Journal of Engineering Sciences

Vol 11, Issue 7,July/2020

ISSN NO:0377-9254

IMPLEMENTATION OF AN EFFICIENT DATA SHARING SCHEME FOR MOBILE CLOUD DATA

[1]MURALIKRISHNA B, [2]CH CHAITANYA

[1]Asst. Professor, Dept. of MCA, PBR VITS, Kavali, A.P, India.

[2]MCA, Dept. of MCA, PBR VITS, Kavali, A.P, India.

Abstract – With the popularity of cloud computing, mobile devices can store/retrieve personal data from anywhere at any time. Consequently, the data security problem in mobile cloud becomes more and more severe and prevents further development of mobile cloud. There are substantial studies that have been conducted to improve the cloud security. However, most of them are not applicable for mobile cloud since mobile devices only have limited computing resources and power. Solutions with low computational overhead are in great need for mobile cloud applications. In this paper, we propose a lightweight data sharing scheme (LDSS) for mobile cloud computing. It adopts CP-ABE, an access control technology used in normal cloud environment, but changes the structure of access control tree to make it suitable for mobile cloud environments. LDSS moves a large portion of the computational intensive access control tree transformation in CP-ABE from mobile devices to external proxy servers. Furthermore, to reduce the user revocation cost, it introduces attribute description fields to implement lazy-revocation, which is a thorny issue in program based CP-ABE systems.

Index terms – CP-ABE, Data Sharing, Cloud Computing.

I. INTRODUCTION

With the development of cloud computing and the popularity of smart mobile devices, people are gradually getting accustomed to a new era of data sharing model in which the data is stored on the cloud and the mobile devices are used to store/retrieve the data from the cloud. Typically, mobile devices only have limited storage space and computing power. On the contrary, the cloud has enormous amount of resources. In such a scenario, to achieve the satisfactory perform-ance, it is essential to use the resources provided by the cloud service provider (CSP) to store and share the data.

Nowadays, various cloud mobile applications have been widely used. In these applications, people (data owners) can upload their photos, videos, documents and other files to the cloud and share these data with other people (data users) they like to share. CSPs also provide data management functionality for data owners. Since personal data files are sensitive, data owners are allowed to choose whether to make their data files public or can only be shared with specific data users. Clearly, data privacy of the personal sensitive data is a big concern for many data owners.

The state-of-the-art privilege management/access control mechanisms provided by the CSP are either not sufficient or not very convenient. They cannot meet all the requirements of data owners. First, when people upload their data files onto the cloud, they are leaving the data in a place where is out of their control, and the CSP may spy on user data for its commercial interests and/or other reasons. Second, people have to send password to each data user if they only want to share the encrypted data with certain users, which is very cumbersome. To simplify the privilege management, the data owner can divide data users into different groups and send password to the groups which they want to share the data. However, this approach requires fine-grained access control. In both cases, password management is a big issue.

Apparently, to solve the above problems, personal sensitive data should be encrypted before uploaded onto the cloud so that the data is secure against the CSP. However, the data encryption brings new problems. How to provide efficient access control mechanism on ciphertext decryption so that only the authorized users can access the plaintext data is challenging. In addition, system must offer data owners effective user privilege management capability, so they can grant/revoke data access privileges easily on the data users. There have been substantial researches on the issue of data access control over ciphertext. In these researches, they have the following common assumptions. First, the CSP is considered honest and curious. Second, all the sensitive data are encrypted before uploaded to the Cloud. Third, user authorization on certain data is

achieved through encryption/decryption key distribution. In general, this can divide these approaches into four categories: simple ciphertext access control, hierarchical access control, access control based on fully homomorphic encryption and access control based on attribute-based encryption (ABE). All these proposals are designed for non-mobile cloud environment. They consume large amount of storage and computation resources, which are not available for mobile devices. According to the experimental results, the basic ABE operations take much longer time on mobile devices than laptop or desktop computers. It is at least 27 times longer to execute on a smart phone than a personal computer (PC). This means that an encryption operation which takes one minute on a PC will take about half an hour to finish on a mobile device. Furthermore, current solutions don't solve the user privilege change problem very well.

Such an operation could result in very high revocation cost. This is not applicable for mobile devices as well. Clearly, there is no proper solution which can effectively solve the secure data sharing problem in mobile cloud. As the mobile cloud becomes more and more popular, providing an efficient secure data sharing mechanism in mobile cloud is in urgent need.

