Secure Data Sharing in Cloud using a Cryptographic Server

Prof. Shital B. Jadhav[1] Neetal A. Revankar[2] Sanjli S. Raorane[3] Vaishnavi D. Sagale[4] Payal D. Oswal[5]

[1]Professor [2,3,4,5]Student

[1,2,3,4,5]Department of Computer Engineering

[1,2,3,4,5]BVCOEW, Pune Bhima, Pune 412216 India

Abstract— Cloud Computing is the future generation internet based computing system which provides easy and customizable services to the users for accessing their data or to work with various cloud applications. One of the major services provided by cloud is data storage. Cloud Computing is a way for storing & accessing the cloud data from anywhere by connecting the cloud application using internet. Cloud Computing security is the main issue rising nowadays. As Cloud computing provides facility for a group of users to share and access the stored data, there is a possibility of having high data risk. A secure and efficient data sharing scheme needs to provide identity privacy, access control, multiple owner and dynamic data sharing without getting affected by number of cloud users revoked. In our project, we propose the Secure Data Sharing in Clouds (SeDaSC) methodology that provides:1) Data confidentiality and integrity; 2) access control; 3) data sharing 4) insider threat Security; 5) forward and backward access control. The SeDaSC encrypts file with a single encryption key. Two different key shares for each of the users are generated, with the user only getting one share. The possession of a single share of a key allows the SeDaSC methodology to counter the insider threats. The other key share is stored by a third party, which is called other cryptographic server.

Key words: Cloud Computing, SeDaSC

I. Introduction

Cloud computing is a type of online network based computing that delivers shared computer handling resources and data to personal computers and other devices on demand. It is a unique way for enabling universal, on-interest access to shared computing assets (like servers, storage, computer network, applications and services), which can be quickly planned and released with reduced management effort. Cloud computing and storage solutions provide users and IT firms with potential to store and process their data in third-force data centers that may be located anywhere worldwide. Cloud computing relies on sharing of resources to get consistency and scale in economy, the same object as in the previous frame.

Data sharing is becoming increasingly important for many users. For businesses and organizations data sharing has become the most important requirement. People love to share information with one another. Whether it is with friends, family, companions or the world, many people benefit greatly through sharing data.

Some of the benefits are:

A. Higher Productivity

Hospitals benefit from data sharing which leads to lowering of healthcare costs. Students can also get benefit from data sharing while working on group projects due to which they can easily interact with each other and get their work done efficiently with collaboration. Businesses can gain profit by working together. Employees also get benefit as they can share work and collaborate with other employees and can also pursue working at home or any other place such as the library.

B. More Enjoyment

Many people of any age, gender or ethnicity can connect with one another and share their life experiences, achievements, photos etc. As well as catch up with other people from various different regions via social networking sites like Facebook, Twitter, Instagram, Orkut.

C. Requirements of Data Sharing in the Cloud

To enable sharing of data in the Cloud, it is important that only authenticated users can access data stored in the Cloud. Following are the ideal requirements of data sharing in cloud:

- The data owner should be able to define a group of users that are authorized to view his/her data.
- Any member of the group should be able to access the data anytime without the data owner's interposition.
- No other user, other than an owner of the data and the members of the group, should gain the access to the data, including the Third Party Auditor (TPA).
- The data owner should be able to abrogate access to data for any user of the group.
- The data owner should be able to add users to the group.
- No member of the group should be allowed to abrogate the rights of other members of the group or join new members to the group.
- The data owner should be able to define who has read/write permissions on the data owner's files.

II. Literature Survey

M. Ali, R. Dhamotharan, E. Khan, S. U. Khan, A. V. Vasilakos, K. Li, and A. Y. Zomaya, "SeDaSC: Secure Data Sharing in Clouds,"[2], 2015, This paper proposes a methodology that provides data confidentiality, secure data sharing without re-encryption, access control for malicious insiders, and forward and backward access control.

C. Chu, S. S. M. Chow, Wen-GueyTzeng, Jianying Zhou, and Robert H. Deng, "Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage," [5], 2013, this paper proposes a new public key cryptosystem that produces a constant size cipher text with private keys to decrypt.

