial time adversary that can distinguish between the games Game$_0$ and Game$_1$ is negligible.

Proof. Assume that there is an adversary algorithm $\mathcal{A}$ that can distinguish Game_0 from Game_1. Then we can build a challenger algorithm $\mathcal{B}$ that can solve the $(q-2)$ problem.

- Init. Algorithm $\mathcal{A}$ gives algorithm $\mathcal{B}$ two challenge keyword sets $\mathbf{W}_0^* = \{W_{0,1}^*, ..., W_{0,m}^*\}$ and $\mathbf{W}_1^* = \{W_{1,1}^*, ..., W_{1,m}^*\}$.
- Setup. In order to generate the public system parameter, algorithm $\mathcal{B}$ implicitly sets $\alpha = xy$. Then algorithm $\mathcal{B}$ randomly chooses $\beta \in \{0, 1\}$, d_1, d_2, d_3, d_4, $\tilde{u}$, $\tilde{h} \in Z_p$, and computes the public parameter $pars = (g, u, h, w, g_1, g_2, g_3, g_4, \hat{e}(g^x, g^y))$ as follows.

$$g = g, \quad w = g^x, \quad g_1 = g^{d_1}, \quad g_2 = g^{d_2},$$
$$g_3 = g^{d_3}, \quad g_4 = g^{d_4}, \quad u = g^{\tilde{u}} \cdot \prod_{i\in[m]} g^{y/b_i^2},$$
$$h = g^{\tilde{h}} \cdot \prod_{i\in[m]} (g^{xz/b_i}) \cdot \prod_{i\in[m]} (g^{y/b_i^2})^{-W_{\beta,i}^*}.$$

- Phase 1 and Phase 2. Algorithm $\mathcal{B}$ has to create the trapdoors for the access structures $(\mathbb{M}, \rho, \{\rho(i)\})$[10] required by algorithm $\mathcal{A}$ that are not satisfied by either $\mathbf{W}_0^*$ or $\mathbf{W}_1^*$.

  Since $\mathbf{W}_\beta^*$ does not satisfy $(\mathbb{M}, \rho)$, there exists a vector $\overrightarrow{w} = (w_1, ..., w_n)^\perp \in Z_p^n$ such that $w_1 = 1$ and $\mathbb{M}_i \cdot \overrightarrow{w} = 0$ for all $i \in [l]$ such that $\rho(i) \in \mathbf{W}_\beta^*$. Algorithm $\mathcal{B}$ computes $\overrightarrow{w}$ using linear algebra. The vector $\overrightarrow{y}$ to be shared is implicitly set as $\overrightarrow{y} = xy\overrightarrow{w} + (0, \tilde{y}_2, ..., \tilde{y}_n)^\perp$, where $\tilde{y}_2, ..., \tilde{y}_n \in Z_p$. This is a properly distributed vector with first component as $xy = \alpha$ and the other components being uniformly random in Z_p. As a result, for each row $i \in [l]$, the share is

$$\begin{aligned} v_i &= \mathbb{M}_i \cdot \overrightarrow{y} = xy(\mathbb{M}_i \cdot \overrightarrow{w}) + (\mathbb{M}_i \cdot (0, \tilde{y}_2, ..., \tilde{y}_n)^\perp) \\ &= xy(\mathbb{M}_i \cdot \overrightarrow{w}) + \tilde{v}_i. \end{aligned}$$

  As mentioned above, for each row i for which $\rho(i) \in \mathbf{W}_\beta^*$, $\mathbb{M}_i \cdot \overrightarrow{w} = 0$. In this case, $v_i = \tilde{v}_i = \mathbb{M}_i \cdot (0, \tilde{y}_2, ..., \tilde{y}_n)^\perp$, which is known to algorithm $\mathcal{B}$, so algorithm $\mathcal{B}$ randomly chooses $t_i \in Z_p$, and outputs $\{T_{i,1}, T_{i,2}, T_{i,3}, T_{i,4}, T_{i,5}, T_{i,6}\}$ as in the Trapdoor algorithm.

