INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

SECURE SHARING AND AUDITING OF PRIVATE DATA IN PUBLIC CLOUD

Ms. Megha Jain[#1], Mrs. Swati Namdev[*2]

Department of Computer Science Career College, Bhopal
jainmegha115@gmail.com
swati.tailor@gmail.com

ABSTRACT: -The advantages of storing important data in cloud storage are cost effective by use of shared computing resources and low infrastructure costs and high availability. However, users is not aware with physical structure and location of the outsourced data, it makes the data integrity protection in cloud computing a challenging task, especially for users with limited computing resources. Moreover, users should be able to access cloud storage as if it is local disk storage, without worrying about the need to verify its integrity. Thus, provide auditability for cloud storage is very important, so that users can ask to a third-party auditor (TPA) to check the integrity of outsourced data and use without worried. To create a secure and effective TPA, the auditing process should not allowed any new vulnerabilities toward user data privacy, and introduce no additional online overheads to user. In this paper, we propose a secure cloud storage system that supports privacy-preserving public auditing. We allow TPA to perform audits for multiple users simultaneously and efficiently. Extensive security and performance analysis show the proposed schemes are provably secure and highly efficient.

1. INTRODUCTION

As the world of computing got more advanced the ways for sharing data started becoming cheaper and cheaper. In recent years a new term has evolved call "Cloud" which is provided by different provides, and which is nothing but facility or service of different resources or components like hardware, platform, storage's, software etc, and it is gaining importance because it frees the user from maintenance perspective on a investment of some money for the use of these services provided by cloud service providers. Now to provide such service to the client, naturally the provider's must have and rather can have access to resources which are used by the people/clients. Among the reasons these access are greatly required are for maintenance perspective. And definitely since billions of clients will be thinking about using such service, the infrastructure ought to be capable enough to support them, and these resources ought to be shared between billions of client's. Service availability, data synchronization between different devices, availability of data via any devices which includes browser facility make cloud more attractive. Now since the info gets shared or stored in providers area, the client gets worried about privacy of its data, although there are certain agreements and SLA which are agreed by cloud provider and client. Now although client have a platform to generally share the info, the expense of securing his/her data or in a nutshell making its data private gets costlier.

As users no longer physically possess the storage of their data, traditional cryptographic primitives for the purpose of data security protection cannot be directly adopted [2]. In particular, simply downloading all the data for its integrity verification is not a practical solution due to the expensiveness in I/O and transmission cost across the network. Besides, it is often insufficient to detect the data corruption only when accessing the data, as it does not give users correctness assurance for those unaccessed data and might be too late to recover the data loss or damage.

Considering the large size of the outsourced data and the user's constrained resource capability, the tasks of auditing the data correctness in a cloud environment can be formidable and expensive for the cloud users [3], [4]. Moreover, the overhead of using cloud storage should be minimized as much as possible, such that a user does not need to perform too many operations to use the data (in additional to retrieving the data). In particular, users may not want to go through the complexity in verifying the data integrity. Besides, there may be more than one user accesses the same cloud storage, say in an enterprise setting. For easier management, it is desirable that cloud only entertains verification request from a single designated party.

Recently, the notion of public auditability has been proposed in the context of ensuring remotely stored data integrity under different system and security models [5], [7], [2], [4]. Public auditability allows an external party, in addition to the user himself, to verify the correctness of remotely stored data. However, most of these schemes [5], [7], [4] do not consider the privacy protection of users' data against external auditors. Indeed, they may potentially reveal user's data to auditors. This severe drawback greatly affects the security of these protocols in cloud computing. From the perspective of protecting data privacy, the users, who own the data and rely on TPA just for the storage security of their data, do not want this auditing process introducing new vulnerabilities of unauthorized information leakage toward their data security [8], [9]. Further demanding the outsourced data not to be leaked to external parties [6]. Simply exploiting data encryption before outsourcing [2] could be one way to mitigate this privacy concern of data auditing, but it could also be an overkill when employed in the case of unencrypted/public cloud data (e.g., outsourced libraries and scientific data sets), due to the unnecessary processing burden for cloud users. Besides, encryption does not completely solve the problem of protecting data privacy against third-party auditing but just reduces it to the complex key management domain. Unauthorized data leakage still remains possible due to the potential exposure of decryption keys. Therefore, how to enable a privacy-preserving third- party auditing protocol, independent to data encryption, is the problem we are going to tackle in this paper. Our work is among the first few ones to support privacy-preserving public auditing in cloud computing, with a focus on data storage. Besides, with the prevalence of cloud computing, a foreseeable increase of auditing tasks from different users may be delegated to TPA. As the individual auditing of these growing tasks can be tedious and cumbersome, a natural demand is then how to enable the TPA to efficiently perform multiple auditing tasks in a batch manner, i.e., simultaneously.

