Searchable Index Schemes for Groups : Security vs. Efficiency *

Hyun-A Park, Yu Jeong Lee, and Dong Hoon Lee

Center for Information Security Technologies (CIST),
Korea University, Anam Dong, Sungbuk Gu, Seoul, Korea
{kokokzi, autumnyj, donghlee}@cist.korea.ac.kr

Abstract. A secure index search protocol makes it possible to search for the index of encrypted documents using specified keywords without decrypting them. These days, personally portable devices of huge storage such as a USB are easily used and hence private and sensitive documents of a user may be securely kept in such personal devices. However, secret documents shared by groups are usually stored in database. In real organizations such as government offices or enterprises with many departments, a group search occurs more often.
In this paper, we propose two search schemes for a hierarchical group under an untrusted server ; A security-centered search scheme(SSIS) and an optimized efficient search scheme(ESIS) for commercial business use. We define 'correlation resistance' as privacy requirement over encrypted search system and prove that SSIS can meet the notion. Also, we experimented two our proposed schemes. In the first try, the performance of both schemes was not good to use for practical business use. It was not until examining the reason of this that we learned the efficient DB schema must be applied into the search system for good performance. However, it was hard to apply efficient DB schema into SSIS because of its data structure. Hence, we applied efficient DB schema into only ESIS. The experiments show that ESIS is approximately 200 times faster than SSIS, which implies that other existing schemes are also not practical because the data structure of them is similar to SSIS. ESIS achieves real practicabilty by loosening its security, but with at least extend. Therefore, in the near future, it's required to develop keyword search system over encrypted data which is secure and applicable to efficient DB schema. In addition, we learned a lesson that works about the efficiency must consider mutual interactive operation with application layer as well as computational efficiency of a proposing scheme.

Keywords : group search, encrypted data, keyword, trapdoor, index string, index list, efficient DB schema, primary key, foreign key

1 Introduction

When documents contain sensitive data, those are usually encrypted and then stored for secrecy. A secure index search protocol enables a legitimate querier to search encrypted documents in a server with a keyword without decrypting them and revealing any information on the documents to any other, especially to the untrusted server. Up until now, there have been many works on the search protocols for encrypted documents.

* This research was supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Assessment)

The search for encrypted data was started by Song, Wagner, and Perrig. They presented a Searchable Symmetric Key Encryption(SSKE) scheme which permits a user, given a trapdoor for a word, to test if a ciphertext block contains the word [20]. However, their scheme requires the server to scan linearly through all the document contents on every query. To solve this problem, Chang and Mitzenmacher proposed two index schemes using the idea of pre-built dictionaries [9]. Song et al. also proposed an index solution that has a single encrypted hash table index for the entire document set. Their encrypted hash table index is computationally efficient [20]. Also, Goh proposed a secure index scheme [10]. He defined a secure index and formulates a security model for indexes known as semantic security against adaptive chosen keyword attack. He also developed a secure index construction called Z-IDX using pseudo-random functions and Bloom filters [9].

As other approaches, Dan Boneh et al. developed the keyword search using a public key system, where they defined the concept of a public key encryption with keyword search (PEKS) and gave two constructions [2]. They showed that PEKS implies Identity Based Encryption, but the converse is currently an open problem.

And B.Waters et al. published the building of an encrypted and searchable audit log [22]. In particular, they implemented an audit log for database queries that uses hash chains for integrity protection and identity based encryption with extracted keywords to enable searching on the encrypted log.

Like this, they only consider the search between a single-user and a server [2,6,9,10,11,14,16,18 ,19,20]. In practical environments, however, a document may be shared by a group or a person, and a server stores such documents. Since there may be the documents requiring security, those documents are encrypted and then stored in order to make them fetchable only by members in a specific group. Especially, because personally portable mass storage devices such as a USB are easily used these days, it would be more realistic to store the secret documents shared by group-members at a server rather than personal secret documents.

We propose two search schemes for a hierarchical group under an untrusted server. One is a security-centered search scheme(SSIS, Secure Searchable Index Schemes) and the other is an optimized efficient search scheme(ESIS, Efficient Searchable Index Schemes) for application into real-world focusing on the more substantial aspects for practical use. As for privacy requirements over encrypted search systems, we define 'correlation resistance' and prove that one of the our two schemes, SSIS can assure correlation-resistant security. We experimented two schemes to evaluate the performance of them. At the first time, however, the performance of both schemes was poor for practical use. After examining the reason of this, we got to know that it's important to apply efficient DB schema into the search system for good performance. However, data structure of SSIS makes it hard to apply efficient DB schema. Hence, we applied efficient DB schema into only ESIS. The experiments show that ESIS is approximately 200 times faster than SSIS, which implies that other existing schemes are also not practical because the data structure of them is similar to SSIS. In fact, ESIS is about 200 times faster than Golle's scheme and 4.5 times faster than Song's scheme. ESIS achieves real practicability yet it looses its security with a minimal limit. That is, ESIS does not reveal keywords or documents to the server, but does expose linkable information between documents containing a common keyword. By analyzing the efficiency based on real experiments like this, we put forward a suggestion about the desirable research view in the future.

1.1 Contribution

Group Search. In a hierarchical group setting, a server contains heterogeneous documents accessible by different groups or persons. Designing secure index search in the hierarchical group setting is not an easy work since a group may be dynamic, i.e., a person may join to and

leave from the group. First, for a leaving member from a group, all documents accessible to the group should not be accessible any more. Second, a newly joining member to a group should be able to search all the previous and current documents. This makes the design even harder. A naive solution will be to decrypt all documents of the group, and re-encrypt them by the new group key. But this requires a large amount of computational overhead.

Park et al.[15] firstly proposed search schemes for groups, which can search both the previous and current documents without re-encrypting documents for each membership change. Unfortunately, the scheme can not provide backward secrecy. Namely, if one of the leaving members reveals his group key to a server, the server can decrypt all the documents previously encrypted. It means complete breakdown of the system. Furthermore, since a user must generate the same number of trapdoors as the number of sessions for one keyword, the scheme does not fit for the mobile system whose membership change is very dynamic.