To address this issue, in this paper, we propose a Lightweight Data Sharing Scheme (LDSS) for mobile cloud computing environment. The main contributions of LDSS are as follows:

(1) we design an algorithm called LDSS-CP-ABE based on Attribute-Based Encryption (ABE) method to offer efficient access control over ciphertext.

(2) We use proxy servers for encryption and decryption operations. In our approach, computational intensive operations in ABE are conducted on proxy servers, which greatly reduce the computational overhead on client side mobile devices. Meanwhile, in LDSS-CP-ABE, in order to maintain data privacy, a version attribute is also added to the access structure. The decryption key format is modified so that it can be sent to the proxy servers in a secure way.

(3) We introduce lazy re-encryption and description field of attributes to reduce the revocation overhead when dealing with the user revocation problem.

(4) Finally, we implement a data sharing prototype framework based on LDSS. The experiments show that LDSS can greatly reduce the overhead on the client side, which only introduces a minimal additional cost on the server side. Such an approach is beneficial to implement a realistic data sharing security scheme on mobile devices.

II. BACKGROUND WORK

This section focuses on the works of ciphertext access control schemes which are closely related to our research.

Access control is an important mechanism of data privacy protection to ensure that data can only be acquired by legitimate users. There has been substantial research on the issues of data access control in the cloud, mostly focusing on access control over ciphertext. Typically, the cloud is considered honest and curious. Sensitive data has to be encrypted before sending to the cloud.

User authorization is achieved through key distribution. The research can be generally divided into four areas: simple ciphertext access control, hierarchical access control, access control based on fully homomorphic encryption and access control based on attribute-based encryption (ABE).

Simple ciphertext access control refers to that after data file encryption, the encryption keys are distributed in a secure way to achieve authorization for trusted users. To reduce the overhead of massive user key distribution, Skillen and Mannan designed a system called Mobiflage that enables PDE (plausibly deniable encryption) on mobile devices by hiding encrypted volumes via random data on a device's external storage. However, the system needs to obtain large amount of information of keys. It borrows the access control method used in conventional distributed storage, separating users into different groups according to access rights and assign different keys to groups. This reduces the overhead of key management, but it cannot satisfy the demand for fine-grained access control.

Hierarchical access control has good performance in reducing the overhead of key distribution in ciphertext access control. As a result, there is substantial research on ciphertext access control based on hierarchical access control method. In hierarchical access control method, keys can be derived from private keys and a public token table. However, the operation on token table is complicated and generates high cost. Besides, the token table is stored in the cloud. Its privacy and security cannot be guaranteed.

Attribute-based encryption algorithm is derived from identity-based encryption. It embeds decryption rules in the encryption algorithm, which avoids frequent key distribution. Lai et al and Bethencourt et al proposed key-policy attribute-based encryption (KP-ABE) and ciphertext-policy attribute-based encryption (CP-ABE). In practical applications, CP-ABE has been extensively studied since it is similar to role-based access control (RBAC) scheme. In CP-ABE, the possession of one attribute key means that the key owner owns corresponding attribute, and attribute keys cannot be reclaimed once they are distributed. As a result, when a data user's attribute is revoked, how to ensure data privacy becomes a difficult issue. Liang et al propose attribute-based proxy re-encryption (ABPRE) scheme to solve this problem. However, in their solution, when a user's attribute is revoked, all other users who own this attribute will lose this attribute at the same time, which cannot satisfy fine-grained access control needs. Tian et al combine CP-ABE and public key cryptography to achieve ciphertext access control.

All the above works focus on the issue of data access control in the cloud. They are mainly for non-mobile devices and cannot be applied for data sharing in mobile cloud environment. Regarding to data privacy in mobile cloud, some works have been done in this field. Huang et al propose MobiCloud, in which traditional Mobile Ad-hoc NETworks (MANETs) is transformed into service-oriented communication architecture. In this architecture, each mobile device is regarded as a service node, and the operations are outsourced to the cloud. However, in MobiCloud, users need to completely trust the cloud, which is not the case in reality. Livshits and Jung designed and implemented a graph theoretic algorithm to place mediation prompts that protect every resource access, while avoiding repetitive prompting and prompting in background tasks or third-party libraries, for the problem of mediating resource accesses in mobile applications. Zhou et al proposed an ABDS scheme to achieve secure data storage in the cloud. However, this scheme is not suitable for data sharing and has no clear solution for attribute revocation.

