Auditing the Shared Data on Cloud using Ring Signatures

[1]V. Gnanasoundari
Department of Information Technology
M.Kumarasamy college of Engineering
Karur, Tamilnadu

[2]S. Deepika
Department of Information Technology
M.Kumarasamy college of Engineering
Karur, Tamilnadu

.

***Abstract*- With cloud information services, it is very common for information to be not only saved in the cloud, but also distributed across multiple users. Unfortunately, the reliability of cloud information is subject to uncertainty due to the lifestyle of hardware/software problems and human errors. Several systems have been designed to allow both information owners and public verifiers to effectively auditing cloud information integrity without accessing the whole information from the cloud server. However, public c auditing on the reliability of distributed information with these existing mechanisms will certainly reveal private information identity privacy to public verifiers. This survey recommends a novel privacy-preserving procedure that facilitates public auditing on distributed information saved in the cloud. It manipulates ring signatures to estimate confirmation meta-data needed to audit the correctness of distributed information. With this procedure, the identification of the signer on each block in distributed information is kept private from public verifiers, who are able to confirm distributed information reliability without retrieving the whole file. In addition, this procedure is able to perform several auditing projects at the same time instead of confirming them one by one. Our trial results illustrate the efficiency and usefulness of our procedure when auditing distributed information reliability.**

Keywords*- *Cloud computing, shared data, Public auditing, privacy preserving.

1. INTRODUCTION

Cloud companies offer customers effective and scalable data storage space alternatives with a much reduced marginal cost than traditional techniques [2]. It is schedule for Customers to leverage cloud storage space alternatives to work together with others in a team, as information discussing becomes a conventional function in most cloud storage space Promotions, such as Drop box, Cloud and Google Generate. The reliability of information in cloud storage space, however, is subject to uncertainty and analysis, as information saved in the cloud can easily be missing or damaged due to the unavoidable hardware/ software problems and individual mistakes. To create this matter even more intense, cloud companies may be reluctant to notify customers about these information mistakes to be able to maintain the popularity of their alternatives and prevent losing profits [3]. Therefore, the reliability of cloud information should be verified before any information usage, such as look for or computation over cloud information The traditional strategy for verifying information correctness is to recover the whole information from the cloud, and then verify data reliability by verifying the correctness of signatures (e.g. RSA) or hash principles (e.g., MD5) of the entire data. Certainly, this traditional strategy is able to successfully check the correctness of cloud information.

The primary purpose is that the size of the cloud information is large in common. Installing the whole cloud information to verify data reliability will price or even spend Customer's quantities of computation and interaction sources, especially when information has been damaged in the cloud. Besides, many uses of cloud information (e.g., informatione exploration and machine learning) do not actually need customers to obtain the entire cloud information to regional gadgets [2]. It is because cloud providers, such as Amazon, can offer customers computation services straight on large-scale information that already existed in the cloud.

However, a new significant privacy issue introduced in the case of shared data with the use of existing mechanisms is the leakage of identity privacy to public verifiers [1].To protect the confidential information, it is essential and critical to preserve identity privacy from public verifiers during public auditing. To solve the above privacy issue on shared data, we propose Oruta1, a novel privacy preserving public auditing mechanism. More specifically, we utilize ring signatures to construct homomorphic authenticators [10] in Oruta, 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 addition, extend this mechanism to support batch auditing, which can perform multiple auditing tasks simultaneously and improve the efficiency of verification for multiple auditing tasks. Meanwhile, Oruta is compatible with random masking [5]; **Oruta stands for "One Ring to Rule Them All". Our contributions in this paper are multirole**.

In this paper, to prove the data freshness (prove the cloud possesses the latest version of shared data)while still preserving identity privacy.Ensures that retrieved data always reflects the most recent updates and prevents rollback attacks. Achieving 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 distributes 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.

KEY MANAGEMENT

In this paper, following are the cryptographic keys to protect data files stored on the cloud.

Public Key: The Public key is a random generated binary key, generated and maintained by the Key manager itself. Particularly used for encryption/ decryption.

Private Key: It is the combination of the username, password and two security question of user's choice. The private key is maintained by client itself. Used for encrypt /decrypt the file.

Access key: It is associated with a policy. Private access key is maintained by the client. The access key is built on attribute based encryption. File access is of read or write.