C. Yang and J. Lai, "Protecting Data Privacy and Security for Cloud Computing Based on Secret Sharing," [6], 2013, This paper proposes the idea of adding symmetric property in secret sharing to successfully minimize the cost to share the shares between the client and the server. Also extended SSC (Secure Cloud Computing) to MSCC (Multi server SCC) fitting the multi-server environment by using a homomorphism property of secret sharing.

Z. Zhu and R. Jiang, "A Secure Anti-Collusion Data Sharing Scheme for Dynamic Groups in the Cloud," [7], 2016, This paper proposes a scheme, in which users can securely obtain their private keys from group manager certificate authorities and secure communication channels.

J. Wei, W. Liu and X. Hu, "Secure Data Sharing in Cloud Computing Using Revocable-Storage Identity-Based Encryption," [8], 2015, This paper proposes a notion called RS-IBE (revocable-storage identity-based encryption), that supports identity revocation and cipher text update simultaneously such that a revoked user is blocked from accessing previously shared data, as well as eventually shared data.

III. EXISTING SYSTEM

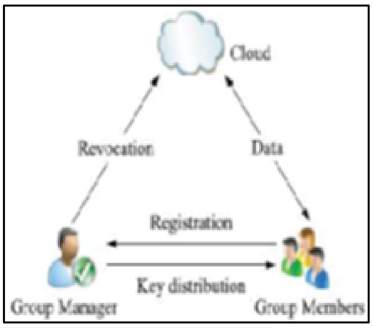


Fig. 1: Existing System

In the Existing System,

- Any user in the group can store and share data files with others by the cloud.
- Entire load of encryption/decryption on group owner.
- Changes of membership make data sharing difficult as the issue of user revocation is not addressed.

IV. PROPOSED SYSTEM

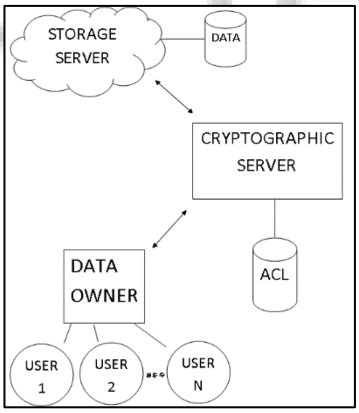


Fig. 2: Proposed System Architecture

Basically our system consists of three main entities: -1) Data Owner 2) A Cryptographic Server (CS) and 3) A Storage Sever. Firstly, the data owner sends the data, the list of the users among whom he wants to share the data, and permissions for each user to the CS. The CS here is a trusted third party (TTP) that is responsible for management of keys, encryption, decryption, and access control. On receiving the data from the data owner, the CS generates an Access Control List (ACL). For key management a random number is generated and its hash value is calculated. This becomes the symmetric key for encryption and decryption. The CS encrypts the data with the generated key and then for each member in the group, the CS splits the key into two parts such that a single part alone cannot regenerate the key. Gradually, the main key is deleted through secure overwriting. One part of the key is given to the corresponding user in the group, whereas the other part is preserved by the CS within the access control list related to the data file. After this hash value is calculated of the encrypted file to detect the tempering of the data and then it is uploaded onto the storage server i.e., Cloud. The user who wishes to access the data sends a download request to the Cryptographic Server. The CS, after authenticating the user, receives the part of the key from the user and afterwards downloads the data file from the storage server. The key is regenerated by operating on the user's part of the key, and the corresponding part of the key for that particular user maintained by the CS. Before decryption hash value is calculated to detect the tempering. After detecting the data file is decrypted and sent to the user. For a new member, the two parts of the key are generated, and the member is added to the ACL. For a departing user, the record of the user is deleted from the ACL. The departing user cannot decrypt the data on its own as he/she only possesses a part of the key not the whole key.

