Scientific Journal Impact Factor (SJIF): 1.711
e-ISSN: 2349-9745
p-ISSN: 2393-8161
International Journal of Modern Trends in Engineering and Research
www.ijmter.com

Centralized Data Verification Scheme for Encrypted Cloud Data Services

Jothi.M[1], Vinoth.P[2]
[1]*PG Scholar, Dept. of CSE*
[2]*Asst. Professor, Dept. of CSE*
[1,2]*Mepco Schlenk Engineering College, Sivakasi, Tamilnadu, India*

Abstract-Cloud environment supports data sharing between multiple users. Data integrity is violated due to hardware / software failures and human errors. Data owners and public verifiers are involved to efficiently audit cloud data integrity without retrieving the entire data from the cloud server. File and block signatures are used in the integrity verification process.
"One Ring to RUle Them All" (Oruta) scheme is used for privacy-preserving public auditing process. In oruta homomorphic authenticators are constructed using Ring Signatures. Ring signatures are used to compute verification metadata needed to audit the correctness of shared data. The identity of the signer on each block in shared data is kept private from public verifiers. Homomorphic authenticable ring signature (HARS) scheme is applied to provide identity privacy with blockless verification. Batch auditing mechanism supports to perform multiple auditing tasks simultaneously. Oruta is compatible with random masking to preserve data privacy from public verifiers. Dynamic data management process is handled with index hash tables. Traceability is not supported in oruta scheme. Data dynamism sequence is not managed by the system. The system obtains high computational overhead
The proposed system is designed to perform public data verification with privacy. Traceability features are provided with identity privacy. Group manager or data owner can be allowed to reveal the identity of the signer based on verification metadata. Data version management mechanism is integrated with the system.

I. INTRODUCTION

Cloud computing is a recent trend in IT that moves computing and data away from desktop and portable PCs into large data centers. It refers to applications delivered as services over the Internet as well as to the actual cloud infrastructure — namely, the hardware and systems software in data centers that provide these services. The key driving forces behind cloud computing are the ubiquity of broad-band and wireless networking, falling storage costs and progressive improvements in Internet computing software. Cloud-service clients will be able to add more capacity at peak demand, reduce costs, experiment with new services and remove unneeded capacity, whereas service providers will increase utilization via multiplexing and allow for larger investments in software and hardware.

Currently, the main technical underpinnings of cloud computing infrastructures and services include virtualization, service-oriented software, grid computing technologies, management of large facilities and power efficiency. Consumers purchase such services in the form of infrastructure-as-a-service (IaaS), platform-as-a-service (PaaS), or software-as-a-service (SaaS) and sell value-added services to users. Within the cloud, the laws of probability give service providers great leverage through statistical multiplexing of varying workloads and easier management — a single software installation can cover many users' needs.

@IJMTER-2014, All rights Reserved

We can distinguish two different architectural models for clouds: the first one is designed to scale out by providing additional computing instances on demand. Clouds can use these instances to supply services in the form of SaaS and PaaS. The second architectural model is designed to provide data and compute-intensive applications via scaling capacity. In most cases, clouds provide on-demand computing instances or capacities with a "pay-as-you-go" economic model. The cloud infrastructure can support any computing model compatible with loosely coupled CPU clusters. Organizations can provide hardware for clouds internally, or a third party can provide it externally. A cloud might be restricted to a single organization or group, available to the general public over the Internet, or shared by multiple groups or organizations.

II. RELATED WORK

Provable data possession (PDP), proposed by Ateniese et al., [9] allows a verifier to check the correctness of a client's data stored at an untrusted server. By utilizing RSA-based homomorphic authenticators and sampling strategies, the verifier is able to publicly audit the integrity of data without retrieving the entire data, which is referred to as public auditing. Unfortunately, their mechanism is only suitable for auditing the integrity of personal data. Juels and Kaliski defined another similar model called roofs of Retrievability (POR), which is also able to check the correctness of data on an untrusted server. The original file is added with a set of randomly-valued check blocks called sentinels. The verifier challenges the untrusted server by specifying the positions of a collection of sentinels and asking the untrusted server to return the associated sentinel values. Shacham and Waters [10] designed two improved schemes. The first scheme is built from BLS signatures and the second one is based on pseudo-random functions.