Renew key: Maintained by the client itself. Each has its own renew key. The renew key is used to renew the policy of each necessary file at easy method.

2. RELATED WORK

The authors [12] take a centralized technique where a single key distribution center (KDC) distributes secret keys and attributes to all the users. Unfortunately, a single KDC is not only a single data of failure but difficult to maintain because of the large number of users that are supported in a cloud environment. The receiver receiving the attributes and secret keys from the attribute authority and is able to decrypt the information if it has matching attributes. All the technique take a centralized approach and allow only one KDC, which is a single point of failure .A proposed a scheme in which there are several KDC authorities (coordinated by a trusted authority) which distribute attributes and secret keys of the users. However, the presence of one proxy and one KDC makes it less robust than decentralized approach. A new scheme given by Maji et al. takes a decentralized approach and provides authentication without disclosing the identity of the users.

BACKGROUND

Assumptions:

a. Users can have either read or write or both accesses to a file stored in the cloud.

b. All communications between users/clouds are secured by the secure shell protocol technique, SSH.

Formats of Access Policies:

a. Boolean functions of attributes,

b. Linear secret sharing scheme (LSSS) matrix of the data [1], or

c. Monotone span programs.

Any access structure can be converted into a Boolean function. An example of a Boolean function is ((a1 Λ a2 Λ a3) V (a4 Λ a5)) Λ (a6 V a7)), where a1, a2, . . . , a7 are attributes.

Let Y: {0; 1} n →{0; 1} be a monotone Boolean function.. A monotone span program for Y over a field IF is an l *t matrix M with entries in IF, along with a labeling function a: [l]→[n]that associates each row of M with an input variable of Y, such that, for every (x1, x2, . . . , xn) ε {0 , 1}n.

a. Distributed access control of the data stored in cloud. Only authorized users with valid attributes can access the data.

b. Authentication of users only store data and modify their data on the cloud.

c. The costs are comparable to the existing centralized approaches, its very expensive operations are mostly done by the cloud.

Attribute-Based Encryption:

a) System Initialization

b) Key Generation and Distribution by KDCs

c) Encryption by Sender

d) Decryption by Receiver

Attribute-Based Signature Scheme:

a) System Initialization

b) User Registration

c) KDC Setup

d) Attribute Generation

e) Sign

f) Verify

3. PROPOESD SCHEME

To solve the above privacy issue on shared data, we propose Oruta, a novel privacy preserving public auditing mechanism. More specifically, we utilize ring signatures to construct homomorphic authenticators in Oruta, 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 addition, we further extend our mechanism to support batch auditing, which can perform multiple auditing tasks simultaneously and improve the efficiency of verification for multiple auditing tasks.

The system style in this paper involves three parties: the reasoning server, a number of users and a community verifier.

There are two kinds of customers in a group: the unique customer and a number of team customers. The original customer originally makes distributed information in the reasoning, and shares it with team customers. Both the unique customer and group users are associates of the team. Every participant of the group is permitted to accessibility and change distributed information. Shared data and its confirmation meta-data (i.e., signatures) are

both stored in the reasoning server. When a community verifier desires to examine the reliability of shared information, it first delivers an audit task to the cloud server. After getting the audit task, the reasoning server reacts to the community verifier with an auditing evidence of the ownership of distributed information.

Then, this community verifier assessments the correctness of the entire data by confirming the correctness of the audit evidence. Essentially, the procedure of community audit is a challenge and- response method between a community verifier and the cloud server

.

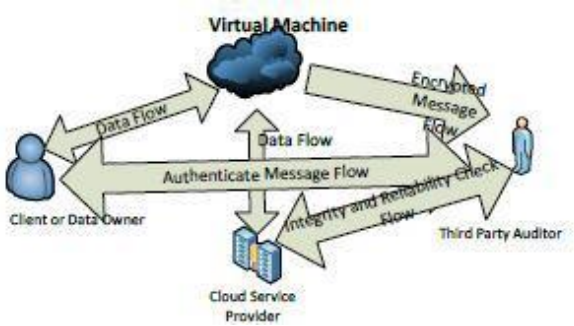

Figure 3.1 Architecture of Cloud server with CU and TPA.

