International Journal of Networking and Computing – www.ijnc.org, ISSN 2185-2847
Volume 9, Number 2, pages 171–187, July 2019

Advanced Searchable Encryption: An Adaptively Secure Keyword Search for Matrix-type Storages

Yuta Kodera
JSPS research Fellow and Graduate School of Natural Science and Engineering, Okayama University, Japan.

Minoru Kuribayashi
Graduate School of Natural Science and Engineering, Okayama University, Japan.

Takuya Kusaka
Graduate School of Natural Science and Engineering, Okayama University, Japan.

and

Yasuyuki Nogami
Graduate School of Natural Science and Engineering, Okayama University, Japan.

Received: January 31, 2019
Revised: May 1, 2019
Accepted: June 11, 2019
Communicated by Toru Nakanishi

Abstract

IoT technologies and cloud storages have been developed remarkably, and many types of data including private information have been gradually outsourced. For such a situation, new convenient functionalities such as arithmetic and a keyword search over ciphertexts are required to allow users to obtain information without leaking queried results, contents of ciphertext, and so on. Especially, searchable encryptions have been paid much attention to realize a keyword search on an encrypted domain. In addition, an architecture of searchable symmetric encryption (SSE) is a suitable and efficient solution for data outsourcing. In this paper, we focus on an SSE scheme which employs a secure index for searching a keyword. In conventional studies, it has been widely considered that the scheme searches whether a queried keyword is contained in encrypted documents or not. On the other hand, we additionally take into account the location of the queried keyword in the documents by targeting a matrix-type data format. The method enables an administrator to search personal information listed per line or column in CSV-like format data.

Keywords: searchable encryption, block cipher, hash function, matrix-type data, location-based search, discrete logarithm problem

1 Introduction

Though various kinds of data format are available in the real world, especially, matrix-type data formats, such as CSV, form a class of convenient format which enables to organize the information

according to the uses' intention. Since CSV provides the sorting facility in each item of the row and columns, the data format is considered to manage several sensitive medical records, such as the symptom of a patient, medicine prescribed for a patient and so on. In the case, the information might be stored as a sorted form, according to the item of row and column, and which can be a daily record by arranging information per day.

However, in proportion to the development of IoT devices, even medical devices equip such a low-performance device inside of them for controlling and communicating with each other. It also brings a threat for an attack from a malicious one on the patient's information. Considering the situation, it is not an appropriate choice for storing and managing of sensitive records without any prominent security considerations.

On the other hand, such daily information will constantly increase, and as a result, the administrator needs to consider accommodation of the data in some virtualized storage such as in cloud storage. However, it makes the administrator impossible to search something except downloading every document over the storage, if the data are encrypted by a standard encryption scheme before uploading to those storages. In addition, it involves the risk of leaking some information that has to be hidden.

Regarding the drawbacks, techniques such as search a keyword without decrypting entire ciphertext have been investigated during these decades, and an approach based on symmetric encryption is an attractive and active topic of them. However, none of such searchable schemes, especially for symmetric encryption based system, focus on the multi-dimensional data format. In other words, it faces difficulty when searching a keyword whose location also involves a meaning.

Therefore, this paper proposes a searchable encryption scheme for secure outsourcing which can consider the location of the information by concentrating on the two-dimensional format: that is matrix-type data.

1.1 Related searchable techniques

An encrypted database management system (DBMS) is designed to avoid information leakage throughout a search such as keywords and documents to an adversary including the server itself. Most of the structure of DBMSs is realized by combining several high functional encryptions and searchable techniques and carries out operations without recovering any ciphertexts on a server.

For example, CryptDB [1] is the first practical DBMS that equips the ability to process an encrypted query so as not to appear plaintexts on the DBMS server. It is realized by using a proxy server that handles a query from a client to convert it to the proper encrypted query for carrying out the queried process. The encryption scheme, which is called *onion*, consisted of four kinds of layers which were encrypted by several functional encryptions, and the proxy server decrypts (or encrypts further) a query to the encrypted one in the appropriate layer of the onion.

Tu et al. proposed Monomi [2] which is known as an efficient encrypted DBMS. However, since the encrypted DBMSs employ functional encryptions which are based on public key encryptions, such as ElGamal encryption [3] and Paillier cryptosystem [4], which means that they require expensive computational costs for handling a query. Thus, it is not a suitable choice to use these DBMSs for big data.

To expand the applicability of DBMSs for big data analysis, an encrypted DBMS, which is called Seabed, is proposed in [5] by using an additive symmetric homomorphic encryption. Moreover, BlindSeer [6] can provide a rich query set with a few pieces of information leakages and it is designed to be capable of scaling to tens of tera bytes' of the data. As real applications, Always Encryted, Skyhigh Network and Encrypted BigQuery are known as industrial DBMSs.

As a platform for building web applications, Mylar, which was proposed by Popa et al. in [7], has also been paid much attention as a pioneer of multi-key searchable encryption (MKSE) protocols. It stimulates researchers to focus on the problems of keyword search over a group of shared documents from the same user in multi-user applications. The related works have become an active research topic in a short period of time.

Though searchable techniques on an encrypted domain have been paid much attention as a fundamental technique for constructing such DBMSs, searchable encryption schemes are classified

by focusing on information to be concealed from an adversary. Oblivious random access machine (ORAM) [8] is a secure searchable system that mainly concentrates on hiding the access pattern of a client to encrypted documents stored in a server. It is realized by reallocating the address of encrypted documents in the server whenever the client access to a document. However, the overhead on the access bandwidth cost is one of drawbacks of the system. One of the research directions in the ORAM is how to make the overhead small. The path ORAM (PORAM), which is proposed by Stefanov et al. in [9] as a lightweight ORAM protocol, is known as one of the most efficient and practical ORAM systems, and implementation on PORAM is so-called PHANTOM [10], which achieves an efficient DBMS. Owing to the result, a new research trend is arisen to reduce a large server occupancy without losing the efficient accessibility such as succinct ORAM proposed by Onodera and Shibuya in [11]. Furthermore, researches on ORAM are an inseparable part of oblivious transfer protocol which is a cryptographic primitive to conceal what pieces of information has been transferred. Though the scope of the research is different from this paper, Cho et al. considers the location of message throughout their research in [12].

On the other hand, searchable encryption (SE) enables a client to publish various queries for searching on an encrypted domain and to outsource or share private data securely with a little sacrifice of small information leakage. The SE schemes can be roughly classified into searchable symmetric encryption (SSE) and public key encryption with keyword search (PEKS). There are several related techniques such as predicate encryption (PE) [13,14], inner product encryption (IPE) [15], anonymous identity-based encryption (AIBE) [16–18], and hidden-vector encryption (HVE) [19]. However, the major objective of the PE, IPE, AIBE, and HVE is access control, which is differ from our targeting SEs. We are interested in the SE systems that can consider the location of information.

Bösch et al. distinguish SEs based on the number of allowable client access in [20] as follows:

1. Single-writer / Single-reader (S/S),
2. Single-writer / Multi-reader (S/M),
3. Multi-writer / Single-reader (M/S),
4. Multi-writer / Multi-reader (M/M).

Most of the oriented SSEs are classified into S/S, which is a suitable architecture for outsourcing data. However, some of recent researches aim to expand SSE to be a multi-user model by employing a proxy server and homomorphic encryptions [21–23]. Further details about SSEs are described in Section 2.4.

On the other hand, PEKSs are classified into S/M, M/S, or M/M, which architectures are suitable for data sharing. There are many directions on the research of PEKS. For example, some schemes [24–26] are constructed to possess resistance against for a quantum attack by founding on lattice-based cryptography. Other direction is to prohibit reusing a token (namely, a trapdoor), Sharma et al. proposed a scheme which can evaluate the freshness of the token in [27]. There also researches about the security of the scheme. For example, Huang et al. proposed a scheme that equips the security against inside keyword guessing attack [28]. In such a research, an adversary is an insider, and the problem is that the adversary may recover the keyword from a given trapdoor by exhaustively guessing the keywords offline. On another direction introduces a certificateless technology which allows us to avoid the key escrow problem in ID-based cryptography. For example, Wu et al. proposed a PEKS construction with a certificateless technique in [29].