A. Modules of the project

1) Module 1

a) Upload Module

Whenever there is need to share data among the group, the owner of the file sends the request for encryption to the CS. The request consists of the file (*F*) and a list (*L*) of users that are to be granted access to the file. *L* also consists of the access rights for each of the users which is used for the generation of the ACL. On receiving the encryption request the CS generates the ACL from the list and creates a group of the users. For each file the ACL is separately maintained. Then the CS generates the symmetric key, *K* using Random key generation Algorithm and encrypts the file using AES algorithm. The result is an encrypted file (*C*).Subsequently, the CS generates ki and ki' for every user and deletes *K* by secure overwriting.

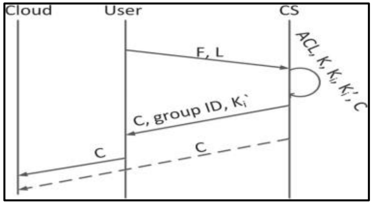


Fig. 3: Upload Module

2) Module 2

a) Download Module

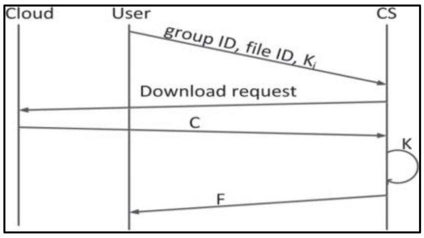


Fig. 4: Download Module

The authorized user either sends download request to the CS or downloads the encrypted file (*C*) from the cloud and then sends the decryption request to the CS. The cloud verifies the authorization of the user through ACL. The decryption request consists of the user key, i.e., *ki'* and the file name. The

CS computes *K* by applying XOR operation over *ki'* and the corresponding *Ki* from the ACL. If the correct *Ki'* is received by the CS, then the decryption and file download will be successful.

3) *Module 3*

a) Update Module

Updating the file is similar to that of uploading the file. The only difference is that, while updating, the activities related to the creation of the ACL and key generation are not carried out. The user downloads the file and makes the required changes and sends an update request to the CS. The request consists the group ID, the file ID, and *Ki'* i.e. the user's key along with the file to be encrypted after changes. The CS verifies whether the user has the WRITE permission to the file from the corresponding ACL. In the request is valid, then the CS computes *K* by XORing *Ki* and *ki'*, encrypts the file, and then the encrypted file is uploaded to the cloud. *K* is deleted afterward.

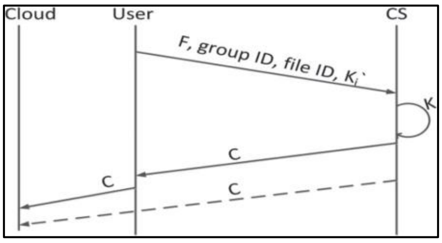


Fig. 5: Update Module

B. *Algorithms*

1) *Algorithm 1: Key generation and encryption*

```
Input:
$F$, the ACL, the SKA, the 256-bit
hash function $H_f$
Compute:
$R = \{0, 1\}^{256}$
        $K = H_f(R)$
        $C = \mathrm{SKA}(F, K)$
for each user $i$ in the ACL, do
$K_i = \{0, 1\}^{256}$
        $K_i = K \oplus K_i$
    Add $K_i'$ for user $i$ in the ACL
    Send $K_i'$ for user $i$
end for
        delete ($K$)
        delete ($K_i'$)
return $C$ to the owner or upload to the cloud.
```

Fig. 6: Key Generation And Encryption

2) *Explaination*

In the first step, a random number 'R' of length 256 bits is generated such that R = $\{0,1\}^{256}$.

In the next step, 'R' is passed through a hash function that could be any hash function with a 256-bit output.

The output of the hash function is termed as 'K' and is used in symmetric key encryption for securing the data.

The output of encryption is stored in 'C'.

For each of the users in the group, CS generates Ki such that Ki = $\{0,1\}^{256.}$ Ki serves as the CS portion of the key and is used to compute K whenever an encryption/decryption request is received by the CS.

Ki' is computed for each of the users in the group as, Ki' = K XOR Ki , serves as the user portion of the key and used to compute K when needed.

Main symmetric key and user portion of the key is deleted from CS.

