An Efficient Scheme of Multi-user Searchable Encryption with Keyword in Cloud Storage

Zhiling He[1],Baomin Wang[2],Wenjun Luo[3]
Chongqing University of Posts and Telecommunications, Chongqing 400065, China
zhilingholo@163.com,luowj@cqupt.edu.cn

***Abstract*—Cloud storage services allow the users to outsource their data to the cloud storage servers to reduce management costs. The cloud servers cannot be fully trusted while those data may contain sensitive personal information. We study a problem of storing and retrieving private data encrypted in a cloud. Multi-user searchable encryption is a promising way to keep user's data confidential against un-trusted cloud service provider and allow users to share their data in privacy. In this paper we propose an efficient, secure and privacy preserving keyword search scheme which supports multiple users with more flexible key management and lower computation cost rather than known approaches.**

Keywords-cloud storage,multi-user, searchable encryption

I. INTRODUCTION

Cloud storage,which dynamically provides reliable services over the Internet, is considered as the most clear development direction of the storage industry. Thus, more and more sensitive information are being centralized to the cloud. However, the cloud customers may worry about their private data would be abused without their permission among others[1]. Encryption is deemed a good approach to protect the confidentiality of users's data. Searchable encryption can enable cloud consumers to use keyword based on searches upon an encrypted database while without revealing any private information to cloud servers. However, the efficient retrieval of encrypted data in un-trusted environment is quite a difficult work. To search over encrypted data securely, searchable encryption techniques have been purposed in recent years[5]. It allows users to preprocess the data and generate an encrypted index for the data. Later, after sending the encrypted data and index to the cloud server, the user may generate a trapdoor and send the trapdoor to the cloud server performing searching request. During the whole process, the cloud sever learns nothing about the keyword related to the trapdoor and the content of the data, but it still be able to return the correct results.

Searchable encryption can meet both the security and operability of the outsourced data. In primitive research, some schemes of single-user searchable encryption have been proposed. These schemes are only applied to the circumstance that the user retrieve his own encrypted data stored in the cloud in advance which means only the data owner can search the data. Multi-user searchable encryption which enable users to share private information with others is more practical. However, a vast number of calculations to be performed are deemed the biggest bottleneck in existing schemes. In this paper, we purpose an efficient multi-user searchable encryption scheme, which processes the following features.

- It support distinct trapdoors. Each authorized user has a distinct key to generate trapdoors for constructing search queries. The keys of one user can easily be revoked without affecting others or the encrypted data on the cloud server.
- In the scheme, a master who manages the users and initial the whole system will also participate in the process of retrieving data, thus reducing the computing burden of the cloud server.
- User authorization is provided. The shared data set can be updated by the authorized users and each user in the group can be both reader and writer.

The rest of this paper is structured as follows: In the next section, we will discuss the related works. We describe our system model and definitions in Section 3. We give our construction and the analyses of our scheme in Section 4.Finally we illustrate our future work and give a conclusion of this paper in Section 5.

II. RELATED WORKS

For providing efficient keyword search and guaranteeing privacy of user data, Song et al [6] proposed the first searchable encryption scheme in 2000. Their scheme is a method of scanning the cipher text. Each search will traverse the whole cipher text, which will lead to a lot of computation and consume a lot of time. Chang et al [10] proposed the multi-user searchable encryption to be an open problem. Curtmola et al [11] analyzed these definitions and gave an adaptively security definition of symmetric searchable encryption. Curtmola et al also give a security definition of multi-user searchable encryption in [11]. And they give a multi-user searchable encryption scheme by combining their symmetric searchable encryption with broadcast encryption. Besides, they use the broadcast encryption to control which user could communicate with the server; the rest part of their scheme is a symmetric searchable encryption. Dong et al [12,13] gave two multi-user searchable encryption schemes. The first of their schemes is based on the RSA proxy encryption. The second is based on the Elgamal proxy encryption.

