Innovative Science and Technology Publications

International Journal of Future Innovative Science and Technology
ISSN: 2454- 194X Volume - 1, Issue - 1

Manuscript Title

SECURE DATA SHARING WITHOUT KEY LEAKAGE IN CLOUD STORAGE USING HYBRID ALGORITHM

Sakthivelmurugan V
(Assistant Professor),
Dept. of Information Technology,
PSNA College of Engineering & Technology, Dindugal,
TamilNadu.
ebenezer88@gmail.com

Balasubramanian P (Faculty),
Indian Institute of Information Technology, Srirangam,
Tiruchirappalli,
TamilNadu.
pearsbala@gmail.com

Shahana R Mtech IT (Final Year),
Dept. of Information Technology,
J. J. College of Engineering & Technology, Tiruchirappalli,
TamilNadu.
shahanabtech@gmail.com

May – 2015

www.istpublications.com

SECURE DATA SHARING WITHOUT KEY LEAKAGE IN CLOUD STORAGE USING HYBRID ALGORITHM

Sakthivelmurugan V
(Assistant Professor),
Dept. of Information Technology,
PSNA College of Engineering & Technology, Dindugal,
TamilNadu.
ebenezer88@gmail.com

Balasubramanian P (Faculty),
Indian Institute of Information Technology, Srirangam,
Tiruchirappalli,
TamilNadu.
pearsbala@gmail.com

Shahana R Mtech IT (Final Year),
Dept. of Information Technology,
J. J. College of Engineering & Technology, Tiruchirappalli,
TamilNadu.
shahanabtech@gmail.com

ABSTRACT

Cloud storage is a medium to store data should be always available to access with more security. To provide security for outsourced data, this is encrypted by using Hybrid Algorithm and then stores it in Cloud Storage Environment. Hybrid Algorithm is a combination of any two or more algorithms that solve the same problem. The main advantage of storing data in the Cloud is possible to share the data among any users more easily. In this paper we show how to securely, efficiently, flexibly share the data with others in the cloud storage. The exclusivity is that one can cumulative any set of secret key and make them as compressed as a single constant size key. In other words, it is a constant size aggregate key for the choice of cipher text set in Cloud Storage but the other encrypted file in Cloud remains confidential. The aggregate key generates into a file format and then sends to the user to avoid key leakage. With this experiment, we show that securely, effectively, flexibly share data among others in Cloud Storage using Hybrid Algorithm.

Index Terms – *Cloud Storage, Hybrid Algorithm (Blowfish, AES), Aggregate Key, Data Sharing.*

1. Introduction

Cloud Computing is an infant one still people are doing more research on it. It is an amazing technology to access resources any were, any time and any place through internet. Cloud Computing is migrated from Parallel & Distributed computing, Grid computing, Pervasive computing. Parallel & Distributed computing is a group of computer connected together. It breaks a task into number of pieces and executes them parallel in a different machine at a time and produce a desire output. Grid computing is a group of resources from various locations to reach a common goal. Cloud computing is a recently evolved computing terminology based on utility and consumption of computing resources. It is a delivery of computing as a service rather than a product. Nowadays people are generated their own data every day. They need to store all the data in some storage. But they find lack of memory to store those data. Moving to cloud storage avoid such a problem [1]. It is easy to apply cloud storage for free account to photo album, file sharing with storage size more than 25GB or few dollars for more than 1TB. Generally cloud storage is a backup of Big Data management. Cloud storage is easy to share data among users and store large amount of data comparably at low cost.

Consider data privacy, information resides in cloud, it presents a unique challenge because

cloud computing resources are distributed, making it difficult to know where data is located and who has access at any given time [2]. Regarding availability of files, there are a series of cryptographic schemes which go as far as allow checking the availability of files on behalf of the data owner without leaking anything about the data or without compromising the data owners anonymity. Mostly cloud users are not hold the strong belief on cloud server security services in terms of confidentiality. So the users are encouraged to encrypt their data with their own keys before uploading them to the any cloud server.

