Auditing Outsourced Data on Cloud Using HLA with Random Masking Technique

Remidicherla Rupa[1]
[1]Asst.Prof, Department of Computer Science & Engineering,
Laqshya Institute of Technology & Sciences, KHAMMAM-507001- Affiliated to JNTUH, TELANGANA-INDIA.

***Abstract* - Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage applications and services). The Cloud server allows user to upload their data on a cloud. But as user's data is stored at the remote place how users will get the confirmation about integrity of stored data. Sometimes Cloud service providers behave unfaithfully towards the cloud users regarding the status of their outsourced data. So we propose Public audit ability that allows an external party, in addition to the user himself, to verify the integrity of outsourced data on cloud. Indeed, they may potentially reveal user data information to the auditors in the auditing process leads to new vulnerabilities and additional online burden. However, most of the schemes do not consider the privacy protection of users' data against external auditors. So to securely introduce an effective TPA, we propose an aggregate 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 using pseudo random function (PRF). Also we use the HLA which is based on the short signature scheme proposed by Boneh, Lynn and Shacham (referred as BLS signature).**

***Keywords* - Cloud Computing, Data Integrity, Homomorphic Encryption, Homomorphic Linear Authentication (HLA), Zero Knowledge, Privacy Preserving, Public Auditing.**

I. INTRODUCTION

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction(NIST def).

This cloud model is composed of five essential Characteristics (On demand self service, Broad network access, Resource pooling, Rapid elasticity, Measured service), three service models (Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS), and four deployment models (public, private, community and hybrid).

II. LITERATURE SURVEY

The Cloud computing is a latest technology which provides various services through internet. The Cloud server allows user to store their data on a cloud without worrying about correctness & integrity of data. Cloud data storage has many advantages over local data storage.

User can upload their data on cloud and can access those data anytime anywhere through internet without any additional burden. The User doesn't have to worry about storage and maintenance of cloud data. But as data is stored at the remote place, here we asked three main questions:

I. How users will get the confirmation about their outsourced data integrity?
II. How to be sure that even if the data-centers of the Cloud Computing provider were attacked, my data won't be stolen or reused?
III. How can my data remain confidential and invisible even to my Cloud provider?

Our basic concept was to encrypt the data before sending them to the Cloud provider. But, this one will have to decrypt them each time he has to work on them. The client will need to provide the private key to the server to decrypt the data before execute the calculations required, which might affect the confidentiality of data stored in the Cloud.

With growing communication networks and digital communication, secure communication and data security is of paramount importance. Today one way to achieve secure communication is by the use of cryptography [1], [2], which concurrently ensures confidentiality of data in communication and in storage. For storing and accessing data securely there exist many ways which can guarantee privacy and confidentiality, such as data encryption and tamper resistance hardware. However, the problem becomes quite complex when it is required to compute publicly private data or to modify a function or algorithm in such a way that they are still executable while their privacy is ensured.

The Homomorphic Encryption method is able to perform operations on encrypted data without decrypting them our proposal is to encrypt data before sending it to the cloud provider, but to execute the calculations the data should be decrypted every time we need to work on it. Until now it was impossible to encrypt data and to trust a third party to keep them safe and able to perform distant calculations on them. So to allow the Cloud provider to perform the operations on encrypted data without decrypting them requires using the cryptosystems based on Homomorphic Encryption.

History of the Homomorphic encryption

In 1978 Ronald Rivest, Leonard Adleman and Michael Dertouzos suggested for the first time the concept of Homomorphic encryption [1]. Since then, little progress has been made for 30 years. The encryption system of Shafi Goldwasser and Silvio Micali was proposed in 1982 was a provable security encryption scheme which reached a remarkable level of safety, it was an additive Homomorphic encryption, but it can encrypt only a single bit. In the same concept in 1999 Pascal Paillier was also proposed a provable security encryption system that was also an additive Homomorphic encryption. Few years later, in 2005, Dan Boneh, Eu-Jin Goh and Kobi Nissim [9] invented a system of provable security encryption, with which we can perform an unlimited number of additions but only one multiplication[12].

During the last few years, homomorphic encryption techniques have been studied extensively and have found application in many different cryptographic protocols operating over open and untrusted networks. Untrusted networks are given only an encrypted version of the data. The network will perform computation on this encrypted data. To ensure that the encrypted data is really being processed securely was addressed by Rivest [3] through homomorphic encryption. However, this scheme has security flaws as pointed out by Brickell and Yacobi [4].

Homomorphic Encryption definition:

Homomorphic Encryption systems are used to perform operations on encrypted data without knowing the private key (without decryption), the client is the only holder of the secret key. When we decrypt the result of any operation, it is the same as if we had carried out the calculation on the raw data [12].

