IOSR Journal of Engineering (IOSRJEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
Vol. 07, Issue 12, December 2017, ||Ver. -II|| PP 01-07

Ensuring Privacy by One Ring to Rule Them All Public Auditing On Cloud Data

[1] R Ramesh kumar, [2] V Umashankari , [3] Vidhya S
[1]Assistant Professor, [2] Assistant Professor, [3]Assistant Professor
Department of Computer Science and Engineering
CMS College of Engineering and Technology,
Coimbatore, Tamil Nadu, India

ABSTRACT
Cloud data services facilitate storage and sharing of data across multiple users. Due to hardware and software failures and human errors,integrity is inevitable. Existing auditing mechanism for data integrity reveals confidential information to public verifiers. So a privacy preserving mechanism wit public auditing on shared data is required. Auditing is done on shared data by computing verification metadata using ring signature. The public verifier verifies the shared data integrity without retrieving the data or without knowing the identity of the signers. Ring signature is exploited 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 verifier.

I. INTRODUCTION

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 or hash values 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 named as WWRL. 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. 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.

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 (e.g., a block signed by Alice can only be correctly verified by Alice's public key). 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).

Failing to preserve identity privacy on shared data during public auditing will reveal significant confidential information e.g., which particular user in the group or special block in shared data is a more valuable target to public verifiers. Specifically, 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.

To solve the above privacy issue on shared data, Oruta is proposed, 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. 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 tables from a previous public auditing solution to support dynamic data. A high-level comparison among Oruta and existing mechanisms is presented.

II. LITERATURE REVIEW

DYNAMIC AUDIT SERVICES FOR INTEGRITY VERIFICATION OF OUTSOURCED STORAGE IN CLOUDS

A dynamic audit service for verifying the integrity of untrusted and outsourced storage is proposed. The audit service, is constructed based on the techniques, fragment structure, random sampling and index-hash table, can support provable updates to outsourced data, and timely abnormal detection. In addition, an efficient approach based on probabilistic query and periodic verification for improving the performance of audit services is proposed.

HOW TO LEAK A SECRET: THEORY AND APPLICATIONS OF RING SIGNATURES

In this work formalization of the notion of a ring signature, which makes it possible to specify a set of possible signers without revealing which member actually produced the signature. Unlike group signatures, ring signatures have no group managers, no setup procedures, no revocation procedures, and no coordination: any user can choose any set of possible signers that includes himself, and sign any message by using his secret key and the others' public keys, without getting their approval or assistance. Ring signatures provide an elegant way to leak authoritative secrets in an anonymous way, to sign casual email in a way that can only be verifed by its intended recipient, and to solve other problems in multiparty computations.

The main contribution lies in the presentation of efficient constructions of ring signatures. The constructions of such signatures are unconditionally signer-ambiguous, secure in the random oracle model, and exceptionally efficient: adding each ring member increases the cost of signing or verifying by a single modular multiplication and a single symmetric encryption. A large number of extensions, modifcations and applications of ring signatures which are described.

PANDA: PUBLIC AUDITING FOR SHARED DATA WITH EFFICIENT USER REVOCATION IN THE CLOUD

With data storage and sharing services in the cloud, users can easily modify and share data as a group. To ensure shared data integrity can be verified publicly, users in the group need to compute signatures on all the blocks in shared data. Different blocks in shared data are generally signed by different users due to data modifications performed by different users. For security reasons, once a user is revoked from the group, the blocks which were previously signed by this revoked user must be re-signed by an existing user. The straightforward method, which allows an existing user to download the corresponding part of shared data and re-sign it during user revocation, is inefficient due to the large size of shared data in the cloud. In this paper, a novel public auditing mechanism for the integrity of shared data with efficient user revocation in mind is proposed. By utilizing the idea of proxy re-signatures, we allow the cloud to re-sign blocks on behalf of existing users during user revocation, so that existing users do not need to download and re-sign blocks by themselves. In addition, a public verifier is always able to audit the integrity of shared data without retrieving the entire data from the cloud, even if some part of shared data has been re-signed by the cloud. Moreover, our mechanism is able to support batch auditing by verifying multiple auditing tasks simultaneously.