Finally the encrypted file is returned to the owner or it is directly uploaded onto the cloud.

```
Input:
$C$, the ACL, the SKA
Compute:
Get $K_i'$ from the requesting user
Get $C$ from the requesting user or download from the cloud
Retrieve $K_i$ from the ACL
If $K_i$ does not exist in the ACL, then
    return the access denied message to the user
else
    $K = K_i \oplus K_i'$
    $F = \mathrm{SKA}(C, K)$
    send $F$ to the user
end if
delete ($K$)
delete ($K_i'$).
```

Fig. 7: Decryption Algorithm

In the first step, get the user portion of the key from the requesting user.

In the next step, get the encrypted file from the requesting user or download it from the cloud.

Retrieve the CS portion of the key from the access control list.

Generate the main symmetric key K by performing the XORing between user's part of key and CS part of key as,

K = Ki XOR Ki'.

Decrypt the encrypted file using the symmetric using AES algorithm.

Send the original file to the user.

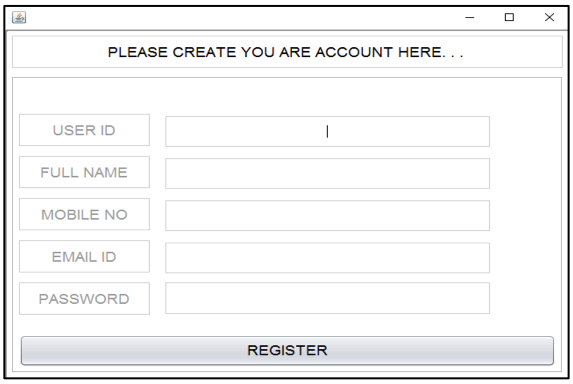
Lastly delete the symmetric key and user part of the key.

V. EXPERIMENTAL RESULTS

To enter the IP address of ACL SERVER

Fig. 8: Welcome Page

A. *New User Registration*



PLEASE CREATE YOU ARE ACCOUNT HERE. . .

