International Journal of Security and Its Applications
Vol.8, No.1 (2014), pp.257-272
http://dx.doi.org/10.14257/ijsia.2014.8.1.24

Secure Data Management Scheme using One-Time Trapdoor on Cloud Storage Environment

Sun-Ho Lee and Im-Yeong Lee[1]

Department of Computer Software Engineering, Soonchunhyang University
{sunho431, imylee}@sch.ac.kr

Abstract

Because of the development of network and computing technology, cloud computing systems, which remotely store data in a third place and retrieves and processes these data with diverse terminals anytime and anywhere, have attracted considerable research attention. If an individual's sensitive information or information about his/her body is stored as data stored in a cloud with remote storage without encryption, then an attacker or unethical server manager can access these stored data without the permission of the data owner. This is a security problem. Therefore, data stored using remote storage, need to be encrypted. However, in the case of data encrypted using a general encryption algorithm, since the entire set of encrypted data is downloaded to the data owner's terminal for a safe search and has to be decrypted for the search, the advantages of remote storage are nullified. To solve this problem, searchable encryption systems have been developed. With a searchable encryption system, encrypted data can be searched safely without the process of decryption. This scheme safely stores the index that can search data and enables search with no information spill. However, the existing scheme, which is available for search, produces the same form of trapdoor for multiple searches of a keyword. A number of queries are transferred to the cloud, and the storage manager infers the keyword through these queries. The fact that the information user searches and the searched data content can be figured out through the queries poses a significant problem. Therefore, this research paper suggests a safe scheme for data management by using a one-time trapdoor so that an unethical server manager or attacker cannot infer the content of the search and the data through the queries when the same user searches the same keyword. This is possible because different trapdoors are produced when the same user searches the same keyword.

Keywords: *Searchable encryption, Data outsourcing, Cloud storage, One-time trapdoor*

1. Introduction

Because of the development of network and computing technology, the cloud computing service has attracted considerable attention. This service stores data at a remote storage location to reduce the operational expense and computational burden of managing data servers for individuals and business, and can retrieve and process the stored data anytime and anywhere. In particular, among cloud computing services, a cloud storage service is provided for free in many companies and is widely used because only the amount of use is charged without any limit on the extension of capacity. However, if an individual's sensitive information is stored in a remote storage location without encryption, an attacker and unethical server manager can access the

[1] Corresponding author: Im-Yeong Lee, imylee@sch.ac.kr

ISSN: 1738-9976 IJSIA

stored data from the server without the consent of the data owner. This is a serious information security issue. To solve this problem, Google has revealed that it will provide a service that will automatically encrypt the data stored by a user on its servers in stages through server-side encryption. However, the big brother problem is that only Google has this technology and manages the key that encrypts the data. To solve this type of problem, searchable encryption systems have been developed. The first public key encryption with keyword search (PEKS) using a bilinear map was proposed by Boneh *et al.*, [1]. The PEKS scheme provides a variety of functions, for example, multi-user and multi-keyword search capability was proposed [2–9]. And proxy re-encryption with keyword search (PRES) for safe sharing of stored data is suggested [10-11]. These scheme that can search the encrypted data without decryption by using the characteristics of a homomorphic encryption. There have been many researches on this encryption scheme.

The process of using a searchable encryption system is as follows: First, the data owner encrypts that data that he/she wants to store at a remote storage location and stores these data. Then, to search the data that have a certain keyword from among the stored data, a trapdoor is produced by using a keyword and a private key. The produced trapdoor is transferred to the server, and the server searches the encrypted data with the trapdoor. In this search process, the server does not know the information of the keyword and the data, and only knows whether the data belong to a certain keyword. Through this process, the server transfers the encrypted data corresponding to the selected keyword to the user. It is incorrect to assume that this searchable encryption system is completely safe for finding data encrypted using a trapdoor. The existing searchable encryption system has the same form with a trapdoor produced to search the same keyword. A considerable number of search queries are transferred to the remote storage and the storage manager has to infer the desired keyword through a query and learn about data searched and stored by the user.

Therefore, in this research, we have developed a scheme for safe data sharing and we suggest a searchable encryption system that uses a one-time trapdoor. With this searchable encryption system, an unethical server manager or attacker cannot infer the content of the search and the searched data when same user searches the same keyword by creating a different trapdoor.

2. Existing Scheme

2.1. Proxy re-encryption with Keyword Search (PRES)

Let us take a look at Ref. [10] proposed by Shao et al. in 2010.

Notation

The notation used in this scheme are as follows.

- q: prime number
- h: random number in G
- l: length of the verification keys
- G: cyclic additive group of order p
- G_T: cyclic multiplicative group of order p
- g: generator of G
- e: bilinear map, $G_1 \times G_1 \rightarrow G_2$
- $H_1()$: hash function, $\{0,1\}^* \rightarrow G$
- $H_2()$: hash function, $\{0,1\}^{\leq l} \rightarrow G$
- $H_3()$: hash function, $\{0,1\}^{\leq l} \rightarrow G$