  For each row $i \notin \mathbf{W}_\beta^*$, algorithm $\mathcal{B}$ randomly chooses $\tilde{t}_{i,1}$, $\tilde{t}_{i,2} \in Z_p$, and implicitly sets

$$t_{i,1} = \frac{-y}{2d_1d_2}(\mathbb{M}_i \cdot \overrightarrow{w}) + \sum_{j\in[m]} \frac{xzb_j(\mathbb{M}_i \cdot \overrightarrow{w})}{\rho(i) - W_{\beta,j}^*} + \tilde{t}_{i,1},$$
$$t_{i,2} = \frac{-y}{2d_3d_4}(\mathbb{M}_i \cdot \overrightarrow{w}) + \sum_{j\in[m]} \frac{xzb_j(\mathbb{M}_i \cdot \overrightarrow{w})}{\rho(i) - W_{\beta,j}^*} + \tilde{t}_{i,2}.$$

  Note that $t_{i,1}$, $t_{i,2}$ are properly distributed due to $\tilde{t}_{i,1}$, $\tilde{t}_{i,2}$. The intuition behind this is that the exponent y raises the power of w to the secret $\alpha = xy$. Though this also results to the exponents xyz/b_j from h, it can be cancelled by the provided exponents xzb_j on the y/b_j^2 part. Thus, algorithm $\mathcal{B}$ can output the elements of the trapdoor $T_{\mathbb{M},\rho}$ as follows.

$$\begin{aligned} T_{i,1} &= g^{v_i} w^{d_1d_2t_{i,1}+d_3d_4t_{i,2}} \\ &= g^{\tilde{v}_i} \cdot \Big(\prod_{j\in[n]} (g^{x^2zb_j})^{\frac{\mathbb{M}_i \cdot \overrightarrow{w}}{\rho(i)-W_{\beta,j}^*}} \cdot w^{\tilde{t}_{i,1}} \Big)^{d_1d_2} \\ &\quad \cdot \Big(\prod_{j\in[n]} (g^{x^2zb_j})^{\frac{\mathbb{M}_i \cdot \overrightarrow{w}}{\rho(i)-W_{\beta,j}^*}} \cdot w^{\tilde{t}_{i,2}} \Big)^{d_3d_4}. \\ T_{i,2} &= g^{d_1d_2t_{i,1}+d_3d_4t_{i,2}} \\ &= (g^y)^{-\mathbb{M}_i \cdot \overrightarrow{w}} \\ &\quad \cdot \Big(\prod_{j\in[m]} (g^{xzb_j})^{\frac{\mathbb{M}_i \cdot \overrightarrow{w}}{\rho(i)-W_{\beta,j}^*}} \cdot g^{\tilde{t}_{i,1}} \Big)^{d_1d_2}, \\ &\quad \cdot \Big(\prod_{j\in[m]} (g^{xzb_j})^{\frac{\mathbb{M}_i \cdot \overrightarrow{w}}{\rho(i)-W_{\beta,j}^*}} \cdot g^{\tilde{t}_{i,2}} \Big)^{d_3d_4}. \\ T_{i,3} &= ((u^{\rho(i)}h)^{-t_{i,1}})^{-d_2} \\ &= \big((g^y)^{(\mathbb{M}_i \cdot \overrightarrow{w})(\rho(i)\tilde{u}+\tilde{h})}\big)^{-d_2} \\ &\quad \cdot \Big(\prod_{j\in[m]} (g^{xzb_j})^{\frac{-(\rho(i)\tilde{u}+\tilde{h})(\mathbb{M}_i \cdot \overrightarrow{w})}{\rho(i)-W_{\beta,j}^*}} \Big)^{-d_2} \\ &\quad \cdot \Big(\prod_{(j,k)\in[m,m]} (g^{(xz)^2b_k/b_j})^{\frac{-(\mathbb{M}_i \cdot \overrightarrow{w})}{\rho(i)-W_{\beta,k}^*}} \Big)^{-d_2} \\ &\quad \cdot \Big(\prod_{j\in[m]} (g^{y^2/b_j^2})^{(\mathbb{M}_i \cdot \overrightarrow{w})(\rho(i)-W_{\beta,j}^*)} \Big)^{-d_2} \\ &\quad \cdot \Big(\prod_{(j,k)\in[m,m]}^{j\neq k} (g^{\frac{xyzb_k}{b_j^2}})^{\frac{-(\mathbb{M}_i \cdot \overrightarrow{w})(\rho(i)-W_{\beta,j}^*)}{\rho(i)-W_{\beta,k}^*}} \Big)^{-d_2} \\ &\quad \cdot ((u^{\rho(i)}h)^{-\tilde{t}_{i,1}})^{-d_2}. \end{aligned}$$

