Hindawi
Security and Communication Networks
Volume 2021, Article ID 2291470, 8 pages
https://doi.org/10.1155/2021/2291470

Research Article

Secure Symmetric Keyword Search with Keyword Privacy for Cloud Storage Services

Taek-Young Youn[1] and Hyun Sook Rhee[2]

[1]*Dankook University, Yongin, Gyeonggi 16890, Republic of Korea*
[2]*Samsung Electronics Co. Ltd., Suwon-si, Gyeonggi-do, Republic of Korea*

Correspondence should be addressed to Hyun Sook Rhee; hyunsook.rhee@gmail.com

Received 1 September 2021; Revised 18 October 2021; Accepted 6 November 2021; Published 30 November 2021

Academic Editor: Ilsun You

As Internet services are widely used in various mobile devices, the amount of data produced by users steadily increases. Meanwhile, the storage capacity of the various devices is limited to cover the increasing amount of data. Therefore, the importance of Internet-connected storage that can be accessed anytime and anywhere is steadily increasing in terms of storing and utilizing a huge amount of data. To use remote storage, data to be stored need to be encrypted for privacy. The storage manager also should be granted the ability to search the data without decrypting them in response to a query. Contrary to the traditional environment, the query to Internet-connected storage is conveyed through an open channel and hence its secrecy should be guaranteed. We propose a secure symmetric keyword search scheme that provides query privacy and is tailored to the equality test on encrypted data. The proposed scheme is efficient since it is based on prime order bilinear groups. We formally prove that our construction satisfies *ciphertext confidentiality* and *keyword privacy* based on the hardness of the bilinear Diffie–Hellman (DH) assumption and the decisional 3-party DH assumption.

1. Introduction

According to the development of IT technologies including communications and computations, the use of small devices for daily human life is increasing. Along with the change, the world's so-called Internet of Everything (for short, IoE) is getting closer to our life. In the IoE world, billions of devices are used for various IT services, including social network websites and applications, which deal with users' personal data for better IT services [1]. Not all data can be stored and managed in small and low-powered devices, and thus, we need to use Internet-connected storage. Although the use of Internet-connected storage can make it possible to utilize much more data without storing it in local storage, we need to care about the security of data which are stored and managed in remote storage.

The main security concern in using Internet-connected storage such as a cloud storage is data privacy [2–4]. The storage inevitably stores and manages the incremental amount of sensitive data on clients. Clients of the storage service must entrust their data to a service provider [5]. Encryption has been the most classical method to provide data privacy. To provide encryption-based access control for clients' sensitive data, a number of value-added encryption techniques have been studied including attribute encryption techniques [6].

A general data protection regulation (GDPR) has forced companies to use encryption of personal data to reduce the probability of a data breach [7]. Accordingly, companies are encrypting, storing. and managing customers' personal information. When an encryption is used for data privacy, we face another obstacle. The storage server should be given a capability that allows server to identify exactly the documents a client wants to retrieve without decrypting them. As one of the basic steps to resolve this difficulty, secure keyword search over encrypted data is receiving much attention. Secure keyword search enables a user to search the encrypted data with a keyword without revealing any information on the data. When an encrypted document is uploaded to a server, a set of ciphertexts of keywords in the document are appended to the encrypted document. Let CT_w denote the ciphertext of keyword w. For a given query (also called *trapdoor*) $T_{w'}$, the server runs the function test with inputs CT_w and $T_{w'}$ to identify whether or not $w = w'$. Only the user who can generate query $T_{w'}$, such that test $(CT_w, T_{w'}) = 1$, can retrieve the encrypted documents containing keyword w. Secure keyword search is a primitive to construct various queries and can be extended to the complex queries such as range queries and inner-product queries [8].

Secure keyword search systems can be classified into two types: asymmetric and symmetric settings. In the asymmetric setting [9–11], known as public key encryption with keyword search (PEKS), ciphertext CT_w of keyword w is generated under a public key and only the owner of the corresponding secret key can generate trapdoor T_w. Hence, PEKS is suitable in a store-and-forward system such as an e-mail system. In the symmetric setting [12–18], ciphertext CT_w of keyword w is generated under a symmetric key and only the owner of the key can generate trapdoor T_w using the symmetric key. Here, the symmetric key is not shared but owned by one client. The symmetric setting is suitable to personal storage service as well as a blog and web-hard service, where the same client uploads and downloads his/her data.

1.1. Necessity of Keyword Privacy.

1.1. Necessity of Keyword Privacy. The formal notion of secure keyword search has considered ciphertext confidentiality, i.e., a semantic security against an attacker who generates the ciphertexts of keywords of her choice. When given a query and a ciphertext of a keyword, the server can decide whether or not the ciphertext is related to the query by running the function test. Therefore, it is not possible to guarantee ciphertext confidentiality without guaranteeing the secrecy of the query (keyword privacy).