To support dynamic data, Ateniese et al presented an efficient PDP mechanism based on symmetric keys. This mechanism can support update and delete operations on data, however, insert operations are not available in this mechanism. Because it exploits symmetric keys to verify the integrity of data, it is not public verifiable and only provides a user with a limited number of verification requests. Wang et al. [2] utilized Merkle Hash Tree and BLS signatures support dynamic data in a public auditing mechanism. Erway et al. [1] introduced dynamic provable data possession (DPDP) by using authenticated dictionaries, which are based on rank information. Zhu et al. [7] exploited the fragment structure to reduce the storage of signatures in their public auditing mechanism. In addition, they also used index hash tables to provide dynamic operations on data. The public mechanism proposed by Wang et al. [5] and its journal version [8] are able to preserve users' confidential data from a public verifier by using random maskings. In addition, to operatemultiple auditing tasks from different users efficiently, they extended their mechanism to enable batch auditing by leveraging aggregate signatures.

Wang et al. [3] leveraged homomorphic tokens to ensure the correctness of erasure codes-based data distributed on multiple servers. This mechanism is able not only to support dynamic data, but also to identify misbehaved servers. To minimize communication overhead in the phase of data repair, Chen et al. [4] also introduced a mechanism for auditing the correctness of data under the multi-server scenario, where these data are encoded by network coding instead of using erasure codes. More recently, Cao et al. [6] constructed an LT codes-based secure and reliable cloud storage mechanism. Compare to previous work [3], [4], this mechanism can avoid high decoding computation cost for data users and save computation resource for online data owners during data repair.

@IJMTER-2014, All rights Reserved

III. PRIVACY PRESERVED PUBLIC AUDITING SCHEME FOR CLOUDS

Cloud service providers offer users efficient and scalable data storage services with a much lower marginal cost than traditional approaches. It is routine for users to leverage cloud storage services to share data with others in a group, as data sharing becomes a 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. To make this matter even worse, 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.

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 of the entire data. Certainly, this conventional approach is 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 users 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 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. 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. 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) who can provide expert integrity checking services. Moving a step forward, Wang et al. designed an advanced auditing mechanism, 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. Unfortunately, current public auditing solutions mentioned above only focus on personal data in the cloud.

We believe that sharing data among multiple users is perhaps one of the most engaging features that motivates cloud storage. Therefore, it is also necessary to ensure the integrity of shared data in the cloud is correct. Existing 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 privacy to public verifiers. For instance, Alice and Bob work together as a group and share a file in the cloud. The shared file is divided into a number of small blocks, where each block is independently signed by one of the two users with existing public auditing solutions. Once a block in this shared file is modified by a user, this user needs to sign the new block using his/her private key. Eventually, different blocks are signed by different users due to the modification introduced by these two different users. Then, in order to correctly audit the integrity of the entire data, a public verifier needs to choose the appropriate public key for each block. As a result, this public verifier will inevitably learn the identity of the signer on each block due to the unique binding between an identity and a public key via digital certificates under public key infrastructure (PKI).

