International Journal of Trend in Scientific Research and Development (IJTSRD)
International Open Access Journal
ISSN No: 2456 - 6470 | www.ijtsrd.com | Volume - 2 | Issue – 4

# A Design of Lightweight Secure Data Sharing

**Neha Fatima, Prof. S. A. Madival**
Appa Institute of Engineering and Technology, PG Department of Computer Science Engineering, Kalaburgi, Karnataka, India

**ABSTRACT**
In this paper, we propose a lightweight data sharing scheme (LDSS) for different cloud platforms. It adopts CP-ABE, an access control technology used in most of the cloud environments; there are certain necessary changes in the structure of access control tree to make it suitable for portable cloud environments. LDSS moves a large portion of the computational intensive access control tree transformation in CP-ABE from different 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. The experimental results show that LDSS can effectively reduce the overhead on large number of devices when users are sharing data through different cloud environments.

***Keywords:*** *Lightweight, cloud, portable, tree, overhead*

## I. INTRODUCTION

Cloud computing means storing data and accessing that data from the internet instead of using traditional hardware for most of the operations. More than 50% of IT companies have moved their Business to the cloud. Sharing of data over the cloud is the new trend that is being set on. The amount of data generated on a day to day life is increasing and to store that all of the data in traditional hardware is not possible because of limited storage capacity [1]. Therefore, transferring the data to the cloud is a necessity where the user can get unlimited storage. Security of that data is the next big concern for most of us. After uploading the data to the cloud user loses its control over that data. 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. Therefore, privacy of the personal sensitive data is a big concern for many data owners.

When any of the people upload the data onto the cloud they are leaving their data in a place where monitoring over that data is out of their control, the cloud service provider can also spy on the personal data of the users. When someone has to share data over the data they have to share the password to each and every user for accessing the encrypted data which is cumbersome [4]. Therefore, to solve this problem data should be encrypted before uploading it onto the cloud which can be safe from everyone. Now the data encryption part brings some new problems such as we have to provide an efficient encryption algorithm such that if the data is in encrypted format it cannot be easily to get break or get accessed by any exploiters [1]. The next big concern is time consumption for encryption. Traditional Hardware with big configuration can encrypt data in short amount of time but limited resource devices suffer from this problem. They require more amount of time of encryption and decryption. So, an efficient crypto system is to be proposed which can worked equally or heterogeneously on all of the devices.

Personal sensitive data should be encrypted before uploaded onto the cloud so that the data is secure against the cloud service providers. However, the data encryption brings new problems. How to provide efficient access control mechanism on cipher text decryption so that only the authorized users can access the plaintext data is challenging [2]. In addition, system must offer data owner's effective user privilege management capability, so they can

grant/revoke data access privileges easily on the data users.

## II. EXISTING SYSTEM

With the development of cloud computing and digitalization, people are gradually getting accustomed to a new era of data sharing model in which the data is stored on the cloud and different devices are used to store/retrieve the data from the cloud. Typically, most of the devices such as mobile through which most of the transaction occurs have limited storage space and computing power. On the contrary, the cloud has enormous amount of resources [9]. In such a scenario, to achieve the satisfactory performance, it is essential to use the resources provided by the cloud service provider to store and share the data. The development of cloud computing and the popularity of smart portable 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 different portable devices are used to store/retrieve the data from the cloud.

Disadvantages:

- 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 cloud service provider are either not sufficient or not very convenient.
- They cannot meet all the requirements of data owners.

## III. PROPOSED SYSTEM

In this paper, we propose a Lightweight Data Sharing Scheme (LDSS) for different cloud computing environment [2] [7]. We design an algorithm called LDSS-CP-ABE based on Attribute Based Encryption (ABE) method to offer efficient access control over cipher text. 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 devices

Advantages:

- We are providing methods for efficient access of the data.
- Performance has been increased with the reduced cost.
- Such an approach is beneficial to implement a realistic data sharing security scheme on devices.
- The results also show that LDSS has better performance compared to the existing ABE based access control schemes over cipher text.