Data sharing is an important functionality in cloud storage. It provides an abundant of benefits to the user. For example, an organization shares 74% of their data with customers and 64% of their data with suppliers. With multiple users from several organizations contributing data to the cloud, the time and cost will be much less compared to having manually exchange the data. The challenging problem in sharing of data to the others is how effectively share encrypted data. Users should be able to delegate the access rights of the sharing data to others so that they can access these data from the server directly. Instead user can download the encrypted data from the storage, decrypt them, then send them to others for sharing, but it loses the value of cloud storage.

Consider any cloud storage, user A puts all her private data, and does not want to expose all their data to everyone. Because of various data leakage possibility user A not relying on the privacy mechanism provided by cloud storage, so she encrypt all the data using her own keys before uploading it. For example one day Alice's friend Bob asks her to share the photos taken over all these years which bob appeared in. Alice can then use the share function, but the problem now is how to delegate the decryption rights for these photos to Bob. A possible option Alice can choose to securely send Bob the secret keys involved. Naturally, there are two ways for her to share photos under the traditional encryption are:

- Alice encrypts all files with a single encryption key and gives Bob the corresponding secret key directly.
- Alice encrypts files with distinct keys and sends Bob the corresponding secret keys.

Obviously, the first method is inadequate all the un-chosen data may be also leaked to Bob. For the second method, there are practical concerns on efficiency. It is difficult with when the number of shared photo increased says hundred. It consumes time encrypt each data separately. And also require more space to store key for the encrypted data. So the cost and complexity increases with the number of decryption key to be shared.

Finally the best solution for the above problem is that Alice encrypts files with distinct public keys, but only sends Bob a single constant size decryption key. Since the decryption key should be sent via a secure channel and kept secret, small key size is always desirable.

2. LITERATURE SURVEY

This section we compare our basic KAC scheme with other possible solutions on sharing in secure cloud storage. They are:

Digital Identity Management System:

We discuss about many security management system. Here the Digital Identity Management scheme is used to provide security to the data which is stored in the cloud storage. It includes the registrar, the user and the cloud service provider [1]. Registrar provides security to the data which is stored in the cloud storage. Also maintains the activity of the user. Cloud service provider provides space to store the data of any user comparably low cost. User stores their own data and shares that data with anybody they like.

DISADVANTAGES:

- Registrar remains online to provide service to the user.
- It is very difficult to place someone to keep monitors the activity and services to the user.

Privacy Preserving Public Auditing System:

We consider a cloud data storage service involving three different entities [3]. They are the cloud user, the third party auditor and the cloud server. Cloud user who has large amount of data files to be stored in the cloud; the cloud server which is managed by the cloud service provider to provide data storage service and has significant storage space and computation resources; the third party auditor who has expertise and capabilities that cloud users do not have and is trusted to assess the cloud storage service reliability on behalf of the user upon request. Users rely on cloud server for cloud data storage and maintenance. They may also dynamically interact with cloud server to access and update their stored data for various application purposes. Users no longer possess their data locally, needs to ensure that their data are being correctly stored and maintained. To know the correctness of data which are verified periodically? Here third party auditing for ensuring the storage integrity of their outsourced data.

We assume the data integrity threats toward users data can come from both internal and external attacks at cloud server. These may include: software bugs, hardware failures, bugs in the network path, economically motivated hackers, malicious or accidental management errors, etc. For their own benefits, such as to maintain reputation, CS might even decide to hide these data corruption incidents to users. Using third-party auditing service provides a cost-effective method for users to gain trust in cloud. We assume the third party auditing, who is in the business of auditing, is reliable and independent. However, it may harm the user if the TPA could learn the outsourced data after the audit. Also we assume that neither CS nor TPA has motivations to collude with each other during the auditing process. To authorize the CS to respond to the audit delegated to TPA's, the user can issue a certificate on TPA's public key, and all audits from the TPA are authenticated against such a certificate.

Here proposed a privacy-preserving public auditing system for data storage security in cloud computing. It utilize the homomorphism linear authenticator and random masking to guarantee that the TPA would not learn any knowledge about the data content stored on the cloud server during the efficient auditing process, which not only eliminates the burden of cloud user from the tedious and possibly expensive auditing task, but also alleviates the users' fear of their outsourced data leakage.

