Scientific Journal of Impact Factor (SJIF): 4.72

e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 5, Issue 01, January -2018

Key Aggregation for Data Sharing in Cloud

PROF. SANTOSH SHAMRAO DARAWADE ,[#1]

[#1,] *Computer Engineering Department, Savitribai Phule Pune University*

[1,]*Assistant Professor,P.K.Techanical Campus,Chakan*

Abstract - *Data sharing is important functionality in cloud storage .To address user concerns over potential data leaks in cloud storage a common approach is for the data owner to encrypt all the data before uploading them to the cloud, such that later the encrypted data may be retrieved and decrypted by those who have the decryption keys. A key challenge to designing such encryption schemes lies in the efficient management of encryption keys. This also implies the necessity of securely distributing to users a large number of keys for both encryption in search and user will have to securely store the received key and submit an equally large number of keywords trapdoors to the cloud in order to perform search over the shared data. The practical problem of privacy preserving data sharing system based on public cloud storage which requires a data owner to distribute a large number of keys to users to enable them to access his/her documents. By addressing this practical problem which is largely neglected in literature, we propose the novel concept of key aggregate searchable encryption (KASE) in which data owner only need to distribute a single key to user for sharing large number of documents and user only needs to submit a single trapdoor to the cloud for querying the shared a large number of documents.*

Keywords *— Searchable Encryption, Data Sharing, Cloud Storage, Data Privacy.*

I. INTRODUCTION

Cloud storage has emerged as a promising solution for providing ubiquitous, convenient, and on-demand accesses to large amounts of data shared over the Internet. Today, millions of users are sharing personal data, such as photos and videos, with their friends through social network applications based on cloud storage on a daily basis. Business users are also being attracted by cloud storage due to its numerous benefits, including lower cost, greater agility, and better resource utilization. However while enjoying the convenience of sharing data via cloud storage, users are also increasingly concerned about inadvertent data leaks in the cloud. Such data leaks, caused by a malicious adversary or a misbehaving cloud operator, can usually lead to serious breaches of personal privacy or business secrets e.g., the recent high profile incident of celebrity photos being leaked in iCloud. To address users concerns over potential data leaks in cloud storage ,a common approach is for the data owner to encrypt all the data before uploading them to the cloud, such that later the encrypted data may be retrieved and decrypted by those who have the decryption keys. Such a cloud storage is often called the cryptographic cloud storage. However, the encryption of data makes it challenging for users to search and then selectively retrieve only the data containing given keywords. A common solution is to employ a searchable encryption scheme in which the data owner is required to encrypt potential keywords and upload them to the cloud together with encrypted data, such that, for retrieving data matching a keyword, the user will send the corresponding keyword trapdoor to the cloud for performing search over the encrypted data. Although combining a searchable encryption scheme with cryptographic cloud storage can achieve the basic security requirements of a cloud storage, implementing such a system for large scale applications involving millions of users and billions of files may still be hindered by practical issues involving the efficient management of encryption keys. The need for selectively sharing encrypted data with different users (e.g., sharing a photo with certain friends in a social network application, or sharing a business document with certain colleagues on a cloud drive) usually demands different encryption keys to be used for different files. However, this implies the number of keys that need to be distributed to users, both for them to search over the encrypted files and to decrypt the files, will be proportional to the number of such files. Such a large number of keys channels, but also be securely stored and managed by the users in their devices. In addition, a large number of trapdoors must be generated by users and submitted to the cloud in order to perform a keyword search over many files. The implied need for secure communication, storage, and computational complexity may render such a system inefficient and impractical. This challenge by proposing the novel concept of key-aggregate searchable encryption (KASE), and instantiating the concept through a concrete KASE scheme. The proposed KASE scheme applies to any cloud storage that supports the searchable group data sharing functionality, which means any user may selectively share a group of selected files with a group of selected users, while allowing the latter to perform keyword search over the former. To support searchable group data sharing the main requirements for efficient key management are twofold. First, a data owner only needs to distribute a single aggregate key (instead of a group of keys) to a user for sharing any number of files. Second, the user only needs to submit a single aggregate trapdoor (instead of a group of trapdoors) to the cloud for performing keyword search over any number of shared files. To the best of our knowledge, the KASE scheme proposed

in this paper is the first known scheme that can satisfy both requirements (the key-aggregate cryptosystem [4], which has inspired our work, can satisfy the first requirement but not the second).
We first define a general framework of key aggregate searchable encryption (KASE) composed of seven polynomial algorithms for security parameter setup, key generation, encryption, key extraction, trapdoor generation, trapdoor adjustment, and trapdoor testing. We then describe both functional and security requirements for designing a valid KASE scheme. We then instantiate the KASE framework by designing a concrete KASE scheme. After providing detailed constructions for the seven algorithms.