As a final note, the scheme proposed in this paper can be viewed as an adaptation of our earlier work [16]. In our earlier work, we note that the data will be re-encrypted by the proxy every time a user updates his data in their schemes. In practical, a user operates on his own data frequently which will lead to a lot of unnecessary computation. We constructed a scheme based on the two proposed schemes of dong et al[13]. With the deepening of the research, we notice that there is an important limitation. When the user requires to search private information, the cloud server need to retrieve and compute every piece of data in the database. The user has to search for his private data among mass data for a long time. In this paper, we purpose that a master, regarded as another trusted server, who manages users and initial the whole system will also participate in the process of retrieving data in order to reduce the computing burden of cloud servers. Besides, we consider the cloud server is honest but curious and it will not collaborate with users. It can be proved that our scheme will reduce the overhead of the cloud server and improve the efficiency of encrypted searching.

III. PRELIMINARIES

The system model of the proposed method is as follows:

- Data: We use *D* to denote the data stored in the cloud.
- Users: Authorized users are able to update and search encrypted data residing on the cloud storage server. After being revoked, the user is no longer able to access the data. We use *U* to represent a group of users.
- Cloud server: The main responsibility of the cloud server is to store and retrieve the encrypted data for users.We use *CS* to stand for a cloud server.
- Key and user management server(*KUMS*): *KUMS*, a fully trusted server, represents a master who will initial the whole system. It is responsible for generating and distributing key sets for each authorized user. KUMS can look up all the key sets of the users and revoke an un-trusted user's permission by revoking his keys.

We assume that the data *D* consists of *m* records $\{d_1,d_2\ldots d_m\}$. The keyword of d_i is denoted as $d_i.w$, the cloud server *CS* hosts the encrypted version of *D*, denoted by $D' = \{ d'_1, d'_2, \ldots, d'_m \}$, where $d'_i = \{E(d_i), I(d_i.w)\}$:The first part is an encryption of d_i and the second part is the output of an index generation function *I( )* on $d_i.w$. Let $E_D=\{E(d_1),E(d_2)\ldots,E(d_m)\}$, and $I_D=\{I(d_1.w), I(d_2.w),\ldots, I(d_m.w)\}$.

Authorized users are fully trusted. With the assistance of *CS*, an authorized user $u \in U$ is given permission to access the shared data stored on the cloud server. We use $q_u(w)$ to denote the query of keyword *w* from user *u*. When the user *u* requires to search the private information, $q_u(w)$ will be sent to *KUMS* first. On receiving a query $q_u(w)$, what *KUMS* should do is to verify the user *u*'s identity and send different feedback information to the user *u* and the cloud server denoted by M_1 and M_2. When the user *u* receives M_1, he will use it to generate a trapdoor, denoted by $T_u(w)$ for data search. Then, *CS* will use the feedback information from *KUMS* and the trapdoor $T_u(w)$ to perform the search and return the result $a_q = \{E(d_i) | d_i \in D, d_i.w = w\}$. Among the whole process, *CS* is modeled as "honest but curious" and the server will not collaborate with the users.

IV. OUR CONSTRUCTION

A. The existed schemes

In this section, we will present our new construction. Before that, we introduce our earlier work [16] based on protocol [13].

The main idea of our earlier work is to separate a user's key K into two parts: K_u and K_s, which satisfy that $K_u * K_s = K$ ($*$ represents one kind of operation: like addition or multiply). The master distributes the master key K_u to user u and distribute the assistant key K_s to the cloud server *CS*. Different user has different K. They can apply for join a similar secret group. When a user u wants to insert the data item, he needs to encrypt his data item d_u and the index $I(d_u \cdot w)$ with his key K_u. Then he uploads the encrypted data item and index to the cloud server. After receiving the encrypted data item from u, the cloud server records the information encrypted by whom rather than re-encrypts the data immediately. The data will be re-encrypted when the data is required, not every time the user sends the data to the cloud server. When a user u' (or u himself) wants to search the data, he need to submit a trapdoor $T_u(w')$ to the cloud server. After receiving the trapdoor, the cloud server can use u''s trapdoor $T_u(w')$ and his assistant key to search the data. If it gets a successful matching, the cloud server can transfer the re-encrypted data item to the form which is able to be pre-decrypted by user u'. After the pre-decrypting, the data is encrypted under the key of u'. Then the user u' could use his user key to decrypt the data received from the server. As we study further, we note that the cloud server need to retrieve and compute every piece of data in the database when it gets the user's trapdoor. In that situation, the cloud server will have a tremendous burden if the data is mass or the amount of requirements increase frequently. We can find that much of complex computing can be saved by executing our new scheme