In summary, current proposals on data access control in the cloud are mostly for non-mobile terminals, which is not suitable for mobile devices. Besides, current solutions don't solve the problem of user privilege change scenarios very well since they bring high revocation cost. This is not applicable for mobile devices which only have limited computing capacity and power. Existing studies on mobile cloud don't have a good solution to secure data sharing when servers are not credible. In a word, there is no proper solution that can solve the problem of secure data sharing in mobile cloud. In this project, it propose a lightweight data sharing scheme (LDSS) for mobile cloud applications.

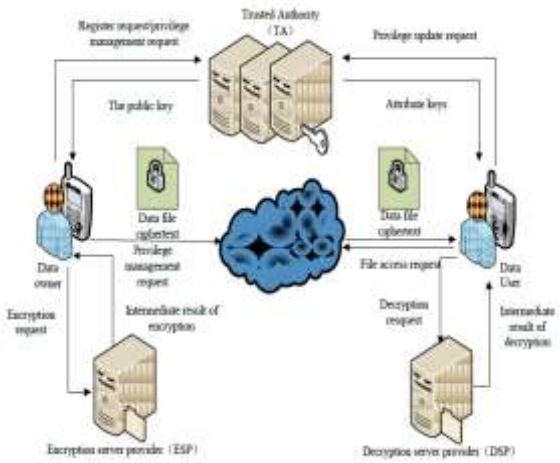

Fig. 1: System Overview

III. PROPOSED WORK

We propose LDSS, a framework of lightweight data sharing scheme in mobile cloud (see Fig. 1). It has the following six components.

(1) Data Owner (DO): DO uploads data to the mobile cloud and share it with friends. DO determines the access control policies.

(2) Data User (DU): DU retrieves data from the mobile cloud.

(3) Trust Authority (TA): TA is responsible for generating and distributing attribute keys.

(4) Encryption Service Provider (ESP): ESP provides data encryption operations for DO.

(5) Decryption Service Provider (DSP): DSP provides data decryption operations for DU.

(6) Cloud Service Provider (CSP): CSP stores the data for DO. It faithfully executes the operations requested by DO, while it may peek over data that DO has stored in the cloud.

As shown in Fig. 1, a DO sends data to the cloud. Since the cloud is not credible, data has to be encrypted before it is uploaded. The DO defines access control policy in the form of access control tree (refer to Definition 2 in Section 3.2) on data files to assign which attributes a DU should obtain if he wants to access a certain data file. In LDSS, data files

are all encrypted with the symmetric encryption mechanism, and the symmetric key for data encryption is also encrypted using attribute based encryption (ABE).

The access control policy is embedded in the ciphertext of the symmetric key. Only a DU who obtains attribute keys that satisfy the access control policy can decrypt the ciphertext and retrieve the symmetric key. As the encryption and decryption are both computationally intensive, they introduce heavy burden for mobile users.

To relieve the overhead on the client side mobile devices, encryption service provider (ESP) and decryption service provider (DSP) are used. Both the encryption service provider and the decryption service provider are also semi-trusted. We modify the traditional CP-ABE algorithm and design an LDSS-CP-ABE algorithm to ensure the data privacy when outsourcing computational tasks to ESP and DSP.

LDSS-CP-ABE algorithm is designed using above definitions. It includes four sub-functions:

Setup(A, V): Generate the master key MK, the public key PK based on attribute set A of the Data Owner and the version attribute V .

KeyGen(Au, MK): Generate attribute keys SKu for a data user U based on his attribute set Au and the master key MK.

Encryption(K, PK, T): Generate the ciphertext CT based on the symmetric key K, public key PK and access control tree T .

Decryption(CT,T,SKu): Decrypt the ciphertext CT using the access control tree T and the attribute keys SKu .

IV. SYSTEM IMPLEMENTATION

a) Data Owner (DO):

When the data owner (DO) registers on TA, TA runs the algorithm Setup() to generate a public key PK and a master key MK. PK is sent to DO while MK is kept on TA itself. DO define its own attribute set and assigns attributes to its contacts. All these information will be sent to TA and the cloud. TA and the cloud receive the information and store it. DO uploads data to the mobile cloud and share it with friends. DO determine the access control policies. DO sends data to the cloud. Since the cloud is not credible, data has to be encrypted before it is uploaded.