  Since $T_{i,3}$, $T_{i,4}$, $T_{i,5}$, $T_{i,6}$ have the term $(u^{\rho(i)}h)^{-t_{i,1}}$ in common, and d_1, d_2, d_3 and d_4 are known to algorithm $\mathcal{B}$, algorithm $\mathcal{B}$ can simply compute $T_{i,4}$, $T_{i,5}$, $T_{i,6}$ as $T_{i,3}$. Thus, algorithm $\mathcal{B}$ successfully responds to algorithm $\mathcal{A}$'s trapdoor queries.
- Challenge. To generate a challenge ciphertext, algorithm $\mathcal{B}$ implicitly sets $\mu = z$ from the $q-2$ assumption, and $z_i = b_i$ for every $i \in [m]$. Notice that these parameters are properly distributed since $z, b_1, ..., b_q$ are information theoretically hidden from the view of algorithm $\mathcal{A}$. In addition, algorithm $\mathcal{B}$ randomly chooses $s_{1,1}$, $s_{1,2}$, ..., $s_{m,1}$, $s_{m,2} \in Z_p$. Thus, algorithm $\mathcal{B}$ can calculate and forward the challenge ciphertext $\text{CT}^* = \big(C^*, D^*, E^*, F^*, \{(D_i^*, E_i^*, F_i^*)\}_{i\in[1,m]}\big)$ to algorithm $\mathcal{A}$, where

$$\begin{aligned} &C^* = Z,\ D^* = g^z,\ E_{i,2}^* = {g_2}^{s_{i,1}},\ F_{i,2}^* = {g_4}^{s_{i,2}}, \\ &D_i^* = w^{-\mu}(u^{W_{\beta,i}^*}h)^{z_i} \\ &\quad = \prod_{j\in[m]}^{j\neq i} g^{xzb_i/b_j} \prod_{j\in[m]}^{j\neq i} (g^{yb_i/b_j^2})^{W_{\beta,i}^*-W_{\beta,j}^*} \\ &\qquad \cdot g^{b_i(\tilde{u}W_{\beta,i}^*+\tilde{h})}, \\ &E_{i,1}^* = {g_1}^{z_i-s_{i,1}} = (g^{b_i-s_{i,1}})^{d_1} = (g^{b_i}/g^{s_{i,1}})^{d_1}, \\ &F_{i,1}^* = {g_3}^{z_i-s_{i,2}} = (g^{b_i-s_{i,2}})^{d_1} = (g^{b_i}/g^{s_{i,2}})^{d_3}. \end{aligned}$$

- Guess. Algorithm $\mathcal{A}$ output a guess β' for β.

10. For easy notation, in the rest of the proof, we use $\rho(i)$ to replace $W_{\rho(i)}$ in the construction.

On the one hand, if $Z = \hat{e}(g,g)^{xyz}$, then algorithm $\mathcal{A}$'s view of this simulation is identical to the original game. On the other hand, if Z is randomly chosen from G_1, then algorithm $\mathcal{A}$'s advantage is nil. Therefore, if algorithm $\mathcal{A}$ can distinguish game Game$_1$ from game Game$_0$ with a non-negligible probability, algorithm $\mathcal{B}$ has a non-negligible probability in breaking the $(q-2)$ assumption.

Lemma 2. Under the decisional linear assumption, the advantage for a polynomial time adversary that can distinguish between the games Game$_{m+1}$ and Game$_m$ for $m \in [1, m]$ is negligible.

