Security-Conserve Common Evaluate for Mutual Information in the Cloud

R. Amudha Valli and Dr.P. Mayilvahanan

Abstract--- With cloud data services, it is common place for data to be not only stored in the cloud, but also shared across multiple users. Unfortunately, the integrity of cloud data is subject to skepticism due to the existence of hardware/software failures and human errors. With our mechanism, the identity of the signer on each block in shared data is kept private from public verifiers, who are able to efficiently verify shared data integrity without retrieving the entire file. In addition, our mechanism is able to perform multiple auditing tasks simultaneously instead of verifying them one by one. However, public auditing on the integrity of shared data with these existing mechanisms will supports public auditing on shared data stored in the cloud that exploit ring signature to compute verification metadata needed to audit the correctness of shared data. So that a third party auditor (TPA) is able to verify the integrity of shared data for users without retrieving the entire data. Meanwhile, the identity of the signer on each block in shared data is kept private from the TPA it also able to perform multiple auditing tasks simultaneously instead of verifying them one by one. In this paper, we proved the data freshness (proved the cloud possesses the latest version of shared data) while still preserving identity privacy. In this proposed system mainly focus on traceability, 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. It's based on ring signatures, where the identity of the signer is unconditionally protected.

Keywords--- Cloud Computing, Public Auditing, Privacy Preserving, Shared Data

I. INTRODUCTION

With cloud computing and storage, users are able to access and to share resources offered by cloud service providers at a lower marginal cost. It is routine for users to leverage cloud storage services to share data with others in a group, as data sharing becomes standard feature in most cloud storage offerings, including Dropbox, iCloud and Google Drive. The integrity of data in cloud storage, however, is subject to skepticism and scrutiny, as data stored in the cloud can easily be lost or corrupted due to the inevitable hardware/software failures and human errors. The traditional approach for checking data correctness is to retrieve the entire data from the cloud, and then verify data integrity by checking the correctness of signatures (e.g., RSA) or hash values (e.g., MD5) of the entire data. Certainly, this conventional approach able to successfully check the correctness of cloud data. However, the efficiency of using this traditional approach on cloud data is in doubt. The main reason is that the size of cloud data is large in general. 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.

The problem of simultaneously achieving scalability, and data confidentiality of access control actually still remains unresolved. We achieve this goal by exploiting and uniquely combining techniques of attribute-based encryption (ABE), proxy re-encryption, and lazy re-encryption. Our proposed scheme also has salient properties of user access privilege confidentiality and user secret key accountability. Extensive analysis shows that our proposed

R. Amudha Valli, MPhil Student, School of Computing, VELS University, Chennai, Tamil Nadu, India. E-mail:amudhamsk@gmail.com
Dr.P. Mayilvahanan, HOD, Dept. of MCA, VELS University, Chennai, Tamil Nadu, India. E-mail:hodmca@velsuniv.org

scheme is highly efficient and provably secures under existing security models. Cloud service providers may be reluctant to inform users about these data errors in order to maintain the reputation of their services and avoid losing profits. Therefore, the integrity of cloud data should be verified before any data utilization, such as search or computation over cloud data. To perform public auditing is designed to check the correctness of data stored in an untrusted server, without retrieving the entire data. In public auditing mechanisms can actually be extended to verify shared data integrity. A new significant privacy issue introduced in the case of shared data with the use of existing mechanisms is the leakage of identity. In this proposed system mainly focus two interesting problem one of them is traceability, 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. It's based on ring signatures, where the identity of the signer is unconditionally protected. The current design of ours does not support traceability. To the best of our knowledge, designing an efficient public auditing mechanism with the capabilities of preserving identity privacy and supporting traceability. Another problem for our further work is how to prove data freshness (prove the cloud possesses the latest version of shared data) while still perpetuate identity privacy. It also contain proofs of Retrievability (POR), which is also able to check the correctness of data on an untrusted server. The first scheme is built from BLS signatures, and the second one is based on pseudo-random functions. Data should be available in encrypted format so its gives more protection for user confidential data or owners data. To support dynamic data presented an efficient PDP mechanism based on symmetric keys. This mechanism can support update and delete operations on info to operate multiple auditing tasks from different users efficiently, they extended their mechanism to enable batch auditing by leveraging aggregate signatures. Another one advantage is similar model called Proofs of Retrievability(POR), which is also able to check the correctness of data on an untrusted server.

II. PROBLEM STATEMENTS

2.1 System Model

As illustrated in Fig. 1, the system model in this paper involves three parties: the cloud server, a group of users and a public verifier. 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 (TPA) 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 challenge and-response protocol between a public verifier and the cloud server.