The above techniques support MKSE protocols oriented to Mylar as fundamental techniques and state-of-the-art works such as [30,31] have been proposed. However, there does not exist any research that aims to expand the searchability to two-dimensional data format such as matrix-type storage (CSV-like format). In other words, any searchable schemes have not been considered in searching a keyword based on its location. Thus, this paper considers a straightforward way to search a keyword that can consider the location of information in the first construction. Then, based on the idea of the first construction, the authors propose a scheme that allows a client to search a keyword over encrypted matrix-type storage without leaking any information except unavoidable leakages.

1.2 Contribution of this work

In this paper, the authors propose a location-based searchable symmetric encryption scheme which allows a client to search a keyword with considering the location in matrix-type format data. It differs from the former SSE schemes in the sense that the proposed scheme handles two-dimensional data. This paper proposes the first SSE scheme that breaks the restriction of search underlying in the conventional methods.

Especially, the uniqueness of the proposed construction can be found by comparing it with the conventional methods by focusing on the trapdoor function. Firstly, most of the conventional schemes build a trapdoor per keyword by combining several cryptographic tools such as a secure hash function and a pseudorandom permutation. However, the authors design the proposed trapdoor function such that a single trapdoor of a keyword is to be a coefficient of a randomly decided degree of a special polynomial. More precisely, it uses a multiplicative cyclic property of a generator element in a field and an irreducible all-one polynomial (AOP).

Though two methods are introduced in this paper, the first method aims to build an index as a simple extension of the conventional scheme, where the secure index can handle the location of a keyword by allowing the leakage of the location of a keyword. Then, the second method, which is the location-based SSE scheme that also can hide the location of a keyword, is constructed based on the concept of the first scheme.

The rest of this paper is organized as follows. Section 2 reviews background information on pseudorandom permutation (PRP), hash function, and SE. Two proposed methods are introduced in Section 3, and the security and consideration are given in Section 4. Section 5 concludes this work and discusses future works.

2 Preliminaries

In the following, notations which are used throughout this paper are explained, and the basic idea of an index-based searchable encryption scheme is introduced in this section.

2.1 Notations

For a set S, let $|S|$ represent the number of elements in S. A vector is represented by a bold face such as $\boldsymbol{v}$. The concatenation of two strings s_1 and s_2 is written as $s_1||s_2$. A combination, which is a selection of m items from a collection whose elements are n, is denoted by $\binom{n}{m}$. Furthermore, let $poly(k)$ and $negl(k)$ denote unspecified polynomial and negligible functions in k, respectively, where k is referred to the security parameter.

We represent the set of documents which include the keyword w by $D(w)$, and $\delta(d)$ indicates the list of keywords that exist in d. The collection of m distinct keywords is denoted by Δ, that is $\Delta = (w_1, w_2, \ldots, w_m)$. For Δ, let 2^{Δ} be the set of all possible documents with words in Δ and let $\boldsymbol{d} \subset 2^{\Delta}$ be a collection of $n = poly(k)$ documents $\boldsymbol{d} = (d_1, d_2, \ldots, d_n)$ and let $ID(d_j)$ be the identifier of a document d_j. In addition, since this paper especially focuses on a matrix-type data format, it is explicitly denoted by D_j when the authors handles matrix-type data as a target document, and whose (u, v) entry is denoted by $w_{u,v}$. On the other hand, w and d implicitly denote an arbitrary document and keyword, respectively. In this paper, w is assumed to be a combination of characters, and the l-th character of w is denoted by $w[l]$, where $1 \leq l \leq len(w)$ and $len(w)$ denotes the length of w.

The system considered in this paper is assumed with the client-server model and a symmetric cryptosystem, which is employed in the first SSE scheme, is a block cipher of block size N_b. A key is denoted by K_{str}, which is the abbreviation of "key" with a subscription "str". The ciphertext of a document d is denoted by c and the collection of ciphertexts is represented by $\boldsymbol{c} = (c_1, c_2, \ldots, c_n)$. In addition, the client uses a trapdoor t or a collection of n trapdoor denoted by $\boldsymbol{t} = (t_1, t_2, \ldots, t_n)$. Throughout this paper, it is assumed that the server always returns correct data, that means the data stored on the server would not be modified by anyone. Finally, for the security discussion, let $\mathcal{A}$ and $\mathcal{S}$ be an algorithm and a simulator, respectively.

2.2 Pseudorandom permutation

A pseudorandom permutation (PRP) [32] is a function which behaves as if the function shuffles a sequence uniformly and randomly. It is an essential cryptographic primitive and defined as follows:

Definition 1 (Pseudorandom permutation) Let $f, f^{-1} : \{0,1\}^k \times \{0,1\}^{N_b}$ be deterministic and efficiently computable function that satisfies

$$f^{-1}(K_{\mathrm{PRP}}, f(K_{\mathrm{PRP}}, \boldsymbol{x})) = \boldsymbol{x} \tag{1}$$

for all keys $K_{PRP} \in \{0,1\}^k$ and $\boldsymbol{x} \in \{0,1\}^{N_b}$.

It is also called block cipher and in what follows, this paper indicates this as a block cipher.

2.3 Hash function

A hash function is a one-way function which maps data of arbitrary size to data of the fixed size. It is useful for security applications to verify the correctness of the two data. For example, Keccak [33], which is the winner of the SHA-3 cryptographic hash algorithm competition, is well-known and used for real applications. A secure hash function is an essential factor of secure systems such as a digital signification and searchable encryption and such a hash function is especially called a cryptographic hash function. Such a cryptographic hash function is required to satisfy some properties as follows:

1. The length of the output of a hash function is always fixed.
2. The algorithm of a hash function must be deterministic, that is the function needs to have reproducibility for the same input.
3. A hash function has to generate a hash value efficiently to be employed by real-time applications.

The security of a hash function is ensured by the one-wayness (OW), the second preimage resistance (2ndPR) and the collision resistance (CR). The OW is the difficulty of predicting the input from output, that is the function should not be invertible. Next, let m_1 and m_2 be distinct messages and let $H(\cdot)$ denote a hash function. Then, the 2ndPR and the CR are the difficulty of finding a pair of m_1 and m_2 such that $H(m_1) = H(m_2)$. In the former case, an attacker can know the original message m_1, and trying to find a message m_2. On the other hand, the latter allows an attacker to select a pair of messages freely and the attacker detects a collision pair (m_1, m_2).

2.4 Searchable symmetric encryption

SSE is a cryptographic primitive which is mainly classified into S/S architecture. A main stream of the approach of SSEs is building an inverted index whose entry is composed of pairs of a trapdoor and the corresponding document identifiers for a keyword. It is introduced by Curtmola et al. in [34] and many state-of-the-art works adopt the approach owing to its efficiency. Thus, the authors introduce the concept of the scheme proposed by Curtmola et al. in this paper. However, to obtain further knowledge of SSE, the readers can refer to [35] which shows several mechanisms of an index and the state-of-the-art works that this paper could not cover. Such a typical SSE scheme realize a search as illustrated in figreffig: see abst and is carried out in the following steps. Firstly, a user scans a collection of documents to generate keywords and builds an index with utilizing trapdoors. Then, the user encrypts documents with a symmetric cryptosystem and uploads the index and encrypted documents to a server. Finally, the user can search a keyword which has already registered in the index. The detailed mechanism of Curtmola's SSE scheme is drawn in Figure 2.

