Hindawi
Mobile Information Systems
Volume 2018, Article ID 3839254, 11 pages
https://doi.org/10.1155/2018/3839254

Research Article

Conjunctive and Disjunctive Keyword Search over Encrypted Mobile Cloud Data in Public Key System

Yu Zhang,[1] **Yin Li**,[1] **and Yifan Wang**[2]

[1]*School of Computer and Information Technology, Xinyang Normal University, Xinyang 464000, China*
[2]*Wayne State University, 42 W. Warren Ave., Detroit, MI 48202, USA*

Correspondence should be addressed to Yu Zhang; willow1223@126.com

Received 13 September 2017; Revised 1 March 2018; Accepted 20 March 2018; Published 2 May 2018

Academic Editor: Paolo Bellavista

The searchable encryption scheme can perform keywords search operation directly over encrypted data without decryption, which is crucial to cloud storage, and has attracted a lot of attention in these years. However, it is still an open problem to develop an efficient public key encryption scheme supporting conjunctive and a disjunctive keyword search simultaneously. To achieve this goal, we introduce a keyword conversion method that can transform the query and index keywords into a vector space model. Through applying a vector space model to a predicate encryption scheme supporting inner product, we propose a novel public key encryption scheme with conjunctive and disjunctive keyword search. The experiment result demonstrates that our scheme is more efficient in both time and space as well as more suitable for the mobile cloud compared with the state-of-art schemes.

1. Introduction

With the rapid development of the cloud computation accompanied by the boosting amount of data, more and more enterprises and individuals are willing to share their own data on the cloud platform. Because the data stored in the cloud may be sensitive, such as medical records, the popularity of the cloud storage inevitably brings its users security concern. Specifically, hacker attack and administrator theft can lead to data leakage. In order to protect the data privacy, encrypting data before outsourcing it on the cloud server is a common way. However, users still confront the problem of how to search the encrypted data stored on the cloud efficiently. A straightforward approach is to download all the encrypted data to the clients and then decrypt them all. After obtaining all the unencrypted data, users can search the document by using common information retrieval technical. Nevertheless, this strategy needs tremendous cost of transportation, storage, and computation, which brings a new issue: how to search encrypted data efficiently without decrypting it first.

Many searchable encryption (SE) schemes were proposed to realize keyword search over encrypted data with various search functions. There are two main categories of SE according to its applications: searchable public key encryption and searchable symmetric key encryption. Over the last few years, many searchable symmetric key encryption schemes have been proposed, which achieve complex search conditions such as Boolean keyword search, personal keyword search, and query result ranking [1–4]. However, the development of searchable public key encryption is relatively slow since it is difficult to support advanced search function without sacrificing security. Specifically, a scheme supporting Boolean keyword search in the public key setting is still in urgent demand. To meet this need, one should solve two subproblems: public key encryption with conjunctive keyword search (PECK) and public key encryption with disjunctive keyword search (PEDK). Park et al. presented the formal model and the security definition of PECK followed by two constructions [6]. Then Hwang and Lee [5] introduced a concept of multiuser PECK as well as a more efficient scheme with less search time. But all these schemes require keyword fields. To eliminate the keyword fields, Boneh and Waters [11] presented a hidden vector encryption (HVE) scheme supporting conjunctive search, comparison queries, and subset queries on encrypted data. Recently, Zhang and Zhang proposed a new PECK scheme with less storage space and query time [12].

However, the research process of PEDK is very slow. In order to support disjunction formulae, Katz et al. gave a predicate encryption supporting inner product (IPE) scheme [7]. This scheme enables more complex evaluations on disjunction, conjunction, and polynomial formulae. In an IPE scheme, the secret key corresponding to a predicate vector $\overrightarrow{v}$ can decrypt the ciphertext associated with an attribute vector $\overrightarrow{x}$ if and only if $\overrightarrow{x} \cdot \overrightarrow{v} = 0$. Unfortunately, the IPE scheme mentioned above was proved in a selective security model. To improve its security level, Lewko et al. presented a fully secure IPE scheme [23] by using dual system encryption introduced in [13].

Although we can create an PEDK scheme by making use of an IPE scheme and a trivial method presented in [6], the time complexities of the encryption and the test in this PEDK scheme are both $O(2^n)$, where n is the number of the keywords in a document. In addition, the storage cost of an index also encounters exponential increase. Therefore, this scheme is not practical when n is large. Besides, if a user wants to perform a conjunctive and a disjunctive keyword query simultaneously, a PEDK and a PECK scheme should be constructed and maintained together. To construct a more efficient PEDK scheme supporting conjunctive keyword search simultaneously, Zhang and Lu proposed an approach of converting an IPE scheme into a public key encryption with conjunctive and disjunctive keyword search (PECDK) scheme and gave a concrete scheme [15]. In their scheme, the size of each document's index, the size of a trapdoor, and the time cost of pairing operations in the test process are all $O(N)$. Since N is a large integer and seen as the number of keywords in a dictionary, there is still a great room to improve the efficiency of the previous PECDK scheme.

In this paper, we first propose a new method that can change an IPE scheme into a PECDK scheme and then give an instance. Our contributions are summarized as follows.

(1) We design a new approach, which converts an index keyword set and a query keyword set into an attribute matrix and a predicate vector, respectively. Technically, we first use the index keyword set to construct an equation of degree n with one unknown. Then, we apply coefficients and the roots of the equation to create a predicate vector and an attribute matrix, respectively.

(2) We propose a construction of PECDK based on the method mentioned in (1) and an efficient IPE scheme proposed in [23]. We also prove the security of our PECDK scheme according to the security definition introduced in [15]. The experiment shows that compared with the previous PECDK scheme, the time complexity of keyword search and that of index construction are both reduced from $O(N)$ to $O(n^2)$, where $n \ll \sqrt{N}$, so is the space cost of encrypted index. Since N is much larger than n, for example, n is less than 20 while N is large than 10000, we can argue that the proposal is suitable for the mobile setting.

1.1. Organization. The rest of this paper is organized as follows. Related work is discussed in Section 2, and Section 3 gives the model of PECDK together with its security model. Section 4 firstly introduces our transformation method, then proposes a concrete PECDK scheme based on our approach, and finally presents its security proof. We present the theoretical and experimental analysis in Section 5. Section 6 covers the conclusion.

2. Related Work

Searchable encryption schemes enable the clients to store the encrypted data to the cloud and execute keyword search over ciphertext domain. Thus, our solution belongs to this field. Due to different cryptography primitives, searchable encryption schemes can be classified into public key system and symmetric key system.

Song et al. first introduced the definition of searchable symmetric encryption and proposed a concrete scheme [1]. Then Goh defined the concept of conjunctive keyword search over encrypted data and presented an effective scheme by taking advantage of the Bloom filter [2]. In addition, he also gave a formal security definition of searchable symmetric encryption. According to this scheme, some improved schemes with less computation and communication cost were proposed in [3–5]. However, the search time cost in these schemes is linear with the number of the documents in a dataset since each document needs an encrypted keyword index. To reduce the time cost of search, some works utilized the tree structure, such as R-tree and kd-tree, to obtain a sublinear search efficiency [8, 9]. Considering the issues that returning all related documents will bring network traffic and the previous schemes fail to sort search results, rank search schemes realizing a quick search of top-k relevant documents were proposed in [16–18]. These works only supported single keyword search due to using order-preserving encryption (OPE) [14]. Recently, some schemes achieving multikeywords rank search were presented in [19, 20].