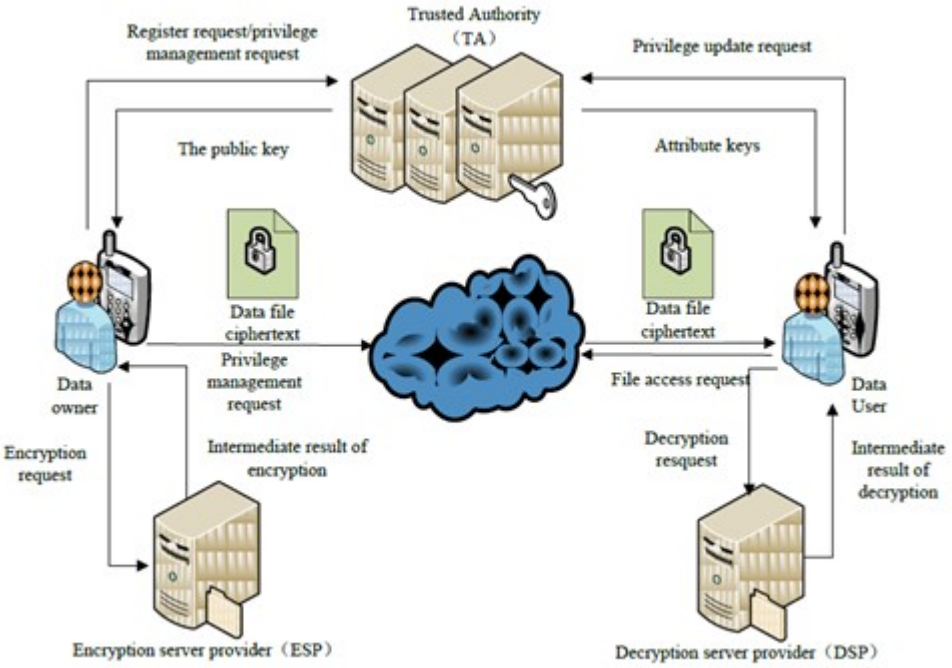


**Fig. 1: LDSS Framework.**

## IV. DESIGN METHODOLOGY

We describe LDSS Framework in "Fig. 1" as shown above [8]. 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 cipher text.

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. The results also show that LDSS has better performance compared to the existing ABE based access control schemes over cipher text.