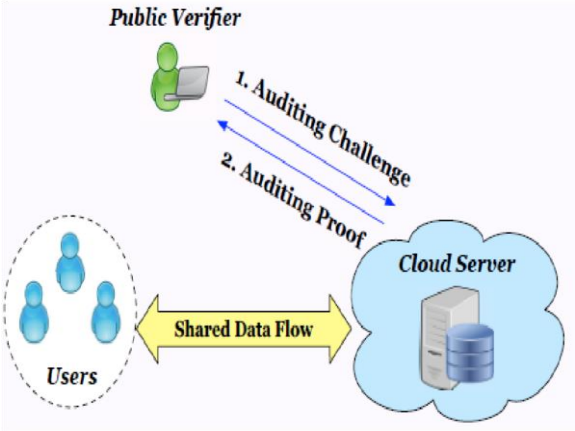


Fig. 1: System Model of Three Parties

2.2 Threat Model

2.2.1 Integrity Threats Two kinds of threats related to the integrity of shared data are possible. First, an adversary may try to corrupt the integrity of shared data. Second, the cloud service provider may inadvertently corrupt (or even remove) data in its storage due to hardware failures and human errors. Making matters worse, the cloud service provider is economically motivated, which means it may be reluctant to inform users about such corruption of data in order to save its reputation and avoid losing profits of its services.

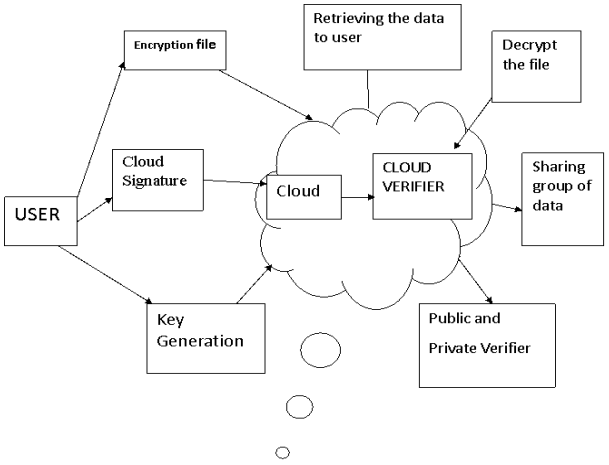
2.2.2 Privacy Threats The identity of the signer on each block in shared data is private and confidential to the group. During the process of auditing, a public verifier, who is only allowed to verify the correctness of shared data integrity, may try to reveal the identity of the signer on each block in shared data based on verification metadata. Once the public verifier reveals the identity of the signer on each block, it can easily distinguish a high value target (a particular user in the group or a special block in shared data) from others.

2.3 Design Objectives

Our mechanism should be designed to achieve the following properties:

1. **Public Auditing**: A public verifiers 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. **Enforceability**: 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**.**
5. **Data freshness**: data freshness is essential to protect against mis-configuration errors or rollbacks caused intentionally. We can develop an authenticated file system that supports the migration of an enterprise-class distributed file system into the cloud efficiently, transparently and in a scalable manner. It's authenticated in the sense that enables an enterprise tenant to verify the freshness of retrieved data while performing the file system operations.
6. **Traceability**: 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. It's based on ring signatures, where the identity of the signer is unconditionally protected.

III. SYSTEM ARCHITECTURE



Key generation is the process of generating keys for encrypting and decrypting the files. Here using two algorithms RSA and DES and keys are randomly generated using random key generation. RSA algorithms use to create a private key and public key.

The public key is made available to anyone. A sender encrypts data with the public key only the holder of the private key can decrypt this data. In this we can change the data as unreadable format by using encryption method which ismentioned above. If any un-authorized persons are opened and trying to do something then they cannot understand that data, so by using this we secure our important files.

Decryption is the reverse process to encryption and decryption is user understandable language. Database decryption is converting the encryption cipher text into the original information using keys generated by the encryption algorithms. Here if you decrypt the file that time you will enter the key after that only your decrypt that particular file why because here that key is provide security for your personal data.

Cloud Verifier a cloud storage framework that provides complete, correct, accurate and verifiable data file service to users. Users can leverage such framework to obtain a correct view of the runtime state of their computing environment and perform responsive reaction upon anomalies.

The Cloud verifier to compare cloud storage data and these data same or not after that same means it provide high security the cloud's hierarchical structure to build transitive trust starting in the cloud platform up to the instances themselves. Platform states are monitored by a Cloud Verifier against the cloud administrator's specified criteria, thereby preventing maliciously modified systems from executing data to user.

Cloud Signature Support is an exclusive technical benefit that provides qualified cloud competency partners with an elevated level of technical support for select Microsoft cloud products. You will receive to original data that will work directly with you, have extensive product-specific knowledge, and are accountable for driving cases from start to finish. Where the identity of the signer is unconditionally protectedthe current design of ours.