Hence, we come to an conclusion that the unknown factors to users is necessary for a group's search system. It is another group key, i.e. independent search key such that must not be changed but fixed despite re-keying(membership change). For this purpose, we introduce a client program. It is an access program into a search system in which the information for group authentication and the search keys for the matching groups are stored. It is similar to SIS-GC of Park et al.

Privacy. A user encrypts documents and generates the indexes corresponding to each encrypted document. These are stored at a server and a user queries with the legitimate trapdoor which is generated with a user's search key and keywords. Where, an index for each document is composed of encrypted principal keywords of the document. Our schemes can provide the following privacy for the search schemes over encrypted documents;

1. The indexes must not leakage any information about its contents without a legitimate trapdoor.
2. In the search process, there must not exist additionally leaking information through the query and its results.

We call the above 'correlation resistance' and one of our schemes, SSIS can meet it. For more details, refer to section 2.2 and 3.5.

Efficiency and Performance Analysis. Unlike the advance of theoretic research papers on this area, the studies about applicable methods for practical use have seldom or never been worked. A preconception that a search system over encrypted data can not guarantee an efficient search process has interrupted the development of a practical search system. Hence, we propose the efficient and optimized scheme for business use, ESIS. Unfortunately, it can not meet the 'correlation resistance'.

At first, we experimented both schemes SSIS and ESIS. Unexpectedly, the performance of SSIS was not so good and even ESIS couldn't meet the satisfied value. After examining the reason, we can find that efficient DB schema plays a significant role of good efficiency. However, we could not apply efficient DB schema into SSIS because the data structure of SSIS is not appropriate for applying efficient DB schema. Eventually, we applied DB schema into only ESIS and the performance of it was great. The experiments show that ESIS is approximately 200 times faster than SSIS.

The data structure of SSIS is similar to that of most existing schemes in which each document and its indexes are stored in row not in field(column). The schemes require at least one computation(test equation) to verify each document by every row. Since the data structure of these existing schemes including SSIS is hard to apply efficient DB schema, we can say that

most existing schemes are not efficient for business use.

Conjunctive Search. Golle et al. proposed efficient conjunctive search[1] schemes which constructed keyword field. The keyword field is inappropriate for a hierarchical group's database because of various kinds of documents. Hence, we introduce index-strings and an index list instead of a keyword field so that these can prohibit any information from leaking by keyword field indexes. SSIS can provide semi-conjunctive search using meta-keywords and ESIS can provide conjunctive search using intersection operation respectively without a keyword field.

In the search process of Golle's scheme, a user gives the field indexes to a server as searching capability [11], where the server must store proto-capabilities[2] alongside each encrypted document until they are used, which are discarded after being used once. These proto-capabilities should be updated for each document by every query time. Consequently, by this method, Golle's scheme enables a server to learn nothing other than the result of query from the combination of accumulated results of queries. On the other hand, SSIS generates a trapdoor using only one random number by every query time and makes up the efficient test equation in the test and search stage(section 4.3), so that it can provide correlation-resistant security with only one computation for each document without the process of updating proto-capabilities for all documents by every query time. If it were not for such an efficient test equation, we would test all the indexes(encrypted keywords) to each document like as PEKS of Boneh et al.[2]. Furthermore, SSIS can provide security of the same level as Golle's scheme and doesn't have to update all documents every query time.

Multipurpose and Access Control. The environment of our schemes is for a huge hierarchical group search system between a server and multi-user. Namely, our system isn't limited to a person's private documents or similar kinds of documents of a special-purpose group such as mail server. However, it is allowed to various kinds of documents aimed at multipurpose ; documents open to all members in a group, documents shared by a subgroup and private documents owned by a person. Our protocol provides access control for these documents. That is, an only person who can authenticate whether he is the legitimate owner of the search key can search, read and update his encrypted documents.

2 Model

We assume that attackers against our search system include not only outside attackers but also inner attackers such as an untrusted server.

We assume a GM(group manager) and client programs. A GM manages the key-change of dynamic groups. A client program is a kind of search program which is established in all user's PC. If a user wants to search some data, he must contact with a client program. In this client program, there exists a key matching table to authenticate users. Being always connected with a GM on-line, the client program can receive the changed keys in real time through a secure channel whenever rekeying happens.

Notation. X_i^s : a group i's group key in the s-th session, K_i: a group i's search key, R_i : a group i's document encryption key, x_j: a user p_j's private key, k_j: a user p_j's search key, r_j : a user p_j's document encryption key, ID_j: a user j's identifier, D_n: the n-th document, d_n: the identifier of the document D_n, $w_{n,t}$: t-th keyword of the n-th document D_n, $I_{n,t}$: t-th index

[1] the search for documents containing all the keywords what a user want to search
[2] a kind of ability that a user allows the server to identify exact documents

of the n-th document D_n, k_c: a client program's secret key, α_q: a random number generated at q-th query.

2.1 Algorithms

Our protocol has the following 6 algorithms.

- **SysParam(1^k)** - Parameter-generation algorithm **SysParam(1^k)** takes an input as a secret parameter k and produces a system parameter λ.
- **KeyGen(λ)** - Key-generation algorithm **KeyGen(λ)** produces a group's secret key set X, a group's search key set K, and a document encryption key set R.
- **IndGen(λ, K, W)** - Taking inputs as λ, K, and keyword set $W = \{w_1, w_2, ...\}$, index-generation algorithm **IndGen(λ, K, W)** outputs indexes encrypting keywords.
- **Auth(ID, pw)** - Authentication algorithm **Auth(ID, pw)** takes inputs as a user's identifier ID and password pw. If there exist the matching pair $(ID, h(pw))$ in the client's key matching table, then output 1, otherwise 0, where h is a one way hash function.
- **Trapdoor(λ, K, W)** - Taking inputs as a group's search key K, system parameter λ , and keyword W, trapdoor algorithm **Trapdoor(λ, K, W)** outputs T_w.
- **TestFind(T_W, I)** - This algorithm takes inputs as trapdoor T_W, and index I. If there exist corresponding value in the index, then outputs 1, otherwise 0.

2.2 Primitives and Notions of Security

Our proposed schemes take primitives as DDH, PRF, PRG and so on. And SSIS provides correlation-resistant security which is achieved through semantic security.

