Secure Index Management Scheme on Cloud Storage Environment

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

Dept. of Computer Software Engineering, Soonchunhyang University, Korea
[1]Dept. of Computer Software Engineering, Soonchunhyang University, Korea
sunho431@sch.ac.kr, [1]imylee@sch.ac.kr

Abstract

Keeping pace with the increase of digital information in use, Cloud storage is in service, which can store one's data from distance through network and various devices and easy to access. Unlike the existing removable storage necessary in order to carry data, it is used many users because it has no limit of memory capacity and no need to carry storage medium. As many users save a great volume of date in Cloud storage, its reliability has become a focus of issue. To protect it from unethical managers and attackers, researches are being conducted on application of a variety of cryptography systems such as searchable encryption and proxy re-encryption to Cloud storage system. However, existing searchable encryption technology is inconvenient in the cloud storage environment in which the user uploads data in person, and those data are shared with others, whenever it is necessary to do, and those with whom data are shared change frequently. In this paper, we propose a searchable re-encryption scheme by which user can share data with others safely by generating searchable encryption index, and re-encrypting it.

Keywords: *Searchable encryption, Proxy re-encryption, Cloud computing, Storage*

1. Introduction

As the volume of digital information has rapidly expanded, storage medium has also developed rapidly to store data. Particularly for mobile storage that can enable us to carry data, tape drivers appeared first in 1951. Since then it has developed to floppy disks, optical media, and flash memory cars and now to USB flash drivers. Because such mobile storage media are easy to carry, it has high risk of loss or theft, which may lead to the discloser of personal information saved in the media. However, those media, because of portability, are also in danger of being stolen and lost, causing data in them to be leaked outside. As the development of network makes it possible for data communication to speed up, cloud computing service made its appearance that can store own data in distant storage and retrieve them to one's own device to have access to them. Recently many companies are providing free storage service of high capacity competitively. Accordingly, more and more people are currently using Cloud storage service t save their data in it. Like this, storing many users' data in the system increases the possibility of 'big brother problem' and risk of disclosure by data attackers and unethical managers.

Data encryption may be one of the measures to tackle such problems, but it has its own hassle of making access to data hard. Therefore, searchable encryption system appeared that can encrypt the indexes of data and allow searching of the indexes without having data information be exposed to attackers and unethical managers [1-11].

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

However, this method is not applicable to Cloud environment where data sharing is frequent among users because of encrypted indexes. Subsequently, searchable re-encryption system entered that re-encrypt encrypted indexes to allow users to search data to be shared without decoding process for safe data sharing in Cloud storage [12]. However the existing systems do not place in consideration the case where those who share data share them with other users and the storage structure of Cloud, so that they handle indexes and data encryption in a single process. Actually Cloud storage system has separate server systems: master server that stores indexes and data information and a server to store data. Therefore, searchable re-encryption system is difficult to be applied to Cloud. Accordingly this study tries to propose a technical measure to allow safe sharing of Cloud users' data, considering Cloud storage structure.

2. Preliminaries

2.1. Distributed File System

Cloud computing is the computing style providing IT related function with service form. Cloud computing is largely divided into 3 classes such as SaaS(Software as a Service), PaaS(Platform as a Service), IaaS(Infrastructure as a Service). In order to provide data safety stored in cloud, at first, safe storing place in IaaS class should be provided. The Service providing service place in cloud computing environment is called cloud storage service. Base technology to provide this cloud storage service is distributed file system.

Let's look into GFS and HDFS which are mostly used in distributed file system.

2.1.1. GFS: GFS(Google File System) is the distributed file system made to provide cloud service in Google. GFS is consisted of client, master server and chunk server, and roles of each object are as follows. Client: this provides self interface similar to file system interface and communicate with master server and chunk server on behalf of application program[13].