As stated in [19, 20], it is not possible to provide keyword privacy of the trapdoor in PEKS which is one of the searchable encryptions in asymmetric setting, due to the ciphertext of a guessed keyword. Hence, an adversary can obtain the test result of the ciphertext of the guessed keyword and a given trapdoor. In [11], Rhee et al. firstly defined the notion of a keyword privacy in asymmetric setting. They proposed the enhanced PEKS scheme that the keyword privacy can only be provided in situations where only the server can test whether the ciphertext and trapdoor are related or not. There has been several works providing keyword privacy in the symmetric setting. Shen et al. [20] firstly proposed a symmetric predicate encryption scheme for an inner-product operation of two vectors, which are used in generating a ciphertext and a token (like a trapdoor in PEKS), and considered keyword privacy in a symmetric predicate encryption scheme. Their scheme is constructed on composite-order bilinear groups, which requires 25 times of exponentiations and 30 times of pairing operations of those in prime-order groups [21]. Recently, Blundo et al. [12] proposed a symmetric hidden vector encryption in asymmetric prime-order bilinear groups. However, there exists no efficiently computable morphism between two different groups used and its security depends on the hardness of nonstandard $(d, m) - Q$ assumption.

However, in general, a predicate encryption differs from a searchable encryption in that a decryption occurs at the same time as the test process. Since it is not common to trust the administrator of the server in various cloud environments, it is necessary to separate the decryption and test processes so that the unreliable server cannot perform the decryption. As noted above, the previous results in symmetric predicate encryption and symmetric hidden vector encryption cannot be immediately adopted for symmetric keyword search.

Also, the protocols for providing access pattern privacy were proposed in [22, 23]. That is, anyone cannot get which documents contain the keyword. But, access pattern privacy does not provide the keyword privacy from the given queries. The keyword privacy is the more intuitional than the pattern privacy. Once the information of keyword from the queries are revealed, then the privacy of the corresponding ciphertext cannot be guaranteed even though the pattern privacy is guaranteed. Also, the protocols providing access pattern privacy do not satisfy the search correctness. That is, these protocols considering access pattern privacy cause a search error and require the additional efforts for fixing the search error.

The comparisons with [20, 21] are shown in Section 4.

1.2. Our Contributions. Our contributions in this paper are twofold:

(1) We firstly define the "trapdoor indistinguishability" for keyword privacy in symmetric keyword search against an active adversary who is able to get trapdoors as well as ciphertexts for any nontarget keyword of his choice. This security of a trapdoor guarantees that the keyword does not reveal any information on any keyword.

(2) We construct a practical and secure keyword search, called secure symmetric keyword search (SSKS), which is tailored for Internet-connected storage service. To construct SSKS, we exploit well-known results of PEKS. Moreover, the proposed scheme achieves both ciphertext confidentiality and keyword privacy. Our construction is efficient since it is based on prime-order bilinear groups unlike the scheme in [20]. The security depends on the hardness of standard assumptions. Ciphertext confidentiality is based on the hardness of the bilinear DH and keyword privacy depends on the hardness of the decisional 3-party DH assumption.

2. Preliminaries

For giving concrete description, we will use pairing-related operations. So, in this section, we describe some fundamental definitions for pairing, hard problems defined over the operation, and formal definitions for scheme descriptions and security features.

2.1. Underlying Mathematical Problems. The pairing operation is defined over an elliptic cubic curve. We will give simple definition of the operation since it is possible to understand our scheme with the knowledge of the property so-called bilinearity of pairing.

Definition 1 (bilinear map). The definition of bilinear groups appears in [9]. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two (multiplicative) cyclic groups of prime order p. We assume that g is a generator of $\mathbb{G}$. e: $\mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ which is a bilinear map with the following properties:

(1) For all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}$, $e(u^a, v^b) = e(u, v)^{ab}$

(2) $e(g, g) \neq 1$ and there is an efficient algorithm to compute map e

To prove the security of our scheme, we use the bilinear Diffie–Hellman assumption and the decision three-party Diffie–Hellman assumption which are defined as follows.

Definition 2 (bilinear Diffie–Hellman assumption (BDH)). The BDH problem [9] is as follows:

$$\text{given}\left(g, g^a, g^b, g^c, Z\right) \in \mathbb{G}^4 \times \mathbb{G}_T \text{as input, compute } e(g, g)^{abc} \in \mathbb{G}_T. \tag{1}$$

The BDH assumption is that all polynomial time algorithms have a negligible advantage in solving the BDH problem.

Definition 3 (decision 3-party Diffie–Hellman assumption (3-party DH)). The decision 3-party Diffie–Hellman problem [24] is as follows:

$$\begin{aligned} &\text{given}\left(g, g^a, g^b, g^c, Z\right) \in \mathbb{G}^5 \text{as input, determine} \\ &\quad if\ Z = g^{abc} or\ Z \text{ is random in } \mathbb{G}. \end{aligned} \tag{2}$$

The 3-party DH assumption is that all polynomial time algorithms have a negligible advantage in solving the decisional 3-party DH problem.

2.2. Formal Definitions for SKSS. We begin by reviewing the formal definition of symmetric keyword search scheme.

Definition 4. A symmetric keyword search scheme (SKSS) can be noted as SSKS = (**KG**, **SEKS**, **STd**, **Test**) which consists of four algorithms. The algorithms are described as follows:

(1) KG (k) takes security parameter $k \in \mathbb{Z}^+$ and generates secret key SK.

(2) SEKS (SK, w) takes input secret key SK and keyword $w \in \mathcal{KW}$, where $\mathcal{KW}$ is a keyword space. It returns ciphertext CT_w.

(3) STd (SK, w) takes input secret key SK and keyword w. It outputs trapdoor T_w.