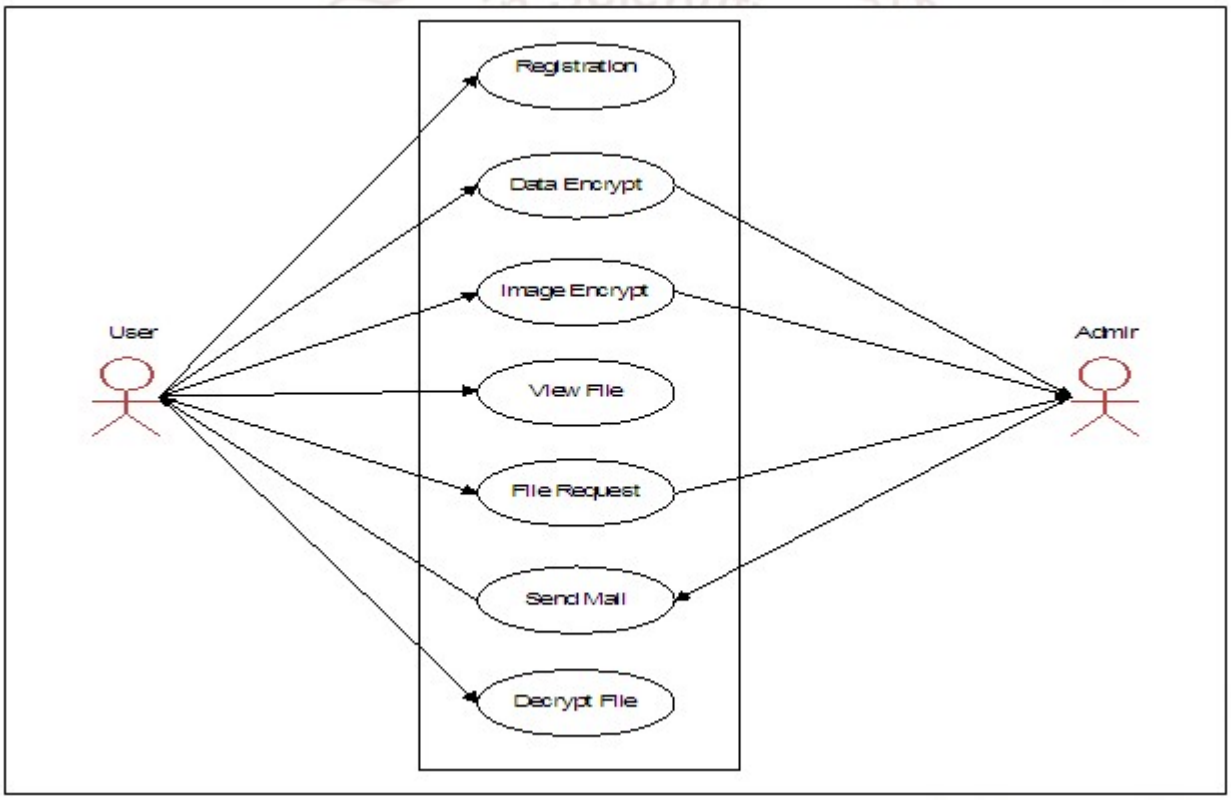


Fig. 2: Use Case diagram.

In "Fig. 2 "we represent the use case diagram which describe both admin and user side operations. The various actions of LDSS Framework are described below in detail [5].

1) **Text Encryption and Decryption:** In this module user encrypted the plain text to encrypted format and uploaded to the cloud. The encryption is done by using a password.
2) **Image Encryption and decryption:** Like the same as the image encryption is also done. And the encrypted images and password will also be uploaded to the cloud.
3) **Text request:** Any user can view the file uploaded in the server. All the files are in encrypted format.
4) **Image request:** Image request is also same as the Text Request. The list of images can view in the application.
5) **View Encrypted Data:** The user uploaded encrypted data can be view in the server side.
6) **View user request:** After user view the encrypted data they can request the password for encrypted

data. This user request can be view in the trusted authority

7) **Provide password:** After view the request Trusted authority validating the user and if the user is valid the Trusted authority provide password for the requested file via email. Using this password user can decrypt the file

## V. IMPLEMENTATION

LDSS Framework has the following six components.

1) **Data Owner (DO):** DO upload data to the cloud and share it with friends. DO determine the access control policies.
2) **Data User (DU):** DU retrieves data from the 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 have stored in the cloud.

LDSS scheme is designed for data sharing in cloud [4]. The whole process of LDSS includes system initialization, file sharing, user authorization, and file access operations. It also has to support attribute revocation and file update operations [7].

### 1. System Initialization:

The specific process is described as follows.

I. 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.
II. DO define its own attribute set and assigns attributes to its contacts. All these information will be sent to TA and the cloud.
III. TA and the cloud receive the information and store it.

### 2. File Sharing:

The specific process is described as follows.

I. DO select a file *M* which is to be uploaded and encrypts it using a symmetric cryptographic mechanism (such as AES, 3DES algorithm) with a symmetric key *K*, generating cipher text *C*.
II. DO assign access control policy for *M* and encrypts *K* with the assistance of ESP using Function 3, generating the cipher text of *K* (*CT*).
III. DO upload C, CT and access control policy to the cloud.

### 3. User Authorization:

The specific process is described as follows.

I. DU logins onto the system and sends, an authorization request to TA. The authorization request includes attribute keys (*SK*) which DU already has.
II. TA accepts the authorization request and checks whether DU has logged on before. If the user hasn't logged on before, go to step (3), otherwise go to step (4).
III. TA calls Function 2 to generate attribute keys (*SK*) for DU.
IV. TA compares the attribute description field in the attribute key with the attribute description field stored in database. If they are not match, go to step (5), otherwise go to step (6).
V. For each inconsistent bit in description field, if it is 1 on data user's side and 0 on TA's side, it indicates that DU's attribute has been revoked, and then TA does nothing on this bit. If it is reversed scenario, it indicates that DU has been assigned with a new attribute, and then TA generates the corresponding attribute key for DU.
VI. TA checks the version of every attribute key of DU. If it's not the same with the current version, then TA updates the corresponding attribute key for DU.