Definition 1. DDH (Decisional Diffie-Hellman). Let G be a group of prime order q and g a generator of G. The DDH problem is to distinguish between triplets of the form (g^a, g^b, g^{ab}) and (g^a, g^b, g^c), where a, b, c are random elements of $\{1, ..., q-1\}$.

Consider the following experiment with a polynomial time adversary A : Flip a coin δ(to get 0 or 1), if $\delta = 1$, set $c = ab$, else choose c at random. The DDH problem is said to be hard if for any polynomial time adversary A, $|Pr(A(G, g^a, g^b, g^c) = \delta) - 1/2|$ is negligible.

Definition 2. PRF(Pseudo Random Function). We say that '$F : K_f \times X \longrightarrow Y$ is (t, q, e) -secure pseudorandom function' if every oracle algorithm A making at most q oracle queries and with running time at most t has advantage $Adv_A < e$. The advantage is defined as $Adv_A =\mid Pr[A^{F_k} = 1] - Pr[A^R = 1] \mid$ where R represents a random function selected uniformly from the set of all maps from X to Y , and where the probabilities are taken over the choice of k and R [20].

Definition 3. PRG G_r(Pseudo Random Generator). We say that '$Gr : K_{Gr} \longrightarrow S$ is a (t, e)-secure pseudorandom generator' if every algorithm A with running time at most t has advantage $Adv_A =\mid Pr[A(Gr(U_{K_{Gr}})) = 1] - Pr[A(U_S) = 1] \mid$. Where $U_{K_{Gr}}$, U_S are random variables distributed uniformly on K_{Gr}, S [20].

Definition 4. Correlation-Resistance. If two ciphertexts are (1)the encryption of the same plaintext, or (2)the encryption of two arbitrary plaintexts - equal or distinct - under the same unknown key, we regard the ciphertexts as being correlated. In the encrypted searching system, 'Correlation-Resistance' means that an adversary A including an inner attacker(a server) and outer attacker must not be able to link the related documents unless they are provided with

searching capability [5].

Definition 5. 'An adversary $A(t, q, \epsilon)$ breaks an Correlation-Resistance'. We say that an adversary $A(t, q, \epsilon)$-breaks 'Correlation-Resistance' if Adv_A is at least ϵ after A takes at most t time and makes q trapdoor queries to the challenger;

1. Trapdoors can be distinguished whether they are encrypted with the same keyword of the same group or not.
2. Indexes can be distinguished whether they are encrypted with the same keyword or not.

Definition 6. IND-CKA(Semantic security against adaptive chosen keyword attack). An adversary A cannot deduce document's contents from its index string and trapdoor. If A cannot determine which document is related to the index string and the trapdoor with probability non-negligibly different from $1/2$, then the index and trapdoor reveal nothing about its contents. We use this formulation of indistinguishability(ind) to prove the semantic security (IND-CKA) of indexes and trapdoors. It's based on Goh's security model [10]. We use the following game between a challenger C and an attacker A to define semantic security against an adaptive chosen keyword attack.

- **Setup.** The challenger C creates a set W of q words and gives this to the adversary A. A chooses a number of subsets from W. This collection of subsets is called W^* and is returned to C. Upon receiving W^*, C runs algorithm **KeyGen** to generate the master key and encrypts each subset W^* running algorithm **IndGen**. Finally, C sends all indexes with their associated subsets to A.
- **Queries.** A is allowed to query C on a word w and receive the trapdoor T_w for w. With T_w, A can invoke algorithm **Test and Find** on an index I to determine if $w \in I$.
- **Challenge.** After making some Trapdoor queries, A decides on a challenge by picking a nonempty subset $D_0 \in W^*$, and generating another non-empty subset D_1 from W. A must not have queried C for the trapdoor of any word in $D_0 \cup D_1$. Next, A gives D_0 and D_1 to C and C chooses $b \overset{\$}{\leftarrow} \{0, 1\}$, invokes algorithm **IndGen** to obtain the index I_b for D_b, and returns I_b to A. The challenge for A is to determine b. After the challenge is issued, A is not allowed to query C for the trapdoors of any word $w \in D_0 \cup D_1$.
- **Response.** A eventually outputs a bit b', representing its guess for b. The advantage of A in winning this game is defined as $Adv_A = |Pr[b = b'] - 1/2|$, where the probability is over A and C's coin tosses.

Definition 7. We say that **'an adversary $A(t, \epsilon, q)$ breaks an index'** if Adv_A is at least ϵ after A takes at most t time and makes q trapdoor queries to the challenger.

Now, we construct two concrete protocols using above algorithm. The protocols consist of 4 processes ; SetUp Process, Group-Authentication Process, Uploading Process, and Search Process.

3 SSIS(Secure Searchable Index Scheme)

SSIS mainly focuses on the efficient correlation resistant security.

3.1 SetUp Process

Algorithm **SysParam($1^{\mathbf{k}}$)** outputs system parameter $\lambda = (G, g, f(\cdot), Gr)$. Where G is a group of order q which is a large prime and g is a generator of a group G. $f : \{0,1\}^{\mathbf{k}} \times \{0,1\}^* \rightarrow \mathbb{Z}_q$

is a pseudo random function. Key generation algorithm **KeyGen(λ)** outputs a group's secret key set $X \in \{0,1\}^{\mathbf{k}}$, a group's search key set $K \in \{0,1\}^{\mathbf{k}}$, and a group's document encryption key set $R \in \{0,1\}^{\mathbf{k}}$. $f(K,\cdot)$ is denoted as $f_K(\cdot)$ and $\{f_K(\cdot)\}_K$ is a pseudo random function family. Gr is a pseudo random generator.

In a client program, there is a key matching table such as Table 1. Table 1 means that for

ID	PW	Search Key	Encryption Key
$ID_1, ID_2, ID_3, ...$	$h(PW_1)$	$E_{k_c}(K_1)$	$E_{k_c}(R_1)$
$ID_{35}, ID_{36}, ID_{39}, ...$	$h(PW_2)$	$E_{k_c}(K_2)$	$E_{k_c}(R_2)$
.....	.....	.....	.....
$ID_{251}, ID_{253}, ID_{274}, ...$	$h(PW_{10})$	$E_{k_c}(K_{10})$	$E_{k_c}(R_{10})$
.....	.....	.....	.....
ID_2	$h(pw_2)$	$E_{k_c}(k_2)$	$E_{k_c}(r_2)$
ID_5	$h(pw_5)$	$E_{k_c}(k_5)$	$E_{k_c}(r_5)$
.....	.....	.....	.....