(4) Test $(CT_w, T_{w'})$ takes input ciphertext CT_w and trapdoor $T_{w'}$. If $W = W'$, output "1"; otherwise, output "0."

For any scheme, it should be guaranteed that it works as intended. More precisely, if w is identical to w', then the test algorithm **Test**, $(CT_w, T_{w'})$ outputs 1. For a SKSS, we define its correctness as follows.

Definition 5 (correctness). For the security parameter k, we define that *SSKS* algorithm satisfies correctness if there is a SKSS scheme *SSKS* = (KG, SEKS, STd, Test) which is defined over a keyword space $\mathcal{KW}$ and secret key *sk*←*SK*; then for any keywords $w, w' \in \mathcal{KW}$,

$$\Pr\left[\text{Test}\left(CT_w, T_{w'}\right)\right] = 1 - \text{neg}(k), \tag{3}$$

where $CT_w \leftarrow \text{SEKS}(SK, w)$ is valid for a keyword w and $T_{w'} \leftarrow \text{STd}(SK, w')$ valid for a keyword w'. Here, if *neg* is negligible, for any constant k, there exists N such that $neg(k) < 1/n^k$ for $n > N$.

2.2.1. Ciphertext Confidentiality. Our definition for a ciphertext confidentiality (SEKS-IND-CPA-security) follows the general framework of those given in [9, 12, 13].

Let $\mathcal{A}$ be an probabilistic polynomial time adversary whose running time is bounded by t, which is a polynomial in a security parameter k. In the experiment of Table 1, $\mathcal{A}$ chooses keywords w_0 and w_1 in the find stage. Given challenge ciphertext $CT^*_{w_b}$ in the guess stage, $\mathcal{A}$ tries to correctly guess b. *St* is used to retain some state information. $\mathcal{A}$ is allowed to obtain trapdoors and ciphertexts by querying trapdoor oracle $\text{STd}(SK, w)$ and encryption oracle $\text{SEKS}(SK, w)$, respectively. But, $\mathcal{A}$ is not allowed to obtain the trapdoor of w_0 or w_1. Otherwise, $\mathcal{A}$ could run Test to find out b.

Here, the trapdoor oracle $\text{STd}(w)$ and the encryption oracle $\text{SEKS}(w)$ are defined as follows.

Oracle $\text{STd}(SK, w)$	Oracle $\text{SEKS}(SK, w)$
$\text{STSet} \leftarrow \text{STSet}(k) \cup \{w\}$	$\text{CSet} \leftarrow \text{CSet}(k) \cup \{w\}$
$T_w \leftarrow \text{STd}(SK, w)$	$CT_w \leftarrow \text{SEKS}(SK, w)$
Return T_w	Return CT_w

The advantage of $\mathcal{A}$ attacking a ciphertext confidentiality is defined as follows:

$$\begin{aligned} \text{Adv}^{\text{seks-ind-cpa}}_{\text{SSKS},\mathcal{A}}(k) &= \Pr\left[\text{Exp}^{\text{seks-ind-cpa-1}}_{\text{SSKS},\mathcal{A}}(k) = 1\right] \\ &\quad - \Pr\left[\text{Exp}^{\text{seks-ind-cpa-0}}_{\text{SSKS},\mathcal{A}}(k) = 1\right]. \end{aligned} \tag{4}$$

Definition 6 (ciphertext confidentiality). We say that SSKS scheme satisfies SEKS – IND – CPA-security against an adaptive chosen plaintext attack if for any polynomial adversary $\mathcal{A}$, the advantage $\text{Adv}^{\text{seks-ind-cpa}}_{\text{SSKS},\mathcal{A}}(k)$ is negligible in security parameter k.

Table 1: Experiment of a ciphertext confidentiality for SSKS.

```
$\mathrm{Exp}_{\mathrm{SSKS},\mathscr{A}}^{\mathrm{seks-ind-cpa}-b}(k)$
$\mathrm{STSet} \leftarrow \varnothing$
$SK \leftarrow KG(k)$
$(w_0, w_1, s) \leftarrow \mathscr{A}^{\mathrm{STd}(SK;),\mathrm{SEKS}(SK;)}$ (find)
$b \leftarrow \{0,1\}$
$CT^*_{w_b} \leftarrow \mathrm{SEKS}(SK, w_b)$
$b' \leftarrow \mathscr{A}^{\mathrm{STd}(SK;),\mathrm{SEKS}(SK;)}$ (guess, $CT^*_{w_b}, St$)
If $\{w_0, w_1\} \cap \mathrm{STSet}(k) = \varnothing$
then return $b'$ else return 0
```

Table 2: Experiment of a keyword privacy for SSKS.

```
$\mathrm{Exp}_{\mathrm{SSKS},\mathscr{A}}^{\mathrm{key-ind-cpa}-b}(k)$
$\mathrm{STSet} \leftarrow \varnothing$
$\mathrm{CSet} \leftarrow \varnothing$
$SK \leftarrow KG(k)$
$(w_0, w_1, s) \leftarrow \mathscr{A}^{\mathrm{STd}(SK;),\mathrm{SEKS}(SK;)}$ (find)
$b \leftarrow \{0,1\}$; $T^*_{w_b} \leftarrow \mathrm{STd}(SK, w_b)$
$b' \leftarrow \mathscr{A}^{ST\,d(SK;),SEKS(SK;)}$ (guess, $T^*_{w_b}, St$)
If $\{w_0, w_1\} \cap \mathrm{STSet}(k) = \varnothing$
and $\{w_0, w_1\} \cap \mathrm{CSet}(k) = \varnothing$
then return $b'$ else return 0
```