II. PROBLEM STATEMENT

Consider a scenario where two employees of a company would like to share some confidential business data using a public cloud storage service . For instance, Alice wants to upload a large collection of financial documents to the cloud storage, which are meant for the directors of different departments to review. Suppose those documents contain highly sensitive information that should only be accessed by authorized users, and Bob is one of the directors and is thus authorized to view documents related to his department. Due to concerns about potential data leakage in the cloud, Alice encrypts these documents with different keys, and generates keyword ciphertexts based on department names, before uploading to the cloud storage. Alice then uploads and shares those documents with the directors using the sharing functionality of the cloud storage. In order for Bob to view the documents related to his department, Alice must delegate to Bob the rights both for keyword search over those documents, and for decryption of documents related to Bob's department.

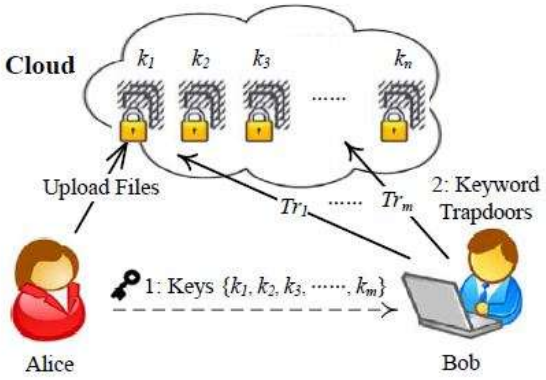

Fig.1. Traditional approach For Group data Sharing

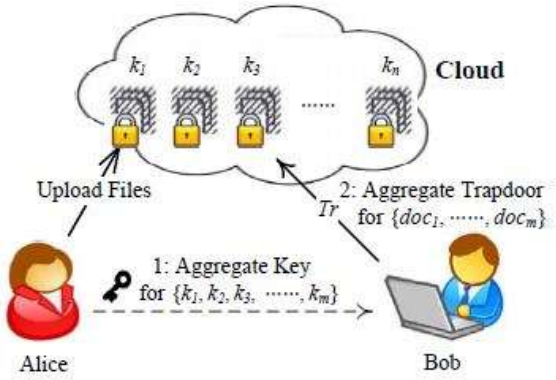

Fig.2. Key-Aggregate Searchable Encryption

With a traditional approach, Alice must securely send all the searchable encryption keys to Bob. After receiving these keys, Bob must store them securely, and then he must generate all the keyword trapdoors using these keys in order to perform a keyword search. As shown in Fig1, Alice is assumed to have a private document set and for each document doci, a searchable encryption key ki is used. In this case Alice must send all the searchable encryption keys to Bob. Then, when Bob wants to retrieve documents containing a keyword w, he must generate keyword trapdoor Tri for each document doci with key ki and submit all the trapdoors cloud server. When m is sufficiently large, the key distribution

and storage as well as the trapdoor generation may become too expensive for Bob's client-side device, which basically defies the purpose of using cloud storage.

In this paper, we propose the novel approach of key-aggregate searchable encryption (KASE) as a better solution, as depicted in Fig.2. In KASE, Alice only needs to distribute a single aggregate key. The cloud server can use this aggregate trapdoor and some public information to perform keyword search and return the result to Bob. Therefore, in KASE, the delegation of keyword search right can be achieved by sharing the single aggregate key. We note that the delegation of decryption rights can be achieved using the key-aggregate encryption approach recently proposed in , but it remains an open problem to delegate the keyword search rights together with the decryption rights, which is the subject topic of this paper. To summarize, the problem of constructing a KASE scheme can be stated as: "To design a key-aggregate searchable encryption scheme under which any subset of the keyword ciphertexts (produced by the SE. Encrypt algorithm to be introduced . From any set of documents is searchable (performed by the SE. Test algorithm) with a constant-size trapdoor (produced by SE. Trpdr algorithm) generated by a constant size aggregate key.