With slower development than searchable symmetric encryption, searchable public key encryption is also difficult to support complex query condition. Boneh et al. brought up the new concept of PEKS and provided several constructions [10] related to the Identity-Based Encryption (IBE) presented in [21]. Based on that, Abdalla et al. specified the computational and statistical consistency of PEKS and proposed a statistically consistent scheme [22]. However, their works only supported a single keyword search. Park et al. raised a concept of PECK in [6]. They defined the security model of this mechanism followed by two constructions. One needs more bilinear pairing operations, while the other needs more private keys. Then Hwang and Lee designed a more efficient scheme as well as introduced a multiuser PECK scheme that enables multiusers keyword search [5]. To avoid the usage of keyword field just like all the constructions mentioned above did, Boneh and Waters presented the hidden vector encryption (HVE), a public key system supporting conjunctive keyword search, comparison queries, and subset queries on encrypted data [11]. For achieving disjunctive keyword search without keyword field,

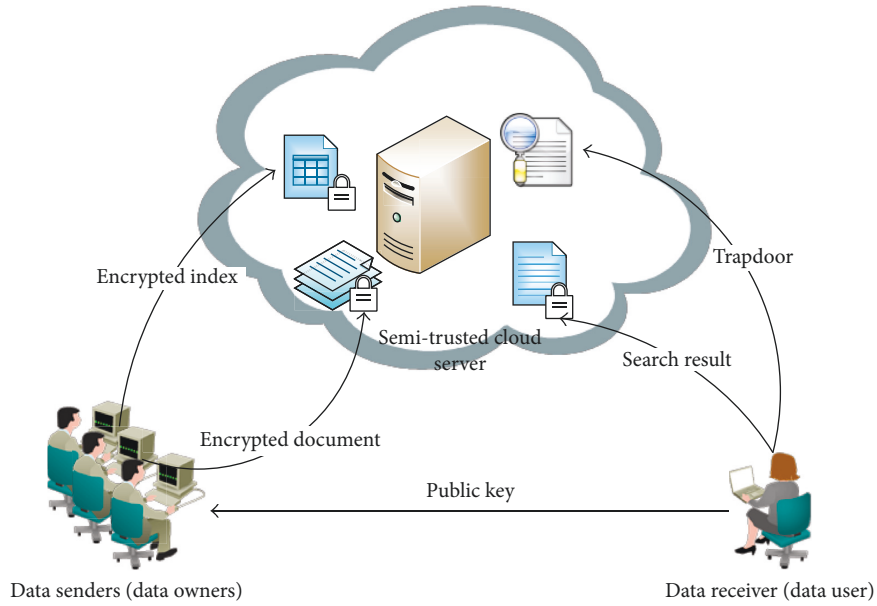

Figure 1: Architecture of the search over encrypted cloud data.

Katz et al. proposed an IPE scheme [7]. To improve the security level and the decryption efficiency, fully secure IPE schemes were proposed in [23, 24]. In addition, the previous schemes fail to support conjunctive keyword search and disjunctive keyword search at the same time. Addressing this issue, Zhang and Lu proposed a PECDK scheme [15].

Another class of searchable encryption is called range search over encrypted data. It can be used to test whether a multidimension point is inside in a hyperrectangle. Related works were presented in [25–27].

3. Preliminaries

3.1. The System Model. Consider a data storage service in cloud, where a data owner has a set of documents D to be outsourced to the cloud server in an encrypted form. To enable efficient query over encrypted documents, we consider the keyword-based index structure for storing the outsourced files. Specifically, the data owner builds an encrypted searchable index set C with each document's keywords, and then both the encrypted index set C and the encrypted document set Enc(D) are outsourced to the cloud server. For each query of an arbitrary keyword set Q, a data user computes a search token TK_Q of the query Q and sends it to the cloud server. Upon receiving TK_Q from the data user, the cloud server queries over the encrypted index set C and returns the candidate encrypted documents. Finally, the data user decrypts the candidate documents and verifies each document by checking the existence of the keyword.

The application scene of the searchable public key encryption involves three roles: data senders, a data receiver, and a cloud server, as illustrated in Figure 1. In this scene, the data receiver performs keyword search, generates a public key (pk) and a secret key (sk), and sends the pk to the public. Anyone who can access the pk is recognized as a data sender. Data senders send their encrypted documents with the related encrypted index to the cloud server. The cloud server stores the encrypted documents and the encrypted searchable indexes for data senders. In addition, we assume that the cloud server is "honest-but-curious" employed by many works on searchable encryption. When the data receiver processes keywords search, one generates a trapdoor of these keywords and sends it to the cloud server. Upon receiving trapdoor from the data receiver, the cloud server queries over each document's encrypted index and returns the candidate encrypted documents.

In this paper, we focus on the searchable public key encryption supporting conjunctive and disjunctive keyword search. Strictly speaking, we present a formal definition PECDK model derived from the model proposed in [15].

Definition 1. There are four polynomial time algorithms in the PECDK scheme: *KeyGen*, *IndexBuild*, *Trapdoor*, and *Test*:

(i) *KeyGen(γ)*: Choosing a security parameter γ, the algorithm outputs the parameter (pk, sk), where pk is the public key and sk is the secret key.

(ii) *IndexBuild(pk, W)*: The algorithm is performed by the sender to produce an encrypted index I_W by using a keyword set $W = \{w_1, w_2, \ldots, w_n\}$ and pk.

(iii) *Trapdoor(sk, Q, sym)*: The receiver runs this algorithm to produce a trapdoor. It takes sk and the keyword query Q, sym as input to generate a trapdoor $T_Q = \{t_Q, \mathrm{sym}\}$.

(iv) *Test(pk, T_Q, I_W)*: It takes the trapdoor $T_Q = \{t_Q, \mathrm{sym}\}$, the secure index I_W, and the public key pk as input. If sym is $\wedge$, it means that the trapdoor is for conjunctive keyword search. In this case, the output is 1 if $Q \subseteq W$ or 0 otherwise. If sym is $\vee$, disjunctive keyword search should be performed. In this case, if $Q \cap W \neq \phi$, it outputs 1 otherwise 0.

3.2. Security Model. Generally speaking, the security of a searchable encryption means that the cloud server can infer as little information as possible from the encrypted data and the search process without sacrificing the search ability. Before introducing the adaptive security definition of our scheme, we first define the privacy leakage, which is revealed to the cloud server inevitably.

3.2.1. Size Pattern. Since the encrypted documents and queries are submitted to the cloud server, the cloud server can obtain the basic size information of these encrypted data easily. This is called the leakage of size pattern.

3.2.2. Access Pattern. For each query, the cloud server can obtain the identifiers of data records that match this query. This is called the leakage of access pattern.

3.2.3. Search Pattern. Given a record set $D = \{D_1, D_2, \ldots, D_n\}$ and a query set $Q = \{Q_1, Q_2, \ldots, Q_t\}$, cloud server can create a $n \times t$ matrix, where the element in the ith row and jth column is 1, if the record D_i matches the query Q_j. The matrix can be seen as the leakage of search pattern.

Actually, Oblivious RAM can be utilized to preserve access and search pattern, but this technique is too inefficient to be used in the real applications. In this paper, we do not consider the problem of how to protect access pattern and search pattern in our scheme.