2.2.2. Keyword Privacy. We newly define keyword privacy (KEY-IND-CPA-security) for SSKS. In the experiment of Table 2, $\mathscr{A}$ tries to correctly guess b of $T^*_{w_b}$. $\mathscr{A}$ is allowed to query the trapdoor oracle and the encryption oracle. But, $\mathscr{A}$ should not be allowed to obtain the ciphertext of w_0 or w_1. Otherwise, $\mathscr{A}$ could run Test to find out b. $\mathscr{A}$ also needs to be restricted in obtaining the trapdoor of w_0 or w_1. Otherwise, $\mathscr{A}$ might find ciphertext CT in the database such that $\mathrm{Test}(CT, T_{w_b}) = 1$. If the trapdoor of w_0 (or w_1) is available, $\mathscr{A}$ then can easily decide the value of b.

The advantage of $\mathscr{A}$ attacking a keyword privacy is defined as follows:

$$\mathrm{Adv}_{\mathrm{SSKS},\mathscr{A}}^{\mathrm{key-ind-cpa}}(k) = \Pr\left[\mathrm{Exp}_{\mathrm{SSKS},\mathscr{A}}^{\mathrm{key-ind-cpa-1}}(k) = 1\right] - \Pr\left[\mathrm{Exp}_{\mathrm{SSKS},\mathscr{A}}^{\mathrm{key-ind-cpa-0}}(k) = 1\right]. \tag{5}$$

Definition 7. (keyword privacy). We say that the SSKS scheme satisfies KEY − IND − CPA-security against an adaptive chosen plaintext attack if for any polynomial adversary $\mathscr{A}$, the advantage $\mathrm{Adv}_{\mathrm{SSKS},\mathscr{A}}^{\mathrm{key-ind-cpa}}(k)$ is negligible in security parameter k.

3. New Symmetric Keyword Search with Keyword Privacy

In this section, we give a detailed description for our symmetric keyword search with keyword privacy.

Since we use pairing operations for our scheme, we use the following notations. Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of prime order p, and let e: $\mathbb{G} \times \mathbb{G} \longrightarrow \mathbb{G}_T$ be a bilinear map. We use hash functions H_1: $\{0,1\}^* \longrightarrow \mathbb{G}$ and H_2: $\mathbb{G}_1 \longrightarrow \{0,1\}^{\log p}$. Our construction works as follows:

KG (k) takes the security parameter $k \in \mathbb{Z}^+$ and picks a random exponent $\alpha \in \mathbb{Z}_p$, generator $g \in \mathbb{G}$, and a random value $u \in \mathbb{G}(u \neq g)$. It outputs secret key

$$SK = [K_1, K_2, K_3] = [\alpha, g^{\alpha}, u]. \tag{6}$$

SEKS (SK, w) takes as input secret key SK and keyword $w \in \mathscr{KW}$, where $\mathscr{KW}$ is a keyword space. It picks a random exponent $s \in \mathbb{Z}_p$ and returns the ciphertext:

$$CT_w = [C_1, C_2, C_3] = [K_3^s, g^s, H_2(e(K_2^s, H_1(w)))]. \tag{7}$$

STd (SK, w) takes as input secret key SK and keyword w. It picks a random exponent $r \in \mathbb{Z}_p$ and outputs the corresponding trapdoor:

$$T_w = [T_1, T_2] = [g^r, H_1(w)^{K_1} \cdot K_3^r]. \tag{8}$$

Test (CT, T_w) takes as input ciphertext CT and trapdoor T_w and parses CT as $[C_1, C_2, C_3]$ and T_w as $[T_1, T_2]$. It checks if the following equality holds:

$$H_2\left(\frac{e(T_2, C_2)}{e(T_1, C_1)}\right) = C_3. \tag{9}$$

If so, output "1"; otherwise, output "0."

4. Analysis

In this section, we analyze the proposed scheme in terms of security against the security notions discussed in Section 2.2. We also compare the proposed scheme with existing schemes to show that our scheme guarantees better security than the existing schemes.

4.1. Security. We now prove that our construction satisfies ciphertext confidentiality and keyword privacy. Ciphertext confidentiality is proved as the same manner in [9].

Theorem 1. *If the (t, ε)-decisional BDH assumption holds in $\mathbb{G}$, then our SSKS scheme is SEKS − IND − CPA-secure.*

Proof. Suppose $\mathscr{A}$ is an adversary that has advantage ϵ in breaking ciphertext confidentiality. We construct $\mathscr{B}$ that solves the BDH problem with probability at least $\varepsilon\prime = \varepsilon/e(q_T q_{H_2})$, where e is the base of the natural logarithm and q_{H_2} (resp., q_T) is the number of hash function H_2 queries (resp., the number of trapdoor queries). Given g, $u_1 = g^a$, $u_2 = g^b$, and $u_3 = g^c \in \mathbb{G}$, the goal of $\mathscr{B}$ is to compute $e(g,g)^{abc} \in \mathbb{G}_T$. $\mathscr{B}$ interacts with $\mathscr{A}$ as follows:

Setup. $\mathscr{B}$ picks a random $t \in \mathbb{Z}_p$, and let $K_2 = u_1 = g^a$ and $K_3 = u = g^t$.