The DO defines access control policy in the form of access control tree on data files to assign which attributes a DU should obtain if he wants to access a certain data file.

b) Data User (DU):

DU logins onto the system and sends, an authorization request to TA. The authorization request includes attribute keys (SK) which DU already has. TA accepts the authorization request and checks the request and a generate attribute keys (SK) for DU.DU sends a request for data to the cloud. Cloud receives the request and checks if the DU meets the access requirement. DU receives the ciphertext, which includes ciphertext of data files and ciphertext of the symmetric key. DU decrypts the ciphertext of the symmetric key with the assistance of DSP. DU uses the symmetric key to decrypt the ciphertext of data files.

c) Trusted Authority:

To make LDSS feasible in practice, a trusted authority (TA) is introduced. It is responsible of generating public and private keys, and distributing attribute keys to users. With this mechanism, users can share and access data without being aware of the encryption and decryption operations. We assume TA is entirely credible, and a trusted channel exists between the TA and every user. The fact that a trusted channel exists doesn't mean that the data can be shared through the trusted channel, for the data can be in a large amount. TA is only used to transfer keys (in a small amount) securely between users. In addition, it's requested that TA is online all the time because data users may access data at any time and need TA to update attribute keys.

d) Cloud Service Provider:

CSP stores the data for DO. It faithfully executes the operations requested by DO, while it may peek over data that DO has stored in the cloud. DU sends a request for data to the cloud. Cloud receives the request and checks if the DU meets the access requirement. If DU can't meet the requirement, it refuses the request; otherwise it sends the ciphertext to DU. CSP manages the Uploaded Files.

V. CONCLUSION

In recent years, many studies on access control in cloud are based on attribute-based encryption algorithm (ABE). However, traditional ABE is not suitable for mobile cloud because it is computationally intensive and mobile devices only have limited resources. This project implements LDSS to address this issue. It introduces a novel

LDSS-CP-ABE algorithm to migrate major computation overhead from mobile devices onto proxy servers, thus it can solve the secure data sharing problem in mobile cloud.

VI. FUTURE ENHANCEMENT

In the future work, we will design new approaches to ensure data integrity. To further tap the potential of mobile cloud, we will also study how to do ciphertext retrieval over existing data sharing schemes.

References

[1] Ruixuan Li, Chenglin Shen, Heng He, Zhiyong Xu, and Cheng-Zhong Xu "A Lightweight Secure Data Sharing Scheme for Mobile Cloud Computing" IEEE Transactions on Cloud Computing, vol. 6, issue. 2, pp. 344 – 357, Jan 2017.

[2] Gentry C, Halevi S. Implementing gentry's fully-homomorphic encryption scheme. in: Advances in Cryptology–EUROCRYPT 2011. Berlin, Heidelberg: Springer press, pp. 129-148, 2011.

[3] Brakerski Z, Vaikuntanathan V. Efficient fully homomorphic encryption from (standard) LWE. in: Proceeding of IEEE Symposium on Foundations of Computer Science. California, USA: IEEE press, pp. 97-106, Oct. 2011.

[4] Qihua Wang, Hongxia Jin. "Data leakage mitigation for discertionary access control in collaboration clouds". the 16th ACM Symposium on Access Control Models and Technologies (SACMAT), pp.103-122, Jun. 2011.

[5] Adam Skillen and Mohammad Mannan. On Implementing Deniable Storage Encryption for Mobile Devices. the 20th Annual Network and Distributed System Security Symposium (NDSS), Feb. 2013.

[6] Wang W, Li Z, Owens R, et al. Secure and efficient access to outsourced data. in: Proceedings of the 2009 ACM workshop on Cloud computing security. Chicago, USA: ACM pp. 55-66, 2009.

[7] Maheshwari U, Vingralek R, Shapiro W. How to build a trusted database system on untrusted storage. in: Proceedings of the 4th conference on Symposium on Operating System Design & Implementation-Volume 4. USENIX Association, pp. 10-12, 2000.

[8] Kan Yang, Xiaohua Jia, Kui Ren: Attribute-based fine-grained access control with efficient revocation in cloud storage systems. ASIACCS 2013, pp. 523-528, 2013.