www.ioirp.com
International Journal of Innovative Research in Computer Science and Engineering (IJIRCSE)
ISSN: 2394-6364, Volume – 1, Issue – 4. March 2015

Privacy Preserving Two Layer Encryption Mechanisms

Shiby Rachel Titus[1], D.Namachivayam[2]
[12]Computer Science / SSM College of Engineering / Anna University / India
[1]shibytitus@gmail.com, [2]namachivayam.d@live.com,

Abstract— Cloud computing allows users to store their datas in cloud and those datas can be shared by different users.So the integrity of cloud data is subject to doubt.Current approach used privacy-preserving mechanism that supports public auditing on shared data stored in the cloud. They exploited ring signatures to compute verification metadata needed to audit the correctness of shared data. With this, the identity of the signer in shared data is also kept private from public verifiers. Since the traditional approach is based on ring signatures, where the identity of the signer is unconditionally protected, it does not support traceability.But designing an efficient public auditing mechanism with the capabilities of preserving identity privacy and supporting traceability is still open.So here, we propose a traceability mechanism which means the ability for the group manager (i.e., the original user) to reveal the identity of the signer based on verification metadata in some special situations and data freshness(cloud possess the latest version of data) is also achieved.

Keywords— Traceability, Privacy, Encryption, Access control

I. INTRODUCTION

Cloud computing platforms provide users scalable data storage services with a low cost than traditional approaches. The integrity of data is subject to doubt due to human errors and hardware or software failures. Therefore, the integrity of cloud data should be verified without any data utilization and without downloading the entire cloud. Traditionally, the data integrity is verified by retrieving the entire data from the cloud and then the correctness of signature is checked. However the efficiency of using this method on cloud data is in doubt [3].

The main reason is that normally the size of cloud data is large. Downloading the entire cloud data to verify data integrity will cost or even waste user's amounts of computation and communication resources, especially when data have been corrupted in the cloud. Besides many uses of cloud data do not necessarily need users to download the entire cloud data to local devices. It is because cloud providers, such as Amazon, can offer users computation services directly on large-scale data that already existed in the cloud.

Recently, many mechanisms [3], [4] have been proposed to allow not only a data owner itself but also a public verifier to efficiently perform integrity checking without downloading the entire data from the cloud, which is referred to as public auditing [2]. In these mechanisms, data is divided into many small blocks, where each block is independently signed by the owner; and a random combination of all the blocks instead of the whole data is retrieved during integrity checking [3]. A public verifier could be a data user who would like to utilize the owner's data via the cloud or a third-party auditor (TPA). Moving a step forward, Wang et al. designed an advanced auditing mechanism [2] (named as WWRL in this paper),so that during public auditing on cloud data, the content of private data belonging to a personal user is not disclosed to any public verifiers. That is, there is a leakage of identity privacy.

Failing to preserve identity privacy on shared data during public auditing will reveal significant confidential information to public verifiers. To solve the above privacy issue on shared data, a novel privacy-preserving public auditing mechanism has been found. Here Ring signature [9] is exploited to construct homomorphic authenticators, so that a public verifier is able to verify the integrity of shared data without retrieving the entire data, while the identity of the signer on each block in shared data is kept private from the public verifier.

In this paper, we propose Traceability mechanism which means the ability for the group manager (i.e., the original user) to reveal the identity of the signer based on verification metadata in some special situations and data freshness is also achieved.

II. PROBLEM STATEMENT

There are two types of users in a group: the original user and a number of group users. The original user initially creates shared data in the cloud, and shares it with group users. Both the original user and group users are members of the group. Every member of the group is allowed to access and modify shared data. Shared data and its verification metadata (i.e., signatures) are both stored in the cloud server. A public verifier, such as a thirdparty auditor providing expert data auditing services or a data user outside the group intending to utilize shared data, is able to publicly verify the integrity of shared data stored in the cloud server. When a public verifier wishes to check the integrity of shared data, it first sends an auditing challenge to the cloud server. After receiving the auditing challenge, the cloud server responds to the public verifier with an auditing proof of the possession of shared data. Then, this public verifier checks the correctness of the entire data by verifying the correctness of the auditing proof. Essentially, the process of public auditing is a challeng eand- response protocol between a public verifier and the cloud server.