Table 1. A key matching table in a client program

example, $ID_1, ID_2, ID_3, ...$ are members of a subgroup g_1 and PW_1 is used as a password for group members of a subgroup g_1. A subgroup g_i's search key K_i and document encryption key R_i are encrypted by a client's secret key k_c. K_i is an unchanged search key of a subgroup g_i regardless of re-keying. The lower part of Table 1 shows key-matching for a private user not as a group-member.

The contents of key-matching table are updated by a GM on-line as soon as re-keying happens. This is processed through a secure channel.

3.2 Authentication Process

To search the databases, a user has to access the client program and authenticate himself. By algorithm **Auth**, a user inputs his ID and a password in log-in pane. Where a password is PW_i or pw_i. The former is a password for group membership authentication and the latter is for private user authentication.

Then, the client program searches the same value pair as $(ID, h(password))$ in the key matching table and returns the searching result to a user as success/failure to access.

3.3 Uploading Process

After successful user-authentication, a user inputs documents and keywords what he wants to store in a server. A client-program decrypts the search key $E_{k_c}(K_i)$ and the document encryption key $E_{k_c}(R_i)$, which are stored along together with ID and password in the key matching table, with his secret key k_c. Then, the client encrypts input documents and keywords with the key R_i and K_i.

In SSIS, an index string for each document is produced using t encrypted keywords. With the r keywords out of t keywords, we make up meta-keywords for conjunctive search. Where the number of t and r can be determined by the group's policy($1 \leq r \leq t$) considering efficiency and functionality. Perfect conjunctive search requires $r = t$. However, since it requires a lot of meta-keywords to generate, we choose appropriate r keywords for semi- conjunctive keyword search.

For example, if t, the total number of keywords is 7 and r is 3, then single keywords are 7, 2-word keywords are $21({}_7C_2)$, 3-word keywords are $35({}_7C_3)$. Accordingly, an index string is composed of 63 encrypted keywords for each document. An index string is constructed as follows;

$$\mathcal{C}_n = \{E_{R_i}(D_n)\}$$

$$\mathcal{I}_n = \{id_{n,0}, id_{n,1}, I_{n,1}, I_{n,2}, ..., I_{n,63}\}$$

$$= \{g^{d_n K_i}, g^{-d_n}, g^{d_n K_i f_{K_i}(w_{n,1})}, ..., g^{d_n K_i f_{K_i}(w_{n,7})}, g^{d_n K_i f_{K_i}(w_{n,1} \| w_{n,2})}, ...,$$
$$g^{d_n K_i f_{K_i}(w_{n,7} \| w_{n,6})}, g^{d_n K_i f_{K_i}(w_{n,1} \| w_{n,2} \| w_{n,3})}, ..., g^{d_n K_i f_{K_i}(w_{n,7} \| w_{n,6} \| w_{n,5})}\}$$

Where, $E_{R_i}(D_n)$ is the encryption of a document D_n using a certain symmetric cryptologic method with document encryption key R_i and $id_{n,0} = g^{d_n K_i}$, $id_{n,1} = g^{-d_n}$ are encryptions of identifier d_n, they can provide a server with searching capability in the Test stage of search process. The identifier d_n for a document D_n is chosen by the client program at random, where g^{d_n} can be set as h_n. When the input keywords are encrypted, the client program salts the keywords in lexicographic order. This makes it possible to use ${}_7C_2$ and ${}_7C_3$ instead of ${}_7P_2$ and ${}_7P_3$ to diminish the number of meta-keywords because $w_{n,1} \parallel w_{n,2}$ is different from $w_{n,2} \parallel w_{n,1}$.

After producing Ciphertexts and indexes, client sends them to a server. A server stores them as follows;

$$\{id_{n,0}, id_{n,1}, I_{n,1}, I_{n,2}, ..., I_{n,63}, \mathcal{C}_n\} = \{h_n^{K_i}, h_n^{-1}, h_n^{K_i f_{K_i}(w_{n,1})}, ..., h_n^{K_i f_{K_i}(w_{n,7})}, h_n^{K_i f_{K_i}(w_{n,1} \| w_{n,2})}$$

,

$$....., h_n^{K_i f_{K_i}(w_{n,7} \| w_{n,6})}, h_n^{K_i f_{K_i}(w_{n,1} \| w_{n,2} \| w_{n,3})}, ..., h_n^{K_i f_{K_i}(w_{n,7} \| w_{n,6} \| w_{n,5})}, E_{R_i}(D_n)\}$$

3.4 Search Process

1. **Trapdoor generation stage.** Algorithm **Trapdoor**$(\lambda, K_i, W_{n,t})$ outputs $T_w = (T_1, T_2)$. If a user succeeds in authentication, a keyword pane for searching is showed up. If r is 3, 3 blanks are showed up. A user inputs keywords and the client program generates trapdoors as follows and sends them to a server.

$$single\ keyword : (T_1, T_2) = (f_{K_i}(w_{n,t}) + \alpha\ (mod\ q),\ K_i\alpha\ (mod\ q)\ )$$

$$two\ keywords : (T_1, T_2) = (f_{K_i}(w_{n,1} \parallel w_{n,2}) + \alpha\ (mod\ q),\ K_i\alpha\ (mod\ q)\ )$$

$$three\ keywords : (T_1, T_2) = (f_{K_i}(w_{n,1} \parallel w_{n,2} \parallel w_{n,3}) + \alpha\ (mod\ q),\ K_i\alpha\ (mod\ q)\ )$$

Where, α is newly generated at random in each query time and the trapdoor is computed by modulus. Hence, a server can not learn the following facts : 1) whether a queried keyword is the same as one of the previously queried keywords. 2) whether the used key K_i for querying is the same as one of the previous queriers' keys, even if the same group member queries for the same keyword. Therefore, it prohibits a server from deducing information from cumulative results.