3.2.4. Query Privacy. The leakage of query privacy means that keywords in the encrypted query will be revealed to the cloud server. It commonly exists in the public key setting since anyone can construct an encrypted index for arbitrary keywords. Because of belonging to public key encryption category, our scheme fails to protect query privacy.

As previous works, we denote the information leakage including size pattern, access pattern, search pattern, and query privacy as leakage function $L(D, I, Q)$, where D is a record set, I is a secure index, and Q is a query set.

3.2.5. Adaptive Security. With the leakage function mentioned above, we introduce an adaptive security definition of the PECDK scheme related to the one proposed in [15] as follows.

Definition 2. A PECDK scheme is adaptively index-hiding against chosen plaintext attacks if for all probabilistic polynomial time adversaries $\mathbb{A}$, the advantage of $\mathbb{A}$ in the following game is negligible.

(1) Setup: The challenger $\mathbb{C}$ runs $KeyGen(\gamma)$ algorithm to generate the public key pk and the secret key sk. Then $\mathbb{C}$ gives pk to the attacker $\mathbb{A}$.

(2) Phase 1: The attacker $\mathbb{A}$ can adaptively ask the challenger $\mathbb{C}$ for the trapdoor T_Q for any query $\{Q, \text{sym}\}$ of his choice.

(3) Challenge: $\mathbb{A}$ selects two keyword sets W^0 and W^1 and sends them to $\mathbb{C}$. Suppose that $\{Q_1, \text{sym}_1\}$, $\{Q_2, \text{sym}_2\}, \ldots, \{Q_t, \text{sym}_t\}$ are the keyword queries in Phase 1, the only restriction is that, for each $i \in [1, t]$, there is $W^0 \cap Q_i = \varnothing$ and $W^1 \cap Q_i = \varnothing$. Then, $\mathbb{C}$ picks a random bit $\beta \in \{0, 1\}$, produces $I_\beta = \text{IndexBuild}(\text{pk}, W^\beta)$, and sends $\{I_\beta, W^0, W^1\}$ to $\mathbb{A}$.

(4) Phase 2: $\mathbb{A}$ can continue to ask for trapdoors for any query of his choice, and these trapdoors subject to the same restriction as before.

(5) Response: $\mathbb{A}$ outputs $\beta' \in \{0, 1\}$ and wins the game if $\beta' = \beta$.

We define $\mathbb{A}$'s advantage in the above game as

$$\text{Adv}^{\mathbb{A}}_{\text{Game}} = \left| \Pr[\beta' = \beta] - \frac{1}{2} \right|. \tag{1}$$

Generally speaking, as long as the information leakage of W^0 and that of W^1 are the same under the leakage function $L(D, I, Q)$, we should insure that the encrypted form of W^0 and that of W^1 are computationally indistinguishable to the adversary.

4. Proposed PECDK Scheme

Based on the system and security model descried in the previous section, in this section, we present the method that converting index and query keyword sets into a vector space model. This model can be applied to an IPE scheme easily.

4.1. Conversion Method. We suppose that any keyword w can be expressed as $\{0, 1\}^*$ and define a function H_1: $\{0, 1\}^* \to Z_p^*$. Since p is a large prime and is larger than the number of the all words, H_1 can be collision-resistance. This means that, if $i \neq j$, then $H_1(w_i) \neq H_1(w_j)$, where w_i and w_j are two distinct keywords.

We first construct an equation of degree n with one unknown by using the index and query keyword sets. After that, we use the roots and coefficients of the equation to create a query vector and an index matrix. Let $W = \{w_1, w_2, \ldots, w_n\}$ and $Q = \{q_1, q_2, \ldots, q_m\}$ are two keyword sets, where $m < n$. The approach is described as follows:

(1) For the keyword set $Q = \{q_1, q_2, \ldots, q_m\}$, the following function can be constructed:

$$\begin{aligned} f(x) &= (x - H_1(q_1))(x - H_1(q_2)) \ldots (x - H_1(q_m)) \\ &= a_m x^m + a_{m-1} x^{m-1} + \ldots + a_0 x^0. \end{aligned} \tag{2}$$

According to the coefficient of the $f(x)$, a vector $\overrightarrow{a} = \{a_0, a_1, \ldots, a_m\}$ can be obtained.

(2) For the keyword set $W = \{w_1, w_2, \ldots, w_n\}$, the following function can be constructed:

$$\begin{aligned} g(x) &= (x - H_1(w_1))(x - H_1(w_2)) \ldots (x - H_1(w_n)) \\ &= b_n x^n + b_{n-1} x^{n-1} + \ldots + b_0 x^0. \end{aligned} \tag{3}$$

It is not difficult to find that the roots of the equation $g(x) = 0$ are $H_1(w_1), H_1(w_2), \ldots, H_1(w_n)$. According to the above roots, a matrix can be constructed:

$$M = \begin{pmatrix} M_{11} & M_{12} & \ldots & M_{1n} \\ M_{21} & M_{22} & \ldots & M_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ M_{n+1,1} & M_{n+1,2} & \ldots & M_{n+1,n} \end{pmatrix} = \begin{pmatrix} H_1(w_1)^0 & H_1(w_2)^0 & \ldots & H_1(w_n)^0 \\ H_1(w_1)^1 & H_1(w_2)^1 & \ldots & H_1(w_n)^1 \\ \vdots & \vdots & \ddots & \vdots \\ H_1(w_1)^n & H_1(w_2)^n & \ldots & H_1(w_n)^n \end{pmatrix}. \quad (4)$$

(3) Note that if there is a keyword $q_i \in Q$ such that $q_i \in W$, according to (2–4), it is not difficult to verify that $f(w_i) = \overrightarrow{a} \cdot \overrightarrow{M_i} = 0$ where $\overrightarrow{M_i} = \{M_{1i}, M_{2i}, \ldots, M_{m+1,i}\}$

As a result, if we can make sure there is $\overrightarrow{a} \cdot \overrightarrow{M_i} = 0$ for each i or some i, a conjunctive or disjunctive query can be obtained when applying the method above. Based on this, a concrete PECDK scheme will be proposed in next section.

4.2. Construction Details. According to the definition of IPE [13], suppose that $\text{Setup}_{\text{IPE}}$, $\text{Enc}_{\text{IPE}}(\text{pk}_{\text{IPE}}, M, \overrightarrow{x})$, $\text{KeyGen}_{\text{IPE}}(\text{pk}_{\text{IPE}}, \text{msk}_{\text{IPE}}, \overrightarrow{v})$, and $\text{Dec}_{\text{IPE}}(\text{pk}_{\text{IPE}}, c, \text{sk}_{\overrightarrow{v}})$ be the four algorithms in the IPE scheme, where pk_{IPE} and msk_{IPE} are the public key and the master secret key generated by using $\text{Setup}_{\text{IPE}}$, $\overrightarrow{x}$ is the attribute vector, $\overrightarrow{v}$ is the predicate vector, c is the ciphertext generated by using Enc_{IPE}, and $\text{sk}_{\overrightarrow{v}}$ is the secret key generated by using $\text{KeyGen}_{\text{IPE}}$. Our PECDK scheme works as follows:

(i) *KeyGen*: By using the $\text{Setup}_{\text{IPE}}$ algorithm, pk_{IPE} and msk_{IPE} can be obtained. The algorithm sets $\text{pk} = \text{pk}_{\text{IPE}}$ and $\text{sk} = \text{msk}_{\text{IPE}}$ and outputs pk and sk.