III. KASE FRAMEWORK

The KASE framework is composed of seven algorithms. Specifically, to set up the scheme, the cloud server would generate public parameters of the system through the Setup algorithm, and these public parameters can be reused by different data owners to share their files. For each data owner, he/she should produce a public/master-secret key pair through the Keygen algorithm. Keywords of each document can be encrypted via the Encrypt algorithm with the unique searchable encryption key. Then, the data owner can use the master-secret key to generate an aggregate searchable encryption key for a group of selected documents via the Extract algorithm.

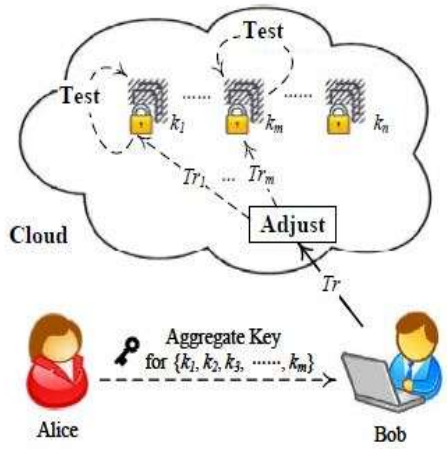

Fig 3:- KASE FRAMEWORK

The aggregate key can be distributed securely (e.g., via secure e-mails or secure devices) to authorized users who need to access those documents. After that, as shown in Fig.4, an authorized user can produce a keyword trapdoor via the Trapdoor algorithm using this aggregate key, and submit the trapdoor to the cloud. After receiving the trapdoor, to perform the keyword search over the specified set of documents, the cloud server will run the Adjust algorithm to generate the right trapdoor for each document, and then run the Test algorithm to test whether the document contains the keyword. This framework is summarized in the following.

- **Setup**(1, n): this algorithm is run by the cloud service provider to set up the scheme. On input of a security parameter 1, and the maximum possible number n of documents which belongs to a data owner, it outputs the public system parameter params.

- **Keygen**: this algorithm is run by the data owner to generate a random key pair (pk,msk).

- **Encrypt**(pk, i): this algorithm is run by the data owner to encrypt the i-th document and generate its keywords' ciphertexts. For each document, this algorithm will create a delta i for its searchable encryption key Ki. On input of the owner's public key pk and the file index i, this algorithm outputs data ciphertext and keyword ciphertexts Ci.

- **Extract**(msk, S): this algorithm is run by the data owner to generate an aggregate searchable encryption key for delegating the keyword search right for a certain set of documents to other users. It takes as input the owner's master-secret key msk and a set S which contains the indices of documents, then outputs the aggregate key kagg.

- **Trapdoor**(kagg, w): this algorithm is run by the user who has the aggregate key to perform a search. It takes as input the aggregate searchable encryption key kagg and a keyword w, then outputs only one trapdoor Tr.

- **Adjust**(params, i, S, Tr): this algorithm is run by cloud server to adjust the aggregate trapdoor to generate the right trapdoor for each different document. It takes as input the system public parameters params, the set S of documents' indices, the index i of target document and the aggregate trapdoor Tr, then outputs each trapdoor Tri for the i-th target document in S.

- **Test**(Tri, i): this algorithm is run by the cloud server to perform keyword search over an encrypted document. It takes as input the trapdoor Tri and the document index i, then outputs true or false to denote whether the document doci contains the keyword w.

IV. WORKING FLOW

- **System setup**. When an organization submits a request, the cloud will create a database containing above four tables, assign a groupID for this organization and insert a record into table company. Moreover, it assigns an administrator account for the manager. Then, the group data sharing system will work under the control of manager. To generate the system parameters params, manager runs the algorithm KASE.Setup and updates the field parameters in table company.

- **User registration** When adding a new member, the manager assigns memberID, membeName, password and a key pair generated by any public key encryption (PKE) scheme for him, then stores the necessary information into the table member. A user's private key should be distributed through a secure channel.