In the stage of user authorization, TA updates attribute keys for DU according to the attribute description field, which is stored with *SK*. It describes which attributes DU has and their corresponding versions. TA also keeps attribute description field of DU in database. When DO changes the attribute of DU, the attribute description field on the TA side is also updated. Thus, when DU logins on the system, the attribute description field on itself may be different from that of TA. TA has to update the attribute keys for DU according to the attribute description field just as described above.

### 4. Access Files:

The specific process is described as follows:

I. DU sends a request for data to the cloud.

II. 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 cipher text to DU.
III. DU receives the cipher text, which includes cipher text of data files and cipher text of the symmetric key. Then DU executes the Function 4 to decrypt the cipher text of the symmetric key with the assistance of DSP.
IV. DU uses the symmetric key to decrypt the cipher text of data files.

**5. Privilege Revoked:**

DO can revoke attributes from a DU. the process is as follows.

I. DO inform TA and the cloud that one attribute has been revoked from a specific DU.
II. TA and the cloud update the information of DU in database.
III. DO mark the corresponding bit of the attribute description field of data files.

**6. Documentation Updates:**

The specific process is as follows.

I. DO checks if there is any bit in the description field of data files has been set to '#'.
II. DO inform TA which attributes should be updated. All the attributes that should be updated form a set is called *Anew*.
III. TA chooses a new value in *G0* for every attribute in *Anew* to replace the original one, and updates the description field of DO in DO-PK/MK table, changing the corresponding attribute description bit to the new value.
IV. TA sends a new *PK* to DO, and DO uses the new *PK* to encrypt data files.

## VI. RESULT

In our paper we describe result analysis from below screenshots of our project. "Fig. 3" states the server login page. In "Fig. 4" we show encryption request and response by ESP, in "Fig. 5" we show data owner uploading file by encryption, in "Fig. 6" we show downloading of file using secret key. In "Fig. 7" we show updating of file only when authorized by trusted authority. In "Fig. 8" we show graphical representation of number of files downloaded by data owner.