To enable privacy-preserving public auditing for cloud data storage, design should achieve the following security and performance guarantees:

- **Public audit ability,** To allow TPA verifies the correctness of the cloud data on demand without retrieving a copy of the whole data or introducing additional online burden to the cloud users.
- **Storage correctness,** To ensure that there exist no cheating cloud server that can pass the TPA's audit without indeed storing users' data intact.
- **Privacy preserving,** To ensure that the TPA cannot derive users' data content from the information collected during the auditing process.
- **Batch auditing,** enables TPA secure and efficient with multiple auditing simultaneously.
- **Lightweight,** To allow TPA perform auditing with minimum communication and computation overhead.

DISADVANTAGES:

- It extends to multi-user's setting, where the TPA can perform multiple auditing tasks in a batch manner for better efficiency.
- It is not possible to rely on third party auditing all the time may threaten sometime.

Security Mediator System

The system model consists of four entities, including the data owner, the data user, the cloud server and the security mediator (SEM). Data owners generate data and upload them to the cloud for sharing [4]. It is a single owner scenario, which means each data file stored in the cloud is managed and modified by only a single data owner. Data users are able to access data uploaded

by data owners, but they are not allowed to modify data in the cloud. Data owners and data users are sometimes collectively termed as cloud users. Cloud server provides data storage and sharing services to data owners and data users. Both the cloud server and data users are public verifiers, who are not the owner of data but need to verify data integrity when it is necessary. Security mediator (SEM) provides security services to data owners by generating signatures on data for owners before these data are outsourced to the cloud.

Due to the existence of external and internal attacks in the cloud, cloud users remain concern the integrity of their data in the cloud. On the other hand, data owners want to preserve not only their identity privacy but also data privacy when they create the data to be shared. To address the above security and privacy threats, cloud storage system should achieve the following objectives:

- **Public Verifiability,** The integrity of cloud data, which is outsourced by data owners, should be verifiable by a public verifier.
- **Verification Efficiency,** A public verifier who does not possess cloud data, should be able to verify the integrity of cloud data without retrieving the entire data from the cloud server.
- **Anonymity,** The identity of a data owner should not reveal to a public verifier during the verification of data integrity. In addition, a SEM should not be able to reveal the identity of a data owner based on cloud data and corresponding signatures.
- **Data Privacy,** During the generation of signatures for a data owner, other parties, even a SEM, should not be able to learn the content of data that the data owner wants to sign.

DISADVANTAGES:

- Security mediator may liable sometimes threat the data stored.
- To extend the multi SEM model, this can avoid the potential single point of failure in the single SEM scenario.

3. SYSTEM MODEL

The system model introduced in this paper consists of two entities, including cloud users and cloud server. Cloud users generate data and upload them to the cloud for sharing. Also cloud server can upload their own and share with the users. Cloud server provides data storage and sharing services to cloud users and their own use also. The cloud server and cloud user activities described in the figure 1 as shown below:

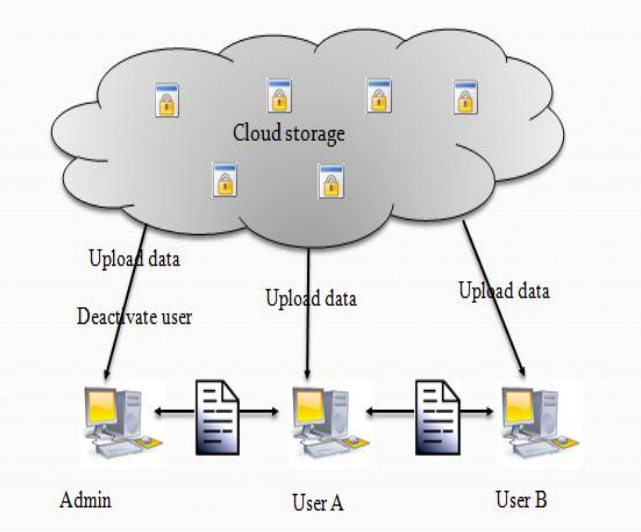

Figure 1 System Model

In general, Cloud Storage is used to store their data in remote and enjoy the on demand high quality applications and services from a shared pool of configurable computing resources, without the burden of local data storage and maintenance. There is a privacy concern in data which is to be stored in remote. To avoid a security issues each data should be encrypted with any algorithm before upload in a cloud storage environment. It includes activity of Admin, public key encryption, file sharing, Aggregate key.