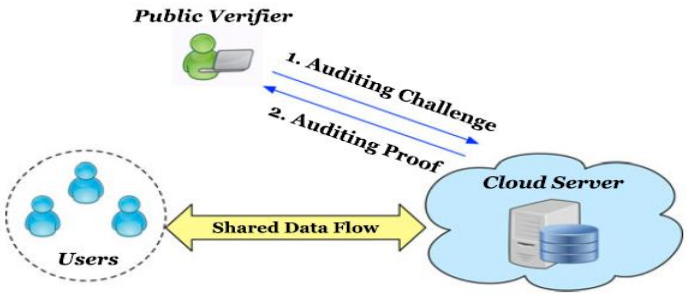

Fig1:Verifying integrity of data

It is designed to achieve following properties: (1) Public Auditing: A public verifier is able to publicly verify the integrity of shared data without retrieving the entire data from the cloud. (2) Correctness: A public verifier is able to correctly verify shared data integrity. (3) Unforgeability: Only a user in the group can generate valid verification metadata (i.e., signatures) on shared data. (4) Identity Privacy: A public verifier cannot distinguish the identity of the signer on each block in shared data during the process of auditing.

A. Cloud Server

The system is designed with Cloud Server, where the datas are stored globally. Oruta mechanism was designed to achieve following properties:

(1) Public Auditing: A public verifier is able to publicly verify the integrity of shared data without retrieving the entire data from the cloud.
(2) Correctness: A public verifier is able to correctly verify shared data integrity.
(3) Unforgeability: Only a user in the group can generate valid verification metadata (i.e., signatures) on shared data.
(4) Identity Privacy: A public verifier cannot distinguish the identity of the signer on each block in shared data during the process of auditing.

B. Group of users

There are two types of users in a group: the original user and a number of group users. The original user initially creates shared data in the cloud, and shares it with group users. Both the original user and group users are members of the group. Every member of the group is allowed to access and modify shared data. Shared data and its verification metadata (i.e., signatures) are both stored in the cloud server. A public verifier, such as a third party auditor providing expert data auditing services or a data user outside the group intending to utilize shared data, is able to publicly verify the integrity of shared data stored in the cloud server.

1. Owner Registration: If the owner has to upload files in a cloud server, he/she should register first. Then only he/she can be able to do it. For that he needs to fill the details in the registration form. These details are maintained in a database.
2. Owner Login: If the owners have to login, they should login by giving their email id and password.
3. User Registration: If a user wants to access the data which is stored in a cloud, he/she should register their details first. These details are maintained in a Database.

4. User Login: If the user is an authorized user, he/she can download the file by using file id which has been stored by data owner when it was uploading.

C. Public verifier

When a public verifier wishes to check the integrity of shared data, he first sends an auditing challenge to the cloud server. After receiving the auditing challenge, the Cloud server responds to the public verifier with an auditing proof of the possession of shared data. Then, this public verifier checks the correctness of the entire data by verifying the correctness of the auditing proof. Essentially, the process of public auditing is a challenge and- response protocol between a public verifier and the cloud server

D. Auditing

A third party auditor TPA (maintainer of clouds) should register first. After third party auditor gets logged in, he/she can see how many data owners have uploaded their files into the cloud. Here, we are providing TPA for maintaining clouds. We only consider how to audit the integrity of shared data in the cloud with static groups. It means the group is pre-defined before shared data is created in the cloud and the membership of users in the group is not changed during data sharing. The original user is responsible for deciding who is able to share her data before outsourcing data to the cloud. Another interesting problem is how to audit the integrity of shared data in the cloud with dynamic groups ie, a new user can be added into the group and an existing group member can be revoked during data sharing still preserving identity privacy.

We are going to propose Traceability mechanism which means the ability for the group manager (i.e., the original user) to reveal the identity of the signer based on verification metadata in some special situations and data freshness is also achieved.

III. ARCHITECTURE DESIGN

A. TWO LAYER ENCRYPTION SCHEME

In the TLE [1], [2], [7], [14], [15] data can be encrypted by two times. Firstly, the data owner can encrypt the data which is called as Coarse-grained encryption and secondly the cloud can re-encrypt the encrypted data which is called as Fine-grained encryption. The two layer encryption is not new but the performance of Coarse-grained and finegrained are best and provide better solution than existing solution. A challenging issue in the TLE is how to decompose ACPs so that fine-grained ABAC enforcement can be delegated to the cloud. Using Policy Decomposition [1], [2], the ACPs can be divided into sub ACPs. Such that the conjunctions of two sub ACPs result the original ACPs. In this process, the data owner first encrypt the data based on one set of sub ACPs and the cloud re-encrypt the encrypted data using other set of sub ACPs. For two encryptions, the user should perform two decryption processes to access the original data. The TLE process overcomes the above all limitations.

The TLE system consists of four entities Owner, User, Idp and cloud as shown in fig (2). The owner and the cloud collectively enforce ACPs by performing two encryptions on each data items. This two layer enforcement allows one to reduce the owner load and delegates as much access control enforcement duties as possible to cloud. Specifically TLE provides a better way to handle data updates, and user dynamics change. The TLE system goes through one additional phase compared to SLE system. The phases are as below:

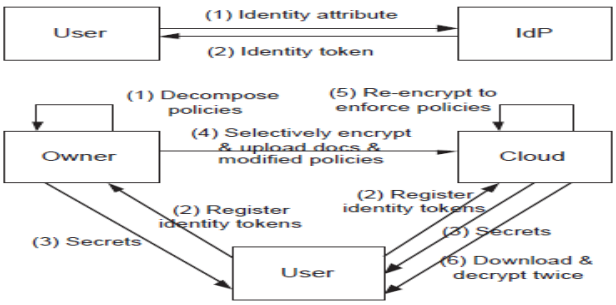

Fig 2:System Architecture

The TLE approach has many advantages.