On the security analysis of an SSE, security definitions similar to the indistinguishable chosen plaintext attack (IND-CPA) have been proposed. Most of the recent works adopt the formal SSE definitions proposed in [34] which guarantee the secrecy of trapdoor and information about a keyword. In addition, Kurosawa and Ohtaki [36] introduced the universal composability (UC). It is a

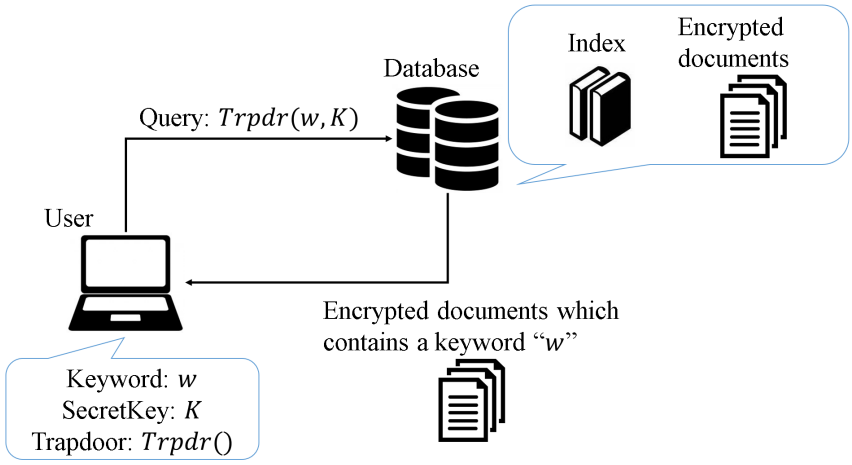

Figure 1: Illustration of a typical index-based SSE scheme

Gen(1^k)**:**
Sample K_1 from $\{0,1\}^k$ and generate a secret-key K_2 for a block cipher. Output $\boldsymbol{K} = (K_1, K_2)$.

Enc$_{\boldsymbol{K}}(\boldsymbol{d})$**:**

Initialization:
Generate the list of distinct keywords Δ. For every keyword $w \in \Delta$, generate the list of identifiers $D(w)$, each of which indicates the document d that contains w.

Building the index I:

1. Let $d_{i,j}$ denote the j-th document of $D(w_i)$ for $1 \leq i \leq |\Delta|$ and $1 \leq j \leq |\delta(D(w_i))|$. Set $I[\phi_{\boldsymbol{K}}(w_i||j)] = ID(d_{i,j})$, where $\phi_{\boldsymbol{K}}$ denotes a pseudorandom permutation.
2. Let $s' = \sum_{w_i \in \Delta} |D(w_i)|$.
3. Let $\max_\delta$ be the maximum number of distinct keywords and let $s = \max_\delta \times |\boldsymbol{d}|$. If $s' < s$, then set values for the remaining $(s - s')$ entries in I such that every $ID(d_j)$ appears the same times.

Preparing the output
Encrypt $\boldsymbol{d}$ and output the index and ciphertexts $\boldsymbol{c}$.

Trpdr$_{\boldsymbol{K}}(w)$**:**
Generate and output the collection of trapdoor $\boldsymbol{t} = (t_1, \ldots, t_n)$. It is noted that $\boldsymbol{t}$ is given by $\boldsymbol{t} = (\phi_{\boldsymbol{K}}(w||1), \ldots, \phi_{\boldsymbol{K}}(w||n))$.

Search$(I, \boldsymbol{t})$**:**
Compare the trapdoor $\boldsymbol{t}$ with the index I and return the ciphertexts $\boldsymbol{c} = (c_1, c_2, \ldots)$ which is inferred to involve the keyword.

Dec$_{\boldsymbol{K}}(c_j)$**:** Decrypt cihpertexts c_j.

Figure 2: The detail of Curtmola's adaptively secure index-based SSE

general-purpose model that claims the protocols remain secure even if they are arbitrarily composed with other instances of the same or other protocols. A server which holds encrypted data is regarded

as an untrusted third party. Under such an assumption, the information leakage must be kept as small as possible with respect to keywords and documents. There is some unavoidable leakage such as history, access pattern and search pattern. For a document set $\boldsymbol{d}$ and a keyword w, they are defined as follows:

Definition 2 (History) A q-query history over $\boldsymbol{d}$ denoted by $\boldsymbol{h} = (\boldsymbol{d}, \boldsymbol{w})$ is a tuple of documents $\boldsymbol{d}$ and q keywords $\boldsymbol{w} = (w_1, w_2, \ldots, w_q)$, which is obtained throughout q times communication between a user and a server.

Definition 3 (Access pattern) The access pattern $Access(\boldsymbol{h})$ induced by $\boldsymbol{h}$ is the result of each search via a trapdoor. It is the collection of documents that contains a keyword w_i $(1 \leq i \leq q)$.

Definition 4 (Search pattern) The search pattern $Search(\boldsymbol{h})$ induced by $\boldsymbol{h}$ is given by a symmetric binary matrix σ whose (i, j)-element is represented by

$$\sigma_{ij} = \begin{cases} 1 & \text{if } w_i = w_j, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Based on these information, Curtmola et al. defined a trace induced by a q-query history as follows:

Definition 5 (Trace) The Trace denoted by $Trace(\boldsymbol{h})$ consisted of lengths of documents in D, the access pattern $Access(\boldsymbol{h})$, and the search pattern $Search(\boldsymbol{h})$.

Curtmola et al. shows that $Trace(\boldsymbol{h})$ is the minimal information leakage [34], and thus $Trace(\boldsymbol{h})$ becomes one of the benchmarks to evaluate the security of an SSE scheme.

This paper adopts real-ideal simulation paradigm introduced in [34], which intuitively asks an adversary to guess which game (real or ideal) plays. It is defined as follows:

Definition 6 (Adaptive semantic security) Let $SSE = (Gen, Enc, Trpdr, KWSearch, Dec), \mathcal{A} = (\mathcal{A}_0, \ldots, \mathcal{A}_q)$ and $\mathcal{S} = (\mathcal{S}_0, \ldots, \mathcal{S}_q)$ be an index-based SSE scheme, an adversary such that $q \in \mathbb{N}$ and a simulator, respectively. For the security parameter $k \in \mathbb{N}$, consider the following probabilistic experiments $\mathbf{Real}_{SSE,\mathcal{A}}(k)$ and $\mathbf{Sim}_{SSE,\mathcal{A}}(k)$, where $st_{\mathcal{A}}$ and $st_{\mathcal{S}}$ are strings that capture the state of $\mathcal{A}$ and $\mathcal{S}$, respectively.

$\mathbf{Real}_{SSE,\mathcal{A}}(k)$:

$K \leftarrow Gen(1^k)$
$(\boldsymbol{d}, st_{\mathcal{A}}) \leftarrow \mathcal{A}_0(1^k)$
$(I, \boldsymbol{c}) \leftarrow Enc(\boldsymbol{d})$
$(w_1, st_{\mathcal{A}}) \leftarrow \mathcal{A}_1(st_{\mathcal{A}}, I, \boldsymbol{c})$
$t_1 \leftarrow Trpdr(w_1)$
for $2 \leq i \leq q$,
$(w_i, st_{\mathcal{A}}) \leftarrow \mathcal{A}_i(st_{\mathcal{A}}, I, \boldsymbol{c}, t_1, \ldots, t_{i-1})$
$t_i \leftarrow Trpdr(w_i)$
let $\boldsymbol{t} = (t_1, \ldots, t_q)$
output $\nu = (I, \boldsymbol{c}, \boldsymbol{t})$ and $st_{\mathcal{A}}$

$\mathbf{Sim}_{SSE,\mathcal{A},\mathcal{S}}(k)$:

$(\boldsymbol{d}, st_{\mathcal{A}}) \leftarrow \mathcal{A}_0(1^k)$
$(I, \boldsymbol{c}, st_{\mathcal{S}}) \leftarrow \mathcal{S}_0(Trace(\boldsymbol{d}))$
$(w_1, st_{\mathcal{A}}) \leftarrow \mathcal{A}_1(st_{\mathcal{A}}, I, \boldsymbol{c})$
$(t_1, st_{\mathcal{S}}) \leftarrow \mathcal{S}_1(st_{\mathcal{S}}, Trace(\boldsymbol{d}, \boldsymbol{w}_1))$
for $2 \leq i \leq q$,
$(w_i, st_{\mathcal{A}}) \leftarrow \mathcal{A}_i(st_{\mathcal{A}}, I, \boldsymbol{c}, t_1, \ldots, t_{i-1})$
$(t_i, st_{\mathcal{S}}) \leftarrow \mathcal{S}_i(st_{\mathcal{S}}, Trace(\boldsymbol{d}, w_1, \ldots, w_i))$
let $\boldsymbol{t} = (t_1, \ldots, t_q)$
output $\nu = (I, \boldsymbol{c}, \boldsymbol{t})$ and $st_{\mathcal{A}}$

Let $\mathcal{D}(\nu, st_{\mathcal{A}})$ be a distinguisher which takes input as the output from the experiments and returns 1 if it succeeds to guess and 0 otherwise. The SSE scheme is adaptively semantically secure if for

all polynomial-size adversaries $\mathcal{A} = (\mathcal{A}_0, \ldots, \mathcal{A}_q)$ such that $q = poly(k)$, there exists a non-uniform polynomial-size simulator $\mathcal{S} = (\mathcal{S}_0, \ldots, \mathcal{S}_q)$ such that for all polynomial-size $\mathcal{D}$,

$$|\Pr[\mathcal{D}(\nu, st_{\mathcal{A}}) = 1 : (\nu, st_{\mathcal{A}}) \leftarrow \mathbf{Real}_{SSE,\mathcal{A}}(k)] - \Pr[\mathcal{D}(\nu, st_{\mathcal{A}}) = 1 : (\nu, st_{\mathcal{A}}) \leftarrow \mathbf{Sim}_{SSE,\mathcal{A},\mathcal{S}}(k)]| \leq negl(k),$$

where the probabilities are over the coins of Gen and Enc.

It is noted that a history is assumed to be non-singular which is defined as follows:

1. There exists at least one history $\boldsymbol{h}' \neq \boldsymbol{h}$ such that $Trace(\boldsymbol{h}') = Trace(\boldsymbol{h})$.
2. Such a history can be found within polynomial-time from given $Trace(\boldsymbol{h})$.

3 A keyword search with the location of data

External storages such as cloud are widely used in current ICT society to outsource or share a massive amount of data even if it contains private information. There arises a strong demand for the privacy sensitive data to search without sacrificing the secrecy of plaintexts against an adversary. Especially, information with respect to an individual person is often listed up as the matrix-type format such that some elements of each person have appeared at each row in the matrix. We focus on the data format, and keyword search methods with a restriction on the position of information are proposed here.

3.1 Basic concept of the location-based SSE scheme

Firstly, let us begin with constructing a simple SSE scheme with allowing several information leakages to make the concept of the location-based SSE clear. Hereafter the authors handle matrix-type data D and a keyword which locates in (u, v) entry of D is explicitly denoted by $w_{u,v}$. On the other hand, $\boldsymbol{d}$ and w_i are used for simply indicating documents including D and a keyword including $w_{u,v}$, respectively, when the authors do not distinguish d and D.

3.1.1 Definition

The scheme consists of the following polynomial-time functions.

Definition 7 (Generation) $\mathrm{Gen}(1^k)$ generates a pair of secret-keys of a block cipher K_{sym} and K_{init}, and make a list of keywords Δ by scanning a collection of documents $\boldsymbol{d}$, where k denotes a security parameter. At the same time, identifiers denoted by $ID(d_j)$ are attached to each document. The function is run by a user.

Definition 8 (Encryption) $\mathrm{Enc}_{K_{sym}}(\boldsymbol{d})$ is a deterministic algorithm run by a user with taking a collection of documents $\boldsymbol{d}$ as its input. This function creates a secure index for searching and the encryption is carried out by a block cipher of block size N_b with using the secret-key K_{sym}.

Definition 9 (Trapdoor) $\mathrm{Trpdr}(w)$ is a deterministic algorithm run by a user. It takes a keyword w as its input and returns a collection of trapdoors t created from the keyword.

Definition 10 (Keyword search) $\mathrm{KWSearch}(t)$ takes a trapdoor t as the input and searching data on a server with an index. Then, the function returns true or false. In addition, it returns the correction of encrypted documents which contain the keyword if and only if the user requests to send back them.

Definition 11 (Decryption) $\mathrm{Dec}_{K_{sym}}(\boldsymbol{c})$ decrypts ciphertexts encrypted by the block cipher of a secret-key K_{sym}.

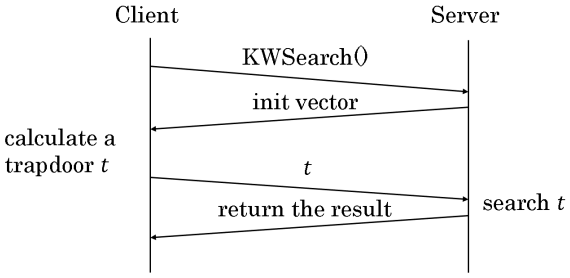

Figure 3: Illustration of the proposed scheme

Table 1: Example of target data for searching

ID	Age	Gender	Blood type
A	23	Men	AB
B	31	Women	B
C	19	Men	O
D	41	Men	A
E	47	Women	B
⋮	⋮	⋮	⋮

The detailed mechanism of the trapdoor function and keyword search scheme is introduced in this section. Figure 3 illustrates the brief flow of the system. The data format treated in this paper is a matrix-type in which each element is arranged at each row and column such as CSV format data.

3.1.2 Encryption and index for searching

An encryption function $\mathrm{Enc}_{K_{sym}}(\cdot)$ creates an index table which associates a keyword w with the collection of documents $D(w)$ and encrypts the documents. The index table has a one-to-one correspondence between the keyword w and the identifier of a document, which is referred to as *normal-index* throughout this paper. At the same time, the function extracts a certain row (column) where the target data is stored in and expands each element of the row (column) to N_b via a hash function $H_1(\cdot)$. Then, every block is encrypted by the block cipher and the ciphertexts are stored in a server. The index created by the above procedure is especially called *CBC-index* in this paper. It is noted that the data is associated with the document identifiers and the encryption is carried out with the cipher block chaining mode. Therefore, K_{init} is handled as the initialization vector of the first encryption for extracted elements.

For simplicity, let us assume the data with CSV format as shown in Table 1 and consider searching the blood type of a specific identifier. Then, only the column with respect to the blood type is picked up, and each element of the column is stored. It is noted that the block cipher used in the system takes XOR operation to the input and the previous ciphertext block before the encryption. The above procedures are illustrated in Figure 4. It is noted that $H_1(\boldsymbol{x})$ denotes an ideal cryptographic hash function which maps $\boldsymbol{x}$ to N_b-bit string and possesses OW, 2ndPR, and CR properties. The hash function must not allow an adversary to obtain a keyword w from the hash value $H_1(w)$. In addition, $H_1(w_{u_1,v_1}) = H_1(w_{u_1,v_1})$ and $H_1(w_{u_1,v_1}) \neq H_1(w_{u_2,v_2})$ are an essential condition for an arbitrary distinct keyword pair $(w_{u_1,v_1}, w_{u_2,v_2})$ in the same row (column).