Proof. Assuming that there is an adversary algorithm $\mathcal{A}$ that can distinguish Game$_m$ from Game$_{m+1}$, we can build a challenger algorithm $\mathcal{B}$ to solve the decisional linear problem.

- Init. Algorithm $\mathcal{A}$ gives algorithm $\mathcal{B}$ two challenge keyword sets $\mathbf{W}_0^* = \{W_{0,1}^*, ..., W_{0,m}^*\}$, $\mathbf{W}_1^* = \{W_{1,1}^*, ..., W_{1,m}^*\}$.
- Setup. In order to generate the public system parameter, algorithm $\mathcal{B}$ implicitly sets $d_1 = x_2$, $d_2 = x_1$. Then algorithm $\mathcal{B}$ randomly chooses d_3, d_4, $\beta \in \{0,1\}$, α, y, $\tilde{w} \in Z_p$, and computes the public parameter $pars = (g, u, h, w, g_1, g_2, g_3, g_4, \hat{e}(g,g)^{\alpha})$.

$$g = g, \quad w = g^{\tilde{w}}, \quad g_1 = g^{x_2}, \quad g_2 = g^{x_1},$$
$$g_3 = g^{x_3}, \quad g_4 = g^{x_4}, \quad u = g^{x_2\alpha},$$
$$h = g^{-x_2\alpha W_{\beta,m}^*} g^y, \quad \hat{e}(g,g)^{\alpha} = \hat{e}(g,g)^{\alpha}.$$

- Phase 1 and Phase 2. In order to create a trapdoor for an access structure $(\mathbb{M}, \rho)$ required by algorithm $\mathcal{A}$ that is satisfied by neither $\mathbf{W}_0^*$ nor $\mathbf{W}_1^*$, algorithm $\mathcal{B}$ performs as follows. It randomly chooses $\overrightarrow{y} = (\alpha, y_2, ..., y_n)^{\perp}$ where $y_2, ..., y_n \in Z_p$. Also, it randomly chooses $t_{1,1}, t_{1,2}, ..., t_{l,1}, t_{l,2} \in Z_p$. For each $i \in [l]$, algorithm $\mathcal{B}$ sets $v_i = \mathbb{M}_i \cdot \overrightarrow{y}$,

$$\tilde{t}_{i,1} = \frac{t_{i,1}\alpha(\rho(i) - W_{\beta,m}^*)}{x_2\alpha(\rho(i) - W_{\beta,m}^*) + y},$$
$$\tilde{t}_{i,2} = t_{i,2} + \frac{yx_1 t_{i,1}}{d_3 d_4(\rho(i) - W_{\beta,m}^*)x_2 + y}.$$

Then it outputs the trapdoor as

$$\begin{aligned} T_{i,1} &= g^{v_i}(g^{x_1 t_{i,1}} g^{t_{i,2} d_3 d_4})^{\tilde{w}} = g^{v_i} w^{d_1 d_2 \tilde{t}_{i,1} + d_3 d_4 \tilde{t}_{i,2}}, \\ T_{i,2} &= g^{x_1 t_{i,1}} g^{t_{i,2} d_3 d_4} = g^{x_1 x_2 \tilde{t}_{i,1} + \tilde{t}_{i,2} d_3 d_4} \\ &= g^{d_1 d_2 \tilde{t}_{i,1} + d_3 d_4 \tilde{t}_{i,2}}, \\ T_{i,3} &= (g^{x_1})^{-\alpha(\rho(i) - W_{\beta,m}^*)t_{i,1}} = ((u^{\rho(i)}h)^{\tilde{t}_{i,1}})^{-d_2}, \\ T_{i,4} &= (g^{x_2})^{-\alpha(\rho(i) - W_{\beta,m}^*)t_{i,1}} = ((u^{\rho(i)}h)^{\tilde{t}_{i,1}})^{-d_1}, \\ T_{i,5} &= (g^{x_1})^{-\alpha(\rho(i) - W_{\beta,m}^*)t_{i,2}} = ((u^{\rho(i)}h)^{\tilde{t}_{i,2}})^{-d_2}, \\ T_{i,6} &= (g^{x_2})^{-\alpha(\rho(i) - W_{\beta,m}^*)t_{i,2}} = ((u^{\rho(i)}h)^{\tilde{t}_{i,2}})^{-d_1}. \end{aligned}$$