@IJMTER-2014, All rights Reserved

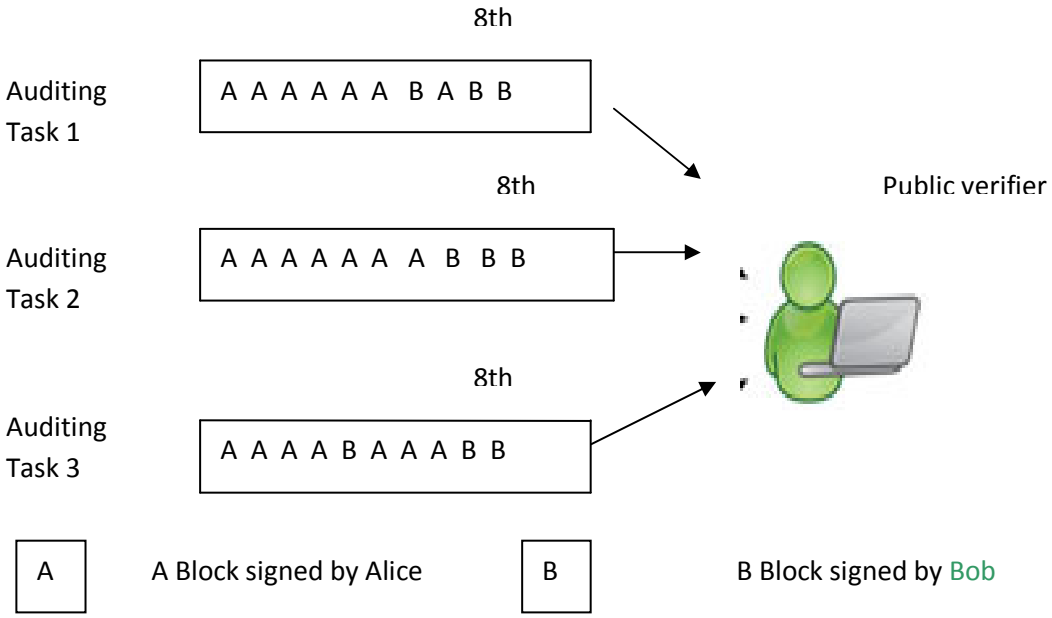

Fig. 1. Shared Data Integrity Auditing by Public Verifier

	PDA	WWRL	Oruta
Public Auditing	✓	✓	✓
Data Privacy	×	✓	✓
Identity Privacy	×	×	✓

TABLE 1: Comparison Among Different Mechanisms

Failing to preserve identity privacy on shared data during public auditing will reveal significant confidential information to public verifiers. Specifically, as shown in Fig. 1, after performing several auditing tasks, this public verifier can first learn that Alice may be a more important role in the group because most of the blocks in the shared file are always signed by Alice; on the other hand, this public verifier can also easily deduce that the eighth block may contain data of a higher value, because this block is frequently modified by the two different users. In order to protect these confidential information, it is essential and critical to preserve identity privacy from public verifiers during public auditing. In this paper, to solve the above privacy issue on shared data, we propose Oruta, a novel privacy-preserving public auditing mechanism. 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. Meanwhile, Oruta is compatible with random masking, which has been utilized in WWRL and can preserve data privacy from public verifiers. Moreover, we also leverage index hash

@IJMTER-2014, All rights Reserved

tables from a previous public auditing solution to support dynamic data. A high-level comparison among Oruta and existing mechanisms is presented.

IV. PROBLEM STATEMENT

"One Ring to RUle Them All" (Oruta) scheme is used for privacy-preserving public auditing process. In oruta homomorphic authenticators are constructed using Ring Signatures. Ring signatures are used to compute verification metadata needed to audit the correctness of shared data. The identity of the signer on each block in shared data is kept private from public verifiers. Homomorphic authenticable ring signature (HARS) scheme is applied to provide identity privacy with blockless verification. Batch auditing mechanism supports to perform multiple auditing tasks simultaneously. Oruta is compatible with random masking to preserve data privacy from public verifiers. Dynamic data management process is handled with index hash tables. The following problems are identified from the existing system.

- Traceability is not supported
- Data dynamism sequence is not managed
- High computational overhead

V. ORUTA SCHEME FOR PRIVACY PRESERVED CLOUD DATA ANALYSIS

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 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. 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.

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 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. 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 from others.

Oruta should be 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 on shared data. (4) Identity Privacy: A public

@IJMTER-2014, All rights Reserved

verifier cannot distinguish the identity of the signer on each block in shared data during the process of auditing.

5.1. Ring Signatures

The concept of ring signatures was first proposed by Rivest et al. in 2001. With ring signatures, a verifier is convinced that a signature is computed using one of group members' private keys, but the verifier is not able to determine which one. More concretely, given a ring signature and a group of d users, a verifier cannot distinguish the signer's identity with a probability more than 1=d. This property can be used to preserve the identity of the signer from a verifier. The ring signature scheme introduced by Boneh et al. is constructed on bilinear maps. We will extend this ring signature scheme to construct our public auditing mechanism.