Definition: An encryption is homomorphic, if: from Enc(a) and Enc(b) it is possible to compute Enc(f (a, b)), where f can be: +, ×, ⊕ and without using the private key.

Among the Homomorphic encryption we distinguish, according to the operations that allows to assess on raw data, the additive Homomorphic encryption (only additions of the raw data) is the Pailler [2] and Goldwasser-Micalli [3] cryptosystems, and the multiplicative Homomorphic encryption (only products on raw data) is the RSA [4] and El Gamal [5] cryptosystems.

III. EXISTING SYSTEM

Since cloud service providers (CSP) are separate administrative entities, data outsourcing is actually relinquishing user's ultimate control over the fate of their data. As a result, the integrity of the data in the cloud is being put at risk due to the following reasons.

- Traditional cryptographic primitives for the purpose of data security protection cannot be directly adopted.
- Without a properly designed auditing protocol, encryption itself cannot prevent data from "Flowing away" towards external parties during the auditing process.
- CSP behave unfaithfully towards the cloud users regarding the status of their outsourced data.
- Infrastructures under the cloud are much more powerful and reliable than personal computing devices; they are still facing the broad range of both internal and external threats for data integrity.
- Expensiveness in I/O and transmission cost across the network

Basic schemes

A. MAC-based solution
B. Homomorphic linear authenticators (HLA).

A. MAC-based solution

- There are two possible ways to make use of MAC to authenticate the data. A trivial way is just uploading the data blocks with their MACs to the server, and sends the corresponding secret key sk to the TPA.
- Later, the TPA can randomly retrieve blocks with their MACs and check the correctness via sk.
- Apart from the high (linear in the sampled data size) communication and computation complexities, the TPA requires the knowledge of the data blocks for verification.

Disadvantages

- The number of times a particular data file can be audited is limited by the number of secret keys that must be fixed a priori. Once all possible secret keys are exhausted, the user then has to retrieve data in full to re-compute and re-publish new MACs to TPA.
- The TPA also has to maintain and update state between audits, i.e., keep track on the revealed MAC keys. Considering the potentially large number of audit delegations from multiple users, maintaining such states for TPA can be difficult and error prone.
- It can only support static data, and cannot efficiently deal with dynamic data at all. However, supporting data dynamics is also of critical importance for cloud storage systems

B. HLA-based solution

To effectively support public audit ability without having to retrieve the data blocks themselves, the HLA technique [6], [7], [8] can be used. HLAs, like MACs, are also some un forgeable verification metadata that authenticate the integrity of a data block.

The difference is that HLAs can be aggregated. It is possible to compute an aggregated HLA which authenticates a linear combination of the individual data blocks.

Disadvantages

- HLA allows efficient data auditing and consumes only constant bandwidth.
- However HLA technique may potentially reveal user data information to TPA and may violate the privacy preserving guarantee as TPA can simply solve a system of linear equations used in the HLA technique [1].
- Moreover, there are legal regulations, such as the US Health Insurance Portability and Accountability Act (HIPAA) [9], further demanding the outsourced data not to be leaked to external parties.

IV. PROPOSED SYSTEM

Public audit ability allows an external party, in addition to the user himself, to verify the correctness of remotely stored data. However, most of these schemes [6], [7], [8] do not consider the privacy protection of users' data against external To fully ensure the data integrity and save the cloud users' computation resources as well as online burden, it is of critical importance to enable public auditing service for cloud data storage, so that users may resort to an independent third party auditor (TPA) to audit the outsourced data when needed.

public key based homomorphic linear authenticator (or HLA for short) [6], [7], [8], which enables TPA to perform the auditing without demanding the local copy of data and thus drastically reduces the communication and computation overhead as compared to the straightforward data auditing approaches. By integrating the HLA with random masking, our protocol guarantees that the TPA could not learn any knowledge about the data content stored in the cloud server during the efficient auditing process. The aggregation and algebraic properties of the authenticator further benefit our design for the batch auditing.

Privacy-Preserving Public Auditing Scheme

To achieve privacy-preserving public auditing, we propose 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 based on pseudo random function (PRF).

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.

Our design makes use of a public key based HLA, to equip the auditing protocol with public audit ability. Specifically, we use the HLA proposed in [1], which is based on the short signature scheme proposed by Boneh, Lynn and Shacham (referred as BLS signature).

Though HLA with random masking solves the problem of privacy-preserving, it increases the burden of maintenance and calculation of masking information on user as well as on TPA.