Admin

Admin is to monitor and maintain the activities of the cloud user. Both the admin and cloud user can upload the file and delegate access rights to the needed user. Admin is mainly for authenticate the user and maintain the user profile. It consists of user registration like username, mobile number, email, and password. Each user contains unique username for the profile which searched with

present data for find match. If any username matches with available data it will provide a chance to alter it. Every user should login to their corresponding account with his unique username and password. Admin always monitor the user which available in the cloud storage. He has rights to deactivate an account of unauthorized user. Once the admin deactivate any users account he loses his rights to access the data and the cloud storage. But the data of the deactivated user still remain in the cloud storage for a period of time. Later it will be deleted to consume space and avoid duplication.

Public-Key Encryption

After the user registration successfully, they can access the delegated files by any user in the cloud. Before store the files into the cloud it should be encrypted by using Hybrid Algorithm [5]. It is the combination of more two algorithms to enhance the security of the files i.e. Advanced Standard Encryption (AES) and Blowfish Algorithm. Advanced Encryption Standard (AES) is a block cipher with a block length of 128 bits [6]. Here used with 128 bits key length for reducing decryption and encryption time complexity. Except for the last round in each case, all other rounds are identical. Each round of processing includes one single-byte based substitution step, a row-wise permutation step, a column-wise mixing step, and the addition of the round key. Blowfish is a symmetric block encryption algorithm. It contains key size of 32 bits. It has a block size 64 bits and 16 rounds.

In general, public key encryption known as asymmetric key encryption, public key encryption uses two different keys at once a combination of a private key and a public key. The private key is known only to your computer, while the public key is given by your computer to any computer that wants to communicate securely with it. To decode an encrypted message, a computer must use the public key, provided by the originating computer, and its own private key. Here using the combination of Advanced Standard Encryption (AES) and Blowfish algorithm to enhance the security to the files before store it into the cloud storage. It possible to encrypts any files using Hybrid Algorithm. It may be a text, image or video file is encrypted using Hybrid Algorithm. Both admin and any authorized users can encrypt their own file before store it into the cloud storage. With this user can store and share their files with others much more easily.

File Sharing

Data sharing is important functionality in cloud storage. It is very easy to share a data which is available in cloud. Once the files stored into cloud storage, any authorized users can access the files after get access privilege from owner of the file. In general, file sharing is the public or private sharing of computer data or space in a network with various levels of access privilege [7]. File sharing allows a number of people to use the same file to read or view it. With the help of cloud storage any authorized user can share number of files effortlessly. User can upload their public or private data to the cloud. Public data is viewed by any authorized user and the private data is viewed only by permitted user, others cannot view. Even though all the data are available, users are only allowed to view permitted files. Cloud storage acts as a personal storage with some security privilege to the confidential files. But the difference is data should be stored in remote.

Aggregate Key

Finally, the group of files is sharing by using Key Aggregate Cryptosystem (KAC). It encrypts a message not only under a public-key, but also under an identifier of cipher text called class. That means the cipher texts are further categorized into different classes [8]. The key owner holds a master-secret called master-secret key, which can be used to extract secret keys for different classes. The extracted key can be an aggregate key which is as compact as a secret key for a single class, but aggregates the power of many such keys i.e., the decryption power for any subset of cipher text classes. A key-aggregate encryption consists of five polynomial time algorithm as follows:

- **Setup:** executed by the data owner to setup an account on an un-trusted server

- **Key Gen:** executed by the data owner to randomly generate a public/master-secret key pair

- **Encrypt:** executed by anyone who wants to encrypt data. On input a public-key, an index denoting the cipher text class, and a message, it outputs a cipher text

- **Extract:** executed by the data owner for delegating the decrypting power for a certain set of cipher text classes to a delegate. On size key selected cipher text can be shared to user that he wants the files. The delegated user can only access to view the selected files. The aggregate constant size key generates into a file format. The generated key file is sent to the user through any channel. The user download file from the channel. With the key file he can decrypt the delegated file from the owner and the other files out of the class remain confidential.input the master-secret key and a set of indices corresponding to different classes, it outputs the aggregate key for set.