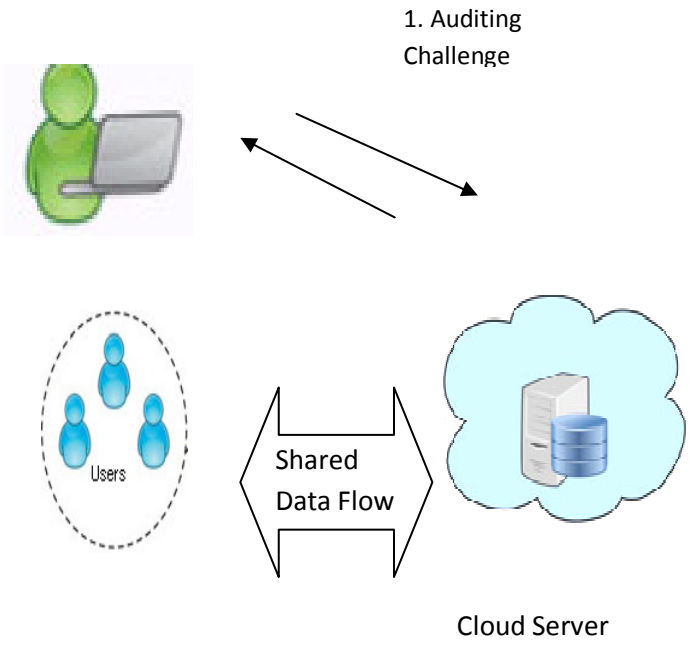

Fig. 2. "One Ring to Rule Them All." (Oruta) Scheme Based Public Verifier

5.2. Homomorphic Authenticators

Homomorphic authenticators are basic tools to construct public auditing mechanisms. Besides unforgeability, a homomorphic authenticable signature scheme, which denotes a homomorphic authenticator based on signatures, should also satisfy the following properties:

Let (pk, sk) denote the signer's public/private key pair, σ_1 denote a signature on block m_1 € Zp, σ_2 denote a signature on block m_2 € Z_p. Blockless verifiability: Given σ_1 and σ_2, two random values α_1, α_2 ε Z_p and a block m'= $\alpha_1 m_1 + \alpha_2 m_2$ € Z_p, a verifier is able to check the correctness of block m0 without knowing block m_1 and m_2.

Non-malleability: Given $\sigma_1 1$ and σ_2, two random values α_1, α_2 € Z_p and a block m'= $\alpha_1 m_1 + \alpha_2 m_{2\varepsilon} Z_p$, a user, who does not have private key s_k, is not able to generate a valid signature σ' on block m' by linearly combining signature σ_1 and σ_2.

@IJMTER-2014, All rights Reserved

Blockless verifiability allows a verifier to audit the correctness of data stored in the cloud server with a special block, which is a linear combination of all the blocks in data. If the integrity of the combined block is correct, then the verifier believes that the integrity of the entire data is correct. In this way, the verifier does not need to download all the blocks to check the integrity of data. Non-malleability indicates that an adversary cannot generate valid signatures on arbitrary blocks by linearly combining existing signatures.

5.3. New Ring Signature Scheme

As we introduced in previous sections, we intend to utilize ring signatures to hide the identity of the signer on each block, so that private and sensitive information of the group is not disclosed to public verifiers. Traditional ring signatures cannot be directly used into public auditing mechanisms, because these ring signature schemes do not support blockless verifiability. Without blockless verifiability, a public verifier has to download the whole data file to verify the correctness of shared data, which consumes excessive bandwidth and takes very long verification times. We design a new homomorphic authenticable ring signature (HARS) scheme, which is extended from a classic ring signature scheme. The ring signatures generated by HARS are not only able to preserve identity privacy but also able to support blockless verifiability. We will show how to build the privacy-preserving public auditing mechanism for shared data in the cloud based on this new ring signature scheme in the next section.