(1) H_1, H_2- *Queries.* As the same manner in [9], $\mathscr{B}$ can simulate H_1 and H_2 queries. If there exists

$\langle w_i, h_i, e_i, c_i\rangle \in H_1$-list, then $\mathscr{B}$ responds with h_i. Otherwise, $\mathscr{B}$ generates a random coin $c_i \in \{0,1\}$ so that $\Pr[c_i = 0] = 1/(q_T + 1)$. If $c_i = 0$, then $\mathscr{B}$ picks a random $e_i \in \mathbb{Z}_p$ and sets $h_i = g^{b\cdot e_i} \in \mathbb{G}$. Otherwise, $\mathscr{B}$ sets $h_i = (u_2)^{e_i} = g^{e_i}$. $\mathscr{B}$ adds the tuple $\langle w_i, h_i, e_i, c_i\rangle$ to the H_1-list and responds to $\mathscr{A}$ with $H_1(w_i) = h_i$. Similarly, if there exists $t \in \mathbb{G}_T$ such that $(t, V) \in H_2$-list, then $\mathscr{B}$ responds with $H_2(t) = V$. Otherwise, $\mathscr{B}$ responds to a query for $H_2(t)$ by picking a random $V \in \{0,1\}^{\log p}$ for t and setting $H_2(t) = V$ and adds (t, V) to the H_2-list.

Query Phase 1. $\mathscr{A}$ makes trapdoor and ciphertext queries as follows:

(2) *Trapdoor Queries.* $\mathscr{B}$ can obtain $h_j \in \mathbb{G}$ such that $H_1(w) = H_1(w_j) = h_j$, where $\langle w_j, h_j, e_j, c_j\rangle \in H_1$-list such that $w_j = w$. If $c_i = 0$, then $\mathscr{B}$ reports failure and terminates. Otherwise, since there exists $e_j \in \mathbb{Z}_p$ such that $h_j = g^{e_j} \in \mathbb{G}$, $\mathscr{B}$ picks a random $r' \in \mathbb{Z}_p$ and sets $K_1 = g^{r'}$ and $K_2 = (u_1)^{e_j} \cdot u^{r'}$. $\mathscr{B}$ gives back $T_w = (K_1, K_2)$ for w to $\mathscr{A}$.

(3) *Ciphertext Queries.* $\mathscr{B}$ can obtain $h_j \in \mathbb{G}$ such that $H_1(w) = H_1(w_j) = h_j$, where $\langle w_j, h_j, e_j, c_j\rangle \in H_1$-list such that $w_j = w$. $\mathscr{B}$ picks a random $s \in \mathbb{Z}_p$ and sets $C_1 = (u)^s$, $C_2 = g^s$ and $C_3 = H_2(e(u_1^s, H_1(w)))$, where $u \in \mathbb{G}$ is the let value in the setup phase. $\mathscr{B}$ gives back CT for w to $\mathscr{A}$.

Challenge. $\mathscr{A}$ outputs challenge keywords w_0 and w_1. To obtain $h_0, h_1 \in \mathbb{G}$ such that $H_1(w_0) = h_0$ and $H_1(w_1) = h_1$, $\mathscr{B}$ queries w_0 and w_1 to H_1-queries. Let $\langle w_b, h_b, e_b, c_b\rangle$ be the corresponding tuples on the H_1-list $(b = 0, 1)$. If both $c_0 = 1$ and $c_1 = 1$, then $\mathscr{B}$ aborts. Otherwise, since there exists $b \in \{0,1\}$ such that $c_b = 0$, $\mathscr{B}$ picks a random $z \in \{0,1\}^{\log p}$ and sets $C_1{}^* = u_3^t$, $C_2^* = u_3$ and $C_3^* = z$, where $t \in \mathbb{Z}_p$ is the selected value in the setup phase. $\mathscr{B}$ responds with $CT^* = [C_1^*, C_2^*, C_3^*]$.

Query Phase 2. $\mathscr{B}$ answers the queries in the same manner as phase 1 under the restriction of trapdoor queries of $w \neq w_0, w_1$.

Output. $\mathscr{A}$ outputs its guess, $b' \in \{0,1\}$. The value $h_i = u_2^{e_i}$ was set with the probability $1/(q_T + 1)$ in the setting of the H_1 queries. Since $\mathscr{A}$ queries the H_2 oracle regarding the value of the form $e(K_2, H_1(w_b))^s)) = e(g^a, (g^{bc})^{e_b})$ with the same probability $1/(q_T + 1)$, there exists one pair of the form $(e(g,g)^{abce_b}, H_2(e(g^a, (g^{bc})^{e_b})) \in H_2$-list. Therefore, $\mathscr{B}$ picks a random pair $(t, V) \in H_2$-list and outputs $t^{(1/e_b)}$ as its guess for $e(g,g)^{abc}$, where e_b is the selected value in the challenge phase. To show that $\mathscr{B}$ correctly outputs $e(g,g)^{abc}$ with probability at least ε', we should analyze the probability that $\mathscr{B}$ does not abort during the simulation.