PRIVACY-PRESERVING PUBLIC AUDITING FOR SECURE CLOUD STORAGE

Using Cloud Storage, users can remotely store their data and enjoy the on-demand high quality applications and services from a shared pool of configurable computing resources, without the burden of local data storage and maintenance. However, the fact that users no longer have physical possession of the outsourced data makes the data integrity protection in Cloud Computing a formidable task, especially for users with constrained computing resources. Moreover, users should be able to just use the cloud storage as if it is local, without worrying about the need to verify its integrity. Thus, enabling public auditability for cloud storage is of critical importance so that users can resort to a third party auditor (TPA) to check the integrity of outsourced data and be worry-free. To securely introduce an effective TPA, the auditing process should bring in no new vulnerabilities towards user data privacy, and introduce no additional online burden to user. In this paper, a secure cloud storage system supporting privacy-preserving public auditing is proposed. The TPA performs audits for multiple users simultaneously and efficiently. Extensive security and performance analysis show the proposed schemes are provably secure and highly efficient.

PROVABLE DATA POSSESSION AT UNTRUSTED STORES

A model for provable data possession (PDP) that allows a client that has stored data at an untrusted server to verify that the server possesses the original data without retrieving it. The model generates probabilistic proofs of possession by sampling random sets of blocks from the server, which drastically reduces I/O costs. The client maintains a constant amount of metadata to verify the proof. The challenge/response protocol transmits a small, constant amount of data, which minimizes network communication. Thus, the PDP model for remote data checking supports large data sets in widely-distributed storage systems. It presents two provably-secure PDP schemes that are more efficient than previous solutions, even when compared with schemes that achieve weaker guarantees. In particular, the overhead at the server is low or even constant, as opposed to linear in the size of the data. Experiments verify the practicality of PDP and reveal that the performance of PDP is bounded by disk I/O and not by cryptographic computation.

EXISTING SYSTEM

RL is used in the case of private cloud. In both the systems public auditing is performed. Identity privacy is not ensured in both the The traditional approach for checking data correctness is to retrieve the entire data from the cloud, The existing system is PDP and WWRL. PDP – Provable Data Prossession. WWand then verify data integrity by checking the correctness of signatures or hash values 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 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 or hash values 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 systems. Data privacy is ensured in the WWRL system. Here failing to preserve identity privacy on shared data during public auditing will reveal significant confidential information e.g., which particular user in the group or special block in shared data is a more valuable target to public verifiers.

DRAWBACKS OF EXISTING SYSTEM

Integrity threat

Adversary may try to corrupt the data.

Corruption of data due to hardware failure and human error.

Privacy threat

Identity of the signer is private and confidential to the group.

Public verifier may try to reveal identity of the signer based on verification metadata

PROPOSED SYSTEM

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. 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.

ADVANTAGES OF PROPOSED SYSTEM

A public verifier is able to correctly verify shared data integrity.

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

The ring signatures generated are used to preserve identity privacy and support blockless verifiability.

IMPLEMENTATION

Implementation is the process of converting a new or revised system design into an operational one when the initial design was done for the system; a demonstration was given to the end user about the working system. This process is used to verify and identify any logical miss – working of the system by feeding various combinations of test data. After the approval of the system by both end user and management the system was implemented. System implementation is stage in the project where the theoretical design is turned into the working system.

The major tasks involves in the implementation are:

Computer based/system testing

Training the user personnel

Full system and making the necessary changes as desired by the user

Change over

Maintenance

The most crucial stage is giving the users confidence that the new system will work effectively and efficiently. The performance of reliability of the system is tested and it gained acceptance. The system was implemented successfully. Implementation is a process that means converting a new system into operation. Proper implementation is essential to provide a reliable system to meet organization requirements. During the implementation stage a live demon was undertaken and made in front of end-users. The various features provided in the system wew discussed during implementation.