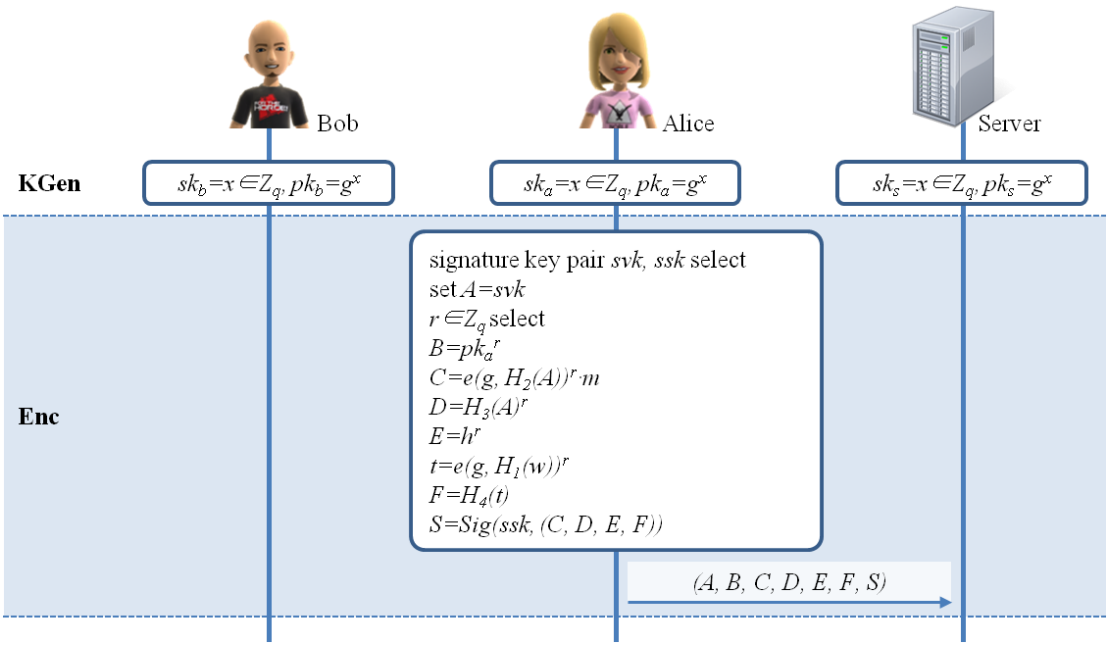

Figure 1. PRES's Data Storage Phase

- $H_4()$: hash function, $G_T \rightarrow \{0,1\}^k$

Data storage Phase

As with most PRES schemes, the protocol of Shao et al. had a total of 7 Steps: KGen, Enc, ReKeyGen, ReEnc, TGen, Test, and Dec. The scheme of PRES can be divided into

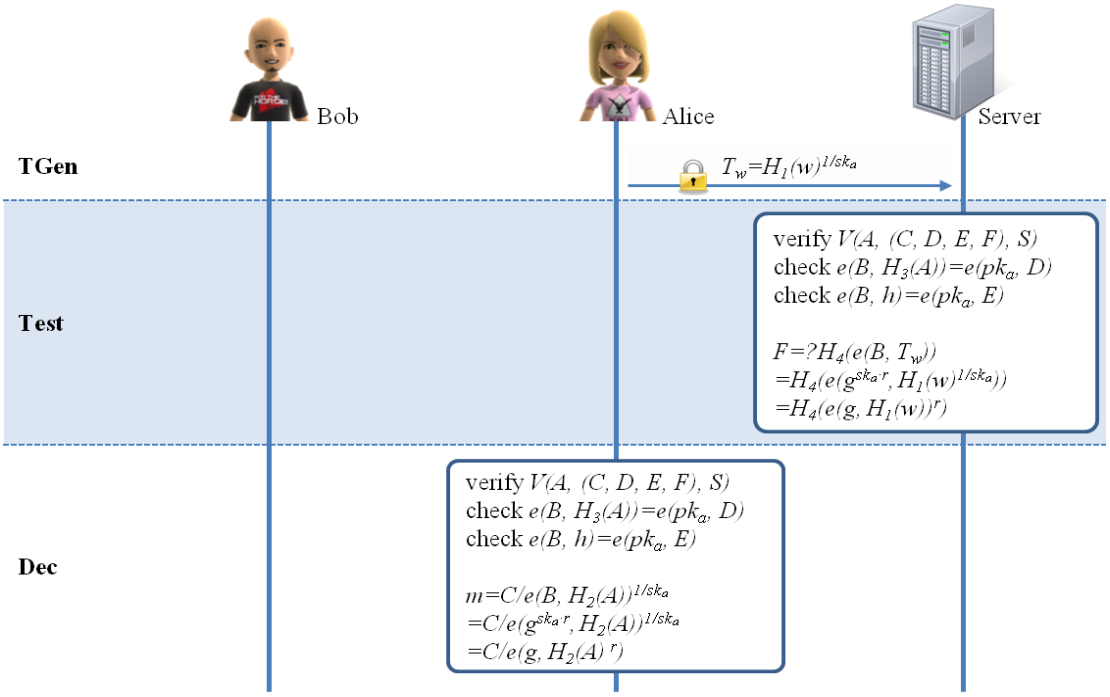

Figure 2. PRES's Data Search Phase

data storage, search, and sharing phases. This study describes a process by phase to understand existing studies. The phase of data storage consists of KGen step generating public key pair and Enc step storing data and keyword in server by encrypting them(see Figure 1).

Data Search Phase

The phase of search consists of TGen which generates trap door with private key and keyword, Test step which searches encrypted index by trap door, and Dec step which decodes searched result data(see Figure 2).

Data Sharing Phase

The phase of Sharing consist of ReKeyGen step generating re-encryption key with Bob, sharing target, and Alice, data owner, and ReEnc step re-encrypting so that Bob also can search Alice's data with re-encryption key. After completing sharing, Bob searches and decrypt sharing data with trapdoor generated by private key and keyword (see Figure 3).