2. PROBLEM STATEMENT

The clients concern about data security, data integrity, and sharing data with specific band be addressed. You can find multiple means of achieving this, example encrypting data on client machine and then storing the information to cloud storage server, computing hash of the information on client machine and storing hash of data in client machine, client trying out the responsibility of sharing the trick key about encryption with specific band of people. Therefore it becomes more tedious for client to keep these information and share such information, more over in the event the device which stores such information is lost or stolen it pose a threat to the total data. Another way could be same storage cloud provider providing the service for secured sharing, hashing, encryption/decryption, but since administrative can have use of both services for maintenance, the security service provided by the cloud storage provider, the information might be compromised.

3. THE PROPOSED SCHEME

Our objective is to build a security service which will be provided with a trusted 3rd party, and would lead to providing only security services and wouldn't store any data in its system. A public auditing scheme consists of four algorithms (KeyGen, SigGen, GenProof, VerifyProof). KeyGen is a key generation algorithm that is run by the user to setup the scheme. SigGen is used by the user to generate verification metadata, which may consist of digital signatures. GenProof is run by the cloud server to generate a proof of data storage correctness, while VerifyProof is run by the TPA to audit the proof. Running a public auditing system consists of two phases,Setup and Audit:

Setup: The user initializes the public and secret parameters of the system by executing KeyGen, and preprocesses the data file F by using SigGen to generate the verification metadata. The user then stores the data file F and the verification metadata at the cloud server, and deletes its local copy. As part of preprocessing, the user may alter the data file F by expanding it or including additional metadata to be stored at server.

Audit: The TPA issues an audit message or challenge to the cloud server to make sure that the loud server has retained the data file F properly at the time of the audit. The cloud server will derive a response message by executing GenProof using F and its verification metadata as inputs. The TPA then verifies the response via VerifyProof.

3.1 Privacy-Preserving Public Auditing Scheme

Overview: To achieve privacy-preserving public auditing, we propose to uniquely integrate the homomorphic linear authenticator with random masking technique. In our protocol, the linear combination of sampled blocks in the server's response is masked with randomness generated by the server. With random masking, the TPA no longer has all the necessary information to build up a correct group of linear equations and therefore cannot derive the user's data content, no matter how many linear combinations of the same set of file blocks can be collected. On the other hand, the correctness validation of the block-authenticator pairs can still be carried out in a new way which will be shown shortly, even with the presence of the randomness. Our design makes use of a public key-based HLA, to equip the auditing protocol with public auditability. Specifically, we use the HLA proposed in [7], which is based on the short signature scheme proposed by Boneh, Lynn, and Shacham, hereinafter referred as BLS signature) [10].

4. KEY MANAGEMENT SCHMES

The process of generating, distributing, and maintaining keys are taken care by key management schemes. The key management scheme developed in this paper is a centralized key management scheme that operates between the data owner and the cloud users. The provision of an access control facility using a centralized key management system is a challenging task. This is due to the fact that key generation and distribution are more complex when messages are distributed to a group of users from the cloud's servers, as users may dynamically join or leave the multicast group. To support this dynamic and secure group communication, it is necessary to allow members to join or depart from the service at any time. When a new member joins the service, it is the responsibility of the data owner to prevent this new member from having access to prior data in order to provide backward secrecy for the earlier secure group communication. Similarly, when an existing group member leaves any group, such a member should no longer have access to data - as this access should only be available to the current group members in order to achieve forward secrecy. In order to handle the issues of forward and backward secrecy, the keys are updated whenever a member joins or leaves the service. The data owner is responsible for generating a new GK after members join or leave. After a member joins the group or leaves the group, the data owner generates a new GK and securely distributes this GK to the group's members. As a result, when a group membership change, the data owner computes a new GK and the access control vector is updated by changing the public and private information available to the data owner. After computing the access control vector, the data owner multicasts this access control vector to the current group of cloud users and each cloud user computes the new GK. The old user cannot find the group key since his/her private key is not used when sending the new group key value to the remaining users. Thus, changing the group key securely after a member joins or leaves takes only limited computation (by each user) and has low communication complexity since it can exploit multicast communication. When a user leaves the group they are excluded from all

future communications and thus will not be able to compute the new Group Key. The proposed scheme provides both forward and backward secrecy, hence the former group members cannot receive future communication and a new user cannot access previous group multicasts.