2. **Test and Search stage.** By algorithm **Test and find**, the server computes the followings using the received trapdoor and the two elements $id_{n,0}$ and $id_{n,1}$ in the index string I_{d_n} for each document. For all $n = \{1, 2, ..., n\}$,

$$(id_{n,0})^{T_1} \cdot (id_{n,1})^{T_2} = (h_n^{K_i})^{(f_{K_i}(w_{n,t})+\alpha)} \cdot (h_n^{-1})^{(K_i\alpha)}$$
$$= h_n^{K_i f_{K_i}(w_{n,t})}$$

And then, the server scans the same value as the above in each index string for all documents. If it exists, the server returns the corresponding ciphertexts $\mathcal{C}_n = \{E_{R_i}(D_n)\}$ to the client-program.

3. **Output stage.** Client decrypts the received ciphertexts $\mathcal{C}_n = \{E_{R_i}(D_n)\}$ using the document encryption key R_i and outputs the results on a user's computer.

3.5 Security of SSIS

The security notion of keyword search on encrypted data is that a server can learn nothing more than the queried trapdoor for some keywords is related to the resulted documents. In particular, SSIS focuses on '$correlation - resistance$' which means that if an attacker queries q trapdoors at most t time to a challenger;

1. A trapdoor generated every query time can not be distinguished whether they are the encryptions for the same keywords of the same group or not.
2. It's impossible to distinguish whether two index strings contain the same keyword of the same group or not.

Theorem 1. Under the assumption that DDH is hard, if Gr is (t, e)-secure pseudo random generator, and $f \in F$ is (t, q, e)-secure pseudo random function, then SSIS provides Correlation-Resistance.

To be more specific, Theorem 1 can be restated as Lemma 1 and Lemma 2 to meet (1) and (2).

Lemma 1. If Gr is (t, e)-secure pseudo random generator, SSIS provides semantic security for trapdoors against adaptive chosen keyword attack.

Lemma 1 is proved in Appendix A because of the lack of space.

Lemma 2. Under the assumption that DDH is hard, if $f \in F$ is (t, q, e)-secure pseudo-random function, then SSIS provides semantic security for indexes against adaptive chosen keyword attack(IND-CKA) over finite field G.

The indexes for a common keyword w of a group i's documents D_1 and D_2 with the search key K_i are $g^{d_1 K_i f_{K_i}(w)} = h_1^{K_i f_{K_i}(w)}$ and $g^{d_2 K_i f_{K_i}(w)} = h_2^{K_i f_{K_i}(w)}$. They have different values respectively because of unique identifiers d_1 and d_2. Hence, although two indexes are for a common keyword, an attacker can not know the fact. The proof of lemma 2 is similar to lemma 1 so that we explain it in an Appendix B, too.

4 ESIS (Efficient Searchable Index Scheme)

ESIS focuses on the design of the most optimal model for practical use. Hence, by minimizing computational overhead in the test/search stage and applying efficient DB schema, it can diminish the difference from plaintext search systems. As a solution, we use an index list.

SetUp Process and Group-Authentication Process are the same as SSIS. Therefore, we address only Uploading Process and Search Process.

4.1 Uploading Process

After user-authentication, a user inputs a document D_n and its keywords $w_{n,1}, w_{n,2}, ...$, like as SSIS.

The client program decrypts user's search key with his secret key and with this, he encrypts the document D_n and its keywords $w_{n,1}, w_{n,2}, ...$ Unlike SSIS, the indexes for meta-keywords are not generated in ESIS. Even so, we can not give up conjunctive search. We deal with more details about conjunctive search in Search Process.

The ciphertext and indexes are generated as follows;

$$\{d_n, E_{R_i}(D_n), f_{k_i}(w_{n,1}), f_{k_i}(w_{n,2}), ...\}$$

The identifier d_n for a document D_n is chosen by the client program at random. $E_{R_i}(D_n)$ is an ciphertext of D_n and $f_{K_i}(w_{n,1}), f_{K_i}(w_{n,2}), ...$ are indexes which are the encryptions of keywords.

The client program sends the ciphertexts and indexes to a server and the server updates an index list. The server adds t rows composed of indexes and their document identifier d_n like as $f_{K_i}(w_{n,1}) : d_n,\ f_{K_i}(w_{n,2}) : d_n, ...,\ f_{K_i}(w_{n,t}) : d_n$ to the index list. The index list is stored at the server like as the left side of Table 2. Then, the server stores an identifier d_n and a document $E_{R_i}(D_n)$ in row like as the right side of Table 2. Namely, ESIS is composed of 2 databases, where d_n plays a role of an address as well as an identifier of D_n.

Index	Identifiers of Documents		Identifiers of Documents	Encrypted Document
$f_{K_i}(w_{1,1})$	d_1		d_1	$E_{R_i}(D_1)$
$f_{K_i}(w_{1,2})$	d_1		d_2	$E_{R_j}(D_2)$
.....	.....		.....	.....
$f_{K_i}(w_{1,t})$	d_1		d_7	$E_{R_p}(D_7)$
$f_{K_j}(w_{2,1})$	d_2		d_8	$E_{R_q}(D_8)$
$f_{K_j}(w_{2,2})$	d_2		d_9	$E_{R_s}(D_9)$
.....	.....		.....	.....
$f_{K_j}(w_{2,t})$	d_2		d_{14}	$E_{R_h}(D_{14})$
.....	.....		.....	.....
$f_{K_h}(w_{14,1})$	d_{14}		d_{561}	$E_{R_i}(D_{561})$
.....	.....		.....	.....
$f_{K_h}(w_{14,t})$	d_{14}		d_n	$E_{R_i}(D_n)$
.....	.....			
$f_{K_i}(w_{n,t})$	d_n			

Table 2. Index list(left) and Encrypted Documents(right)

Since an index list is made in this way, we can apply efficient DB schema using primary key and foreign key into ESIS.