We define the following events:

(1) $(\mathrm{E}_1)\mathscr{B}$ does not abort as a result of any of $\mathscr{A}$'s trapdoor queries

(2) $(\mathrm{E}_2)\mathscr{B}$ does not abort during challenge phase

(3) $(\mathrm{E}_3)\mathscr{A}$ does not issue a query for either one of

$$H_2(e(K_2, H_1(w_0))^s)))\text{and}H_2(e(K_2, H_1(w_1))^s))) \tag{10}$$

in the real attack

We can show in the same manner as in [9] that

$$\Pr[\mathrm{E}_1] \geq \frac{1}{e}, \Pr[\mathrm{E}_2] \geq \frac{1}{q_T} \text{and} \Pr[\mathrm{E}_3] \geq 2\varepsilon. \tag{11}$$

We omit the detailed description.

To prove keyword privacy, we consider a hybrid game which differs on what challenge trapdoor T_b^* is given by the challenger to $\mathscr{A}$. We suppose that T_β^* is the real trapdoor $T_{w_\beta^*}$ $(\beta \in \{0,1\})$, which is the challenge ciphertext given to the adversary during a real security game or is a random $R \in \mathbb{G}$. □

Theorem 2. *If the (t, ε)-decisional 3-party DH assumption holds, then our SSKS scheme is KEY − IND − CPA-secure.*

Proof. Suppose that there exists a t-time adversary $\mathscr{A}$ with nonnegligible difference ϵ between its advantage for challenge T_{w^*} and its advantage for challenge $R \in \mathbb{G}$. We construct an algorithm $\mathscr{B}$ that solves the decisional 3-party DH problem in $\mathbb{G}$. Given a random challenge (g, g^a, g^b, g^c, Z), $\mathscr{B}$ outputs 1 if $Z = g^{abc}$ and 0 otherwise. $\mathscr{B}$ interacts with $\mathscr{A}$ as follows:

Init. $\mathscr{A}$ outputs challenge keywords $w_0^*, w_1^* \in \mathscr{KW}$. $\mathscr{B}$ flips a coin to obtain $\beta \in \{0,1\}$, internally.

Setup. $\mathscr{B}$ chooses a random $\tau \in \mathbb{Z}_p$ and sets unknown secret values $K_2 = g^{ab}$ and $K_3 = (g^a)^\tau$. $\mathscr{B}$ chooses random exponents $e_\beta^*, e_1, e_2, \ldots, e_{q_{H_1}} \in \mathbb{Z}_p$ and sets $H_1(w_i) = H(w_\beta^*) = (g^c)^{e_\beta^*} \in \mathbb{G}$ and $H_1(w_i) = (g^{1/a})^{e_i} \in \mathbb{G}\,(w_i \neq w^*)$.

Query Phase 1. $\mathscr{A}$ makes trapdoor and ciphertext queries of the form $w_i \neq w_\beta^*$ as follows:

(1) *Trapdoor Queries.* If $w_i \neq w_\beta^*$, then $\mathscr{B}$ chooses a random exponent $r_i \in \mathbb{Z}_p$ and sets $T_{w_i} = [g^{r_i}, (g^b)^{e_i} \times u^{r_i}]$. If $w_i = w_\beta^*$, then $\mathscr{B}$ aborts.

(2) *Ciphertext Queries.* If $w_i = w_\beta^*$, then $\mathscr{B}$ aborts. If $w_i \neq w_\beta^*$, then $\mathscr{B}$ chooses a random exponent $s_i \in \mathbb{Z}_p$ and sets $CT_{w_i} = [(g^a)^{\tau s_i}, g^{s_i}, H_2(e((g^b)^{s_i}, g^{e_i}))]$.

Challenge Phase . $\mathscr{B}$ chooses a random exponent $r^* \in \mathbb{Z}_p$ and outputs the challenge trapdoor for keyword w_β^* as follows:

$$T^* = \left[g^{r^*}, Z^{e_\beta^*} \times u^{r^*}\right]. \tag{12}$$

Query Phase 2. $\mathscr{B}$ answers the queries in the same manner as phase 1 under the restriction of trapdoor and ciphertext queries of $w \neq w_0, w_1$.

Table 3: Comparison among symmetric predicate encryptions.

Scheme	Group order	Type of pairing	Hardness assumption	Class of encryption
Proposed scheme	p	Symmetric	BDH and decision 3-party DH	Searchable
Shen et al. [20]	pqrs	Symmetric	C3DH and DLinear [20]	Predicated
Blundo et al. [12]	p	Asymmetric	$(d, m) - \mathrm{Q}$ [12]	Predicated

p, q, r, and s: prime values.

Table 4: Comparison with symmetric searchable encryptions.

Scheme	CT *Ind*	Keyword *Ind*	Searching *Error*
Proposed scheme	Satisfied	Satisfied	Not related
Watanabe et al. [18]	Satisfied	Not satisfied	Related
Demertz et al. [17]	Satisfied	Not satisfied	Related

CT *Ind*: ciphertext indistinguishability; Keyword *Ind*: keyword indistinguishability.

Table 5: Comparison of security and performance between our scheme and the others.