4.1 KEY GENERATION

A key generation process is responsible for generating the random private keys assigned to the registered cloud users. This process also generates and computes GKs with respect to these private keys under a common subgroup. An important issue in maintaining the integrity in communication is to propose techniques for generating a GK by the data owner and enabling the group members to independently derive this GK without revealing the identity of the individual members of the group. There are two types of techniques that are used for GK generation. In the first method, users generate their own secret keys from which they compute a common GK, which will act as a public key for a group of members. This method is a distributed key management scheme. In the second method, a trusted third party (in this case the data owner) generates the GK and distributes it to the group members in a secure way. This second method is a centralized key management scheme. In both schemes, several computations are necessary to compute the subgroup and GKs. Moreover, both schemes need to store the public parameters and various key values used for computing the GK. In order to overcome the challenges of computational complexity and minimize memory requirements, a
new key management scheme with reduced computational cost and memory requirements is needed. Therefore, this research proposes a new computationally efficient technique that uses simple mathematical functions and an optimal number of multiplications and additions in order to efficiently generate a GK. Operations such as multiplication, division, and exponential operations are expensive; however, most key management schemes primarily use multiplication, multiplicative inverse, and exponential operations and hence they require more computation. The proposed scheme minimizes this complexity by using only multiplication and addition.

4.2 KEY DISTRIBUTION

A key distribution scheme for secure group communication is responsible for distributing the private keys and GKs to the registered users in the cloud network. GKs can be distributed either by the data owner to the participating members or the members themselves will distribute the keys generated by them as necessary for computing the GK. In a centralized key management scheme, the GK is distributed by the data owner, whereas in the distributed approach any one of the group members can distribute the GK. This project focuses on centralized key distribution schemes, since the major security challenges lie mostly in the design of a new effective centralized key distribution scheme.. Moreover, all of the existing key management schemes are unsuitable for providing a group oriented service in a cloud network. Hence, a new and effective centralized key distribution scheme is needed that is suitable for providing a group oriented service in a cloud, while reducing computational complexity.

4.3 KEY RECOVERY

In a centralized key management scheme, secure multicast key recovery process is used by group members to construct the original GK computed by the data owner. In contrast, in distributed GK management, the key recovery process is used by group members to individually compute the GK based on values received from other group members. In both of these schemes, the members of the group should perform a minimum number of mathematical operations to recover the newly generated or updated GK. Moreover, the key recovery process should minimize the number of parameters needed for recovering the common GK whenever there is a change in the group's membership.

5. CONCLUSIONS

In this paper, a privacy preserving algorithm is implemented to improve the privacy preservation of each user who is accessing the data from a public cloud. In addition, a multicast key management scheme is implemented to provide an ACV for all the users who belong to a particular group. This ACV is used to compute a common

group key to perform decryption of the document on the user's side. The proposed algorithm is computation, and communication efficient for the cloud user.

6. REFERENCES

[1] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving Public Auditing for Storage Security in Cloud Computing," Proc. IEEE INFOCOM 10, Mar. 2010.

[2] A. Juels and J. Burton, S. Kaliski, "PORs: Proofs of Retrievability for Large Files," Proc. ACM Conf. Computer and Comm. Security (CCS '07), pp. 584-597, Oct. 2007.

[3] Cloud Security Alliance, "Security Guidance for Critical Areas of Focus in Cloud Computing," http://www.cloudsecurityalliance.org, 2009.

[4] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing," IEEE Trans.Parallel and Distributed Systems, vol. 22,no. 5, pp. 847-859, May 2011.

[5] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores," Proc. 14th ACM Conf. Computer and Comm. Security (CCS '07), pp. 598-609, 2007.

[6] M.A. Shah, R. Swaminathan, and M. Baker, "Privacy-Preserving Audit and Extraction of Digital Contents," Cryptology ePrint Archive, Report 2008/186, 2008.

[7] H. Shacham and B. Waters, "Compact Proofs of Retrievability," Proc. Int'l Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (Asiacrypt), vol. 5350, pp. 90-107, Dec. 2008.

[8] C. Wang, K. Ren, W. Lou, and J. Li, "Towards Publicly Auditable Secure Cloud Data Storage Services," IEEE Network Magazine,vol. 24, no. 4, pp. 19-24, July/Aug. 2010.

[9] M.A. Shah, M. Baker, J.C. Mogul, and R. Swaminathan, "Auditing to Keep Online Storage Services Honest," Proc. 11th USENIX Workshop Hot Topics in Operating Systems (HotOS '07), pp. 1-6, 2007.

[10] D. Boneh, B. Lynn, and H. Shacham, "Short Signatures from the Weil Pairing," J. Cryptology, vol. 17, no. 4, pp. 297-319, 2004.