USER ID

FULL NAME

MOBILE NO

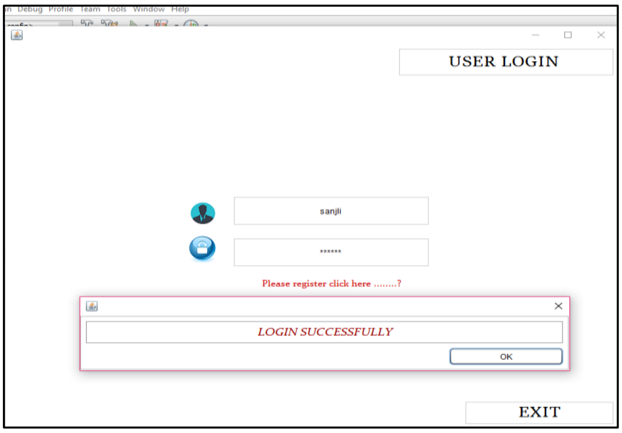
EMAIL ID

PASSWORD

REGISTER

Fig. 9: Registration

B. To verify Login name and password



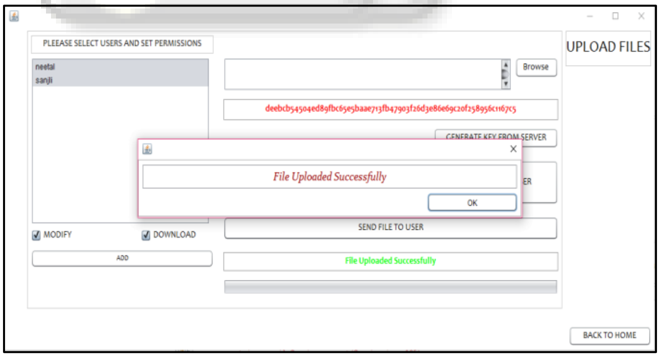
LOGIN SUCCESSFULLY

Fig. 10: Login Page

C. User account on successful login

Fig. 11: Homepage after Login

D. File upload for sharing among users by setting access permissions for each user



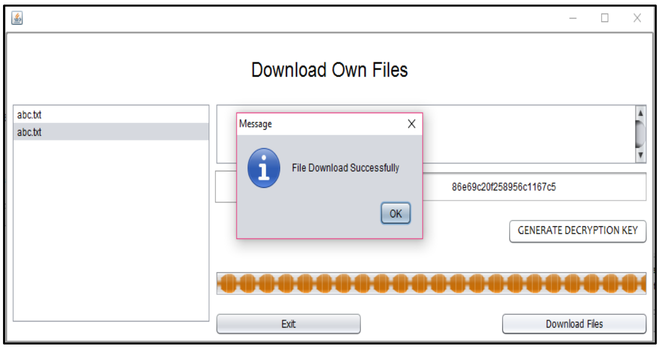
deebcb54504ed89fbc65e5baae713fb47903f26d3e86e69c20f258956c1167c5

File Uploaded Successfully

File Uploaded Successfully

Fig. 12: File Upload

E. Download own files

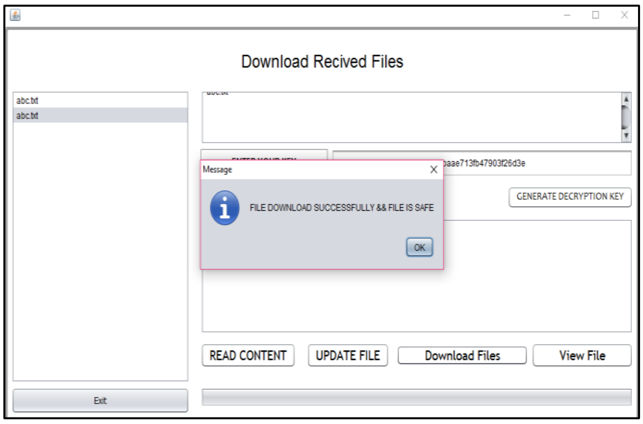


File Download Successfully

86e69c20f258956c1167c5

Fig. 13: File Download (Own Files)

F. Download the received files by user's part of key

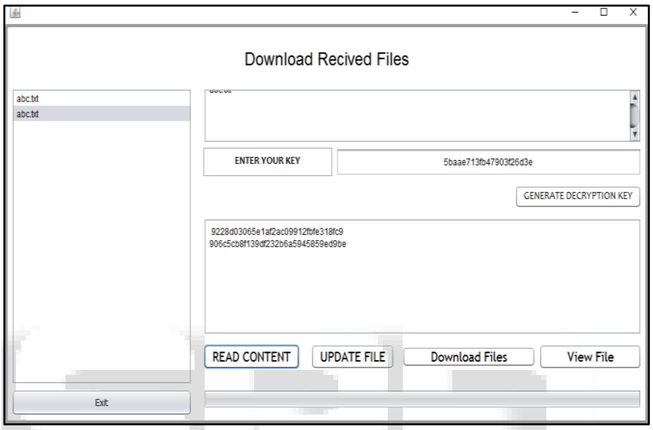


baae713fb47903f26d3e

FILE DOWNLOAD SUCCESSFULLY && FILE IS SAFE

Fig. 14: File Download (Received Files)

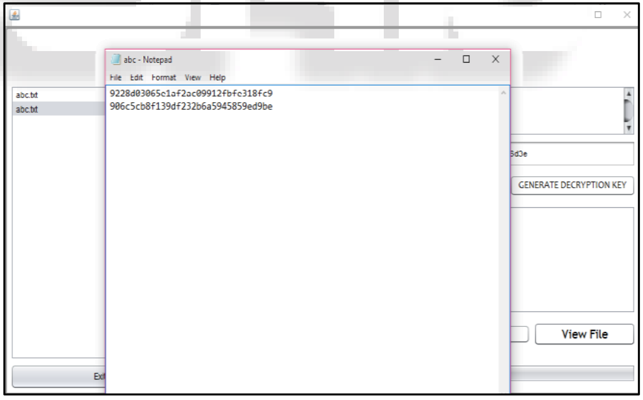
G. Read and view the received file by the user.