MODULE DESCRIPTION

CONSTRUCTION AND SECURITY ANALYSIS OF HARS

Homomorphic authenticators also called homomorphic verifiable tags are basic tools to construct public auditing mechanisms. Besides unforgeability i.e., only a user with a private key can generate valid signatures 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,s1 denote a signature on block $m1 \in Zp$, s2 denote a signature on block $m2 \in Zp$.

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.

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.

Given any block m, its block identifier id, and its ring signature σ = (σ1,....σd), a verifier is able to correctly check the integrity of this block under HARS.
For any adversary A, it is computationally infeasible to forge a ring signature under HARS, as long as the Co-CDH assumption on (G1;G2) holds.

ORUTA CONSTRUCTION

The public auditing mechanism includes five algorithms: KeyGen, SigGen, Modify, ProofGen and ProofVerify. In KeyGen, users generate their own public/private key pairs. In SigGen, a user (either the original user or a group user) is able to compute ring signatures on blocks in shared data by using its own private key and all the group members' public keys. Each user in the group is able to perform an insert, delete or update operation on a block, and compute the new ring signature on this new block in Modify. ProofGen is operated by a public verifier and the cloud server together to interactively generate a proof of possession of shared data. In ProofVerify, the public verifier audits the integrity of shared data by verifying the proof.

We first assume the group is static, which means the group is pre-defined before shared data is created in the cloud and the membership of the group is not changed during data sharing. Specifically, before the original user outsources shared data to the cloud, he/she decides all the group members.

SECURITY ANALYSIS OF ORUTA

Security properties of Oruta, including its correctness, unforgeability, identity privacy and data privacy. A public verifier is able to correctly audit the integrity of shared data under Oruta. For an untrusted cloud, it is computationally infeasible to generate a forgery of an auditing proof under Oruta as long as the DL assumption holds.

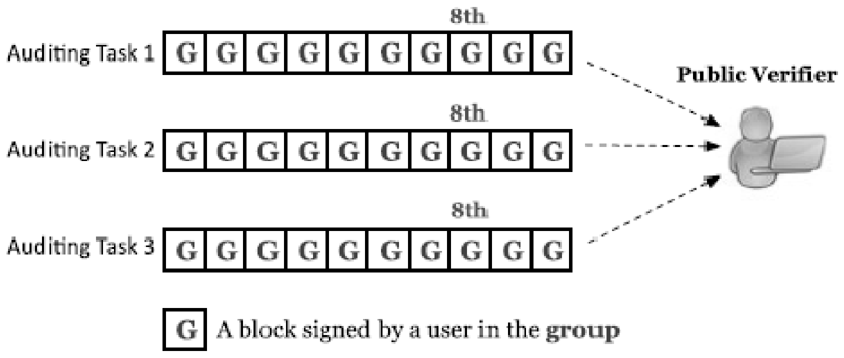

Fig.3.2 Public Auditing

A public verifier is able to correctly audit the integrity of shared data under Oruta.
For an untrusted cloud, it is computationally infeasible to generate a forgery of an auditing proof under Oruta as long as the DL assumption holds.
During public auditing, the probability for a public verifier to distinguish the identities of all the signers on the c selected blocks in shared data is at most 1=d^c.
Given an auditing proof = $\{\lambda, \boldsymbol{\mu}, \emptyset, \{id_j\}_{j\epsilon J}\}$, it is computationally infeasible for a public verifier to reveal any private data in shared data under oruta as long as the dl assumption holds.

BATCH AUDITING

A public verifier may need to verify the correctness of multiple auditing tasks in a very short time. Directly verifying these multiple auditing tasks separately would be inefficient. By leveraging the properties of bilinear maps, we can further extend Oruta to support batch auditing, which can verify the correctness of multiple auditing tasks simultaneously and improve the efficiency of public auditing.