4.2 Search Process

1. **Trapdoor generation stage.** Algorithm **Trapdoor**$(\lambda, K_i, W_{n,t})$ outputs T_w. After user-authentication, a user inputs keywords what he wants to search. Then, the client program computes trapdoors using the same method as index generation, and sends them to a server.

$$T_w = (f_{K_i}(w_{n,1}),\ f_{K_i}(w_{n,2}), ...)$$

2. **Test and Search stage.** By algorithm **Test and Find**, the server scans the identifiers of the same value as the trapdoor in the index list(left) and then scans the encrypted documents corresponding to the identifiers at the right side of Table 2. If a query is for single keyword, the server only has to search one index. If a query is for conjunctive keyword search, the server searches for each keyword individually in the same way as single keyword and stores the results in a set for each keyword. After doing the intersection-operation, the server sends the resulted documents to the client program. There is no computation to test and search but scanning.

3. **Output stage.** The client program decrypts the received ciphertexts with a user's document encryption key and outputs the plaintexts on user's computer.

5 Evaluation of Performance

To evaluate the efficiency of encrypted search systems more precisely, we also experimented the plaintext versions of proposed schemes in which encryption is removed.

The setting of our experiments is described in Appendix C.

Table 3 is the result of our experiment. P-SSIS and P-ESIS are the plaintext versions of SSIS and ESIS. Table 3 indicates the required time(unit; ms) that each scheme implements each process such as up-loading, single keyword search(1-keyword), conjunctive keyword search(2-keywords), where '2500, 5000, 7500, 10000' means the number of documents.

At first, we experimented without applying efficient DB schema. The performance of even P-SSIS and P-ESIS as well as SSIS and ESIS was very poor. The reason of this is efficient DB schema. The database for efficient search system must be supported by the efficient DB schema no matter whether it is encrypted or not. However, SSIS could not be applied to efficient DB schema due to the structure of stored data. The details are explained below. Hence, we applied efficient DB schema into only ESIS. As it can be seen in the Table 3, the performance of ESIS is greatly superior.

Scheme	P-SSIS		SSIS		P-ESIS		ESIS	
Uploading	54		154		28		49	
Search	single	conjunc.	single	conjunc.	single	conjunc.	single	conjunc.
2500	94	102	6171	14148	46	49	73	90
5000	172	203	12359	28250	50	54	110	120
7500	203	250	18516	42554	61	65	138	172
10000	281	320	37828	57578	69	75	185	213

Table 3. Performance. (time unit:ms)

As expected, the fastest uploading time(for one document) is the plaintext version of ESIS which does not use meta-keywords and any encryption method. As the number of documents grows, the performance of ESIS is much less influenced than SSIS. Because the searching method of ESIS is the direct access to corresponding documents without testing whether each document contains the queried keyword or not. Compared with plaintext search systems, the gap of P-ESIS and ESIS is much less than that of P-SSIS and SSIS. Comparing single keyword search with conjunctive search, the difference in ESIS using intersection method is less than SSIS using meta-keywords. Because there are many fields to scan by every row on conjunctive search using meta-keywords.

From a practical point of view, ESIS is an optimized scheme for commercial business. The first reason for this is the structure of index list applicable to efficient DB schema. Using 'primary key' : index(encrypted keyword) and document identifier and 'foreign key' : document identifier, it enables the server to immediately access the corresponding data to a user's query. However, in most of the existing schemes, indexes(a collection of the encrypted keywords) for each document are stored in row not in field(column)[3] and they require at least one computation(test equation) for verification by every row for each document [2,6,9,10,11,12,14,15,16,18,19,20,22]. To apply primary and foreign key, indexes should be stored in one field not in row and there must not be computations to test whether this keyword is contained intended documents. Evidently, we found that the structure of stored data of most existing schemes including SSIS is inappropriate for applying efficient DB schema. From this reason, we could guess that the performance of the existing schemes is also poor.

To demonstrate it, we experimented two existing schemes ; Song's scheme and Golle's scheme. Table 4 indicates the performance of single keyword search over 10,000 documents. The

	Song	Golle	SSIS	ESIS
1-keyword	825	38922	37828	185

Table 4. Comparison of Performance

performance of ESIS is the most superior and the next is Song's scheme which is SSKE(searchable symmetric key encryption). The implementing time of ESIS is about 200 times faster than Golle's scheme and about 4.5 times faster than Song's scheme.

Furthermore, the kind of used function greatly influences the searching time. Among the recently proposed keyword search schemes, there are some schemes based on bilinear function [1, 2, 5, 11, 14, 16,18]. According to http://indigo.ie/ mscott, the calculation of a 512-bit Tate pairing, which is faster than Weil pairing in general, takes 20ms on a 1GHz Pentium III and also a 1024-bit RSA decryption/signature takes 20ms on a 233 Mhz Pentium II. However, considering the CPU of used computers, we can say that RSA decryption/signature is at least 4 times faster than Tate pairing. Moreover, since decryption/signature operation includes additional computations, the calculation of pairing takes at least 4 times or more time than that of RSA. Compared with symmetric cryptologic method, it will be more inefficient. In conclusion, we get to know that bilinear function is not appropriate for real-world application especially for the search process on-line. Hence, our schemes never use bilinear function. ESIS is based on the only symmetric cypto-system. Especially, in the search process which is one of the most important processes in respect of time-consumption, its computational overhead is not so different from general plaintext search system. There is no computation to test whether the document contains the queried keyword or not. Since it requires the additional time only to encrypt the keywords and to decrypt the resulted documents in a client program, we can say that it's the optimized scheme to diminish the difference from plaintext search system as indicated in Table 3.

However, as for security, ESIS can't provide correlation-resistance because document identifiers within an index list can be classified by common keywords. For example, we assume $w_{1,1}$=$w_{5,7}$, then $f_{K_1}(w_{1,1}) = f_{K_1}(w_{5,7})$. Although the server can not know that these values $f_{K_1}(w_{1,1}) = f_{K_1}(w_{5,7})$, $E_{R_1}(D_1)$ and $E_{R_1}(D_5)$ are encryptions of $w_{1,1}$, D_1 and D_5, he can know that two documents $E_{R_1}(D_1)$, $E_{R_1}(D_5)$ have the common keyword. That is, it does't reveal

[3] The computation or scanning within one field is relatively faster than in one row. However, the computation or scanning for many fields within one row is not fast.

keywords or documents to the server but does expose linkable information between documents containing a common keyword.