- **Decrypt:** executed by a delegate who received an aggregate key generated by Extract. It outputs the decrypted result.

Here the sizes of public-key, master-secret key, aggregate key and cipher text are all of constant size. With the constant.

4. CONCLUSION

With the popularity of outsourcing of data to the cloud storage, data privacy is central concern to the user. With more mathematical tools and cryptographic schemes are more flexible and often involve multiple keys for single application. With multiple keys, it occupies more space to store such keys find difficult. To avoid such a problem, consider how to compress secret keys in public key cryptosystems which support delegation of secret keys for different cipher text classes in cloud storage. No matter which one among the power set of classes, the delegate can always get an aggregate key of constant size. In the future extends the number of predefined bound of the cipher text of given set of class.

REFERENCES

[1] S.S.M. Chow, Y.J. He, L.C.K. Hui, and S.-M. Yiu, "SPICE – Simple Privacy-Preserving Identity-Management for Cloud Environment," Proc. 10th Int'l Conf. Applied Cryptography and Network Security (ACNS), vol. 7341, pp. 526-543, 2012.

[2] L. Hardesty, Secure Computers Aren't so Secure. MIT press, http://www.physorg.com/news176107396.html, 2009.

[3] C. Wang, S.S.M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy- Preserving Public Auditing for Secure Cloud Storage," IEEE Trans. Computers, vol. 62, no. 2, pp. 362-375, Feb. 2013.

[4] M.J. Atallah, M. Blanton, N. Fazio, and K.B. Frikken, "Dynamic and Efficient Key Management for Access Hierarchies," ACM Trans. Information and System Security, vol. 12, no. 3, pp. 18:1-18:43, 2009.

[5] S.S.M. Chow, Y.J. He, L.C.K. Hui, and S.-M. Yiu, "SPICE – Simple Privacy-Preserving Identity-Management for Cloud Environment," Proc. 10th Int'l Conf. Applied Cryptography and Network Security (ACNS), vol. 7341, pp. 526-543, 2012.

[6] L. Hardesty, Secure Computers Aren't so Secure. MIT press, http://www.physorg.com/news176107396.html, 2009.

[7] C. Wang, S.S.M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy- Preserving Public Auditing for Secure Cloud Storage," IEEE Trans. Computers, vol. 62, no. 2, pp. 362-375, Feb. 2013.

[8] M.J. Atallah, M. Blanton, N. Fazio, and K.B. Frikken, "Dynamic and Efficient Key Management for Access Hierarchies," ACM Trans. Information and System Security, vol. 12, no. 3, pp. 18:1-18:43, 2009.

[9] D. Boneh, X. Boyen, and E.-J. Goh, "Hierarchical Identity Based Encryption with Constant Size Ciphertext," Proc. Advances in Cryptology Conf. (EUROCRYPT '05), vol. 3494, pp. 440-456, 2005.

[10] T.H. Yuen, S.S.M. Chow, Y. Zhang, and S.M. Yiu, "Identity-Based Encryption Resilient to Continual Auxiliary Leakage," Proc. Advances in Cryptology Conf. (EUROCRYPT '12), vol. 7237, pp. 117-134, 2012.

[11] C.-K. Chu and W.-G. Tzeng, "Identity-Based Proxy Re-encryption without Random Oracles," Proc. Information Security Conf. (ISC '07), vol. 4779, pp. 189-202, 2007.

[12] F. Guo, Y. Mu, and Z. Chen, "Identity-Based Encryption: How to Decrypt Multiple Ciphertexts Using a Single Decryption Key," Proc. Pairing-Based Cryptography Conf. (Pairing '07), vol. 4575, pp. 392-406, 2007.