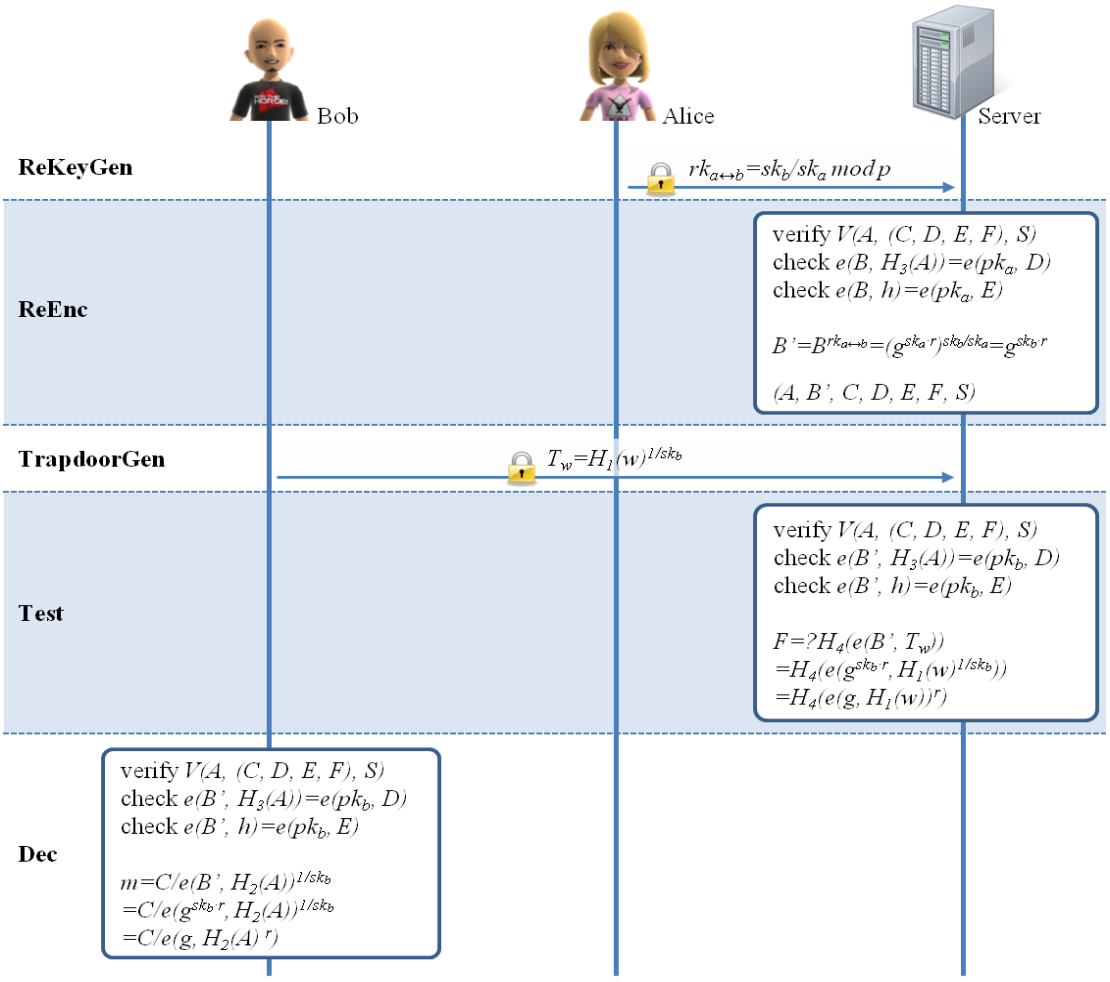

Figure 3. PRES's Sharing Phase

The PRES scheme requires a sharing target Bob's private key to be informed to Alice when generating a re-encryption key, which indicates that the authorization on search and decrypting is handed over to Alice. Further, in searching the same data, as the trapdoor generated has the same pattern each time, contents of data can be Inferred from search query. Furthermore, data encryption requires much performance time as it uses multiplying operation on pairing.

2.2. Proxy Re-encryption with Private Searching (PRPS)

Let us take a look at Ref. [11] proposed by Chen et al. in 2011.

Notation

The notation used in this scheme are as follows.

- ***q*:** prime number
- G_1**:** cyclic additive group of order p
- G_2**:** cyclic multiplicative group of order p
- ***g*:** generator of G
- ***e*:** bilinear map, $G_1 \times G_1 \rightarrow G_2$
- H_1**():** hash function, $\{0,1\}^* \rightarrow G_1^*$
- H_2**():** hash function, $G_2 \rightarrow \{0,1\}^{\log q}$
- H_3**():** hash function, $\{0,1\}^* \rightarrow G_1^*$
- H_4**():** hash function, $G_2 \rightarrow \{0,1\}^n$

Data Storage Phase

This phase consists of KGen step generating public key beforehand, and Enc step storing data and keyword in server by encrypting them(see Figure 4).

Data Sharing Phase:

The sharing phase consist of ReKeyGen phase generating re-encryption key with Bob,