Master server: this manages meta-data of file system such as name space, access control information, mapping information between file and chunk, chunk location information, etc. These meta-data are stored in the memory of master server and quickly inform the location of data to client. Also, they control overall system operation such as creating chunk copy, adjusting number of copies, returning unused store space, chunk server health check, etc.

Chunk server: chunk server manages chunk which is stored data unit and supports input and output of data requested by client. Chunk server regularly reports Heartbeat message to master server. Also, this detects data error using checksum and deletes error detected chunk. How GFS is operated can be fully supposed by component role of prior GFS. When storing file, client sends file information to be stored by own to master server and master server sends chunk server location and handle of actually storing file to client. Afterward, the client divides own data into chunk with fixed size. And then it sends divided chunk to chunk server. When reading file, client searches own data in master server and receives chunk server location where these data are stored. Afterward, it receives chunk through communication with chunk server and can have original data by summing these.

2.1.2. ANALYSIS OF DISTRIBUTED FILE SYSTEM. Other many distributed file systems are used and their structures are not far different. Here, we should keep a close eye on the structure separating control and data store for fast data process by distributed file

system. In order to provide safe storing place in cloud storage using distributed file system, data and encoded index for data search should be separately stored.

2.2. Bilinear Pairing

Cloud computing The bilinear map was originally suggested as a tool to attack elliptical curve encryption, by reducing the problem of discrete algebra on elliptical curve into the problem of discrete algebra on finite field, and thus reducing the difficulty of it. However, it began to be used recently not as an attacking tool, but as an encryption tool for information protection. Bilinear pairing is equivalent to a bilinear map. The following terms are used, as stated in this paragraph, and this theory is defined below.

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

Bilinear: Define a map $e = G_1 \times G_1 \rightarrow G_2$ as bilinear if $e(aP, bP) = e(P,Q)^{ab}$ where all $P,Q \in G1$, and all $a, b \in Z$.

Non-degenerate: The map does not relate all pairs in $G_1 \times G_1$ to the identity in G_2. Observe that since G_1 and G_2 are groups of prime order, this implies that if P is a generator of G1, then e(P,P) is a generator of G_2.

Computable: There is an efficient algorithm to compute e(P,Q) for any P , $Q \in G_1$. The following definition was constructed based on the bilinear map $e(aP, bQ) = e(P,bQ)^a = e(aP,Q)^b = e(P,Q)^{ab} = e(abP,Q) = e(P, abQ)$. From this map, for ellipses, the D-H decision problem can be easily solved using the following equation. $e(aP, bQ) = e(cP,P) \Rightarrow ab = c$. Therefore, the following is the basis for resolving the difficulties of the bilinear map used as an encryption tool by many encryption protocols.

Definition 2 When elements G_1,P, aP,bP, cP (BDHP, Bilinear Diffie–Hellman Problem) are given, this refers to $e(P,P)^{abc}$ calculation problem. In this research, the admissible bilinear map was used as the basis of the secret numbers production in the key construction process between heterogeneous devices. This problem can be solved, if the ellipse curve discrete mathematics problem can be solved. For example, a can be calculated from aP , then $e(P,P)^{abc}$ can be calculated through $e(bP, cP)^a$.

2.3. Requirements

The following requirements should be met for safe search and sharing to be secured under Cloud storage environment.

Confidentiality: Data transmitted between remote data server and client terminal should be identifiable only by proper persons.

Search speed: The client who has limited system resources should be able to quickly search documents including word files from documents stored in cloud storage systems.

Traffic efficiency: Communication volume should be small for the energy efficiency between client and server, and efficiency of network resources.

Calculation efficiency: Calculation efficiency should be provided for index generation and execution of search, and for sharing data with other users safely.

Sharing efficiency among users: it must make encrypted data saved in distant data be protected and shared to those users who share them safely and efficiently from an unreliable server.

3. Definition

In this clause, we consider structural characteristics of distributed file system analyzed beforehand and define by what scenarios our schemes are operated and what roles each step takes of for satisfying requirements.