5baae713fb47903f26d3e

9228d03065e1af2ac09912fbfe318fc9
906c5cb8f139df232b6a5945859ed9be

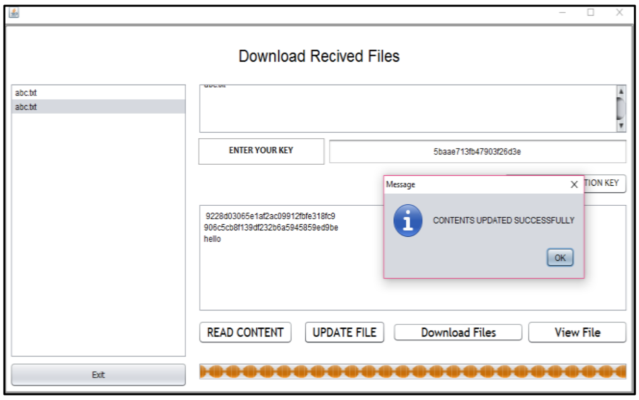
Fig. 15: Read Contents of File



```
9228d03065c1af2ac09912fbfc318fc9
906c5cb8f139df232b6a5945859ed9be
```

Fig. 16: View File

H. Update the contents of the received file



5baae713fb47903f26d3e

9228d03065e1af2ac09912fbfe318fc9
906c5cb8f139df232b6a5945859ed9be
hello

CONTENTS UPDATED SUCCESSFULLY

Fig. 17: Update File

VI. CONCLUSION

In our paper, we propose the SEDASC methodology which is cloud storage security scheme for group data. It provides data confidentiality, data integrity, internal threat security, secure data sharing without re-encryption, access control for malicious insiders. It also provides assured deletion by deleting the parameters required to decrypt a file. The encryption and decryption process are carried out at the CS that is a trusted third party in the SeDaSC methodology. The proposed methodology can be also employed to mobile cloud computing as the compute-intensive tasks are performed at the CS. In the future, the proposed methodology can be extended by limiting the trust level in the CS which will further enhance the system to cope with insider threats.

REFERENCES

[1] T. Jiang, X. Chen, and J. Ma, “Public Integrity Auditing for Shared Dynamic Cloud Data with Group User Revocation,” IEEE Transactions on Computers vol: pp no: 99 year 2015.

[2] M. Ali, R. Dhamotharan, E. Khan, S. U. Khan, A. V. Vasilakos, K. Li, and A. Y. Zomaya, “SeDaSC: Secure Data Sharing in Clouds,” IEEE Systems Journal year 2015.

[3] S. Seo, M. Nabeel, X. Ding, and E. Bertino, “An Efficient Certificateless Encryption for Secure Data Sharing in Public Clouds,” IEEE Trans. Knowl. Data Eng., vol. 26, no. 9, Sep. 2014.

[4] S. Sundareswaran, A. C. Squicciarini, and D. Lin, “Ensuring Distributed Accountability for Data Sharing in the Cloud,” IEEE Transactions on Dependable And Secure Computing vol.9 no.4 year 2012.

[5] C. Chu, S. S. M. Chow, Wen-GueyTzeng, Jianying Zhou, and Robert H. Deng, “Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage,” IEEE Transactions on Parallel And Distributed Systems year 2013.

[6] C. Yang and J. Lai, “Protecting Data Privacy and Security for Cloud Computing Based on Secret Sharing,” International Symposium on Biometrics and Security Technologies year 2013.

[7] Z. Zhu and R. Jiang, “A Secure Anti-Collusion Data Sharing Scheme for Dynamic Groups in the Cloud,” IEEE Transactions on Parallel And Distributed Systems, vol. 27, no. 1, Jan. 2016.

[8] J. Wei, W. Liu and X. Hu, “Secure Data Sharing in Cloud Computing Using Revocable-Storage Identity-Based Encryption,” IEEE Transactions on Cloud Computing vol. 14, no. 8, Aug. 2015.