5.4. Construction of HARS

HARS contains three algorithms: KeyGen, RingSign and RingVerify. In KeyGen, each user in the group generates his/her public key and private key. In RingSign, a user in the group is able to generate a signature on a block and its block identifier with his/her private key and all the group members' public keys. A block identifier is a string that can distinguish the corresponding block from others. A verifier is able to check whether a given block is signed by a group member in RingVerify. Details of this scheme are described.

VI. PUBLIC AUDITING FOR SECURED SHARED DATA IN THE CLOUD

The proposed system is designed to perform public data verification with privacy. Traceability features are provided with identity privacy. Group manager or data owner can be allowed to reveal the identity of the signer based on verification metadata. Data version management mechanism is integrated with the system. The system is divided in to five major modules. They are data center, third party auditor, client data dynamism handler and batch auditing. The cloud data center manages the shared data values. Auditing operations are initiated by the Third Party Auditor. Client application is designed to manage data upload and download operations. Data update operations are managed under data dynamism module. Batch auditing is designed for multi user data verification process.

6.1. Data Center

The data center application is designed to allocate storage space for the data providers. Data center maintains data files for multiple providers. Different sized storage area is allocated for the data providers. Data files are delivered to the clients.

6.2. Third Party Auditor

The Third Party Auditor (TPA) maintains the signature for shared data files. TPA performs the public data verification for data providers. Data integrity verification is performed is using Secure Hashing Algorithm (SHA). Homomorphic linear authenticator and random masking techniques are used for privacy preservation process.

6.3. Client

@IJMTER-2014, All rights Reserved

The client application is designed to access the hard data values. The cloud user initiates the download process. Data access information is updated to the data center. Data center transfers the data as blocks.

6.4. Data Dynamism Handler

Shared data values are managed with blocks. Block update and delete operations are handled with signature update process. Block insertion operations are also supported in data dynamism process. Block signatures are also updated in data dynamism process.

6.5. Batch Auditing

Data integrity verification is carried out under auditing process. Batch auditing is applied to perform simultaneous data verification process. Batch auditing is tuned for multi user environment. Data dynamism is integrated with batch auditing process.

VII. CONCLUSION

Public auditing schemes are used to verify the data integrity in cloud servers. Oruta (One Ring to Rule Them All) scheme is used to support privacy ensured data verification process. Data dynamism and batch auditing are supported in Oruta. Oruta scheme is enhanced with Traceability and Data freshness features. Privacy ensured data verification is performed. Simultaneous data verification scheme is provided in the system. Computational and communication cost is reduced by the system. The system supports data dynamism for secured cloud storage environment. Traceability and version management mechanism is integrated with the system.

REFERENCES

[1] C. Erway, A. Kupcu and R. Tamassia, "Dynamic Provable Data Possession," Proc. 16th ACM Conf. Computer and Comm. Security, pp. 213-222, 2009.
[2] Q. Wang, Ren and Lou, "Enabling Public Verifiability and Data Dynamic for Storage Security in Cloud Computing," Proc. 14th European Conf. Research in Computer Security, 2009.
[3] C. Wang, Q. Wang, K. Ren and W. Lou, "Ensuring Data Storage Security in Cloud Computing," Proc. 17th Int'l Workshop Quality of Service (IWQoS'09), pp. 1-9, 2009.
[4] B. Chen, Curtmola, G. Ateniese, and R. Burns, "Remote Data Checking for Network Coding-Based Distributed Storage Systems," Proc. ACM Workshop Cloud Computing Security Workshop, 2010.
[5] 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.
[6] N. Cao, S. Yu, Z. Yang, W. Lou, and Y.T. Hou, "LT Codes-Based Secure and Reliable Cloud Storage Service," Proc. IEEE INFOCOM, 2012.
[7] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn and S.S Yau, "Dynamic Audit Services for Integrity Verification of Outsourced Storages in Clouds," Proc. ACM Symp. Applied Computing, 2011.
[8] 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.
[9] Henry C.H. Chen and Patrick P.C. Lee, "Enabling Data Integrity Protection in Regenerating-Coding-Based Cloud Storage- Theory and Implementation" IEEE Transactions On Parallel And Distributed Systems, Vol. 25, No. 2, February 2014
[10] 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.

@IJMTER-2014, All rights Reserved