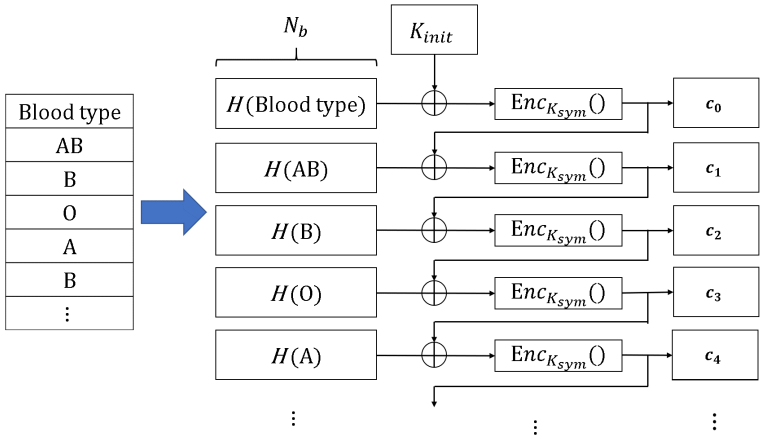

Figure 4: Mechanism of the proposed method

3.1.3 Procedure for searching a keyword

This section introduces how to find a target keyword located in the indicated position. Firstly, the normal-index is for searching a keyword without its location. This functionality allows a user to confirm whether a keyword w is contained in the collection of documents $\boldsymbol{d}$ or not. The above search method is called *normal search* in this paper. Secondly, CBC-index is a set of a ciphertext which is specialized to search a keyword $w_{u,v}$ in a matrix-type data D based on the location. The search is referred by *location-based search* in this paper. The detailed flow of the location-based search is illustrated in Figure 5. Let us assume that a user would like to search whether a keyword $w_{u,v}$ is

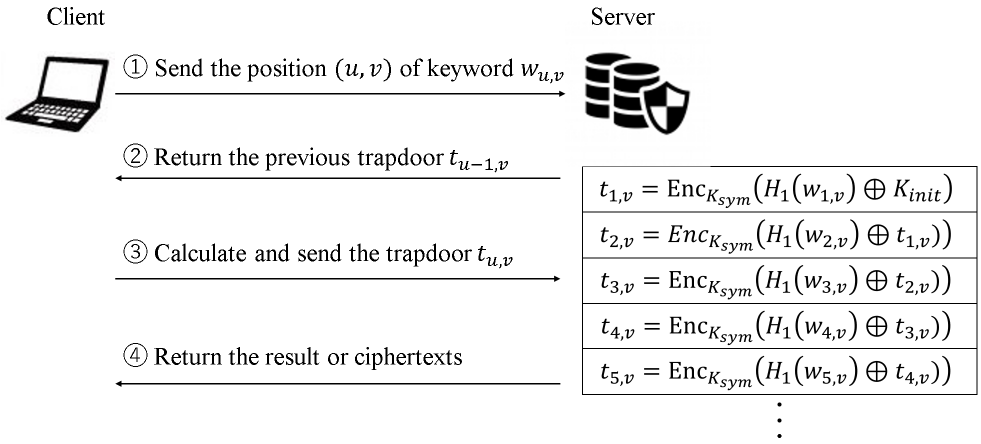

Figure 5: Detail of keyword search based on the location

located in the (u, v) entry of D. The user sends the position of a keyword, that is (u, v), then the server will return the trapdoor $t_{u-1,v}$. If $t_{u-1,v}$ truly corresponds to $\mathrm{Enc}_{K_{sym}}(H_1(w_{u-1,v}) \oplus t_{i-2,v})$, then the user can calculate the correct trapdoor of the target keyword $w_{u,v}$ as $t_{u,v} = \mathrm{Enc}_{K_{sym}}(H_1(w_{u,v}) \oplus t_{u-1,v})$. After the server receives the trapdoor $t_{u,v}$, it begins to compare the trapdoor in the server with the received $t_{u,v}$. If the server finds the same trapdoor in the stored data, then it returns true. Otherwise, it returns false to the user. It is noted that the encrypted documents which contain the keyword are turned back to the user if and only if the user requests to send back them.

Let us consider the above procedure with the previous example. Here, let us search the blood type of ID B. For example, the administrator in the medical field knows the name of a patient and that of some corresponding item the one would like to search. Thus, it is natural that we assume the identifiers and items are managed by associating with the number of rows and columns, respectively, in a matrix-type document used in such medical and financial fields. For simplicity, we discuss with numeric numbers to indicate a row and a column, and the location u and v are larger than 1, respectively. It is considered that the assumption is quite natural even for practical use because the identifiers are distributed by the user and it is uniquely given to each person. Since ID B locates in the third row in Table 1, a client firstly sends the position $(u, v) = (2, 3)$ of a keyword $w_{u,v}$ to obtain the previous trapdoor. Then, the server receives $v = 3$ and refers to and returns the trapdoor located in the position $(u - 1, v)$. Thirdly, the user calculates $t_{2,3} = \mathrm{Enc}_{K_{sym}}(H_1(\boldsymbol{x}) \oplus t_{1,3})$ by using received trapdoor $t_{1,3}$, where $\boldsymbol{x}$ denotes a blood type indicated by the user. Finally, the server returns the queried result as true if $t_{2,3}$ is found on the server. Otherwise, the user understands that the keyword is not contained in the row of the data and he may search another keyword again. In this way, the construction allows a user to search for information located in a specific position by building entries of an index for each keyword with location.

3.2 Adaptively secure location-based SSE

The CBC-index allows us to search whether a keyword locates on u-th row and v-th column. However, since it uses a piece of raw information about the locations of a keyword and chains the previous trapdoor, adversaries can obtain the location and an initialization vector for calculating the target trapdoor. There is a threat that a set of several queries and the results may enable adversaries to infer a relationship between trapdoors. In this sense, the scheme with CBC-index is not suitable for a system that handles several search queries. Therefore, this section proposes the second construction of an index to realize a location-based SSE which achieves to conceal until the location of a keyword.

3.2.1 Mechanism of the index

The scheme uses cyclotomic polynomial over $\mathbb{F}_p$. Before referring to the detail of construction, we introduce mathematical conditions here. Let $\Phi_{p_2}(x)$ be a cyclotomic polynomial and let p_1, p_2 be distinct primes such that $\gcd(p_1, p_2) = 1$ and $p_1 \pmod{p_2}$ becomes a generator in $\mathbb{F}_{p_2}$ for all $p_1 > p_2$. Then, $\Phi_{p_2}(x)$ is an irreducible polynomial over $\mathbb{F}_{p_1}$ and is given by AOP as follows:

$$\Phi_{p_2}(x) = \sum_{j=0}^{p_2-1} x^j. \tag{3}$$

Let g_1 be a generator in $\mathbb{F}_{p_1}$. The scheme uses a series of polynomials $\Phi_{p_2}(g^i x)$, where $i (1 \leq i < p_1 - 1)$. Since g_1 is a generator in $\mathbb{F}_{p_1}$, polynomials $\Phi_{p_2}(g_1^i x)$ are distinct and each coefficient of the certain $x^j (j > 0)$ forms a cyclic group under the modulus p_1. For example, let $p_1 = 7, p_2 = 5$ and $g_1 = 3$, then the following polynomials hold:

$$\begin{aligned}
\Phi_{p_2}(3^1 x) &= 4x^4 + 6x^3 + 2x^2 + 3x + 1, \\
\Phi_{p_2}(3^2 x) &= 2x^4 + 1x^3 + 4x^2 + 2x + 1, \\
\Phi_{p_2}(3^3 x) &= 1x^4 + 6x^3 + 1x^2 + 6x + 1, \\
\Phi_{p_2}(3^4 x) &= 4x^4 + 1x^3 + 2x^2 + 4x + 1, \\
\Phi_{p_2}(3^5 x) &= 2x^4 + 6x^3 + 4x^2 + 5x + 1,
\end{aligned}$$

Since the coefficients of polynomials $\Phi_{p_2}(g_1^i x)$ can be considered as entries of a matrix, it is useful to associate with the location of a keyword and the keyword itself. For convenience, let us replace variables (i, j) by (u, v) and let u, v are indicators of row and column, respectively. It is noted

that the above replacement implicitly means that p_1 and p_2 show the allowable maximum row and column in this scheme. Therefore, they have to be large at least they can cover the maximum row and column size of matrix-type data. Assume that every character which consists words is uniquely associated with a non-zero number (e.g. ASCII code) less than p_1. Let $H_2(x)$ be a secure hash function of the bit length N and a trapdoor is generated by a client in the following steps:

1. Let $w_{u,v}$ be a keyword located in the (u, v) cell. Calculate the hash value $H_2(w_{u,v}||u||v)$, and split it into the bit length m, where m is derived by $m = \lceil \log_2(p_1) \rceil$. It denotes the allowable maximum bit size for a trapdoor. It is noted that the bit length of the hash value need to be longer than m.