[13] W.-G. Tzeng, "A Time-Bound Cryptographic Key Assignment Scheme for Access Control in a Hierarchy," IEEE Trans. Knowledge and Data Eng., vol. 14, no. 1, pp. 182-188, Jan./Feb. 2002.

[14] Y. Sun and K.J.R. Liu, "Scalable Hierarchical Access Control in Secure Group Communications," Proc. IEEE INFOCOM '04, 2004.

[15] B. Alomair and R. Poovendran, "Information Theoretically Secure Encryption with Almost Free Authentication," J. Universal Computer Science, vol. 15, no. 15, pp. 2937-2956, 2009.

[16] F. Guo, Y. Mu, and Z. Chen, "Identity-Based Encryption: How to Decrypt Multiple Ciphertexts Using a Single Decryption Key," Proc. Pairing-Based Cryptography Conf. (Pairing '07), vol. 4575, pp. 392-406, 2007.

[17] S.S.M. Chow, Y. Dodis, Y. Rouselakis, and B. Waters, "Practical Leakage-Resilient Identity-Based Encryption from Simple Assumptions," Proc. ACM Conf. Computer and Comm. Security, pp. 152-161, 2010.

[18] B. Wang, S.S.M. Chow, M. Li, and H. Li, "Storing Shared Data on the Cloud via Security-Mediator," Proc. IEEE 33rd Int'l Conf. Distributed Computing Systems (ICDCS), 2013.

[19] F. Guo, Y. Mu, Z. Chen, and L. Xu, "Multi-Identity Single-Key Decryption without Random Oracles," Proc. Information Security and Cryptology (Inscrypt '07), vol. 4990, pp. 384-398, 2007.

[20] G. Ateniese, R. Burns, R. Curtmola, J. Herring, O. Khan, L. Kissner, Z. Peterson, and D. Song, "Remote Data Checking Using Provable Data Possession," ACM Trans. Information and System Security, vol. 14, article 12, May 2011.

[21] H. Shacham and B. Waters, "Compact Proofs of Retrievability," Proc. 14th Int'l Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT '08), 2008.

[22] K. Ren, C. Wang, and Q. Wang, "Security Challenges for the Public Cloud," IEEE Internet Computing, vol. 16, no. 1, pp. 69–73, 2012.

[23] D. Boneh, R. Canetti, S. Halevi, and J. Katz, "Chosen-Cipher text Security from Identity-Based Encryption," SIAM J. Computing, vol. 36, no. 5, pp. 1301-1328, 2007.

[24] R. Canetti and S. Hohenberger, "Chosen-Ciphertext Secure Proxy Re-Encryption," Proc. 14th ACM Conf. Computer and Comm. Security (CCS '07), pp. 185-194, 2007.

[25] B. Wang, S.S.M. Chow, M. Li, and H. Li, "Storing Shared Data on the Cloud via Security-Mediator," Proc. IEEE 33rd Int'l Conf. Distributed Computing Systems (ICDCS), 2013.

[26] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and Verifiably Encrypted Signatures from Bilinear Maps," Proc. 22nd Int'l Conf. Theory and Applications of Cryptographic Techniques (EUROCRYPT '03), pp. 416-432, 2003.

[27] M.J. Atallah, M. Blanton, N. Fazio, and K.B. Frikken, "Dynamic and Efficient Key Management for Access Hierarchies," ACM Trans. Information and System Security, vol. 12, no. 3, pp. 18:1-18:43, 2009.

[28] S.S.M. Chow, C.-K. Chu, X. Huang, J. Zhou, and R.H. Deng, "Dynamic Secure Cloud Storage with Provenance," Cryptography and Security, pp. 442-464, Springer, 2012.

[29] F. Guo, Y. Mu, and Z. Chen, "Identity-Based Encryption: How to Decrypt Multiple Ciphertexts Using a Single Decryption Key," Proc. Pairing-Based Cryptography Conf. (Pairing '07), vol. 4575, pp. 392-406, 2007.

[30] S.S.M. Chow, Y. Dodis, Y. Rouselakis, and B. Waters, "PracticalLeakage-Resilient Identity-Based Encryption from Simple Assumptions," Proc. ACM Conf. Computer and Comm. Security, pp. 152-161, 2010.