3.1. Writing Scenario

In suggested method considering cloud storage structure, encode index possible for sharing and search is stored in master server. User encodes keyword necessary at data search later to be able to search by oneself only and sends this to master server. Master server sends chunk information for data storage to user and user divides data into chunks and stores in designated chunk server.

3.2. Reading Scenario

User sends trapdoor which is able to search data without exposing keyword information to master server. Master server searches data having keyword by using trapdoor in encoded index. And then it sends chunk information having corresponding data to user. User acquires data by summing these after receiving each chunk from chunk server where is storing data.

3.3. Sharing Scenario

In order to share data with desired user and in order for shared user to freely share data with another user, re-encryption should be done for shared user to be able to search encoded index only. The user acquired index of sharing data can always search corresponding data by keyword and download them.

4. Proposed Scheme

The proposed scheme defined above satisfies requirements by performing detailed calculation as follows.

4.1. System Parameters

p: prime number
G: Cyclic additive group of order p
g: generator of G
e: bilinear map, $G \times G \rightarrow G_T$
EC_k(): symmetric ecryptiony by key k
DC_k(): symmetric decryptiony by key k
K_d: symmetric key k for data encryption
d: data for encryption
c_i: i_{th} chungk of data

e_i: i_{th} encrypted chungk of data
w: keyword
m: plain data
sk_*: *'s secret key
pk_*: *'s public key
w_*: $*_{th}$ keyword of data
H_1( ): hash function, $\{0,1\}^* \rightarrow G$
H_2( ): hash function, $\{0,1\}^* \rightarrow G$
H_3( ): hash function, $G_T \rightarrow \{0,1\}^*$
T_*: trapdoor searching keyword *
$rk_{a \rightarrow b}$: re-encryption key changing A's crypt to B's crypt

4.2. KeyGen

TA generates a pair of keys and sends them safely to cloud storage user.
$x \in Z_q$ selection
sk=x setting up
$pk=g^x$ setting up

4.3. Enc(sk_a, pk_a, w)

Data owner A generates the cipher-text which can be used for secure search
$A=pk_a^r$ ($r \in Z_p$)
$B=e(g,g)^{sk_a \cdot r}$
$C=H_3(e(g,H_1(w))r$
$D=e(g, H2(sk))^r \cdot K_D$
E_a=(A, B, C, D) output as encrypted index
$d=\{c_1, c_2, ..., c_l\}$
$e=E_{Kd}(C_i)$ (i=1~l)

4.4. ReKeyGen(sk_a, pk_b)

When the data owner wants to share his data with other users, he generates keys for re-encryption. When user A wants to share his data with user B, A generates re-encryption key using A's secret key and B's public key as follows.
$rk_{a \rightarrow b} = pk_b^{-sk_a} \bmod p$

4.5. ReEnc($rk_{a \rightarrow b}$, E_a)$\rightarrow E_b$

The cloud storage service server, with re-encryption key inputted by the user, the target crypt intended to be re-encrypted, and public key, performs re-encryption as follows.
$A' = A^{sk_b/sk_a}$
$B' = e(A, rk_{a \rightarrow b})$
$E_b = (A', B', C, D)$

4.6. TrapdoorGen(sk_b, w)

The user wanting to search the data generates trapdoor with keywords and his secret key.
$T_w=H_1(w)^{-sk_b}$

4.7. Test(E, T_w)→'yes' or 'no'

To confirm whether the data contain the keywords he intends to find, the user performs the following tests, by his public key, trapdoor, and crypt inputted from the server.

C_i =? $H_2(e(A', T_w))$

4.7. Dec

Index search for legitimate users to decrypt the data as follows.

$K_d = D/e(A', H_2(sk))^{-sk}$

$c_i = D_{Kd}(e_i)(i=1\sim l)$

$d_i = \{c_1, c_2, ..., c_l\}$

5. Analysis

The proposed method satisfies the following requirements.