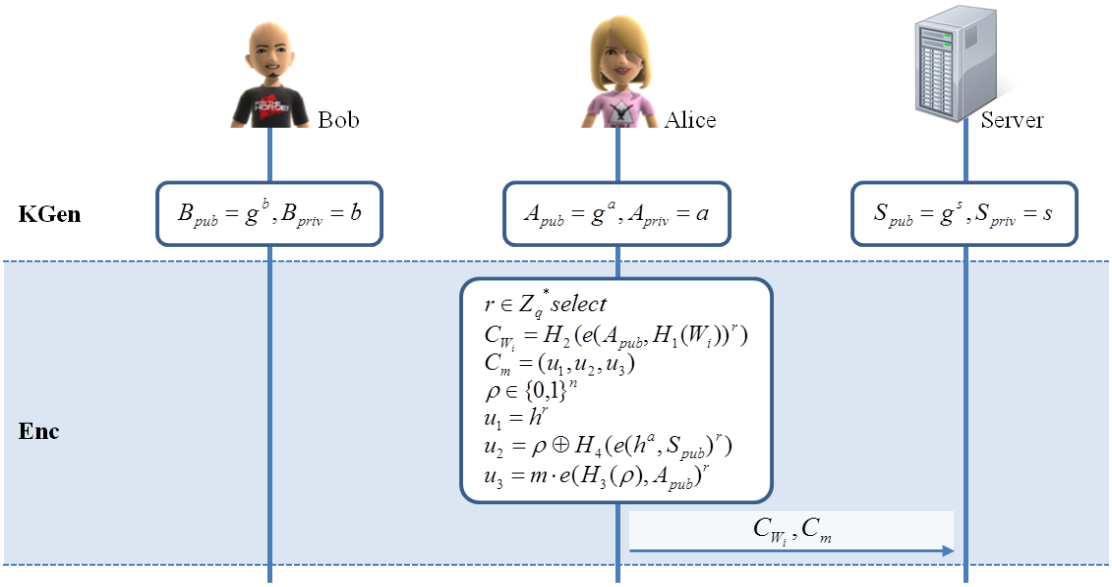

Figure 4. PRPS's Data Storage Phase

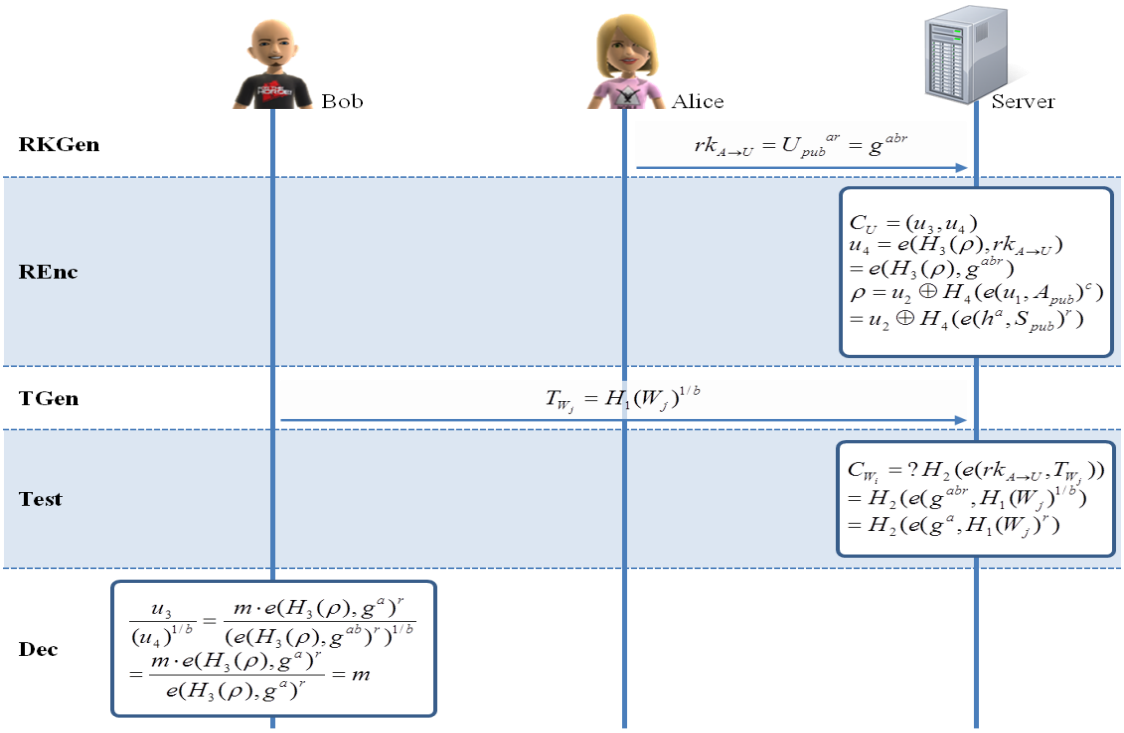

Figure 5. PRPS's Data Storage Phase

sharing target, and Alice, data owner, and ReEnc phase re-encrypting so that Bob also can search Alice's data with re-encryption key. After sharing completed, Bob searches and decrypts sharing data with trapdoor generated by Bob's private key and keyword(see Figure 5).

In this scheme, there is no mention about scheme enabling Alice, data owner, to search her index. Provided that she should use additional schemes to search her data, they are supposed to have index storage space and operation inefficiency. Further, this has a problem of re-encryption that enables Bob to see Alice's other data through collusion of Bob and server. In this scheme, like PRES, the trapdoor has the same pattern to search the same keyword, and data encryption takes long performance time as the method processes data encryption through multiplying operation on paring.

3. Preliminaries

In this section, we provide the necessary preliminary details.

3.1. Bilinear Maps

A bilinear map was proposed originally as a tool for attacking elliptical curve encryption by reducing the problem of discrete algebra on an elliptical curve to the problem of discrete algebra in a finite field, thereby reducing the problem's complexity. However, recently, this scheme has been used as an encryption tool for information protection, instead of an attacking tool. Bilinear pairing is equivalent to a bilinear map. The relevant terms are defined below, and the theory can be described as follows:

Definition 1: Characteristics that satisfy an admissible bilinear map are as follows.

- **Bilinear:** Define a map $e = G \times G \rightarrow G_T$ as bilinear if $e(P^a, P^b) = e(P, Q)^{ab}$ where all P, $Q \in G$, and all $a, b \in Z$.
- **Non-degenerate:** The map does not relate all pairs in $G \times G$ to the identity in G_T. Note that G and G_T *are g*roup**s** of prime order, which implies that if P is a generator of G, $e(P, P)$ is a generator of G_T.
- **Computable:** There is an efficient algorithm to compute $e(P, Q)$ for any $P, Q \in G$. The following definition was constructed on the basis of the bilinear map $e(P^a, Q^b) = e(P, Q^b)^a = e(P^a, Q)^b = e(P, Q)^{ab} = e(P^{ab}, Q) = e(P, Q^{ab})$. With this map, the D-H decision problem can be solved readily for ellipses by using the following equation: $e(P^a, Q^b) = e(P^c, P) \Rightarrow ab = c$. Therefore, the following is the basis for resolving the difficulties of the bilinear map, which is used as an encryption tool by many encryption protocols.

Definition 2: When the elements G, P, P^a, P^b, P^c (BDHP, Bilinear Diffie–Hellman Problem) are given, this relates to the $e(P, P)^{abc}$ calculation problem. In this study, the admissible bilinear map was used as the basis for secret number production during the key construction process between heterogeneous devices. This problem can be solved if the ellipse curve discrete mathematics problem can be solved. For example, the value of a can be calculated from P^a, and thus, $e(P, P)^{abc}$ can be calculated using $e(P^b, P^c)^a$.

3.2. Security Requirement

The following requirements should be satisfied for safe storage and search of data in a remote storage environment.