2. Calculate the polynomial $\Phi_{p_2}(g_1^u x)$ and multiply $w_{u,v}[0]$ to the $(p_2 - 1 - v)$-th coefficient of $\Phi_{p_2}(g_1^u x)$, where $w_{u,v}[0]$ denotes the character at the beginning of a keyword located in (u, v) cell.

3. Convert coefficients of the polynomial calculated in the first step to a binary sequence. Note that every coefficient is transformed into a binary representation with m-bit padding.

4. Embed the first m-bits obtained from the hash value of $w_{u,v}$ into the v-th coefficient of the polynomial throughout XOR operation.

5. Concatenate binary sequences generated in the previous step with uniform shuffling algorithm (e.g pseudorandom permutation, Fisher Yates Shuffle [37] and so on), and output it as the trapdoor of u-th row elements.

Let l be a control parameter determined by a client. Since the coefficients in small degrees are not frequently shuffled even if a client randomly chooses a generator in $\mathbb{F}_{p_1}$, the coefficients of degree less than l is used for a dummy entry. In addition, l also has a role to adjust so that m becomes a factor of $p_2 - l$, and the remaining part can be ignored in this method. An illustration of the trapdoor is shown in Figure 6.

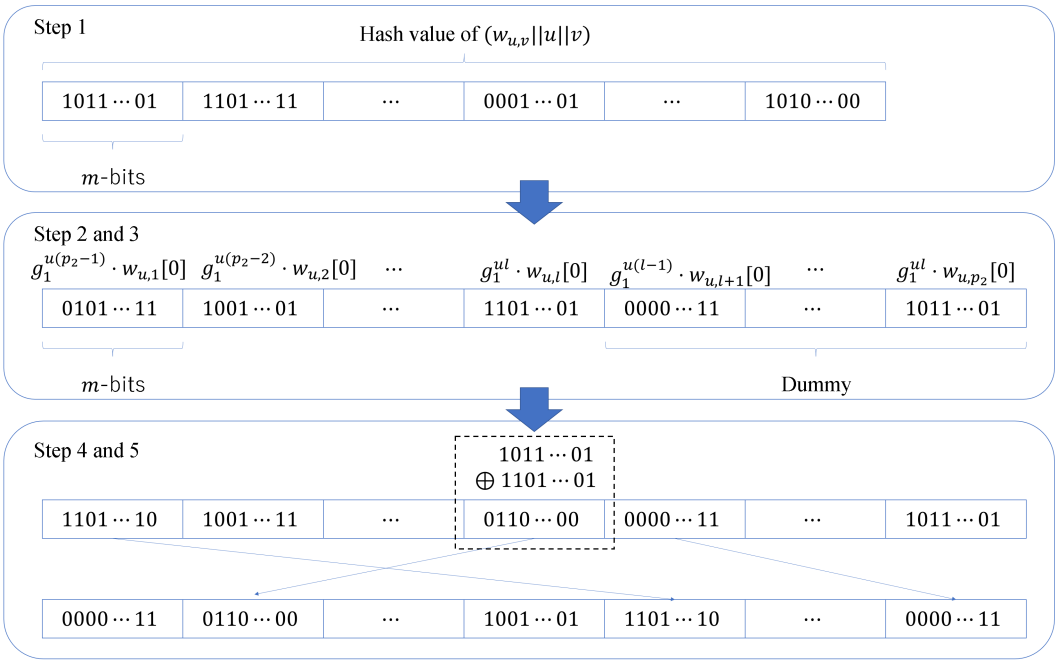

Figure 6: Illustration of the generating procedure of trapdoors

To mix the rows of trapdoors generated in the above procedure, a method for shuffling trapdoors is required. Let $g_2 \in \mathbb{F}_{p_1}$ be a different generator from g_1, and we propose to store the u-th trapdoor by sorting based on the ascending order of $g_2^u \pmod{p_1}$. Namely, u-th trapdoor is stored in the g_2^u-th address of a memory.

3.2.2 Query and search

To search a keyword located in (u, v) cell in the original matrix from the document is discussed in this section. Recall that the client has parameters p_1, p_2, g_1, g_2 and a keyword $w_{u,v}$, and assume that the client knows which cell is the target cell that may contain $w_{u,v}$. It is natural in the target document format such as medical data, for example, data in (u, v) cell is managed by an identifier and an item associated with u and v.

A query is composed of two binary sequences of length m and an integer. Under the setting, the client prepares a query as follows:

1. Calculate $g_1^{u(p_2-1-v)} \cdot w_{u,v}[0] \pmod{p_1}$ and convert it to a binary sequence with m-bit padding.
2. Calculate the hash value $H_2(w_{u,v}||u||v)$ and extract the first m-bits from the hash value.
3. Calculate g_2^u to indicate the address of a storage.
4. Output the set of m-bits sequences together with g_2^u as a query $\boldsymbol{q}$.

Let t be a trapdoor stored in g_2^u-th address. Let $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ be m-bits sequences derived from the coefficient of the polynomial and the hash value, respectively. The server searches $\boldsymbol{s}_2$ by removing $\boldsymbol{s}_1$ from t with XOR operation and returns the search result to the client (e.g. document identifiers, and the location of corresponding m-bits found in the target trapdoor to check the correctness of the column). Since a trapdoor is composed of the concatenation of binary sequences of length m, the order of a search is $\mathcal{O}(n)$.

4 Security and consideration

4.1 Security of the location-based SSE

In this section, the authors discuss the security of the proposed location-based SSE (the second construction) based on the real-ideal simulation paradigm. Assume that the hash function satisfies the requirements as a secure hash function, and the symmetric encryption is PCPA-secure, which means the encryption scheme has pseudorandomness against chosen plaintext attacks. In addition, assume that distinct primes p_1, p_2 are sufficiently large so that an adversary can not solve the discrete logarithm problem (DLP). Throughout the discussion, the authors describe a polynomial-size simulator $\mathcal{S}$ such that the output of $\mathbf{Real}_{SSE,\mathcal{A}}(k)$ and $\mathbf{Sim}_{SSE,\mathcal{A},\mathcal{S}}(k)$ are computationally indistinguishable for all polynomial-size adversaries $\mathcal{A} = (\mathcal{A}_0, \ldots, \mathcal{A}_q)$. Consider that the simulator adaptively generates a string $\nu = (I, \boldsymbol{c}, \boldsymbol{t}) = (I, c_1, \ldots, c_n, t_1, \ldots, t_n)$ as follows:

$\mathcal{S}_0(1^k, Trace(\boldsymbol{d}))$**:** Since the size of all the documents can obtain from the trace of $\boldsymbol{d}$, it may estimate p_1, p_2 from the maximum number of rows and columns in $\boldsymbol{d}$ such that $(p_1, p_2) = 1$ and p_1 becomes a generator in $\mathbb{F}_{p_2}$. However, it is noted that the estimated p_1 and p_2 are not always the correct prime integers since they are not public information. It then sets I to be a lookup table with inserted document identifiers at random locations. Then, $\mathcal{S}_0$ includes I in $st_{\mathcal{S}}$ and outputs $(I, \boldsymbol{c}, st_{\mathcal{S}})$, where c_i is sampled from $\{0,1\}^{|d_i|}$. Since $st_{\mathcal{A}}$ does not have the secret key for the symmetric encryption scheme except a negligible probability, each c_i is indistinguishable from a real ciphertext.