(ii) *IndexBuild*: Given a keyword set $W = \{w_1, w_2, \ldots, w_n\}$, the algorithm generates a matrix $M = \{\overrightarrow{M_1}, \overrightarrow{M_2}, \ldots, \overrightarrow{M_n}\}$ by using (4). Then it generates $I_W = \{c_1, c_2, \ldots, c_n\}$ in which $c_i = \text{Enc}_{\text{IPE}}(\text{pk}, 1, \overrightarrow{M_i})$.

(iii) *Trapdoor*: Given a keyword query $\{Q, \text{sym}\}$, the algorithm creates $\overrightarrow{a}$ by using (3). Then it generates t_Q by making use of $\text{KeyGen}_{\text{IPE}}(\text{sk}, \overrightarrow{a})$. Then it outputs a trapdoor $T_Q = \{t_Q, \text{sym}\}$.

(iv) *Test*: Given a T_Q, a I_W, and the pk, there are two situations.

(1) If the symbol in T_Q is $\vee$, the algorithm works as follows.

(a) Choosing a counter i and setting $i = 1$.

(b) If $i > n$, then go to step (c), otherwise the algorithm computes $R = \text{Dec}(\text{pk}, I_W, T_Q)$. If $R = 1$, then the algorithm outputs 1 and ends. Otherwise, it sets $i = i + 1$ and goes to the step (b).

(c) The algorithm outputs 0 and ends.

(2) If the symbol in T_Q is $\wedge$, the algorithm works as follows.

(a) Choosing two counters i and j and setting $i = 1$ and $j = 1$.

(b) If $i > n$, then go to step (c), otherwise the algorithm computes $R = \text{Dec}(\text{pk}, I_W, T_Q)$. If $R = 1$, then the algorithm sets $j = j + 1$. Otherwise, it does nothing. After that, it sets $i = i + 1$ and goes to the step (b).

(c) If $j = m$, the algorithm outputs 1 and ends. Otherwise, it outputs 0 and ends.

4.3. Security Proof. The proposed PECDK scheme can be constructed by making use of the fully secure IPE scheme. Therefore, we have the following proposition.

Proposition 1. *If the IPE scheme is secure, then our PECDK scheme is secure.*

Proof Sketch: If there is a PPT algorithm $\mathbb{A}$ which can break the PECDK scheme, we can say that $\mathbb{A}$ can break the IPE scheme. To create pk and sk in the PECDK scheme, the challenger $\mathbb{C}$ uses the $\text{Setup}_{\text{IPE}}$ algorithm to generate pk_{IPE}, sk_{IPE}, and sets $\text{pk} = \text{pk}_{\text{IPE}}$ and $\text{sk} = \text{sk}_{\text{IPE}}$. $\mathbb{A}$ can adaptively query trapdoors of keyword set $\{Q_1, Q_2, \ldots, Q_t\}$. It is not difficult to find that these trapdoors can be seen as a group of decryption keys for the IPE scheme. Then $\mathbb{A}$ outputs two challenge keyword sets W^0 and W^1, under a constraint that $Q_i \cap W^0 = \varnothing$ and $Q_i \cap W^1 = \varnothing$, where $i \in [1, t]$. $\mathbb{C}$ flips a coin $\beta \in \{0, 1\}$ and gives an index I_{W^β} to $\mathbb{A}$. This index can be seen as a set of challenge ciphertexts of IPE. After this, $\mathbb{A}$ can continue querying trapdoors which subject to the restriction described above. These trapdoors can still be seen as a group of decryption keys for IPE. Finally, $\mathbb{A}$ gives a guess β'. Note that if $\mathbb{A}$ can break the PECDK scheme, the value of $|\Pr[\beta' = \beta] - (1/2)|$ is not negligible. It means that the two challenge indices can be distinguished. Because the challenge indices in the PECDK scheme is equal to the challenge ciphertext in the IPE scheme, according to the security definition for IPE, it means that $\mathbb{A}$ can break the IPE scheme.

5. Performance Evaluation

5.1. Cost Analysis. We denote the previous PECDK scheme [15] as PECDK-2 and the proposal as PECDK-1. Let $F = \{f_1, f_2, \ldots, f_D\}$ be the file set, and W_i be the keyword set for the corresponding file f_i, where $i \in [1, D]$. For each keyword set W_i, we build an index of file f_i. We assume that n is the number of keywords in W_i, m is the number of keywords in a query Q, and N is a large number such as the number of keywords in a dictionary, where $N \gg n$. The index and query structures for PECDK-1 and PECDK-2 are illustrated in Figures 2(a) and 2(b), respectively. Note that the index

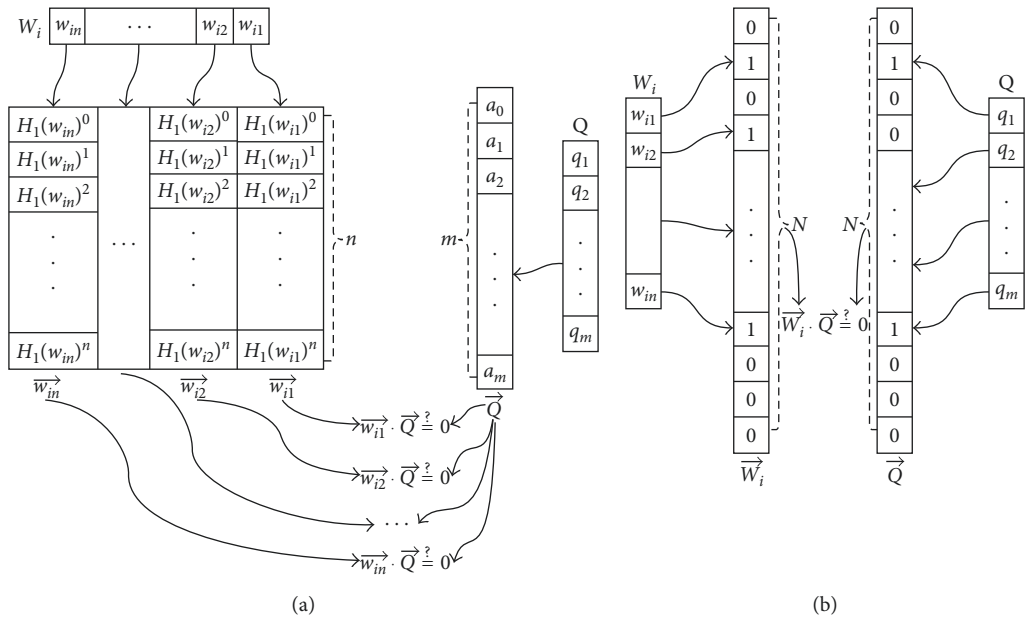

Figure 2: Index structure for a single file f_i's keyword set W_i, query structure for a query keyword set Q, and testing process for W_i and Q in PECDK-1 (a) and PECDK-2 (b), where $i \in [1, D]$.

structure in Figure 2 is only for one file f_i, and the index for whole files is a simple combination for each file's index.

From Figure 2(a), we can find that W_i is converted into a matrix in which each column is associated with a keyword w_{ij} in W_i and represented by a vector $\overrightarrow{w_{ij}}$, where $j \in [1, n]$. Correspondingly, the query Q is changed to a vector $\overrightarrow{Q}$ based on the approach introduced in Section 4. The test algorithm is to verify whether there exists at least one j such that $\overrightarrow{w_{ij}} \cdot \overrightarrow{Q}$ equals 0. Therefore, PECDK-1 is independent with the vocabulary length N. According to the above analysis, we know that the time complexity for index building, trapdoor generation, and testing is $O(n^2)$, $O(m)$, and $O(m \cdot n)$, respectively.