1. **Service Request to TPA:** The user registers with the original identity and enrolls with the Third Party Authenticator (TPA).The user sends request to the Third Party Authenticator(TPA) for registration.
2. **TPA Policy Creation:** The TPA along with token provides the rules and regulation to be followed by Creator, Reader and Writer.
3. **User File Upload:** The file creator after getting proper authentication encrypts the file and uploads his files in the cloud.
4. **KDC Key Generation:** The Key Distribution Centers which are decentralized generate different keys to different types of users after getting tokens from users.
5. **Key Revocation:** Whenever there is mishehavior detected upon a user his key is revoked and that particular user can neither use or re-enter the cloud environment.
6. **Cloud Admin:** Cloud admin has the list of Key Distribution Centres(KDCs) and Third Party Authenticator(TPA). The cloud admin sets the norms to be followed by TPA and KDC. It monitors the key generation policies and informs abnormal behaviours

(1) **Public Auditing**: A public verifieris able to publicly verify the integrity of shared datawithout retrieving the entire data from the cloud.

(2)**Correctness**: A public verifier is able to correctly verifyshared data integrity.

(3) **Unforgeability**: Only a user inthe group can generate valid verification metadata (i.e., signatures) on shared data.

(4)**Identity Privacy**: A publicverifier 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

4. SYSTEM MODEL

4.1 Architecture

n this paper, to solve the above privacy issue on shared data, we propose Oruta,

a novel privacy-preserving public auditing mechanism. More specifically, we utilize ring signatures [21] to construct homomorphic authenticators [10] in Oruta, 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, a public verifier.

Fig. 2. Our system model includes the cloud server, a group of users and as illustrated in Fig. 2, 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 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 challenge and-response protocol between a public verifier and the cloud server [9].

THREAT MODEL

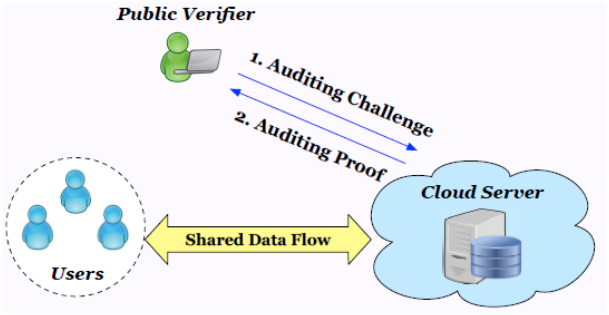

Fig.1 System model of three parties.

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 highvaluetarget (a particular user in the group or a special block in shared data) from others.

6. CONCLUSION AND FUTURE WORK

In this paper, we propose a privacy-preserving public auditing with data freshness verification mechanism for shared data in the cloud. Freshness verification should be extremely efficient for existing file system operations and induce minimal latency.

To ensure freshness, it is necessary to authenticate not just data blocks, but also their *versions*. Each block has an associated version counter that is incremented every time the block is modified. This version number is bound to the file-block's MAC: To protect against cloud replay of stale file-blocks (rollback attacks), the counters themselves must be authenticated. The interesting problems we will continue to study for our 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 the current design of ours does not support traceability.

REFERENCES

[1] E. Candès and T. Tao, ``Decoding by linear programming," IEEE Trans. Inf. Theory, vol. 51, no. 12, pp. 4203_4215, Dec. 2005.

[2] E. Candès and T. Tao, ``Near-optimal signal recovery from random projections: Universal encoding strategies," IEEE Trans. Inf. Theory, vol. 52, no. 12, pp. 5406_5425, Dec. 2006.

[3] E. Candès and M. Wakin, ``An introduction to compressive sampling," IEEE Signal Proc. Mag., vol. 25, no. 2, pp. 21_30, Mar. 2008.

[4] (2009). Security Guidance for Critical Areas of Focus in

[5] Cloud Computing, [Online]. Available:http://www.cloudsecurityalliance.org

[6] K. Ramanathan, J. Giraudi, and A. Gupta, "Creating Hierarchical User Profiles Using Wikipedia," HP Labs, 2008.

[7] K. Ja¨rvelin and J. Keka¨la¨inen, "IR Evaluation Methods for Retrieving Highly Relevant Documents," Proc. 23rd Ann. Int'l ACM SIGIR Conf. Research and Development Information Retrieval (SIGIR), pp. 41-48, 2000.

[8] R. Baeza-Yates and B. Ribeiro-Neto, Modern Information Retrieval. Addison Wesley Longman, 1999.