B. The details of our scheme

Our new scheme can be consisted by the following algorithms:

- Setup(1^k): Input a security parameter k, output the system parameters para=($G, g, q, f, H, h, g^a h^a$), in which G is a cyclic group with order q; g is a generator of G; f is a pseudo-random function; H is a collision-free hash function; $h = g^R$,where $R \in_R Z_q^*$; selects $a \in Z_q^*$, $x \in Z_q^*$ randomly. Setting para to be public and keeping $S_k = (a, x, s)$ to be private.
- Authorization (K_{KUMS}, *u*): *KUMS* generate a user key $x_{u_1} \in Z_q^*$ for $u \in \mathrm{U}$, computes the complementary key $x_{u_2} = x - x_{u_1}$, then computes $a_1 \cdot a_2 \equiv a \pmod q$. Sending (x_{u_1}, a_1) to u, (u, x_{u_2}, a_2) to the cloud

server *CS* as the complementary key corresponding to user *u*.

- Encryption (x_{u_1} , d , $d.w$):User u computes $E(d)=(g^r, g^{rx_{u1}}d)$, where $r \in_R Z_q^*$. For the keyword $d.w$, u computes $I(d.w)=(g^{\sigma})^{a_1}$. Then u sends $d'=\{E(d), I(d.w)\}$ to *CS*.
- Re-encryption (u , x_{u_2} , d'): When *CS* has received the data d' , *CS* will first find the user u's complementary key x_{u2}, store data item(u, x_{u2}, d''), which $d''=\{E''(d), I''(d.w)\}$, $E''(d)=E(d)$, $I''(d.w)=H(I(d.w)^{a_2})$.
- Pre-searching($q_u(w)$): If a user *u* wants to search the data, *u* submits a requirement $q_u(w)$ to *KUMS* first. *KUMS* computes $M_1=(gh)^{-r}$, $M_2=(g^a h^a)^r$, sends feedback information M_1 to the user *u*, M_2 to *CS*.
- Trapdoor (x_{u_1} , w', M_1):After receiving the data M_1 , the user u computes a trapdoor $T=T_u(w')^{a_2}=(g^{\sigma_{w'}-r}h^{-r})^{a_1 a_2}=(g^{\sigma_{w'}-r}h^{-r})^a$, in which $\sigma_{w'}=f_s(w')$. Then the user *u* sends $T_u(w')$ to *CS* as his search request of w' .
- Search (M_2, $T_u(w')$, *I*): Upon receiving the search request from *u* and feedback information M_2 , *CS* sets $a_q=\varnothing$, and then *CS* finds the corresponding complementary key of u and computes $T=T_u(w')^{a_2}=(g^{\sigma_{w'}-r}h^{-r})^{a_1 a_2}=(g^{\sigma_{w'}-r}h^{-r})^a$. For each data item d_i , *CS* computes $H(T\cdot M_2)$ and compares the result with $I''(d.w)$, if they are equal, then adds the corresponding data item to a_q . *CS* finds the destiny of a_q , suppose the destiny to be u_d . For each data item in a_q , recall that each data item is formed like { u, x_{u_2}, d'' }, *CS* compares u_d with u , if $u=u_d$,writes the data item back to a_q ; if $u \neq u_d$, *CS* computes $\psi = x_{u_2}-x_{d_2}$, where x_{d_2} is the complementary key of u_d , *CS* can compute $g^{r\psi}g^{rx_{u1}}d = g^{r(x_{u1}+x_{u_2}-x_{d2})}d = g^{rx_{d1}}d$, then writes $\{g^r, g^{rx_{d1}}d\}$ back to a_q . Finally, *CS* sends a_q to u_d . The user u_d could use her key to compute $(g^r)^{-x_{u1}}\cdot g^{rx_{u1}}d=d$.
- Revoke (u): *DO* sends a message to tell *CS* to revoke user u . *CS* deletes the complementary key of u .