- **Confidentiality:** Only an authorized individual can identify the communication data between the remote storage and the client's terminal. A trapdoor produced for the data search needs to be created in different forms when the same keyword is searched.
- **Efficiency of Traffic:** For the efficiency of network resource and energy between the client and the server, the wire traffic should be small. In particular, for the use of a disposable trap door, less than three times the wire traffic should be provided. The existing OTP requires less than three times the wire traffic.
- **Efficiency of Operation:** For index production and search, a certain operational efficiency needs to be ensured. In particular, a client who has a limited computing ability should be considered. The efficiency of operations related to encryption, decryption, production of trapdoor, and data sharing conducted by the client needs to be ensured.
- **Efficiency of Sharing: T**he searchable encryption system that provides the existing sharing function allows the primary co-owner to conduct a data search and is inefficient because the encryption index needs to be created again when one wants to share the shared data with another person. Therefore, free sharing irrespective of order should be available.

4. Proposed Scheme

In this paper, a practical proxy re-encryption scheme with a keyword search capability is proposed considering the structural characteristics of an entrusted cloud storage center. This paper describes what steps should be taken in a secure data storage, searching, and sharing phase. Figure. 6 is flowchart of the proposed scheme.

4.1. Notation

The notation used in proposed scheme are as follows.

- *: object (a: Alice, b: Bob, s: Server)
- ||: Concatenation
- p: prime number
- n: amount of data
- m: number of keywords related to the data
- G: cyclic additive group of order p
- G_T: cyclic multiplicative group of order p
- g: generator of G
- e: bilinear map, $G \times G \rightarrow G_T$
- sk_*: *'s private key in Z_p

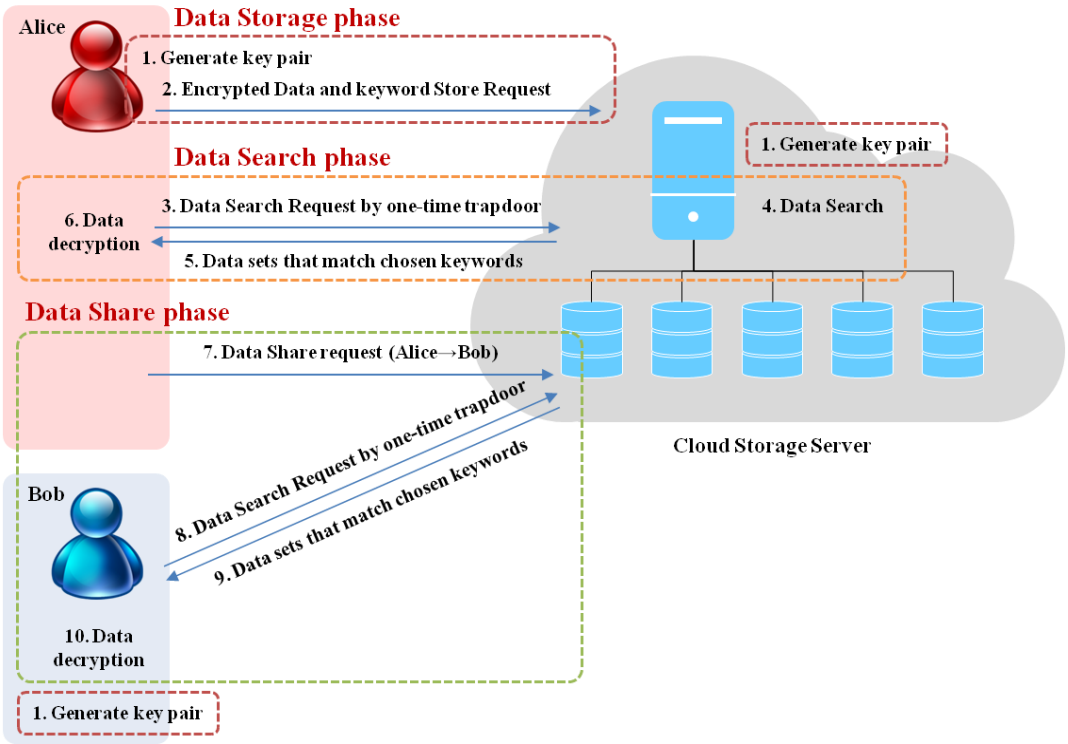

Figure 6. Flowchart of Proposed Scheme

- pk_*: *'s public key in G
- pd_i: i th plain data
- ed_i: i th encrypted data
- k_{i*}: i th data encryption key (i = 1~n)
- $w_{i,j}$: j th keyword on i th data (j = 1~m)
- $Enc_k()$: symmetric key encryption by key k
- $Dec_k()$: symmetric key decryption by key k
- W_i: set of keyword on i th data$_*$
- $H_1()$: hash function, $\{0,1\}^* \rightarrow G$
- $H_2()$: hash function, $G \rightarrow G$
- T_*: trapdoor searching keyword *

4.2. Definition

The detailed steps of the proposed scheme are as follows:

- **KeyGen:** The storage outsourcing users generate public key pairs, which are created prior to the use of the service. The storage outsourcing server should not know the user's private key. If the private key is leaked, the attacker can generate a trapdoor by acting as the owner of the private key. Thus, we generate a key pair based on the Discrete Logarithm Problem (DLP).
- **Enc(sk, W, pd)$\rightarrow E$, ed:** The data owner creates the encrypted index, E, and encrypted data, ed, which only the owner can search by inputting his/her own private key, sk, and set of keywords, W, which are sent to the master server.
- **TGen(sk, w)$\rightarrow T_w$:** To search the data safely, the user creates a trapdoor, T_w, which does not leak information related to the keyword w, which is being searched for by using the private key sk. The trapdoor is sent to the master server. The storage outsourcing administrator should not be able to access information via a trapdoor.
- **Test(E, T_w)$\rightarrow$"yes" or "no":** Using the trapdoor generated by the user's private key and the search keyword, the server performs a test to confirm whether the encrypted data contain the keywords. If the cipher text contains the specified keyword, the server sends a "yes" to the user and a "no" if not. Thus, the server cannot learn anything about the keywords or the data.
- **REnc(sk_a, pk_b, E_a)$\rightarrow E_b$:** The data owner a creates a parameter to generate a data index for sharing that can be searched by b. This parameter is created using the data owner's private key sk_a, and the public key pk_b of the user who will be sharing the data. The master server creates a new index, E_b, which b can use to search via the trapdoor.
- **Dec(sk, E, ed)$\rightarrow$pd:** The rightful owner of the encrypted data uses his/her private key to decrypt the encrypted data.