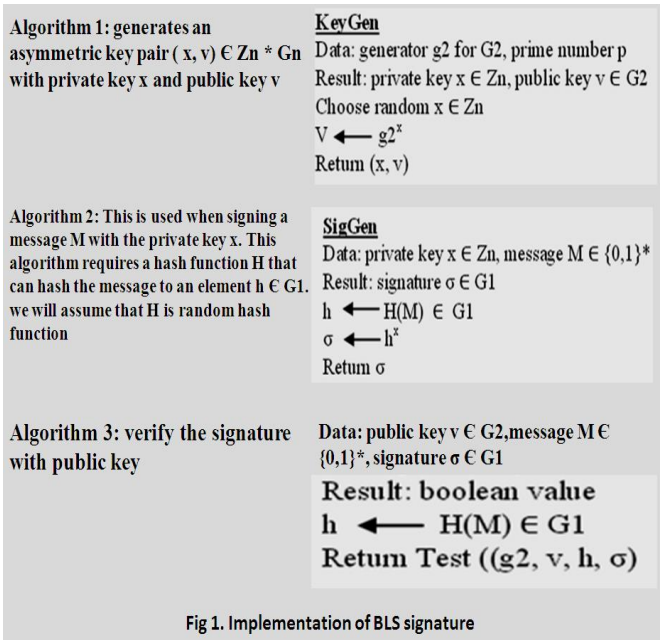

Fig 1. Implementation of BLS signature

4.2 The System and Threat Model [10]

The cloud data storage service contains 3 different entities as cloud user, Third party auditor & cloud server / cloud service provider.

Cloud user (CU): is a person who stores large amount of data or files on a cloud server.

Cloud server (CS) & Cloud service provider: is a place where we are storing cloud data and that data will be managed by the cloud service provider.

Third party auditors (TPA): TPA will do the auditing on users request for storage correctness and integrity of data.

Zero knowledge: TPA will audit the users' data without seeing the contents. It uses public key based homomorphic linear authentication (HLA) [6], [7] which allows TPA to perform auditing without requesting for user data.

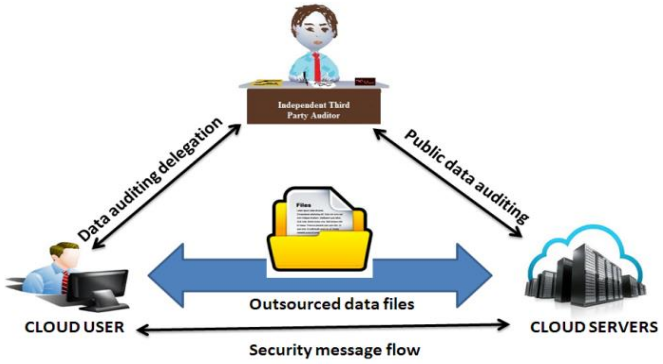

Fig 2 The architecture of cloud data storage

4.3 Design Goals

1) Public audit ability: Allows third party auditor to check data correctness without accessing local data.
2) Storage Correctness: The data stored on a cloud is as it. No data modification is done.
3) Privacy preserving: TPA can't read the users' data during the auditing phase.
4) Batch Auditing: Multiple users auditing request is handled simultaneously.
5)Light Weight: Less communication and computation overhead during the auditing phase.

V. ALGORITHMS INVOLVED

There are two phases, **setup** and **audit.**

Setup

- **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 MAC, signatures, or other related information that will be used for auditing.

Audit

- **GenProof:** is run by the cloud server to generate a proof of data storage correctness.
- **VerifyProof:** is run by the TPA to audit the proof from the cloud server.

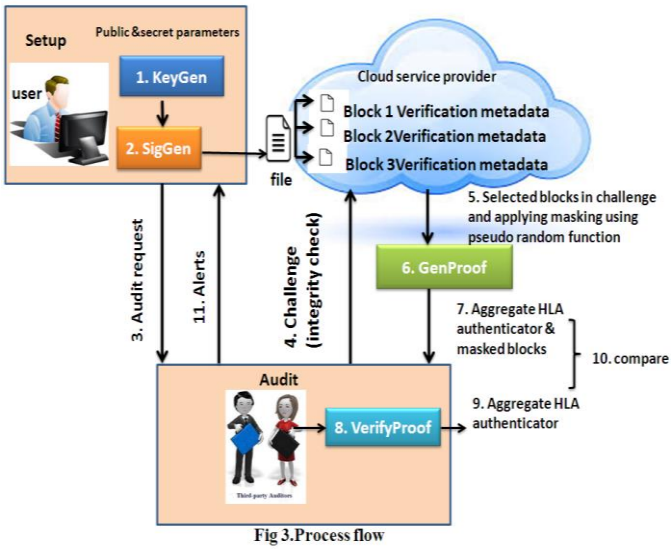

Fig 3.Process flow

Batch Auditing

It also supports batch auditing through which efficiency is improved. It allows TPA to perform multiple auditing task simultaneously and it reduces communication and computation cost. Through this scheme, we can identify invalid response. It uses bilinear signature (BLS proposed by Boneh, Lynn and Shacham) to achieve batch auditing. System performance will be faster.