1. When user dynamics changes, only the outer layer of the encryption needs to be updated. Since the outer layer encryption is performed at the cloud, no data transmission is required between the data owner and the cloud.
2. Further, both the data owner and the cloud service utilize a broadcast key management whereby the actual keys do not need to be distributed to the users.
3. Instead, users are given one or more secrets which allow them to derive the actual symmetric keys for decrypting the data.

The detailed description of six phases for two layer encryption are described below

1. *Identity token issuance*:-

Idps are trusted third parties that issues identity tokens to user based on their identity attributes. It should be noted that Idps need not be online after they issue identity tokens.

2. *Policy Decomposition*:-

Using the Policy decomposition, the owner decomposes each ACPs into two sub ACPs. Such that the owner enforces the minimum number of attributes to assure confidentiality of data from the cloud. The two sub ACPs are noted as $ACPB_{owner}$ is used by owner to enforce the confidentiality and the $ACPB_{cloud}$ is used by the cloud.

3. *Identity token registration*:

Users register their identity tokens in order to obtain secrets to decrypt the data. Users only register those tokens which are related to owner's sub ACPs and remaining identity tokens related to cloud.

4. *Data encryption and uploading:*

Firstly, owner can encrypt the data based on owner sub ACPs and then that data uploaded on the cloud along with the public information which is generated by the Attribute Based-Group Key Management : KeyGen (ABGKM:: KeyGen) algorithm and remaining sub ACPs used on the cloud.

5. *Data downloading and decryption*:

Users download encrypted data from the cloud and decrypt the data using the derived Key. Users can decrypt two times encrypted data, first to remove the encryption layer added by the cloud and then the encryption layer added by the owner.

6. *Encryption evolution management*:

Regularly users credential may change. Further, already encrypted data may go through various changes or updates. In such situation, already encrypted data must be re-encrypted with a new key.

IV. CONCLUSION

Current approaches to enforce ACPs on outsourced data using selective encryption require organizations to manage all keys and encryptions and upload the encrypted data to the remote storage. Such approaches incur high communication and computation cost to manage keys and encryptions whenever user credentials or organizational authorization policies/data change. In this paper, we proposed a two layer encryption based approach to solve this problem by delegating as much of the access control enforcement responsibilities as possible to the Cloud while minimizing the information exposure risks due to colluding Users and Cloud. Our approach is based on a privacy preserving attribute based key management scheme that protects the privacy of users while enforcing attribute based ACPs.As the experimental results show, decomposing the ACPs and utilizing the two layer of encryption reduce the overhead at the Owner. As future work, we plan to investigate the alternative choices for the TLE approach further. We also plan to further reduce the computational cost by exploiting partial relationships among ACPs.

Acknowledgment

I would like to thank my guide Mr.D. Namachivayam for assisting me in this paper work

References

[1] M. Nabeel and E. Bertino, "Privacy Preserving delegated access control on Public Clouds", IEEE Transactions on Knowledge and Data Engineering, 2013

[2] M. Nabeel and E. Bertino, "Privacy preserving delegated access control in the storage as a service model," in EEE International Conference on Information Reuse and Integration(IRI), 2012.

[3] M. Nabeel, N. Shang, and E. Bertino, "Privacy preserving policy based content sharing in public clouds," IEEE Transactionson Knowledge and Data Engineering, 2012.

[4] A .Reddy, GudivadaLokesh and N. Vikram "Privacy Preserving Delegated Access Control in Public Clouds", International Journal of Computer Science Trends and Tech (IJCST)- Vol. 2 Issue 4, July Aug 2014.

[5] Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," Proceedings of the 13th ACM conference on Computerand communications security. New York, NY, USA: ACM, pp. 89–98, 2006

[6] B. Wang, B. Li, and H. Li, "Oruta: Privacy- Preserving Public Auditing for Shared Data in The Proc. IEEE Fifth Int'l Conf. Cloud Computing, pp. 295- 302, 2012.

[7] C. Wang, Q. Wang, K. Ren, and W, Lou,"Privacy- Preserving Public Auditing for Data Storage Security in Cloud Computing," Proc. IEEE INFOCOM, pp.525-533, 2010.

[8] 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. 598-610, 2007

[9] 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), pp. 90-107, 2008

[10] C. Wang, S.S. 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