**Fig. 3: Server Login Page.**


**Fig. 4: Encryption Requests and Response by ESP.**

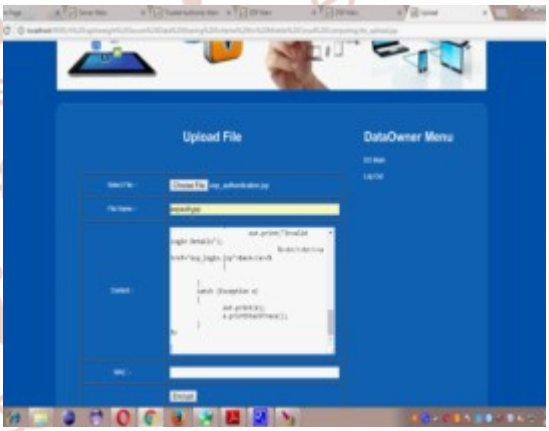


**Fig. 5: Data Owner Uploading File.**

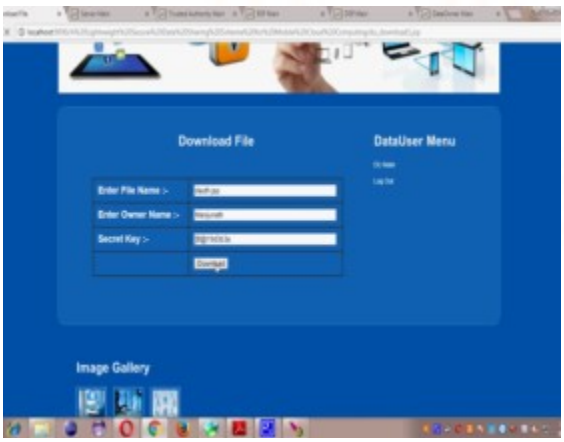


**Fig. 6: Downloading File Using Secret Key.**

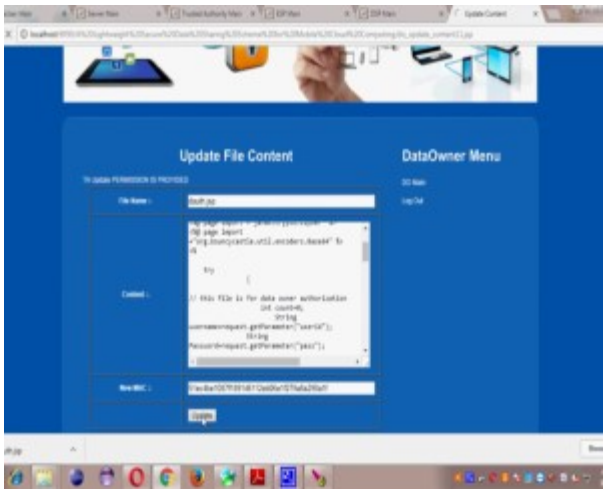


**Fig. 7: Updating of File.**

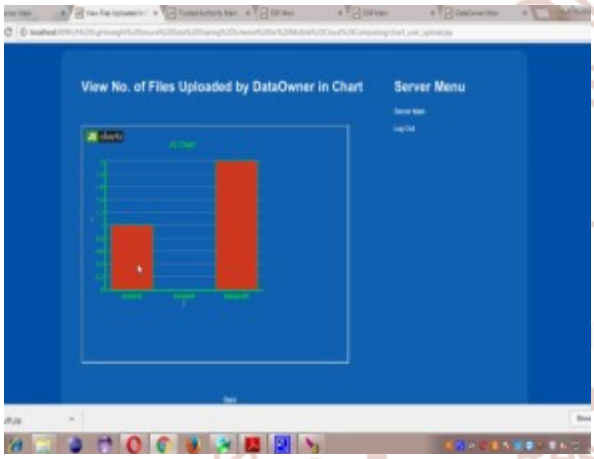


**Fig. 8: Graphical representation of number of Files Uploaded by data owner.**

## VII. CONCLUSION AND FUTURE WORK

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 several cloud platforms because it is computationally intensive and few devices only have limited resources. In this paper, we propose LDSS to address this issue. It introduces a novel LDSS-CP-ABE algorithm to migrate major computation overhead from devices onto proxy servers, thus it can solve the secure data sharing problem in cloud. The experimental results show that LDSS can ensure data privacy in cloud and reduce the overhead on users' side in cloud. In the future work, we will design new approaches to ensure data integrity. To further tap the potential of different cloud platforms, we will also study how to do cipher text retrieval over existing data sharing schemes.

## REFERENCES

1. 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.
2. 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.
3. Cong Wang, Kui Ren, Shucheng Yu, and Karthik Mahendra Raje Urs. Achieving Usable and Privacy-assured Similarity Search over Outsourced Cloud Data. IEEE INFOCOM 2012, Orlando, Florida, March 25-30, 2012
4. Yu S., Wang C., Ren K., Lou W. Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing. INFOCOM 2010, pp. 534-542, 2010
5. Junzuo Lai, Robert H. Deng ,Yingjiu Li ,et al. Fully secure key-policy attribute-based encryption with constant-size ciphertexts and fast decryption. In: Proceedings of the 9th ACM symposium on Information, Computer and Communications Security (ASIACCS), pp. 239-248, Jun. 2014.
6. Jia W, Zhu H, Cao Z, et al. SDSM: a secure data service mechanism in mobile cloud computing. in: Proceedings of 30th IEEE International Conference on Computer Communications. Shanghai, China: IEEE, pp. 1060-1065, 2011.
7. Kan Yang, Xiaohua Jia, Kui Ren, Ruitao Xie, Liusheng Huang: Enabling efficient access control with dynamic policy updating for big data in the cloud. INFOCOM 2014, pp.2013-2021, 2014.
8. Yu S., Wang C., Ren K., et al. Attribute based data sharing with attribute revocation. in: Proceedings of the 5th International Symposium on Information, Computer and Communications Security (ASIACCS), New York, USA: ACM press pp. 261-270, 2010.
9. Stehlé D, Steinfeld R. Faster fully homomorphic encryption. in: Proceedings of 16th International Conference on the Theory and Application of Cryptology and Information Security. Singapore: Springer press, pp.377-394, 2010.
10. Shamir A. How to share a secret. Communications of the ACM,1979, 22 (11): 612-613