Recall that a trapdoor is generated by the following steps: 1) calculating $g_1^{u(p_2-1-v)} \cdot w_{u,v}[0] \pmod{p_1}$ with a secret information g_1 and binarize it with m-bit padding, 2) calculating the hash value $H_2(w_{u,v}||u||v)$ and extracting the first m-bit from the output, 3) taking XOR of each m-bit and output it. Assume that a trapdoor is distinguishable from a real trapdoor.

Then, one can distinguish the output of $H_2(w_{u,v}||u||v)$ by observing only m-bit sequence. Secondly, assume that the one can remove the masking obtained from $H_2(w_{u,v}||u||v)$ and reveal the secret information contained in $g_1^{u(p_2-1-v)} \cdot w_{u,v}[0] \pmod{p_1}$. It is noted that target information would be the input of the hash function, the location (u, v), and a character $w_{u,v}[0]$ used for the keyword.

However, since we assume that the hash function is not invertible, the discussion is omitted here. In this case, the one can distinguish a randomly chosen primitive element g_1, then the exponential part can be derived by factorizing $g_1^{u(p_2-1-v)} \cdot w_{u,v}[0] \pmod{p_1}$. In other words, such an adversary has to try to solve the DLP by selecting a generator $g \in \mathbb{F}_{p_1}$ in brute-force way. However, since there are $\binom{\varphi(p-1)}{2}$ kinds of generators in $\mathbb{F}_{p_1}$, the probability of presuming the generator would be negligible, where $\varphi(\cdot)$ is the Euler's totient function. Therefore, since g_1 is not contained in $st_{\mathcal{A}}$, I is indistinguishable from a real index. Furthermore, since $g_2 \in \mathbb{F}_{p_1}$ is also not contained in $st_{\mathcal{A}}$, the location of the trapdoor is indistinguishable from real address owing to the DLP.

$\mathcal{S}_1(1^k, Trace(\boldsymbol{d}, w_1))$**:** $\mathcal{S}_1$ randomly picks an entry from I and remember association between t_1 and w_1 by including it in $st_{\mathcal{S}}$. It then outputs $(t_1, st_{\mathcal{S}})$. Except the negligible probability, $st_{\mathcal{A}}$ does not include g_1 and g_2, t_1 is indistinguishable from a real trapdoor otherwise one can distinguish an m-bits extracting from a random string of the hash function and the representation of a scalar value by estimating the bases g_1 and g_2.

$\mathcal{S}_u(1^k, Trace(\boldsymbol{d}, w_1, \ldots, w_i))$ **for** $2 \leq i \leq q$**:** $\mathcal{S}_i$ firstly checks whether the keyword w_i appears before or not by evaluating the access pattern $\mathcal{M}$. If w_i appears previously, $\mathcal{S}_i$ uses it as t_i. On the other hand, if w_i has not appeared previously, $\mathcal{S}_i$ generates a trapdoor in the same way as $\mathcal{S}_1$ does. Then, it outputs $(st_{\mathcal{S}}, t_i)$ and clearly, t_i is indistinguishable from a real trapdoor.

Therefore, the location-based SSE is adaptively semantically secure.

4.2 Consideration

This section gives further practical security analysis and consideration for a real use. The DLP is a problem that asks to find the exponent of g^i in a finite field with a large prime number from given pair (g, g^i), where g is a generator in the field. There are several efficient methods for solving DLP such as Pohlig-Hellman algorithm [38] and Pollard Rho algorithm [39]. However, the DLP is still difficult if the characteristic p is sufficiently large.

Since the number of generators in $\mathbb{F}_p$ is given by $\varphi(p-1)$, there are $\binom{\varphi(p-1)}{2}$ kinds of candidates of generator pairs. In addition, the probability of collisions is given by $\frac{1}{2^{\frac{m}{2}}}$ according to the birthday bound [40]. Considering the current security level in RSA, it would be close to 0 since each bit size of the trapdoor of a keyword can be given by $m = \lceil \log_2(p) \rceil = 3072$ even for 128-bit security.

Considering the real uses, a form $m = 8\xi, (1 \leq \xi)$ is a reasonable choice for implementation, and a collision occurs in the probability $1/2^4 \simeq 0.0625$ at the worst case $(u = 1)$. This is because current a high-level programing language such as C can manage 64-bit operation without multiple precision arithmetic libraries. In addition, the bit size of trapdoor is calculated by $\sum_{i=1}^{|\boldsymbol{d}|} m \times N_{d_i}$, where $|\boldsymbol{d}|$ denotes the number of documents in a document set $\boldsymbol{d}$ and $d_i \in \boldsymbol{d}$.

Since this paper considers much complex search rather than the conventional methods that purpose to find whether a keyword is contained in documents or not, the index size would be large. The authors think that it is a trade-off between compactness and usability. However, there is a possibility that the trapdoor size of the proposed method becomes smaller than that of conventional schemes since a size in the proposed method is m-bits. For example, for simplicity, assume that the trapdoor of a method consisted of N-bits of hash values. Then, the trapdoor of the proposed method becomes small if $N \times |\Delta| \geq \sum_{i=1}^{|\boldsymbol{d}|} m \times N_{d_i}$ is satisfied even though the searchability becomes high.

5 Conclusion

A searchable encryption scheme with considering the location of the data in storage was proposed in this paper. This functionality targets matrix-type data, which is used to list up information for an individual person in the real world such as the medical field and financial field. This is also for overcoming the difficulty of searching a duplicated information by distinguishing the location, which is underlying in the conventional keyword search.

The proposed method firstly introduced in this paper provides a search which can consider the location of a keyword. Since the trapdoor of a keyword is calculated by using a hash function and a block cipher, a client can prepare a query efficiently. However, the scheme is not sufficiently secure because it directly indicates the location of a keyword in an interaction and use another trapdoor as an initial value for the trapdoor. Based on the first concept, the authors proposed the location-based scheme and showed that it is adaptively and semantically secure if prime numbers used in the method is sufficiently large so that an adversary cannot solve the DLP.

Considering the current computational performance, the scheme proposed in the latter can implement with a low collision probability. However, the computational cost for preparing a query becomes larger than the first method. For further compactness, the authors think that a location-based search can struct with an elliptic curve, which difficulty is called ECDLP and can achieve the same security level even though a small security parameter. In addition, a secure update, which means rebuilding the index without any leaks, is required. These improvements are left for future works.

Acknowledgment

This work was partly supported by a JSPS Research Fellowships for Young Scientists KAKENHI 19J1179411.

References

[1] R. A. Popa, C. M. S. Redfield, N. Zeldovich and H. Balakrishnan, “CryptDB: protecting confidentiality with encrypted query processing,” ACM, SOSP ’11, pp. 85-100, 2011.

[2] S. Tu, M. F. Kaashoek, S. Madden and N. Zeldovich, “Processing Analytical Queries over Encrypted Data,” VLDB Endowment, Proc. VLDB Endow., vol. 6, no. 5, pp. 289-300, 2013.

[3] A. J. Menezes, P. C. v. Oorschot and S. A. Vanstone, “Handbook of Applied Cryptography (5th ed.),” CRC Press, p. 294-298, 2001.