Here support traceability. Each user makes his encryption key public, and keeps the corresponding decryption key private signatures People and organizations buy or lease storage capacity from the providers to store user, organization, or Application data Cloud storage has several advantages over traditional data storage.

File Generation to store and share a data file in the cloud, a group member performs the following operations: Getting the revocation list from the cloud. In this step, the member sends the group identity I group as a request to the cloud. Then, the cloud responds the revocation list RL to the member. Verifying the validity of the received revocation list.

On receiving the data, the cloud first checks its validity. If the algorithm returns true, the group signature is valid; otherwise, the cloud abandons the data. In addition, if several users have been revoked by the group manager, the cloud also performs revocation verification.

IV. PUBLIC AUDITING MECHANISM

4.1 Overview

Using HARS and its properties, we now construct Oruta, a privacy preserving public auditing mechanism for shared data in the cloud. With Oruta, the public verifier can verify the integrity of shared data without retrieving the entire data. Meanwhile, the identity of the signer on each block in shared data is kept private from the public verifier during the auditing.

BLS Algorithm

BLS means Boneh–Lynn–Shacham(BLS) Algorithm allows a user to verify that a signer inauthentic. The scheme uses for verification and signatures are group elements in some. It provides defense against attacks against allowing shorter signatures than signatures. Designed a public verification scheme for cloud data, where data is encoded with erasure codes, so that the content of a user’s data can be recovered even if some part of data is collapsed.

De Cipher Algorithm

A user exchange encrypted messages using a public-key cryptosystem would place their public encryption procedure, E, in a public file. The user's corresponding decryption procedure, De cipher algorithm is provide four properties that the encryption and decryption.

1. Deciphering the enciphered form of a message M yields M. That is, D(E(M)) = M
2. Encryption and Decryption are easy to compute.

3. Publicly revealing Encryption does not reveal an easy way to compute D. As such, only the user can decrypt messages which were encrypted with E. Likewise, only the user can compute D efficiently.
4. Deciphering a message M and then enciphering it results in M. That is,E(D(M)) = M.

V. CONCLUSION AND FUTURE WORK

One of them is traceability, 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. Since Oruta is based on ring signatures, where the identity of the signer is unconditionally protected, the current design of ours does not support traceability. To the best of our knowledge, designing an efficient public auditing mechanism with the capabilities of preserving identity privacy and supporting traceability is still open. Contractual and legal protections can, of course, play a valuable role in laying the foundations of secure storage infrastructure. We believe that the technical assurances provided by PORs, however, will permit even more rigorous and dynamic enforcement of service policies and ultimately enable more flexible and cost-effective storage architectures.

Future Work

Our feature work is a fully homomorphic signature scheme would be a useful parallel to existing fully homomorphic encryption systems. Current constructions of fully homomorphic encryption are obtained by applying a "bootstrapping" process to a scheme that allows a limited amount of computation on encrypted data. It is unclear whether Gentry's bootstrapping process can be applied to signature schemes such as ours.

REFERENCE

[1] B. Wang, B. Li, and H. Li, ―Oruta: Privacy-Preserving Public Auditing for Shared Data in the Cloud,‖ in Proceedings of IEEE Cloud 2012, 2012, pp. 295–302.
[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, ―A View of Cloud Computing,‖ Communications of the ACM, vol. 53, no. 4, pp. 50–58, April 2010.
[3] K. Ren, C. Wang, and Q. Wang, ―Security challenges for the Public Cloud,‖ IEEE Internet Computing, vol. 16, no. 1, pp. 69–73, 2012.
[4] D. Song, E. Shi, I. Fischer, and U. Shankar, ―Cloud Data Protection for the Masses,‖ IEEE Computer, vol. 45, no. 1, pp. 39–45, 2012.
[5] C. Wang, Q. Wang, K. Ren, and W. Lou, ―Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing,‖ in Proceedings of IEEE INFOCOM 2010, 2010, pp. 525–533.
[6] B. Wang, M. Li, S. S. Chow, and H. Li, ―Computing Encrypted Cloud Data Efficiently under Multiple Keys,‖ in Proc. of CNSSPCC' 13, 2013, pp. pp.90–99.
[7] R. Rivest, A. Shamir, and L. Adleman, ―A Method for Obtaining Digital Signatures and Public Key Cryptosystems,‖ vol. 21, no. 2, pp. 120–126, 1978.
[8] The MD5 Message-Digest Algorithm (RFC1321). [Online].Available: https://tools.ietf.org/html/rfc1321
[9] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, ―Provable Data Possession at Untrusted Stores,‖ in Proceedings of ACM CCS'07, 2007, pp. 598–610.
[10] H. Shacham and B. Waters, ―Compact Proofs of Retrievability,‖ in Proceedings of ASIACRYPT'08. Springer poVerlag,2008,pp.90–107.