4.3. Data Storage Phase

The proposed scheme considers the storage outsourcing structure, and therefore, the encrypting index used for sharing and searching is stored on the master server. We assume that each user has received a key pair before using the storage outsourcing service (refer to Step 1). The user encrypts the necessary keywords during data searching, and hence, he/she can perform his/her own search later and send this to the master server (refer to Step 2). The master server sends chunk information to the user for data storage, who then divides the data into chunks and stores it on the designated chunk server (see Figure 7).

Step 1. Key generation (KeyGen): Each storage outsourcing service user generates a key pair.

$x \in Z_p$ selection

$sk = x$ setting up

$pk = g^x$ setting up

Step 2. Index and data encryption (Enc): The data owner generates an encrypted index, which can be used for conducting a secure search.

$A = pk_a^{hk}$ $(hk = H(k))$

$b_i = H_1(w_i)^{-hk}$

$B = \{b_1, b_2, \ldots, b_n\}$

$C = e(g, H_2(pk_a))^{hk} \cdot k$

$D = EC_k(m)$

$E_a = (A, B, C, D)$ output encrypted index and data for the cloud storage

4.4. Data Search Phase

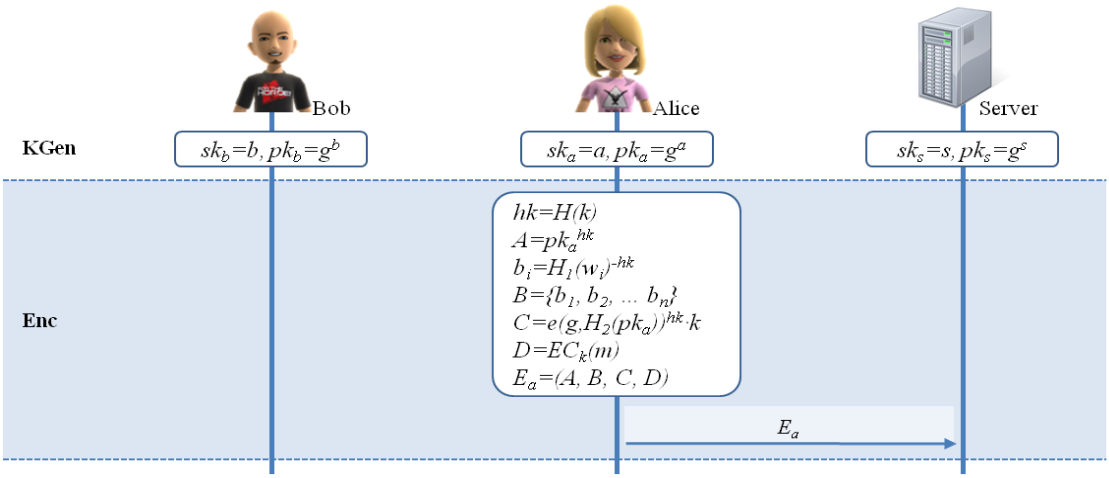

Figure 7. Data Storage Phase

The user sends a trapdoor that can search data without exposing keyword information to the master server (refer to Step 1). The master server searches for the data with the keyword in the encrypted index using the trapdoor and then sends the chunk information that corresponds to the data to the user (refer to Step 2). The retrieved data are decrypted by the legitimate user (refer to Step 3). The user acquires the data by summing each chunk received from the chunk server that stores the data (see Figure 8).

Step 1. Trapdoor generation (TGen): A user, *a*, who wants to search the data generates a trapdoor using the keywords and his/her secret key.

$r \in Z_p^*$

$X=e(g, H_1(w)^{ska \cdot r})$

$T_w=r \,||\, X$

Step 2. Test: To confirm that the data contain the keywords sought by the user, the user performs the following tests with the public key, trapdoor, and crypt obtained from the server.

$X=?e(A^r, b_i)$

$e(g, H_1(w)^{ska \cdot r})=e(pk_a{}^{hk \cdot r}, H_1(w)^{-hk})$

$e(g, H_1(w)^{ska \cdot r})=e(g^{a \cdot -hk \cdot r}, H_1(w)^{-hk})$

$e(g, H_1(w)^{a \cdot r})=e(g, H_1(w)^{a \cdot r})$

Step 3. Decryption (Dec): The user can perform the following decryption using his/her private key and the crypt obtained from the server.