C. *The analysis of our scheme*

Security analysis: Comparing with our earlier scheme, we add another public parameter named *h*. The discrete logarithm problem (DLP) for a cyclic group *G* has told us as follows:

Given $g \in G$, it is hard to find a integer x such that $g^x = a$, where a is a random number.

Based on that theory, with knowing g, h , the other users and *CS* can not get valid information through $(g^{\sigma})^{a_1}$, $(gh)^{-m}$, $(g^a h^a)^m$, $(g^{\sigma_{\omega'}-m}h^{-m})^{a_1}$, $(g^{\sigma_{\omega'}-m}h^{-m})^a$. In our scheme, the function f is known to the entire participant. So, σ is public computable. We can take g^{σ} as a generator of the cyclic group, and the private key a_1 as the x in the DLP. As we known, in the present situation, there has not an efficient way to solve the DLP.

In our scheme, the generation of user's trapdoor need the feedback information M_1 from *KUMS*. We can see $M_1=(gh)^{-r}$. Because *r* is selected randomly, so the user's trapdoor $T=(g^{\sigma_{w'}-r}h^{-r})^a$ is totally random. Thus, a user can use different trapdoor to search the data every time. It can efficiently resist attack.

Another aspect is the same as our earlier scheme. According to our definition, the cloud server will not collaborate with the malicious users. What *CS* needs to convert the cipher text is only the complementary keys of users. *CS* knows $x_{u_1}+x_{u_2}=x_{d_1}+x_{d_2}=x$, but it can't deduce x_{u_2}, x_{d_2} or x by its knowledge of x_{u_1}, x_{d_1} . It's just like to solve the equation $x_{d_1}+x_{d_2}=x$ with knowing x_{d_2} . As we can see, there will be countless solutions to the equation. So, the probability for *CS* to deduce the user key is negligible. But *CS* can get the convert key by computing $x_{u_2}-x_{d_2}$, and finish the process of converting the cipher text alone.

In a conclusion, our scheme is secure if there is no an efficient way to break the DLP. And in the process of dealing the cipher text, the cloud server cannot get any knowledge of the user key. Further more, *CS* can know nothing about user's secret information stored on it.

Performance analysis: Suppose the modulus we use is *K*-bit length, and the output of the hash function *H* is *L* bit. Then we can get the following results in Fig.1.

TABLE I. THE COMPARISON OF FOUR SCHEMES

	Length of cipher text	***Length of index***	***Trapdoor length***
Our scheme	2K	K	K
[16]	2K	K+L	2K
[13]	2K	2K+L	2K+L
[12]	K	K+L	K

From the Fig.1, we can see that our new scheme gets the better performance than our earlier work. Besides, we can easily figure out that the storage cost and communication overhead of our new scheme is less than the scheme of [13] but a little more than the scheme of [12]. That's because in our scheme and [13] take the Elgamal encryption as our data encryption scheme. Elgamal encryption scheme is a probabilistic encryption scheme, while RSA encryption is a deterministic encryption scheme. Using the Elgamal encryption to encrypt the data, the cloud server can't distinguish two data items. That provides a better security for the privacy of users.

Moreover, in our earlier scheme[16], every piece of data item should be computed during the process of search leading to a great burden to the cloud server. In our new scheme, we just need to execute a process of data-matching on the cloud server dealing with a user's requirement for search. So, our scheme could be more efficient in practical by saving a lot of unnecessary computation.

V. CONCLUSION AND FUTURE WORK

In this paper, we presented a new multi-user searchable encryption scheme based on our earlier work. In the scheme, user's data can be securely stored on the un-trusted cloud server. The cloud server can perform encrypted searches and return the correct data item to users without learning any plaintext of the data. Users can also share their secret information with others in one group holding unique keys. Besides, when the data is requested, the storage server could perform different operations on the data, if the requestor and the source user of the data are the same, then the data could be sent back to the requestor directly without any computation; if the requestor and the source user are not the same, the storage server could compute a transform key with the complementary keys of the requestor and the source user, then it can convert the data to a cipher text that can be decrypted by the requestor. Moreover, comparing with our earlier work, we allow a master who will manage the users and initial the whole system to take part in the process of data search. It is in line with the storage feature of cloud computing and reduces a huge burden of a cloud server. In the existed schemes, most of them are accurate searchable encryptions. It means users can only search the data by giving precise keywords. Our future work are pointed to the fuzzy searchable encryption[18]. Fuzzy searchable encryption is more practical supporting close tags to generate trapdoor. Thus, users can get more related information during the process of data search.