Scheme	Functionalities			Performances		
	CT *Ind*	Keyword *Ind*	Class	Size (CT)	Size (trap)	Comp (test)
Shen et al. [20]	Satisfied	Satisfied	Predicate	4G	$(2 + 2n)G$	$(2 + 2n)e$ + De
Blundo et al. [12]	Satisfied	Satisfied	Predicate	2G	2G	2 p + 2e + De
Proposed scheme	Satisfied	Satisfied	Searchable	2G + log p	2G	2p

CT *Ind*: ciphertext indistinguishability; Keyword *Ind*: keyword indistinguishability; Size (CT): size of PEKS (dPEKS) ciphertext; Size (trap): size of trapdoor (dTrapdoor); Comp (test): computation cost of test (dTest); G: element in $\mathbb{G}$; p: pairings; e: exponentials; De: decryption corresponding to En.

Guess. $\mathscr{A}$ outputs guess $\beta' \in \{0, 1\}$ in response to the challenge trapdoor. If $\beta = \beta'$, then $\mathscr{B}$ outputs 1. Otherwise, $\mathscr{B}$ outputs 0.

$\mathscr{B}$'s advantage in solving the decisional 3-party DH problem is directly taken from $\mathscr{A}$'s advantage to distinguish between $T_{w^*_\beta}$ and R, except with negligible probability less than or equal to $q_T q_C / p$. □

4.2. Comparison. Since the symmetric searchable encryption scheme is related to symmetric predicate encryption schemes and symmetric searchable encryption providing access pattern privacy, we give the following two categories for correct comparison.

4.2.1. Symmetric Predicate Encryption. There have been several studies providing keyword privacy in the symmetric setting. In [20], symmetric predicate encryption schemes for an inner-product operation of two vectors were proposed and it considered keyword privacy in a symmetric predicate encryption scheme.

Their scheme is constructed on composite-order bilinear groups, which requires 25 times of exponentiations and 30 times of pairing operations of those in prime-order groups. Recently, Blundo et al. [12] proposed a symmetric hidden vector encryption in asymmetric prime-order bilinear groups. However, there exists no efficiently computable morphism between two different groups used and its security depends on the hardness of nonstandard $(d, m) - Q$ assumption. The comparisons with [20, 21] are shown in Table 3. However, as mentioned before, the predicate encryption cannot be immediately adopted for symmetric keyword search on cloud storage services.

4.2.2. Symmetric Searchable Encryption. In the previous works for the symmetric searchable encryption [12–18], only the ciphertext confidentiality was guaranteed. In searchable encryption schemes, if the trapdoor T_w is given, then the server can run the test function with T_w and ciphertext $C_{w'}$. When the keyword w from a trapdoor T_w is revealed, the confidentiality of the ciphertext C_w associated with T_w cannot be guaranteed. Once the keyword privacy is ensured, even if we know that the given trapdoor and the ciphertext are associated, we cannot know which keyword is related to the trapdoor and the ciphertext. As shown in Table 4, we firstly defined the keyword privacy in a symmetric searchable encryption scheme and presented an efficient SSKS scheme with ciphertext confidentiality as well as keyword privacy.

4.2.3. Efficiency. Since the access patter privacy is the main security goal of this work, as mentioned in the beginning of Section 4.2, we will compare our scheme with predicate encryption schemes. In Table 5, we compare our scheme and the other schemes in terms of functionalities and performances. The table shows that our scheme provides both ciphertext confidentiality and keyword privacy as well as enables to search on encrypted data without any decryption. However, predicate encryption schemes require decryption operations to implement test function in predicate encryption schemes. Therefore, the predicate encryption schemes have the restriction that the test function can be processed by only a trusted party. In comparison with Shen et al.'s scheme and Blundo et al.'s scheme, our scheme requires the similar size in terms of the ciphertext and the smallest size in terms of the trapdoor. Most of all, in the test phase, our scheme needs the smaller computation than other schemes.

5. Conclusions

In this paper, we proposed a practical and secure keyword search, called secure symmetric keyword search (SSKS), which is tailored for cloud storage service. We firstly defined the keyword privacy in a symmetric searchable encryption scheme and presented SSKS scheme that guarantees ciphertext confidentiality as well as keyword privacy. Our SSKS systems and the new security model for key privacy in the symmetric setting can be further exploited to construct symmetric searchable encryption schemes providing extended queries such as conjunctive and inner-product queries [25–27].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was partly supported by the Institute of Information and Communications Technology Planning and Evaluation (IITP) grant funded by the Korea Government (MSIT) (No. 2021-0-00779, Development of High-Speed Encryption Data Processing Technology That Guarantees Privacy Based Hardware, 50%) and National R&D Program through the National Research Foundation of Korea (NRF) funded by Ministry of Science and ICT (NRF-2021R1F1A1056115, 50%).

References

[1] Y. A. Shichkina, G. V. Kataeva, Y. A. Irishina, and E. S. Stanevich, "The use of mobile phones to monitor the status of patients with Parkinson's disease," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 11, no. 2, pp. 55–73, 2020.

[2] G. Lacava, A. Marotta, F. Martinelli et al., "Cybsersecurity issues in robotics," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, vol. 12, pp. 1–28, 2021.

[3] H. Anada1 and Y. Ueshige, "Anonymous deniable predicate authentication scheme with revocability," *Journal of Internet Services and Information Security (JISIS)*, vol. 11, no. 3, pp. 1–15, 2021.

[4] D. Caputo, L. Verderame, A. Ranieri, A. Merlo, and L. Caviglione, "Fine-hearing Google Home: why silence will not protect your privacy," *JoWUA*, vol. 11, no. 1, pp. 35–53, 2020.