$k=C/e(A, H_2(pk_a)^{-ska})$

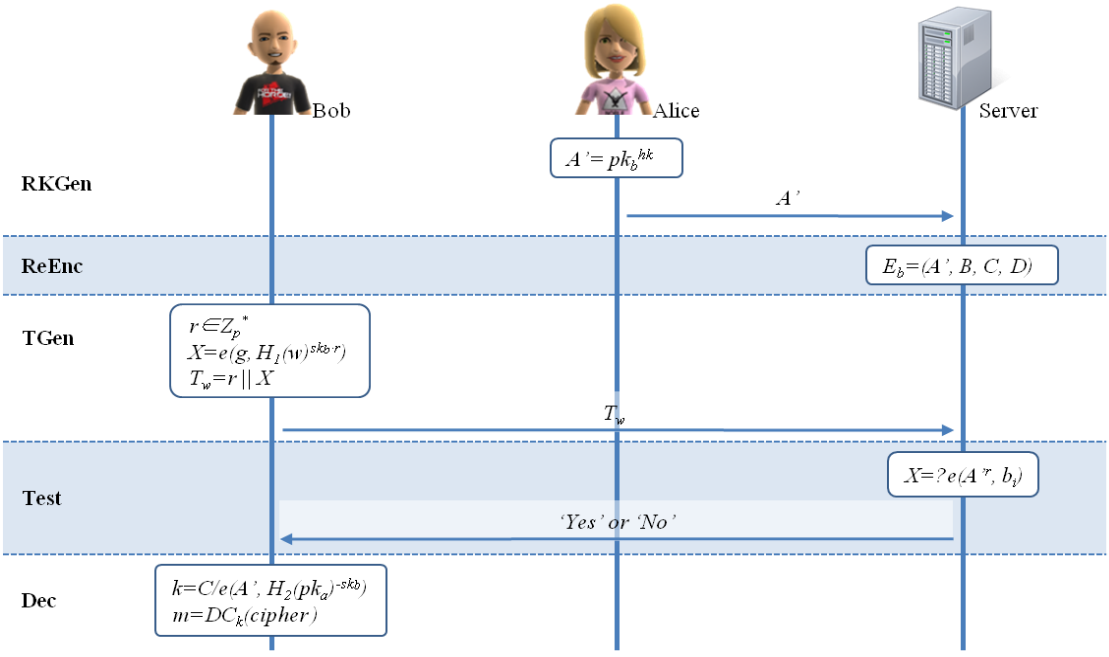

Figure 9. Data sharing Phase

$=C/e(pk_a^{hk}, H_2(pk_a)^{-ska})$

$=C/e(g^{ska \cdot hk}, H_2(pk_a)^{-ska})$

$=C/e(g, H_2(pk_a))^{hk}$: output decryption key

$m=DC_k(D)$: output decrypted data

4.5. Data Sharing Phase

To share data with the desired user and to allow the shared users to share data freely with another user, re-encryption needs to be performed to allow the shared users to search the encrypted index only.

Many parameters are required to implement proxy re-encryption and a separate searchable encryption scheme for secure data sharing in a storage outsourcing environment, which reduces the storage volume efficiency. Therefore, we propose an algorithm that provides both functions simultaneously. First, parameter A is generated to allow index sharing with another user, which is sent to the storage outsourcing provider by the data owner (refer to Step 1). Next, the storage outsourcing provider changes the owner's index with respect to the data sharing target. Shared (re-encrypted) data searching is then possible, as shown in Steps 2–4. A user who acquires the data sharing index can always search for the corresponding data using keywords and download them(see Figure 9).

Step 1. Re-encryption Key Generation (RKGen): If the data owner wants to share his/her data with other users, he/she generates keys for re-encryption. If user *a* wants to share his/her data with user *b*, *a* generates parameter *A′* using *a*'s secret key and *b*'s public key, as follows:

$A' = pk_b^{hk}$

Step 2. Re-encryption (ReEnc): The server make the encrypted data for user *b*.

$E_b = (A', B, C, D)$

Step 3. Trapdoor generation (TGen): The user *b* who wants to search the data generates a trapdoor using the keywords and his/her secret key.

$r \in Z_p^*$

$X=e(g, H_1(w)^{skb \cdot r})$

$T_w=r \,||\, X$

Step 4. Test: To confirm that the data contain the keywords that the user seeks, the server performs the following tests using *b*'s trapdoor. It checks the equality $b_i =? e(A', T_w)$. If this is true, the output is "Yes" but "No" if not.

$X=?e(A''^{r}, b_i)$

$e(g, H_1(w)^{skb \cdot r})=e(pk_b^{hk \cdot r}, H_1(w)^{-hk})$

$e(g, H_1(w)^{skb \cdot r})=e(g^{b \cdot -hk \cdot r}, H_1(w)^{-hk})$

$e(g, H_1(w)^{b \cdot r})=e(g, H_1(w)^{b \cdot r})$

Table 1. Analysis of Proposed Scheme

	PRES [10]	PRPS [11]	Proposed Scheme
Enc (Client)	(n+4)e+(n+1)p+(n+2)h+s+a	(n+4)e+(n+2)p+(2n+2)h+a	(n+2)e+p+(n+2)h+m+b
RKGen (Client)	m	e+m	e
ReEnc (Server)	e+4p+h+s	e+2p+2h	
Tgen (Client)	e+h	e+h	e+p+m
Test (Server)	(n+4)p+(n+1)h+(1)s	p+h	e+np
Dec (Client)	5e+p+h+a +s	e+a	e+p+h+m+b
Free share	O	X	O
Anti collusion	X	X	O

n: number of keyword on document, e: modular exponentiation, p: pairing, h: hash, m: multiplication/division, a: data encryption/decryption using multiplication/division on pairng, b: data encryption/decryption using symmetric cipher, s: sign/verification

Step 5. Decryption (Dec): The user can perform the following decryption with his/her private key.

$k=C/e(A', H_2(pk_a)^{-skb})$

$=C/e(pk_b^{hk}, H_2(pk_a)^{-skb})$

$=C/e(g^{skb \cdot hk}, H_2(pk_a)^{-skb})$

$=C/e(g, H_2(pk_a))^{hk}$: output decryption key

$m=DC_k(D)$: output decrypted data

5. Analysis of Proposed Scheme

The proposed scheme satisfies the following requirements:

- **Confidentiality:** With the proposed system, it is difficult to infer the content of communication when a malicious third person wiretaps the communication between the client and the server by using pairing. Further, it is difficult to infer the keyword and the data content searched by an unethical server manager through a search query when the trapdoor searches the same keyword by using an one-time trapdoor, because a different form of the trapdoor is produced each time.
- **Efficiency of Traffic:** For data sharing, only one round of the communication process is needed.