- **User login** Like most popular data sharing products (e.g., Dropbox and citrix), our system relies on password verification for authenticating users. To further improve the security, multi-factor authentication or digital signatures may be used when available.

- **Data uploading** To upload a document, the owner runs KAE.Encrypt to encrypt the data and KASE.Encrypt to encrypt the keyword ciphertexts, then uploads them to the cloud. The cloud assigns a docID for this document and stores the encrypted data in the path filePath, then inserts a record into the table docs.

- **Data sharing**. To share a group of documents with a target member, the owner runs KAE. Extract and KASE. Extract to generate the aggregate keys, and distributes them to this member, then inserts/updates a record in table sharedDocs. If the shared documents for this member are changed, the owner must reextract the keys and update the field docIDSet in table sharedDocs.

- **Keyword Search**. To retrieve the documents containing an expected keyword, a member runs KASE. Trapdoor to generate the keyword trapdoor fordocuments shared by each owner, then submits each trapdoor and the related owner's identity OwnerID to the cloud. After receiving the request, for each trapdoor, the cloud will run KASE .Adjust the trapdoor for each document in the docIDSet and run KASE. Test to perform keyword search. Then, the cloud will return the encrypted documents which contains the expected keyword to the member.

- **Data retrieving**. After receiving the encrypted document, the member will run KAE. Decrypt to decrypt the document using the aggregate key distributed by the document's owner.

V. CONCLUSION

Considering the practical problem of privacy preserving data sharing system based on public cloud storage which requires a data owner to distribute a large number of keys to users to enable them to access his/her documents, we for the first time propose the concept of key-aggregate searchable encryption (KASE) and construct a concrete KASE

scheme. Both analysis and evaluation results confirm that our work can provide an effective solution to building practical data sharing system based on public cloud storage.

In a KASE scheme, the owner only needs to distribute a single key to a user when sharing lots of documents with the user, and the user only needs to submit a single trapdoor when he queries over all documents shared by the same owner. However, if a user wants to query over documents shared by multiple owners, he must generate multiple trapdoors to the cloud. How to reduce the number of trapdoors under multi-owners setting is a future work. Moreover, federated clouds have attracted a lot of attention nowadays, but our KASE cannot be applied in this case directly. It is also a future work to provide the solution for KASE in the case of federated clouds.

VI. REFERENCES

[1] P. Amaravathi , V. Josephraju, K. Challayamma, "Provide Secure And Privacy- Preserving Access Control To Users Anonymously Utilize The Cloud Resource Using Mona",2010.

[2] Younis A.Younis, Madjid Merabti and Kashif Kifayat"Secure Cloud Computing for Critical Infrastructure: A Survey",2013.

[3] Fangquan Cheng, Qian Wang, Qianwen Zhang, and Zhiyong Peng"Highly Efficient Indexing for Privacy-Preserving Multi-keyword Query over Encrypted Cloud Data",2010.

[4] Shobha D. Patil, S. B. Sonkamble ,"Survey Paper On Modoc: Multi Owner Data Sharing Over Cloud",2011.

[5] Ming Li,Shucheng Yu,Yao Zheng,Kui Ren,and Wenjing Lou"Scalable and Secure Sharing of Personal Health Records in Cloud Computing using Attribute-based Encryption",2011

[6] LalVikram Singh, Amol V. Bole, Shailesh Kumar Yadav, "Security Issues of Cloud Computing- A Survey",2011.

[7] S. Yu, C. Wang, K. Ren, and W. Lou,"Achieving Secure, Scalable, and Fine- Grained Data Access Control in Cloud Computing", Proc. IEEE INFOCOM, pp. 534-542, 2010.

[8] R. Lu, X. Lin, X. Liang, and X. Shen,"Secure Provenance: The Essential of Bread and Butter of Data Forensics in Cloud Computing', Proc. ACM Symp. Information, Computer and Comm.Security, pp. 282-292, 2010.12

[9] X. Liu, Y. Zhang, B. Wang, and J. Yan. "Mona: secure multi-owner data sharing for dynamic groups in the cloud", IEEE Transactions on Parallel and Distributed Systems, 2013, 24(6): 1182-1191.

[10] C. Chu, S. Chow, W. Tzeng, et al."Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage", IEEE Transactions on Parallel and Distributed Systems, 2014, 25(2): 468-477.