From Figure 2(b), we suppose that the vocabulary is defined as $w_1, w_2, \ldots, w_N$. For the keyword set W_i, the index of W_i is initialed by a zero vector with length N which is denoted as $\overrightarrow{W_i}$. After initialization, if $w_{ij} = w_k$, the k-th position of $\overrightarrow{W_i}$ is set to be 1, where $j \in [1, n]$ and $k \in [1, N]$. Moreover, the vector building process for query Q is the same as that for W_i. The test process in PECDK-2 is to check whether $\overrightarrow{W_i} \cdot \overrightarrow{Q}$ equals 0. According to the above description, it is clear that the time complexities of index building, trapdoor generation and testing are all $O(N)$.

Let $|G|$ represents the size of an element in G, $|T_e|$ be the time cost for a pairing operation [28], and $|T_G|$ be the time cost for the power operation on G. The results of the comparison with the PECDK-2 scheme are shown in Table 1.

Since, in the encryption and test phrase, the time cost of the pairing and the power operation are much more than other operations, we do not take account of other operations. According to Table 1, we can argue that compared with the previous PECDK (PECDK-2) scheme, the proposed one (PECDK-1) has better performance on time and space complexity when $N > n^2$.

In the following, we will argue that our statement where $N \gg n$ is true in real world. We investigate data collections used in recent works [31, 32] for information retrieval. The statistical information for these data sets are shown in Table 2. From this table, we found that the vocabulary size (N) is commonly linear

Table 1: Comparison with the previous PECDK scheme.

	PECDK-2	PECDK-1
pk size	$O(N^2)\|G\|$	$O(n^2)\|G\|$
sk size	$O(N^2)\|G\|$	$O(n^2)\|G\|$
Trapdoor size	$O(N)\|G\|$	$O(m)\|G\|$
Index size	$O(N)\|G\|$	$O(n^2)\|G\|$
Encryption time	$O(N)\|T_G\|$	$O(n^2)\|T_G\|$
Test time	$O(N)\|T_e\|$	$O(m \cdot n)\|T_e\|$

Table 2: Statistics for data collections used in the domain of information retrieval.

Dataset name	Documents	Vocabulary size
AP88-89	164,597	247,350
WSJ87-92	173,252	216,539
DOTGOV	1,247,442	3,051,601
MedTrack	100,866	55,065
Yelp2013	335,018	211,245
Yelp2014	1,125,457	476,191
Yelp2015	1,569,264	612,636
IMDB review	348,415	115,831
Yahoo answer	1,450,000	1,554,607
Amazon review	3,650,000	1,919,336

Table 3: Statistics for documents' information in the OHSUMED collection (#w denotes the number of words per document).

Field name	#w	Vocabulary size
Title	5~20	33059
Abstract	50~200	83496

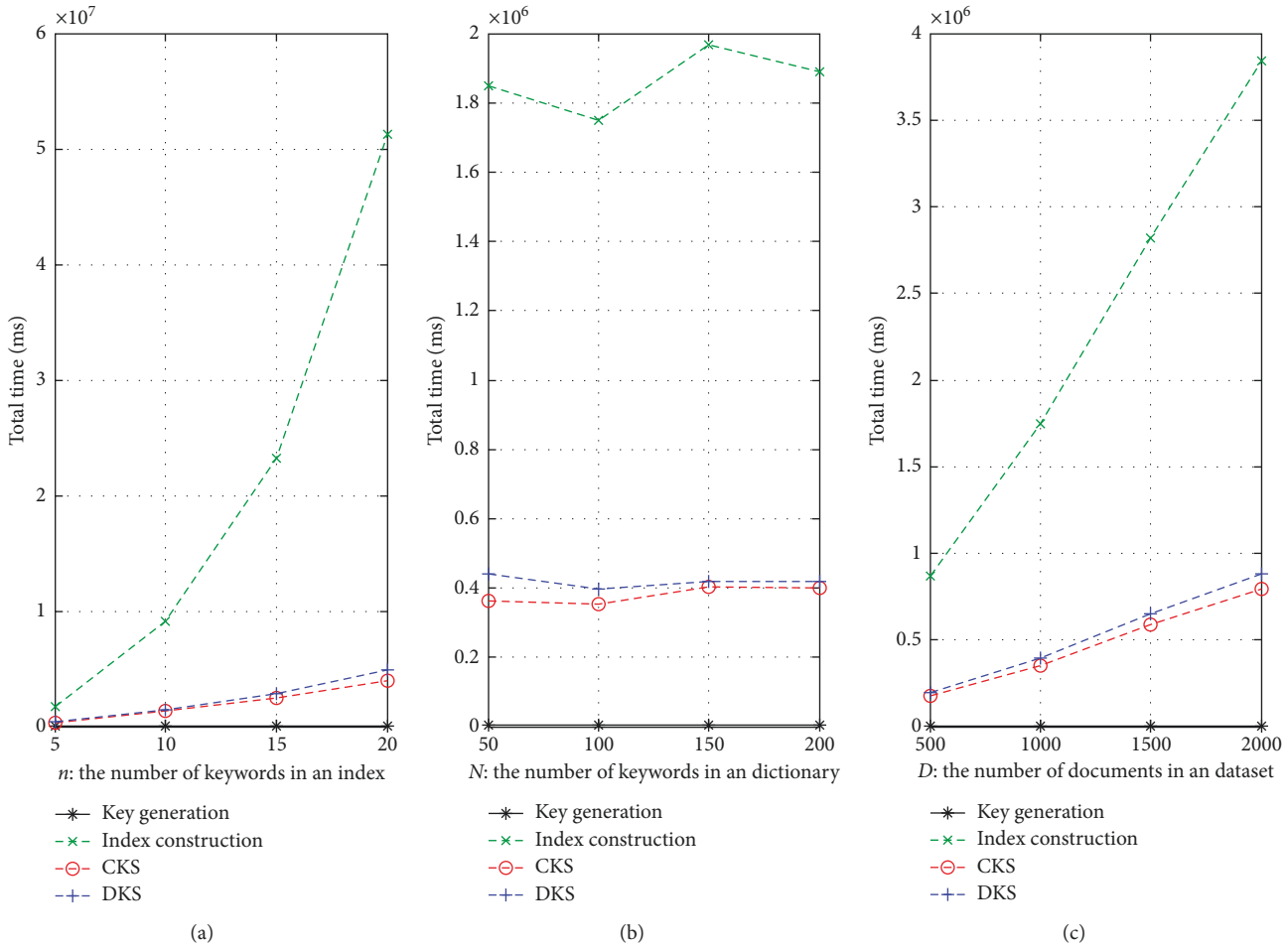

Figure 3: Impact of n, N, and D on the time cost of key generation, index construction, conjunctive keyword search (CKS), and disjunctive keyword search (DKS) in PECDK-1 (a) for different sizes of n with $D = 1000$ and $N = 100$, (b) for different numbers of N with $D = 1000$ and $n = 5$, and (c) for different numbers of D with $N = 100$ and $n = 5$.

with $O(10^6)$. However, the number of keywords in a document (n) is usually only 3~5 (e.g., the scientific paper). Even if the abstract field is used as keywords, the number of keywords in a single document is approximately 200. So we can empirically think that N is much bigger than n.