Confidentiality: By using pairing, the proposed method makes it difficult for a vicious third party to decode communication contents even if he bugs communication between client and server.

Search speed: By doing single pairing calculation and hash calculation, user can check whether the document contains keywords. The method provides quick search speed.

Traffic efficiency: Since keyword search and re-encryption need only one round of communication process, the method provides efficiency in communication volume.

Calculation efficiency: Based on lighter pairing calculation, the method allows user to generate index, search documents, and do re-encryption, providing calculation efficiency

Sharing efficiency among users: By re-decoding them, it must make encrypted and saved in unreliable distant data server be shared safely and efficiently regardless of time of use.

6. Conclusion

With the advent of cloud storage service, many users can store and get access to data by using it. To secure the security of data stored in such a storage place, researches designed to apply searchable encryption technology to cloud storage have begun recently. However, most of existing researches have problems in the sense that, since they are based on e-mail environment, and, thus decide objects with which data can be shared, they become inefficient in adding more objects to share data. In Cloud storage environment, users upload date to use by themselves. And they share them in a safe manner that they want. Therefore, data information like indexes and data are separated, so that the existing methods are hard to be compatible with Cloud storage system. Therefore, considering such requirements in the cloud storage environment, we set up security requirements, and proposed a method of providing the two functions

simultaneously – Proxy Re-encryption function and searchable encryption function. This method provides efficiency in terms of calculation volume.

To make it flexible and easy to search data in cloud storage, it seems that the search method using multiple keywords will become an important issue. Therefore, it will be necessary to do research in the future on a re-encryption system where the index composed of multiple keywords with variable length can be encrypted, and can be searched flexibly.

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2010-0022607).

References

[1] D. X. Song, D. Wagner and A. Perrig, "Practical Techniques for Searching on Encrypted Data", Symposium on Security and Privacy, **(2000)** May 14-17; California, USA.
[2] E. J. Goh, "Secure Indexes", ePrint Crpytography Archive, **(2004)**.
[3] R. Curtmola, J. Garay, S. Kamara and R. Ostrovsky, "Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions", Proceedings of the 13th ACM conference on Computer and communications security, **(2006)** Oct 30-Nov 3; Virginia, USA.
[4] 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, **(2004)** May 2-6; Interlaken, Switzerland.
[5] D. Boneh and B. Waters, "Conjunctive, Subset and Range Queries on Encrypted Data", Proceedings of the 4th Theory of Cryptography Conference, **(2007)** Febraury 21-24; Amsterdam, Netherlands.
[6] 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, **(2007)** July 2-4; Tokyo, Japan.
[7] 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, **(2008)** April 21-23; Sydney, Australia.
[8] S. Kamara and K. Lauter, "Cryptographic Cloud Storage. Proceedings of Workshops on Financial Cryptography and Data Security", **(2010)** January 25-28; Canary Islands, Spain.
[9] 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, **(2011)** June 6-8; Trento, Italy.
[10] 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)**.
[11] Y. Yang, "Towards Multi-user Private Keyword Search for Cloud Computing", Proceeding of International Conference on Cloud Computing, **(2011)** July 4-9; Singapore, Singapore.
[12] X. Chen and Y. Li, "Efficient Proxy Re-encryption with Private Keyword Searching in Untrusted Storage", I.J. Computer Network and Information Security, vol. 3, no. 2, **(2011)**.
[13] S. Ghemawat, H. Gobioff and S. Leung, "The Google File System", Proceedings of the nineteenth ACM symposium on Operating systems principles, **(2003)** December 5; Newyork,, USA.
[14] D. Borthakur, "The Hadoop Distributed File Aystem: Architecture and Design", http://svn.apache.org/repos/asf/hadoop/common/tags/release-0.16.1/docs/hdfs_design.pdf, **(2007)**.

Authors

Sun-Ho Lee

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

Im-Yeong Lee

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