[5] G. Baldi, Y. Diaz, T. Dimitrakos et al., "Session-dependent usage control for BigData," *Journal of Internet Services and Information Security(JISIS)*, vol. 10, no. 3, pp. 76–92, 2020.

[6] S. Lawa and R. Krishnan, "Policy review in attribute based access control: a policy machine case study," *Journal of Internet Services and Information Security(JISIS)*, vol. 10, no. 2, pp. 67–81, 2020.

[7] E. Union, *General Data Protection Regulation, Europe*, 2016.

[8] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data,"vol. 4392, pp. 535–554, in *Proceedings of the Theory of Cryptography Conference*, vol. 4392, pp. 535–554, Springer, Amsterdam, The Netherlands, February 2007.

[9] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," *Advances in Cryptology-EUROCRYPT 2004*, vol. 3027, pp. 506–522, 2004.

[10] J. Baek, R. Safavi-Naini, and W. Susilo, "Public key encryption with keyword search revisited," *Proceedings of the ACCSA2008, LNCS*, vol. 5072, pp. 1249–1259, 2008.

[11] H. S. Rhee, J. H. Park, W. Susilo, and D. H. Lee, "Trapdoor security in a searchable public-key encryption scheme with a designated tester," *Journal of Systems and Software*, vol. 83, no. 5, pp. 763–771, 2010.

[12] C. Blundo, V. Iovino, and G. Persiano, "Private-key hidden vector encryption with key confidentiality," *Cryptology and Network Security*, vol. 5888, pp. 259–277, 2009.

[13] J. Katz, A. Sahai, and B. Waters, "Predicate encryption supporting disjunctions, polynomial equations, and inner products," *Proceedings of the EUROCRYPT2008, LNCS*, vol. 4965, pp. 146–162, 2008.

[14] K. Kurosawa and Y. Ohtaki, "UC-secure searchable symmetric encryption," in *Financial Crypotography and Data Security, FC 2012*, A. D. Keromytis, Ed., Springer, Berlin, Germany, pp. 285–298, 2012.

[15] D. Cash and S. Tessaro, "The locality of searchable symmetric encryption," in *Advances in Cryptology – EUROCRYPT 2014*, P. Nguyen and E. Oswald, Eds., vol. 8441, pp. 351–368, Springer, Berlin, Germany, 2014.

[16] G. Asharov, G. Segev, and I. Shahaf, "Tight tradeoffs in searchable symmetric encryption," in *Lecture Notes in Computer Science, Advances in Cryptology—CRYPTO 2018*, H. Shacham and A. Boldyreva, Eds., Springer International Publishing, Cham, Switzerland, pp. 407–436, 2018.

[17] I. Demertzis, D. Papadopoulos, and C. Papamanthou, "Searchable encryption with optimal locality: achieving sublogarithmic read efficiency," in *Lecture Notes in Computer Science, Advances in Cryptology – CRYPTO 2018*, H. Shacham and A. Boldyreva, Eds., Springer International Publishing, Cham, pp. 371–406, 2018.

[18] Y. Watanabe, T. Nakai, K. Ohara et al., "How to make a secure index for searchable symmetric encryption, revisited," *Cryptology ePrint Archive*, 2021, https://eprint.iacr.org/2021/948.pdf.

[19] I. R. Jeong, J. O. Kwon, D. Hong, and D. H. Lee, "Constructing PEKS schemes secure against keyword guessing attacks is possible?" *Computer Communications*, vol. 32, no. 2, pp. 394–396, 2009.

[20] E. Shen, E. Shi, and B. Waters, "Predicate privacy in encryption systems," *Theory of Cryptography*, vol. 5444, pp. 457–473, 2009.

[21] S. Garg, A. Sahai, and B. Waters, *Efficient Fully Collusion-Resilient Traitor Tracing Scheme*, Cryptology ePrint Archive, Chicago, IL, USA, 2009, http://eprint.iacr.org/2009/532/.

[22] M. D. Islam, M. Kuzu, and M. Kantarcioglu, "Access pattern disclosure on searchable encryption: ramification, attack and mitigation," in *Proceedings of the NDSS Symposium'12*, February 2012.

[23] X. Zhou, H. Pang, and K.-L. Tan, "Hiding data accesses in steganographic file system," in *Proceedings of the 20th*

International Conference on Data Engineering, pp. 572–583, IEEE, Boston, MA, USA, April 2004.

[24] D. Boneh, A. Sahai, and B. Waters, “Fully collusion resistant traitor tracing with short ciphertexts and private keys,” *Proc. EUROCYYPT2006, LNCS*, vol. 4004, pp. 573–592, 2006.

[25] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, “Searchable symmetric encryption: improved definitions and efficient constructions,” *Journal of Computer Security*, vol. 19, no. 5, pp. 895–934, 2011.

[26] G. Asharov, M. Naor, G. Segev, and I. Shahaf, “Searchable symmetric encryption: optimal locality in linear space via two-dimensional balanced allocations,” in *Proceedings of the ACM Symposium on Theory of Computing, STOC 2016*, pp. 1101–1114, ACM, New York, NY, USA, 2016.

[27] D. X. Song, D. Wagner, and A. Perrig, “Practical techniques for searches on encrypted data,” in *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 44–55, Berkeley, CA, USA, May 2000.