Moreover, we have investigated the OHSUMED collection [30] that was created for information retrieval research. In this collection, each document contains several fields, such as title, abstract, sequential identifier, and so on. In our experiment, we use the field of "title" and "abstract," as well as a file (file name: ohsumed.87) containing more than 54000 documents for statistics. The results of this experiment are described in Table 3. Whether we use the "title" or "abstract" field as keywords, the keywords size n is far less than the vocabulary size N. In conclusion, we think that the statement of $N \gg n$ is true.

In addition, since PECDK-1 needs to test each keyword w_{ij} in the W_i against the query Q one by one, it will leak the information of the keyword set $Q \cap W_i$. Therefore, the proposed scheme is proven to be secure under a weaker security definition (Definition 2) than the previous one [15].

5.2. Experiment Results. For our experiments, we build artificial plaintext index with different number of keywords in a dictionary (i.e., $N = 50, 100, 150, 200$), different number of documents (i.e., $D = 500, 1000, 1500, 2000$), and different number of keywords in a document (i.e., $n = 5, 10, 15, 20$). In this index, we denote each keyword as a unique integer in a range $[0, N]$. We encrypted the indices with PECDK-1 and PECDK-2, respectively, and the encrypted indices were stored on our machine. We then executed random queries over these encrypted indices. We implemented our constructions in JAVA with Java Pairing Based Cryptography library (JPBC) [29]. Our experiments were run on Intel(R) Core(TM) i7-3520M CPU at 2.90 GHz processor and 3537 MB memory size. In our implementation, the bilinear map is instantiated as Type A pairing (base field size is 512-bit), which offers a level of security equivalent to 1024-bit DLOG [29].

5.2.1. Performance Comparison Between PECDK-1 and PECDK-2. For a query with five keywords, Figures 3 and 4 show the following:

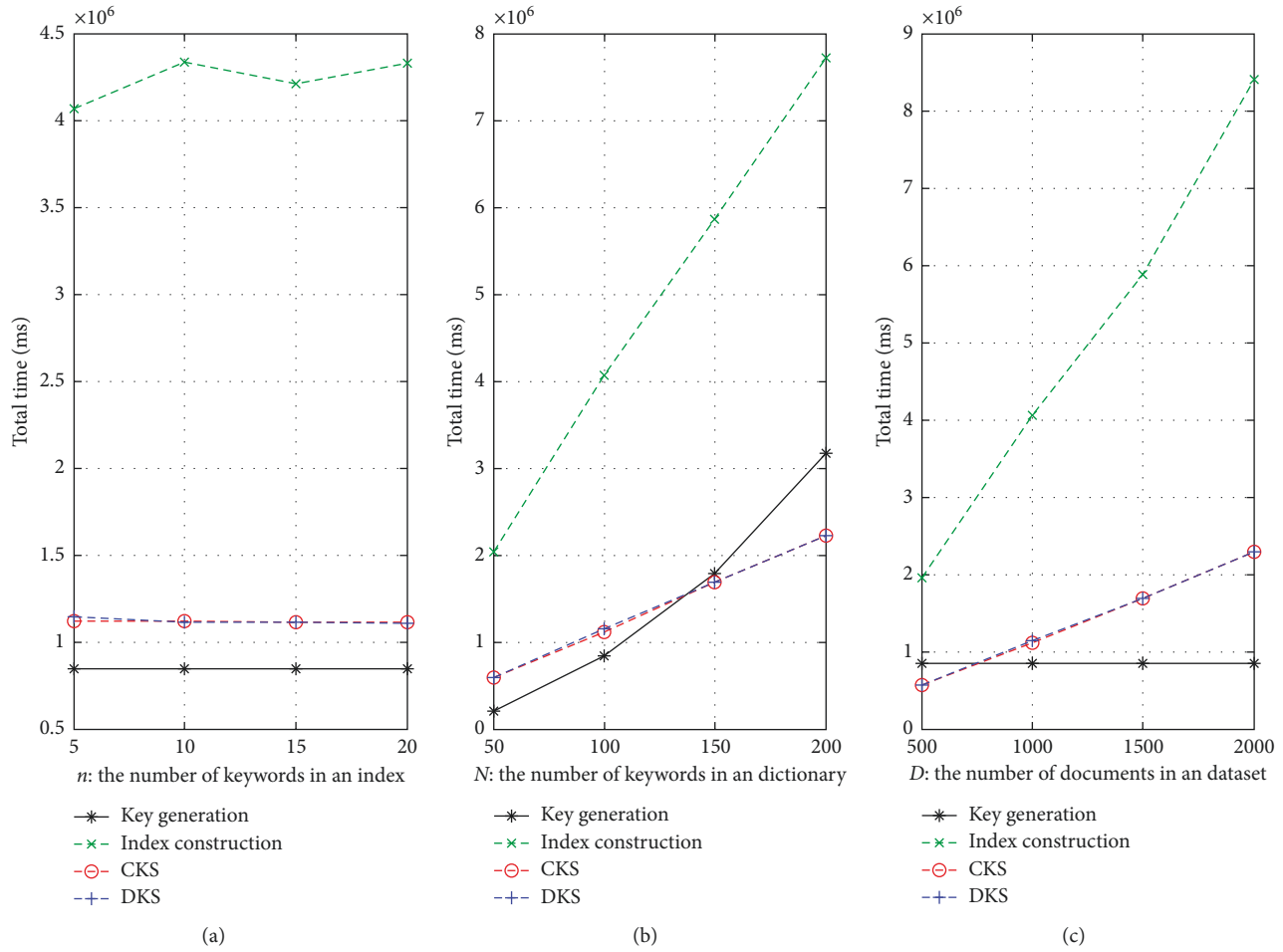

Figure 4: Impact of n, N, and D on the time cost of key generation, index construction, conjunctive keyword search (CKS), and disjunctive keyword search (DKS) in PECDK-2 (a) for different sizes of n with $D = 1000$ and $N = 100$, (b) for different numbers of N with $D = 1000$ and $n = 5$, and (c) for different numbers of D with $N = 100$ and $n = 5$.

(1) Figures 3(b) and 3(c) and Figures 4(a) and 4(c) denote that the execution time of key generation in PECDK-1 is irrespective of N and D, while that in PECDK-2 is independent with n and D.

(2) Figures 3(b) and 4(a) indicate that the running time of index construction and keywords search in PECDK-1 is independent with N, and that in PECDK-2 is not connected with n.

(3) Figures 3(c) and 4(c) illustrate that both the search time and the indexing time in PECDK-1 and PECDK-2 are nearly linear with D.

(4) Figures 3(a) and 4(b) shows that PECDK-1 is influenced mainly by n, while PECDK-2 is affected primarily by N.

According to the item (4), since n and N are two different parameters, we need to investigate the influence extent of the time and space complexity from these two parameters.

5.2.2. Key Generation. The running time of key generation in PECDK-1 is less than $O(n^2)$ while in PECDK-2 is less than $O(N^2)$. Since n is much less than N, PECDK-1 is more efficient than PECDK-2 in key generation phase.

Table 4: Impact of n on time consumption of trapdoor generation ($m = 5$, $N = 100$).

Size of n	5	10	15	20
PECDK-1 (ms)	576	963	1338	1805
PECDK-2 (ms)	5564	5562	5565	5603

Table 5: Impact of N time consumption of trapdoor generation ($m = 5$, $n = 5$).