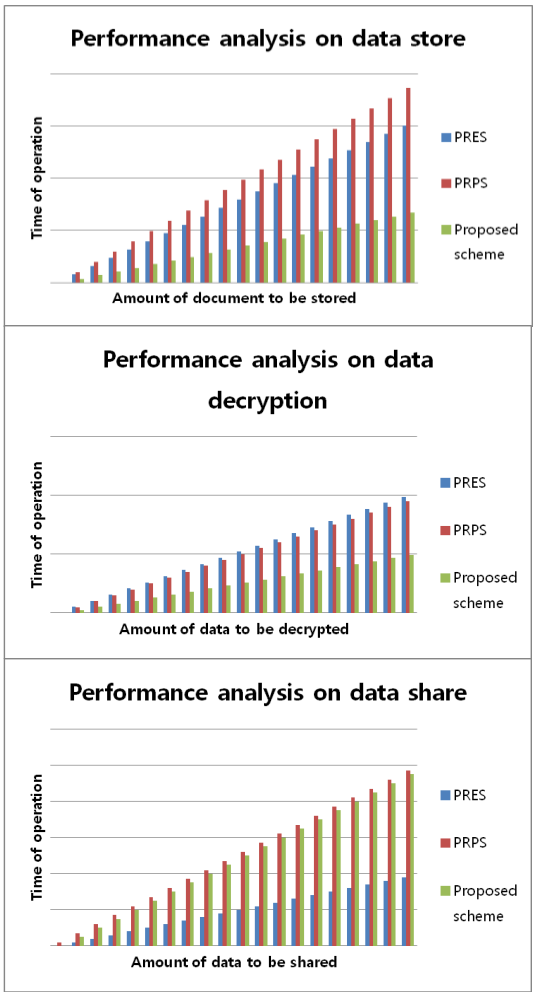

Figure 10. Performance Analysis on Data Management by Client

Since no additional communication occurs when a disposable trap door is produced, this system provides the same efficiency of wire traffic as that of the existing scheme.

- **Efficiency of Operation:** To minimize the amount of client operation, which has a limited computing ability, the proposed system accomplishes an overall efficiency improvement of encryption, decryption, and operation of data sharing, performed by the client(see Figure 10). Unlike the existing system, the proposed system carries out data encryption with a symmetric key, not with a encyption based on pairing, and decreases the time required for operating encryption and decryption by producing the index that manages the relevant key (see Table 1).
- **Efficiency of Sharing:** The process of sharing data is simple. The user who lawfully shares data does not need to produce the index again and can share data with one exponentiation when the user wants to share data with another user.

6. Conclusion

The advent of storage outsourcing services has allowed many users to store and access data remotely. Recent studies on the application of searchable encryption technologies to data storage outsourcing have attempted to ensure the security of these data. However, most available searchable encryption technologies are inefficient when data sharing objects are added because they are based on e-mail environments, which determine the objects with which data can be shared. In a storage outsourcing environment, users upload data on their own and share the data in a safe manner. Therefore, the indexes and data are separated, and hence, the available technologies are compatible with data storage outsourcing systems. After considering the requirements of the data storage outsourcing environment, we specified the security requirements and proposed a scheme that provides both functions simultaneously: a proxy re-encryption function and a searchable encryption function. The proposed scheme provides a free sharing feature, which has the same calculation efficiency as the existing schemes. Search schemes based on multiple keywords are expected to become important for ensuring flexibility and for facilitating search during data storage outsourcing. In the future, it will be necessary to develop a re-encryption system where an index containing multiple keywords with variable lengths can be encrypted and searched flexibly.

Acknowledgments

This research was supported by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program (NIPA-2013-H0301-13-1003) supervised by the NIPA (National IT Industry Promotion Agency).

References

[1] D. Boneh, G. Crescenzo, R. Ostrovsky and G. Persiano, “Public Key Encryption with Keyword Search”, Proceedings of International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, **(2004)** May 2-6.
[2] D. Boneh and B. Waters, “Conjunctive, Subset and Range Queries on Encrypted Data”, Proceedings of the 4th Theory of Cryptography Conference, Amsterdam, Netherlands, **(2007)** Febraury 21-24
[3] Y. H. Hwang and P. J. Lee, “Public key encryption with conjunctive keyword search and its extension to a multi-user system”, Proceeding of First International Conference on Pairing-Based Cryptography, Tokyo, Japan, **(2007)** July 2-4.

[4] F. Bao, R. H. Deng, X. Ding and Y. Yang, "Private Query on Encrypted Data in Multi-User Settings", Proceeding of the 4th international conference on Information security practice and experience, Sydney, Australia, **(2008)** April 21-23.
[5] S. Kamara and K. Lauter, "Cryptographic Cloud Storage", Proceedings of Workshops on Financial Cryptography and Data Security, Canary Islands, Spain, **(2010)** January 25-28.
[6] M. Ion, G. Russello and B. Crispo, "Enforcing Multi-user Access Policies to Encrypted Cloud Databases", International Symposium on Policies for Distributed Systems and Networks, Trento, Italy, **(2011)** June 6-8.
[7] B. Zhang and F. Zhang, "An efficient public key encryption with conjunctive-subset keywords search", Journal of Network and Computer Applications, vol. 34, no. 1, **(2011)**.
[8] Y. Yang, "Towards Multi-user Private Keyword Search for Cloud Computing", Proceeding of International Conference on Cloud Computing, Singapore, Singapore, **(2011)** July 4-9.
[9] S.-H. Lee and I.-Y. Lee, "Secure Index Management Scheme on Cloud Storage Environment", International Journal of Security and Its Applications, vol. 6, no. 3, **(2012)**.
[10] J. Shao, Z. Cao, X. Liang and H. Lin, "Proxy re-encryption with keyword search", Information Sciences, vol. 180, **(2010)**.
[11] X. Chen and Y. Li, "Efficient Proxy Re-Encryption with Private Keyword Searching in Untrusted Storage", International Journal of Computer Network and Information Security, vol. 3, no. 2, **(2011)**.

Authors

Sun-Ho Lee, received his B.S. and M.S. degrees from Department of Computer Software Engineering, Soonchunhyang University, Korea, in 2009 and 2011, respectively. He is now a Ph.D. candidate in Department of Computer Software Engineering, Soonchunhyang University, Korea. His research interests include Searchable encryption, Secure USB flash drive, and Cloud computing Security.

Im-Yeong Lee, is the corresponding author. He received his B.S. degree from Department of Electronic Engineering, Hongik University, Korea, in 1981 and his M.S. and Ph.D. degrees from Department of Communication Engineering, Osaka University, Japan, in 1986 and 1989, respectively. From 1989 to 1994, he was a senior researcher at ETRI (Electronics and Telecommunications Research Institute), Korea. Presently, he is a professor in Department of Computer Software Engineering, Soonchunhyang University, Korea. His research interests include Cryptography, Information theory, and Computer & network security.