6 Conclusion

Our proposed schemes, SSIS and ESIS are about search system for hierarchical group's encrypted documents.

Especially, SSIS can provide correlation-resistant security with only one computation for each document without the process of updating all the documents by every query time. Also, our schemes are easy to update any time because update process is the same as uploading process. On the other hand, ESIS is the most optimized scheme for commercial business use by applying efficient DB schema. Through experiments of performance, we got to know that the structure of stored data of existing schemes including SSIS is inappropriate for commercial business use. In addition, we learned an important lesson that in efficiency works, we must consider mutual interactive operation with application layer as well as computational overhead.

In conclusion, we need to develop practical keyword search systems over encrypted documents which is secure and applicable to efficient DB schema.

Acknowledgement

This paper is the revised version after failing in my first submission to a conference. I thank the anonymous referees of the conference for greatly careful and acute comments with all my heart. I could make up my mind again and could learn many things. Lastly, I thank two programmers Yu Jeong Lee and Jong Uk Hong.

References

1. Michel Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H.a Shi. Searchable Encryption Revisited: Consistency Properties, Relation to Anonymous IBE, and Extensions. Crypto05, LNCS 3621, pp.205-222, 2005
2. D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano. Public-key encryption with keyword search. Eurocrypt04, LNCS. Springer-Verlag, 2004
3. M. Burrnesterand Y. Desmedt. A secure and efficient conference key distribution system. The Advances in Cryptology - EUROCRYPT, 1994
4. K. Bennett, C. Grothoff, T. Horozov, I. Patrascu. Efficient sharing of encrypted data, ACISP02, 2002
5. L. Ballard, M. Green, B. de Medeiros, F. Monrose, Correlation-Resistant Storage via Keyword-Searchable Encryption, SPAR Technical Report TR-SP-BGMM-050705
6. B. Chor, N. Gilboa, M. Naor. Private Information Retrieval by Keywords Technical Report TR CS0917, 1997
7. D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. ACM 4(2), 1981
8. B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. Private information retrieval. Journal of the ACM, 45(6):965-981, 1998.
9. Y.-C. Chang and M. Mitzenmacher. Privacy preserving keyword searches on remote encrypted data. Cryptology ePrint Archive, 2004
10. Eu-Jin Goh, Secure Indexes, Cryptology ePrint Archive, 2004
11. P. Golle, J. Staddon, B. Waters, Secure Conjunctive Keyword Search Over Encrypted Data, ACNS04, 2004
12. H. Hacigumus, B. Iyer, and S. MehrotraEfficient. Execution of Aggregation Queries over Encrypted Relational Databases, DASFAA2004, LNCS 2793, pp.125-136, 2004

13. L. Kissner and D. Song. Private and Threshold Set-Intersection, CMU TR, 2004
14. Ogata and K. Kurosawa, Oblivious Keyword Search. Journal of Complexity, Volume 20, pp.356 - 371, 2004
15. Hyun-A Park, Jin Wook Byun, and Dong Hoon Lee, Secure Index Search for Groups, TrustBus 2005, LNCS3592 pp.128-140, 2005
16. D. Park, K. Kim, and P. Lee, Public Key Encryption with Conjunctive Field Keyword Search, WISA 2004, LNCS 3325, pp.73-86, 2004
17. A. Perrig, D. Song, and J.D. Tygar. ELK, a new protocol for efficient large-group key distribution, IEEE Symp. on Security and Privacy, pp.247-262, 2001.
18. Hyun Sook Rhee, Jin Wook Byun, and Dong Hoon Lee, Oblivious Conjunctive Keyword Search, WISA2005, LNCS3886, pp.318-327, 2006
19. R. Sion and B. Carbunar, Conjunctive keyword search on encrypted data with completeness and computational privacy, Cryptology ePrint Archive, 2005
20. D. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data, IEEE Symposium on Security and Privacy, pp.44-55, 2000
21. UC Berkeley—CS 170: Ecient Algorithms and Intractable Problems Handout 10 Lecturer : D. Wagner, 2003
22. B.Waters, D. Balfanz, G. Durfee, and D. Smetters. Building an encrypted and searchable audit log. NDSS Symposium, pp.205-214, 2004

A The proof of lemma 1.

We prove lemma 1 with contraposition. We assume that SSIS is not IND-CKA for trapdoors. Then, there exists an algorithm $A(t, \epsilon, q)$ which breaks SSIS. We construct an algorithm β which can solve the problem about whether G_r is pseudo random or random generator with non-negligible probability using an algorithm A. Where, algorithm β can query the oracle $\mathcal{O}_{G_r}$ and obtain the value of random number α_q. Algorithm β simulates algorithm A as follows;

- **Setup.** Algorithm β picks a set W of q words uniformly at random and sends W to A. A chooses a polynomial number of subsets of W. We call this collection W^*. A sends them to β again. Upon receiving W^*, β runs algorithm **KeyGen** to generate search keys K_i and invokes algorithm **IndGen** for each document D_n and its keywords set $W_n \in W^*$. Where, an index string I_{d_n} is produced and an unique identifier d_n for each document D_n is assigned. After producing all the index strings and ciphertexts, β sends them to A.
- **Queries.** If A queries the trapdoor for a word $w \in D$, β runs algorithm **Trapdoor** for a word w and produces T_w and sends it to A. Where, T_w is obtained through the queries to the oracle $\mathcal{O}_{G_r}$ and again with T_w as input value, A runs algorithm **Test and Find** to check $w \in I_{d_n}$.
- **Challenge.** After making some queries, A chooses distinctive $w_{n,0},\ w_{n,1} \in W_n$. A can not query any trapdoor for $w_{n,0},\ w_{n,1} \in W_n$. Next, A gives $w_{n,0}$ and $w_{n,1}$ to β and β chooses $b \overset{\$}{\leftarrow} \{0,1\}$, and then invokes algorithms **Trapdoor** for $w_{n,b}$. The resulted value of this algorithm, $T_{w_{n,b}}$ is returned to A.
- **Response.** A finally outputs a bit b', representing its guess for b. If $b = b'$, then β outputs 1. Otherwise, β outputs 0. Where, 'β outputs 1' means that G_r is a pseudo random generator.