Size of N	50	100	150	200
PECDK-1 (ms)	592	576	589	606
PECDK-2 (ms)	2862	5564	8224	10860

5.2.3. Index Construction. Figure 3(b) shows that the time cost of index construction in PECDK-2 is linear with N. Specifically, when $n = 5$, $N = 200$ and $D = 1000$, it takes around 7720.292 seconds to build the index. Therefore, we

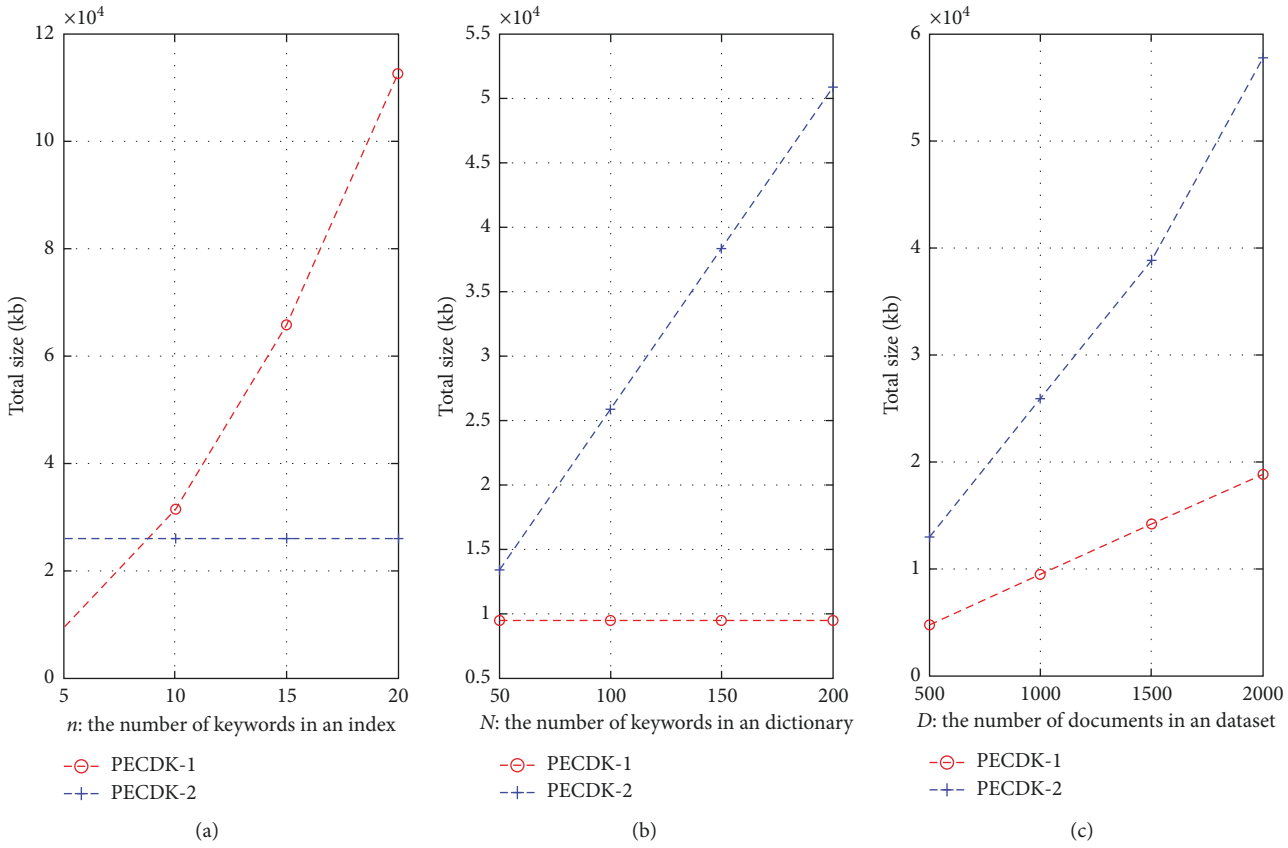

FIGURE 5: Impact of n, N, and D on the storage cost of the index in PECDK-1 and PECDK-2 (a) for different sizes of n with $D = 1000$ and $N = 100$, (b) for different numbers of N with $D = 1000$ and $n = 5$, and (c) for different numbers of D with $N = 100$ and $n = 5$.

can argue that under the condition that $N = 10000$ and $D = 1000$, the time of building index requires 7720.292*500 seconds, which is significantly longer than the time of index construction in PECDK-1 with $n = 20$ and $D = 1000$. According to this, we can conclude that the PECDK-1 scheme is more suitable and practical since n is always less than 20 and N is always larger than 10000.

5.2.4. Trapdoor Generation. Tables 4 and 5 show that the time of trapdoor generation in PECDK-1 is linear with n, while that in PECDK-2 is linear with N. Since n is much less than N, the time cost of building a trapdoor in PECDK-1 is less than that in PECDK-2. The trapdoor generation, unlike the key generation, is not a one-time cost and is performed by the data receiver. Therefore, the PECDK-1 scheme is more suitable on mobile cloud where data users possess little computation capacity.

5.2.5. Search Efficiency. Generally speaking, Figures 3(c) and 4(c) indicate that the time cost of conjunctive keyword search and that of disjunctive keyword search in PECDK-1 are both increasing with n^2, while those in PECDK-2 are both linearly increasing with N. Specifically, when $n = 5$, $N = 200$ and $D = 1000$, it takes around 2220 seconds to make a conjunctive or disjunctive keyword search in PECDK-2. According to the property that search time is linear with N, we can also conclude that the time cost of keyword search needs 2220∗500 seconds when $N = 10000$ and $D = 1000$. But the PECDK-1 scheme only costs approximately 5000 seconds to realize the conjunctive or disjunctive keyword search when $n = 20$ and $D = 1000$. As the same reason mentioned above, the PECDK-1 scheme is more practical than the PECDK-2 one.

5.2.6. Storage Overhead. As shown in Figures 5(a) and 5(b), we put forward the following arguments:

(1) The parameters N and n are independent with the storage cost of the index in PECDK-1 and that in PECDK-2, respectively.

(2) The storage of the index in PECDK-1rises with the square of n while that in PECDK-2 is linearly related to N. Due to the reason that n is less than 20 and N is larger than 10000 in common, we deem that the proposal is more useful in practice. As illustrated in Figure 5(c), we can find that both storage overhead in PECDK-1 and that in PECDK-2 are linear with D. Because n is significantly less than N and the increase rate in PECDK-1 is larger than that in PECDK-2, the storage cost in the proposed scheme is less than the previous one when D is large.

5.3. More Comments. Although the index structure, such as inverted document and R-tree, can raise the search efficiency, it fails to support dynamic operations in the public key system. Because anyone who can access the pk can construct an index in this system as well, it is difficult to combine indices obtained from data senders into a structured index. Our proposal can dynamically support document update in nature since each document is associated with an encrypted index.

In addition, because of the simple index structure, we can easily accelerate the search process by utilizing the technique of parallel computation. Thus, we argue that our scheme is practical in the cloud platform.

6. Conclusion

In this paper, we proposed a new approach to construct an efficient PECDK scheme with better performance in time and space complexity under an adaptive security model. To reveal the efficiency of the proposed scheme, we compared it with the existing PECDK scheme presented in [15] through theoretical analysis and experimental results.