- Challenge. To generate a challenge ciphertext, algorithm $\mathcal{B}$ implicitly sets $s_{m,1} = x_3$, $z_m = x_3 + x_4$ from the decisional linear assumption. In addition, algorithm $\mathcal{B}$ randomly chooses μ, $s_{1,1}, ..., s_{m-1,1}, s_{1,2}, ..., s_{m,2} \in Z_p$, $z_1, ..., z_{m-1} \in Z_p$. Thus, algorithm $\mathcal{B}$ can calculate the challenge ciphertext as follows.

1) For $i = m$, algorithm $\mathcal{B}$ outputs

$$\begin{aligned} C^* &= \hat{e}(g,g)^{\alpha\mu}, \quad D^* = g^{\mu}, \\ D_m^* &= w^{-\mu}(u^{W_{\beta,m}^*}h)^{z_m} = w^{-\mu}Z^y, \\ E_{m,1}^* &= {g_1}^{z_m - s_1} = g^{x_2 x_4}, \\ E_{m,2}^* &= {g_2}^{s_{m,1}} = g^{x_1 x_3}, \\ F_{m,1}^* &= Z^{d_3} \cdot {g_3}^{-s_{m,2}}, \quad F_{m,2}^* = {g_4}^{s_{m,2}}. \end{aligned}$$

2) For any $i \in [m-1]$, algorithm $\mathcal{B}$ outputs

$$D_i^* = w^{-\mu}(u^{W_{\beta,i}^*}h)^{z_i}, \quad E_{i,1}^* = {g_1}^{z_i - s_{i,1}},$$
$$E_{i,2}^* = {g_2}^{s_{i,1}}, \; F_{i,1}^* = {g_3}^{z_i - s_{i,2}}, \; F_{i,2}^* = {g_4}^{s_{i,2}}.$$

- Guess. Algorithm $\mathcal{A}$ output a guess β' for β.

On the one hand, if $Z = g^{x_3 + x_4}$, then algorithm $\mathcal{A}$'s view of this simulation is identical to the original game. On the other hand, if Z is randomly chosen from G, then algorithm $\mathcal{A}$'s advantage is nil. Therefore, if algorithm $\mathcal{A}$ can distinguish game Game$_m$ from game Game$_{m+1}$ with a non-negligible probability, algorithm $\mathcal{B}$ has a non-negligible probability in breaking the decisional linear assumption.

Lemma 3. Under the decisional linear assumption, the advantage for a polynomial time adversary that can distinguish between the games Game$_{m'+m+1}$ and Game$_{m'+m}$ for $m' \in [1, m]$ is negligible.

proof. This proof follows almost the same as that of Lemma 2, except that the simulation is done over the parameters g_3 and g_4 instead of g_1 and g_2.

This completes the proof of Theorem 1.

Hui Cui obtained her PhD degree in the School of Computing and Information Technology, University of Wollongong, Australia. Now she works in the Secure Mobile Centre under the School of Information Systems, Singapore Management University, Singapore.

Zhiguo Wan obtained his PhD degree at the School of Computing, National University of Singapore, Singapore. Now he works at the School of Computer Science and Technology, Shandong University, China.

Robert H. Deng obtained his PhD degree from the Illinois Institute of Technology, Chicago, United States. He is now a professor at the School of Information Systems, Singapore Management University, Singapore.

Guilin Wang obtained his PhD degree from the Institute of Software, Chinese Academy of Sciences, China. He is now working in the Shield Lab, Central Research Institute, Huawei International Pte Ltd, Singapore.

Yingjiu Li obtained his PhD degree from the George Mason University, Virginia, United States. He is now an associate professor of the School of Information Systems, Singapore Management University, Singapore.