Data Dynamics

It also supports data dynamics where user can frequently update the data stored on a cloud. It supports block level operation of insertion, deletion and modification. Author of [6] proposed scheme which support simultaneous public audability and data dynamics. It uses Merkle Hash Tree (MHT) which works only on encrypted data for block tag authentication.

VI. RELATED WORK

Portions of the work presented in this paper have previously appeared in [10]. TPA is stateless i.e. no need to maintain or update the state information of audit phase. Public key based homomorphic linear authentication with random masking technique is used to achieve privacy preserving public auditing. TPA checks the integrity of the outsourced data stored on a cloud without accessing actual contents. Existing research work of proof of retrievability (PoR) or Proofs of Data Possession (PDP) technique doesn't consider data privacy problem. PDP scheme first proposed by Ateniese et al. used to detect large amount corruption in outsourced data [6].

It uses RSA based Homomorphic authentication for auditing the cloud data and randomly sampling a few blocks of files. A Second technique proposed by Juels as Proofs of retrievability (PoR) allows user to retrieve files without any data loss or corruptions [15]. It uses spot checking & error correcting codes are used to ensure both "Possession" and "Retrievability". To achieve Zero knowledge privacy, researcher proposed Aggregatable Signature Based Broadcast (ASBB). It provides completeness, privacy and soundness. It uses 3 algorithms as Keygen, Gentag and Audit.

We extend our main scheme to support data dynamics and provide discussions on how to generalize our privacy preserving public auditing scheme.

VII. ACKNOWLEDGEMENT

I would like to sincerely thank my brother Mr.R.DAYRAM, M.Sc in Computer Science, Microsoft Certified Professional, for his support and encouragement towards publishing this paper**.**

VIII. CONCLUSION

In this paper, we propose a privacy-preserving public auditing for the outsourced data integrity in security in Cloud Computing. We utilize the homomorphic 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.

IX. REFERENCES

[1] Ronald L. Rivest, Leonard Adleman, and Michael L. Dertouzos. On Data Banks and Privacy Homomorphisms, chapter On Data Banks and Privacy Homomorphisms, pages 169-180. Academic Press, 1978.

[2] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In 18 The Annual Eurocrypt Conference (EUROCRYPT'99), Prague, Czech Republic , volume 1592, 1999

[3] Julien Bringe and al. An Application of the Goldwasser-Micali Cryptosystem to Biometric Authentication, Springer-Verlag , 2007.

[4] R. Rivest, A. Shamir, and L. Adleman. "A method for obtaining digital signatures and public key cryptosystems". Communications of the ACM, 21(2) :120-126, 1978. Computer Science, pages 223-238. Springer, 1999.

[5] Taher ElGamal. "A public key cryptosystem and a signature scheme based on discrete logarithms". IEEE Transactions on Information Theory, 469-472, 1985.

[6] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in *Proc. of CCS'07*, Alexandria, VA, October 2007, pp. 598–609.

[7] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in *Proc. of ESORICS'09, volume 5789 of LNCS*. Springer-Verlag, Sep. 2009, pp. 355–370.

[8] H. Shacham and B. Waters, "Compact proofs of retrievability," in *Proc. of Asiacrypt 2008*, vol. 5350, Dec 2008, pp. 90–107.

[9] 104th United States Congress, "Health Insurance Portability and Accountability Act of 1996 (HIPPA)," Online at http:// aspe.hhs.gov/admnsimp/pl104191.htm, 1996.

[10] Cong Wang, *Student Member, IEEE,* Sherman S.-M. Chow, Qian Wang, *Student Member, IEEE,* Kui Ren, *Member, IEEE,* and Wenjing Lou, *Member, IEEE"* Privacy-Preserving Public Auditing for Secure Cloud Storage".

[11] Rana M Pir, Lecturer, Leading university, Sylhet Bangladesh © 2014 IJEDR | Volume 2, Issue 4 | ISSN: 2321-9939, "Cloud Storage Security Using Encryption and Third-Party Storage Auditing Service".

[12] Maha TEBAA, Saïd EL HAJJI, Abdellatif EL GHAZI, Proceedings of the World Congress on Engineering 2012 Vol I WCE 2012, July 4 - 6, 2012, London, U.K," Homomorphic Encryption Applied to the Cloud Computing Security".

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

[14]CMerkle, "Protocols for public key cryptosystems," in Proc.of IEEE Symposium on Security and Privacy, Los Alamitos, CA, USA, 1980.

[15] A. Juels and J. Burton S. Kaliski, "Pors: Proofs of retrievability for large files," in *Proc. of CCS'07*, Alexandria, VA, October 2007, pp. 584–597.