Since n and m are much smaller than N, we think that the proposed scheme is beneficial for mobile applications with computation and memory limitations. In the future, we plan to create a new index structure to reduce the time cost of search and index construction.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors gratefully acknowledge the National Natural Science Foundation of China under Grant nos. 61402393 and 61601396 and Shanghai Key Laboratory of Integrated Administration Technologies for Information Security (no. AGK201607).

References

[1] D. Song, D. Wagner, and A. Perrig, "Practical techniques for searching on encrypted data," in *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pp. 44–55, Berkeley, CA, USA, 2000.

[2] E. J. Goh, "Secure indexes," *IACR Cryptology ePrint Archive*, vol. 2003, p. 216, 2003.

[3] Z. Fu, X. Wu, C. Guan et al., "Toward efficient multi-keyword fuzzy search over encrypted outsourced data with accuracy improvement," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 12, pp. 2706–2716, 2017.

[4] Z. Fu, K. Ren, J. Shu et al., "Enabling personalized search over encrypted outsourced data with efficiency improvement," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 9, pp. 2546–2559, 2016.

[5] Y. H. Hwang and P. J. Lee, "Public key encryption with conjunctive keyword search and its extension to a multi-user system," in *Proceedings of the Pairing*, vol. 4575 of Lecture Notes in Computer Science, pp. 2–22, Springer, Tokyo, Japan, July 2007.

[6] D. J. Park, K. Kim, and P. J. Lee, "Public key encryption with conjunctive field keyword search," in *Proceedings of the WISA*, vol. 3325 of Lecture Notes in Computer Science, pp. 73–86, Springer, Wuhan, China, 2004.

[7] J. Katz, A. Sahai, and B. Waters, "Predicate encryption supporting disjunctions, polynomial equations, and inner products," in *Proceedings of the Advances in Cryptology, EUROCRYPT*, vol. 4965 of Lecture Notes in Computer Science, pp. 146–162, Springer, Istanbul, Turkey, April 2008.

[8] D. Cash, S. Jarecki, C. Jutla, H. Krawczyk, M.-C. Roşu, and M. Steiner, "Highly-scalable searchable symmetric encryption with support for Boolean queries," in *Proceedings of the Advances in Cryptology, CRYPTO*, pp. 353–373, Santa Barbara, CA, USA, August 2013.

[9] D. Cash, J. Jaeger, S. Jarecki et al., "Dynamic searchable encryption in very-large databases: data structures and implementation," in *Proceedings of the Network and Distributed System Security Symposium*, San Diego, CA, USA, February 2014.

[10] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Proceedings of the EUROCRYPT*, vol. 3027 of Lecture Notes in Computer Science, pp. 506–522, Springer, Interlaken, Switzerland, May 2004.

[11] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Proceedings of the TCC*, vol. 4392 of Lecture Notes in Computer Science, pp. 535–554, Springer, Amsterdam, Netherlands, February 2007.

[12] B. Zhang and F. Zhang, "An efficient public key encryption with conjunctive-subset keywords search," *Journal of Network and Computer Applications*, vol. 34, no. 1, pp. 262–267, 2011.

[13] B. Waters, "Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions," in *Proceedings of the Advances in Cryptology, CRYPTO*, vol. 5677 of Lecture Notes in Computer Science, pp. 619–636, Springer, Santa Barbara, CA, USA, August 2009.

[14] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill, "Order-preserving symmetric encryption," in *Proceedings of the Advances in Cryptology, EUROCRYPT*, pp. 224–241, Cologne, Germany, April 2009.

[15] Y. Zhang and S. Lu, "POSTER: efficient method for disjunctive and conjunctive keyword search over encrypted data," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp. 1535–1537, Hong Kong, China, December 2014.

[16] A. Swaminathan, Y. Mao, G. M. Su et al., "Confidentiality-preserving rank-ordered search," in *Proceedings of the ACM Workshop on Storage Security and Survivability, Storages*, pp. 7–12, Alexandria, VA, USA, October 2007.

[17] S. Zerr, D. Olmedilla, W. Nejdl et al., "Zerber+R:top-k retrieval from a confidential index," in *Proceedings of the International Conference on Extending Database Technology: Advances in Database Technology*, pp. 439–449, Saint Petersburg, Russia, March 2009.

[18] C. Wang, N. Cao, K. Ren, and W. Lou, "Enabling secure and efficient ranked keyword search over outsourced cloud data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 8, pp. 1467–1479, 2012.

[19] Z. Fu, X. Sun, Q. Liu et al., "Achieving efficient cloud search services: multi-keyword ranked search over encrypted cloud data supporting parallel computing," *IEICE Transactions on Communications*, vol. 98, no. 1, pp. 190–200, 2015.

[20] Z. Xia, X. Wang, X. Sun et al., "A Secure and dynamic multi-keyword ranked search scheme over encrypted cloud data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 2, pp. 340–352, 2016.

[21] B. Dan and M. Franklin, *Identity-Based Encryption from the Weil Pairing*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2001.

[22] M. Abdalla, M. Bellare, D. Catalano et al., "Searchable encryption revisited: consistency properties, relation to anonymous IBE, and extensions," in *Proceedings of the CRYPTO*, vol. 3621 of Lecture Notes in Computer Science, pp. 205–222, Springer, Barbara, CA, USA, August 2005.

[23] A. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, "Fully secure functional encryption: attribute-based encryption and (hierarchical) inner product encryption," in *Proceedings of the Advances in Cryptology, EUROCRYPT*, vol. 6110 of Lecture Notes in Computer Science, pp. 62–91, Springer, French Riviera, France, May-June 2010.

[24] T. Okamoto and K. Takashima, "Achieving short ciphertexts or short secret-keys for adaptively secure general inner-product encryption," *Designs Codes and Cryptography*, vol. 77, pp. 138–159, 2015.

[25] B. Wang, Y. Hou, M. Li, H. Wang, and H. Li, "Maple: scalable multi-dimensional range search over encrypted cloud data with tree-based index," in *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security*, pp. 111–122, Kyoto, Japan, June 2014.

[26] B. Wang, M. Li, and H. Wang, "Geometric range search on encrypted spatial data," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 4, pp. 704–719, 2016.

[27] H. Ren, H. Li, H. Chen et al., "Efficient privacy-preserving circular range search on outsourced spatial data," in *Proceedings of the 2016 IEEE International Conference on Communications (ICC)*, pp. 1–7, Kuala Lumpur, Malaysia, May 2016.

[28] A. Joux, "The Weil and Tate pairings as building blocks for public key cryptosystems," in *Proceedings of the International Algorithmic Number Theory Symposium (ANTS)*, vol. 2369, pp. 20–32, Sydney, NSW, Australia, July 2002.

[29] A. D. Caro, "The java pairing based cryptography library (JPBC)," pp. 2–24, 2013, http://gas.dia.unisa.it/projects/jpbc/laatstnagekeken op.

[30] W. Hersh, C. Buckley, T. J. Leone, and D. Hickam, "OHSUMED: an interactive retrieval evaluation and new large test collection for research," in *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 192–201, Dublin, Ireland, July 1994.

[31] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the Conference on North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1480–1489, San Diego, CA, USA, June 2017.

[32] G. Zuccon, B. Koopman, P. Bruza, and L. Azzopardi, "Integrating and evaluating neural word embeddings in information retrieval," in *Proceedings of the Australasian Document Computing Symposium*, pp. 1–8, Parramatta, NSW, Australia, December 2015.