We show that β can solve the problem about whether G_r is pseudo random or random generator with non-negligible probability. Accordingly, the advantage of β in winning this experiment is ;

$$
\begin{aligned}
Adv_\beta &= Pr[Exp_\beta^{PR} = 1] = Pr[b' = b] \\
&= Pr[b' = b | b = 1] \cdot Pr[b = 1] + Pr[b' = b | b = 0] \cdot Pr[b = 0] \\
&= Pr[b' = b | b = 1] \cdot \frac{1}{2} + Pr[b' = b | b = 0] \cdot \frac{1}{2} \\
&= Pr[b' = 1 | b = 1] \cdot \frac{1}{2} + Pr[b' = 0 | b = 0] \cdot \frac{1}{2}
\end{aligned}
$$

$$= Pr[b' = 1|b = 1] \cdot \frac{1}{2} + (1 - Pr[b' = 1|b = 0]) \cdot \frac{1}{2}$$
$$= \frac{1}{2} + \frac{1}{2}(Pr[Exp_A^{ind-cka-1} = 1] - Pr[Exp_A^{ind-cka-0} = 1])$$
$$= \frac{1}{2} + \frac{1}{2}Adv_A^{ind-cka} = \frac{1}{2} + \frac{1}{2}\epsilon$$

Accordingly, we proved lemma 1 by constructing an algorithm β with $Adv_\beta > \frac{1}{2}$, non-negligible probability. □

B The proof of lemma 2.

We prove it with contraposition, too. We assume that SSIS is not IND-CKA index. Then, there exists an algorithm $A(t, \epsilon, q)$ which breaks SSIS. We construct an algorithm Δ which use algorithm A to solve the DDH and to determine f is a pseudo random function or a random function with non-negligible probability. An algorithm Δ can access to oracle $\mathcal{O}_f$ to produce an index string. The unknown function f takes input $w \in D_n$ and returns $f(w)$. Δ obtains the value of f with the queries to the oracle $\mathcal{O}_f$. $\mathcal{O}_f$ takes as input $W_n = \{w_{n,1}, w_{n,2}, ..., w_{n,t}\}$ and outputs a tuple $\{g^{K_i}, g^{-1}, g^{K_i f(w_{n,j})}\}$.(where, $1 \leq j \leq t$.) Algorithm Δ simulates A as follows;

- **Setup.** Algorithm Δ picks a set W of q words uniformly at random and sends W to A. A chooses a polynomial number of subsets of W. We call this collection W^*. A sends them to Δ again. Upon receiving W^*, Δ runs algorithm **KeyGen** to generate search keys K_i and queries an oracle $\mathcal{O}_f$ for each keyword set $W_n \in W^*$. Δ runs algorithm **IndGen** and then **IndGen** takes as input $\{g^{K_i}, g^{-1}, g^{K_i f(w_{n,j})}\}$ and outputs an index string $I_{d_n} = \{g^{K_i d_n}, g^{-d_n}, g^{K_i f(w_{n,j}) d_n}, ...\}$, where an unique identifier d_n for each document D_n is assigned. Let $I_{d_n} = \{g^{\alpha K_i}, g^{-\alpha}, g^{K_i f(w_{n,j})\alpha}\}$. If f is a pseudo random function, we set $f(w_{n,j})$ is $f_{K_i}(w_{n,j}) = \beta$. Otherwise, if f is a random function, $f(w_{n,j})$ is set as randomly chosen number r_j. We define $\gamma = \alpha\beta$. After producing all the index strings, Δ sends their index strings to A.
- **Queries.** If A queries the trapdoor for a word w, Δ runs algorithm **Trapdoor** for a word w and produces T_w and sends it to A. And then, A runs algorithm **TestFind** to check $w \in I_{d_n}$.
- **Challenge.** After making some queries, A chooses $W_{d_0} \in W^*$, and generates another subset W_{d_1} from W. A must not have queried Δ for the trapdoor of any word in $W_{d_0} \cup W_{d_1}$. Next, A gives W_{d_0} and W_{d_1} to Δ and Δ chooses $b \stackrel{\$}{\leftarrow} \{0, 1\}$, invokes an oracle $\mathcal{O}_f$ and algorithm **IndGen** for W_{d_b}. In this processing, the unique identifier d_b for W_{d_b} is assigned and an index string I_{d_b} is produced. Δ returns it to A.
- **Response.** A eventually outputs a bit b', representing its guess for b. If $b = b'$, then Δ outputs 1. Otherwise, Δ outputs 0. 'Δ outputs 1' means that he can solves DDH, at the same time he can determine f is a pseudo random function or a random function. Because he can know either $g^\gamma = g^{\alpha\beta}$ or $g^\gamma = g^{\alpha r} = r'$(i.e. random number) if and only if he is able to determine $f(w_{n,j})$ is β or r.

We show that Δ can solve DDH and determine f is a pseudo random function or random function with non-negligible probability. Namely, the advantage of Δ in winning this experiment is;

$$Adv_\Delta = Pr[Exp_\Delta^{\mathcal{O}_f} = 1] = Pr[Exp_\Delta^{DDH} = 1] = Pr[b' = b]$$

The subsequent processes are the same as lemma 1.

$$= \frac{1}{2} + \frac{1}{2}Adv_A^{ind-cka} = \frac{1}{2} + \frac{1}{2}\epsilon$$

Consequently, we showed lemma 2 by constructing an algorithm Δ with non-negligible Adv_Δ. □

C The setting of experiments

The setting of PC and data set is as follows :

C.1 PC

- Local Server Operating System : Win-XP
- Client Operating System : Win-XP
- Database Server : MS SQL Server 2000
- PC : Intel Pentium 4 CPU 2.66GHz, 512RAM
- Library : WinAPI C Library, OpenSSL Crypto Library(AES-CBC-128), MS-SQL DB Library for C

C.2 Data Set

- the total number of documents : 10,000
- the number of keywords for each documents : 7
- the size of each keyword : 32byte(=32×16=256bits)

We consider the case that different documents contain common keywords so that we set for a common keyword to be repeated at least every 435 documents in total 10,000 documents.