REFERENCES

[1] R. Chow, et. al., Controlling Data in the Cloud: Outsourcing Computation without Outsourcing Control. Proc. IEEE International Conf. on Cloud Computing, pp. 85-90, 2010.

[2] S. Kamara and K. Lauter, Cryptographic Cloud Storage. Proc. Financial Cryptography: Workshop on Real-Life Cryptographic Protocols and Standardization 2010.

[3] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovskey, Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions. Proc. ACM Conf. on Computer and Communications Security, CCS'06, pp. 79-88, 2006.

[4] Y. Chang and M. Mitzenmacher, Privacy Preserving Keyword Searches on Remote Encrypted Data. Proc. Applied Cryptography and Network Security, ACNS'05, LNCS 3531, pp. 442-455, 2005.

[5] Tang Q. Search in Encrypted Data: Theoretical Models and Practical Applications[J]. 2012.

[6] Song, D. X., Wagner, D., & Perrig, A. (2000). Practical techniques for searches on encrypted data. In Security and Privacy, 2000. S&P 2000. Proceedings. 2000 IEEE Symposium on (pp. 44-55). IEEE.

[7] L. Ballard, S. Kamara, and F. Monrose, Achieving Efficient Conjunctive Keyword Searches over Encrypted Data. Proc. International Conf. on Information and Communications Security, ICICS'05, pp. 414-426, 2005.

[8] P. Golle, J. Staddon, and B. Waters, Secure Conjunctive Keyword Search over Encrypted Data. Proc. Applied Cryptography and Network Security, ACNS'04, pp. 31-45, 2004.

[9] C. Wang, et. al., Secure Ranked Keyword Search over Encrypted Cloud Data . Proc. IEEE International Conf. on Distributed Computing Systems, ICDCS'10, pp. 253-262, 2010.

[10] Yan-Cheng Chang, Michel M. Privacy preserving Keyword Searches On Remote encrypted data[C]. In Applied Cryptography and Network Security (ACNS '05), volume 3531 of Lecture Notes in Computer Science, pages 442-455. Springer, 2005.

[11] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky. Searchable symmetric encryption: Improved definitions and efficient constructions[C]. In ACM Conference on Computer and Communications Security (CCS '06), pages 79-88. ACM, 2006.

[12] C. Dong, G. Russello, and N. Dulay. Shared and searchable encrypted data for untrusted servers. In V. Atluri, editor, DBSec, volume 5094 of Lecture Notes in Computer Science, pages 127–143. Springer, 2008.

[13] C. Dong, G. Russello, and N. Dulay. Shared and searchable encrypted data for untrusted servers. Journal of Computer Security, 2011.

[14] Feng Bao, Robert H.Deng, Xuhua Ding, Yanjiang Yang. Private Query on Encrypted Data in Multi-User Settings. ISPEC'08 Proceedings of the 4th international conference on Information security practice and experience, 2008.

[15] Yang, Y., Lu, H., & Weng, J., Cloud Computing Technology and Science: Multi-user private keyword search for cloud computing. Proc. IEEE International Conf. on Cloud Computing, pp. 264-271, 2011.

[16] Wang Yingkang. & Wenjun Luo. Multi-user Searchable Encryption in Cloud Storage[J]. Telecommunications Science,2012,11:103-107.

[17] Rajan, R., & Coimbatore, A. V. V. P. Efficient And Privacy Preserving Multi User Keyword Search For Cloud Storage Services[J], International Journal of Advanced Technology & Engineering Research, 2012,2(4):48-51.

[18] Wang J, Ma H, Tang Q, et al. Efficient verifiable fuzzy keyword search over encrypted data in cloud computing[J]. Computer Science and Information Systems, 2013, 10(2): 667-684.