[4] P. Paillier, “ Paillier Encryption and Signature Schemes,” Springer, Encyclopedia of Cryptography and Security, 2005.

[5] A. Papadimitriou, R. Bhagwan, N. Chandran, R. Ramjee, A. Haeberlen, H. Singh, A. Modi and S. Badrinarayanan, “Big Data Analytics over Encrypted Datasets with Seabed,” USENIX Association, OSDI 16, pp. 587-602, 2016.

[6] V. Pappas, F. Krell, B. Vo, V. Kolesnikov, T. Malkin, S. G. Choi, W. George, A. Keromytis and S. Bellovin, “Blind Seer: A Scalable Private DBMS,” IEEE Computer Society, SP ’14, no. 16, pp. 359-374, 2014.

[7] R. A. Popa, E. Stark, S. Valdez, J. Helfer, N. Zeldovich and H. Balakrishnan, “Building Web Applications on Top of Encrypted Data Using Mylar,” USENIX Association, NSDI ’14, pp. 157-172, 2014.

[8] O. Goldreich, “Towards a theory of software protection and simulation by oblivious RAMs,” ACM, STOC ’87, pp. 182-194, 1987.

[9] E. Stefanov, M. v. Dijk, E. Shi, C. Fletcher, L. Ren, X. Yu and S. Devadas, "Path ORAM An Extremely Simple Oblivous RAM Protocol," ACM, CCS '13, pp. 299-310, 2013.

[10] M. Maas, E. Love, E. Stefanov, M. Tiwari, E. Shi, K. Asanovic, J. Kubiatowicz and D. Song, "PHANTOM: practical oblivious computation in a secure processor," ACM, CCS '13, pp. 311-324, 2013.

[11] T. Onodera and T. Shibuya, "Succinct Oblivious RAM," Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, STACS 2018, vol. 96, pp.52:1-52:16, 2018.

[12] C. Cho, N. Döttling, S. Garg, D. Gupta, P. Miao and A. Polychroniadou, "Laconic Oblivious Transfer and its Applications," Springer, CRYPTO 2017, LNCS, vol. 10402, 2017.

[13] J. Katz, A. Sahai and B. Waters, "Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products," Springer, EUROCRYPT 2008, LNCS, vol 4965, 2008.

[14] S. Xu and F. Ye, "A Predicate Encryption Based Anomaly Detection Scheme for E-Health Communications Network," IEEE, ICC 2018, pp. 1-6, 2018.

[15] S. Agrawal, B. Libert and D. Stehlé , " Fully Secure Functional Encryption for Inner Products," Springer, CRYPTO 2016, LNCS, vol. 9816, 2016.

[16] X. Boyen and B. Waters B, "Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles)," Springer, CRYPTO 2006, LNCS, vol. 4117, 2006.

[17] X. Ma, X. Wang and D. Lin, "Anonymous Identity-Based Encryption with Identity Recovery," Springer, ACISP 2018, LNCS, vol. 10946, 2018.

[18] Z. Brakerski, A. Lombardi, G. Segev and V. Vaikuntanathan, "Anonymous IBE, Leakage Resilience and Circular Security from New Assumptions," Springer, EUROCRYPT 2018, LNCS, vol. 10820, 2018.

[19] A. D. Caro, V. Iovino and G. Persiano, "Fully Secure Hidden Vector Encryption,"Springer, Pairing 2012, LNCS, vol. 7708, 2012.

[20] C. Bösch, P. Hartel, W. Jonker and A. Peter, "A Survey of Provably Secure Searchable Encryption," ACM Computing Surveys, vol. 47, no. 2, article. 18, 2014.

[21] X. Sun, T. Waang, Z. Sun, P. Wang, J. Yu and W. Xie, "An Efficient Quantum Somewhat Homomorphic Symmetric Searchable Encryption," Springer, IJTP, vol. 56, no. 4, pp.1335-1345, 2017.

[22] G. Wang, C. Liu, Y. Dong, P. Han, H. Pan and B. Fang, "IDCrypt: A Multi-User Searchable Symmetric Encryption Scheme for Cloud Applications," IEEE, IEEE Access, vol. 6, pp. 2908-2921, 2018.

[23] L. Chen and N. Zhang, "Efficient Verifiable Multi-user Searchable Symmetric Encryption for Encrypted Data in the Cloud," Springer, LNICST, vol. 197, CloudComp 2016, SPNCE 2016. 2018.

[24] C. Hou, F. Liu, H. Bai and L. Ren, "Public-Key Encryption with Keyword Search from Lattice," IEEE, 3PGCIC 2013, pp. 336-339, 2013.

[25] R. Behnia, A. A. Yavuz, M. O. Ozmen, "High-Speed High-Security Public Key Encryption with Keyword Search," Springer, DBSec 2017. LNCS, vol 10359, 2017.

[26] R. Behnia, M. O. Ozmen and A. A. Yavuz, "Lattice-Based Public Key Searchable Encryption from Experimental Perspectives," IEEE, IEEE Trans. Depend. Sec. Comput., 2018.

[27] D. Sharma and D. C. Jinwala, "Multiuser Searchable Encryption with Token Freshness Verification," Security and Communication Networks, vol. 2017, Article ID 6435138, 16 pages, 2017.

[28] Q. Huang and H. Li, "An Efficient Public-Key Searchable Encryption Scheme Secure against Inside Keyword Guessing Attacks," ELSEVIER, Information Sciences, vol. 403-404, pp. 1-14, 2017.

[29] T. Wu, C. Chen, K. Wang, C. Meng and E. K. Wang, "A provably secure certificateless public key encryption with keyword search," Taylor & Francis, JCIE, vol. 42, no. 1, pp. 20-28, 2019.

[30] A. Hamlin, A. Shelat, M. Weiss and D. Wichs, "Multi-Key Searchable Encryption, Revisited," Springer, PKC 2018, LNCS, vol. 10769, 2018.

[31] C. V. Rompay, R. Molva and M. Önen, "Fast Two-Server Multi-User Searchable Encryption with Strict Access Pattern Leakage," Springer, ICICS 2018, LNCS, 2018.

[32] J. Katz and Y. Lindell, "Introduction to Modern Cryptography (2nd ed.)," Chapman & Hall/CRC, 2014.

[33] G. Bertoni, J. Daemen, M. Peeters, G. V. Assche, "Keccak," Springer EUROCRYPT 2013. LNCS, vol. 7881, 2013.

[34] R. Curtmola, J. Garay, S. Kamara and R. Ostrovsky, "Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions," Journal of Computer Security, IOS Press, vol. 19, no. 5, pp. 895-934, 2011.

[35] G. S. Poh, J. Chin, W. Yau, K. R. Choo and M. S. Mohamad, "Searchable Symmetric Encryption: Designs and Challenges," ACM Computing Surveys, vol. 50, no. 40, 2017.

[36] K. Kurosawa and Y. Ohtaki, "UC-Secure Searchable Symmetric Encryption," Springer, FC 2012, vol. 7397, pp. 285-298, 2012.

[37] R. A. Fisher and F. Yates, "Statistical Tables for Biological, Agricultural and Medical Research," Oliver and Boyd, http://hdl.handle.net/2440/10701, 1963.

[38] S. Pohlig and M. Hellman, "An improved algorithm for computing logarithms overGF(p)and its cryptographic significance (Corresp.)," in IEEE Transactions on Inf. Theory, vol. 24, no. 1, pp. 106-110, 1978.

[39] J. M. Pollard, "Monte Carlo Methods for Index Computation Mod P," Mathematics of Computation, Vol. 32, No. 143, pp. 918-924, 1978.

[40] J. Patarin and A. Montreuil, "Benes and Butterfly Schemes Revisited," Springer, ICISC 2005, vol. 3935, pp. 92-116, 2005.