REDUCE SIGNATURE STORAGE AND SUPPORT DYNAMIC OPERATION

Important issue we should consider in the construction of Oruta is the size of storage used for ring signatures. According to the generation of ring signatures in HARS, a block m is an element of Z_p and its ring signature contains d elements of G_1, where G_1 is a cyclic group with order p. It will be very frustrating for users, because cloud service providers, such as Amazon, will charge users based on the storage space they use.

To reduce the storage of ring signatures on shared data and still allow the public verifier to audit shared data efficiently,we exploit an aggregated approach from to expand the size of each block in shared data into $k \times |p|$ bits.

To enable each user in the group to easily modify data in the cloud, Oruta should also support dynamic operations on shared data. A dynamic operation includes an insert, delete or update operation on a single block . However, since the computation of a ring signature includes an identifier of a block as presented in HARS, traditional methods, which only use the index of a block as its identifier i.e., the index of block m_j is j, are not suitable for supporting dynamic operations on shared data efficiently.The reason is that, when a user modifies a single block in
shared data by performing an insert or delete operation, the indices of blocks that after the modified block are all changed, and the changes of these indices require users, who are sharing the data, to re-compute the signatures of these blocks, even though the content of these blocks are not modified.

By utilizing an index hash table, which is a data structure indexing each block based on its hash value, our mechanism can allow a user to efficiently perform a dynamic operation on a single block, and avoid this type of re-computation on other blocks.

PERFORMANCE ANALYSIS

Computation Cost

During an auditing task, the public verifier first generates some random to values construct an auditing challenge, which only introduces a small cost in computation. Then, after receiving the auditing challenge, the cloud server needs to compute an auditing proof $\{\lambda, \boldsymbol{\mu}, \emptyset, \{id_j\}_{j\epsilon J}\}$.

Communication Cost

The communication cost of Oruta is mainly introduced by two aspects: the auditing challenge and auditing proof. For each auditing challenge $\{j, y_j\}_{j\epsilon J}$, the communication cost is c(|q| + |n|) bits, where |q| is the length of an element of Zq and |n| is the length of an index. Each auditing proof $\{\lambda, \boldsymbol{\mu}, \emptyset, \{id_j\}_{j\epsilon J}\}$ contains $(k + d)$ elements of G_1, k elements of Z_p and c elements of Z_q, therefore the communication cost of one auditing proof is $(2k + d)|p| + c|q|$ bits.

III. CONCLUSION

In this project, we propose Oruta, a privacy-preserving public auditing mechanism for shared data in the cloud. We utilize ring signatures to construct homomorphic authenticators, so that a public verifier is able to audit shared data integrity without retrieving the entire data, yet it cannot distinguish who is the signer on each block. To improve the efficiency of verifying multiple auditing tasks, we further extend our mechanism to support batch auditing.

REFERENCES

[1]. B. Wang, B. Li, and H. Li, “Oruta: Privacy-Preserving Public Auditing for Shared Data in the Cloud,”(2012) Proc. IEEE Fifth Int’l Conf. Cloud Computing, pp. 295-302.

[2]. G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song,(2007) “Provable Data Possession at Untrusted Stores,” Proc. 14th ACM Conf. Computer and Comm.Security (CCS ’07),pp. 598-610.

[3]. L. Rivest, A. Shamir, and Y. Tauman,(2011) “How to Leak a Secret,” Proc. Seventh Int’l Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT’01), pp. 552-565.

[4]. B. Wang, B. Li, and H. Li, (Dec 2013) “Panda: Public Auditing for Shared Data with Efficient User Revocation in the Cloud,” IEEE Trans. Services Computing, DOI: 10.1109/TSC.2013.2295611.

[5]. C. Wang, S.S. Chow, Q. Wang, K. Ren, and W. Lou, (Feb 2013) “Privacy-Preserving Public Auditing for Secure Cloud Storage,” IEEE Trans. Computers, vol. 62, no. 2, pp. 362-375.

[6]. Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S.S Yau,(2011) “Dynamic Audit Services for Integrity Verification of Outsourced Storages in Clouds,” Proc. ACM Symp. Applied Computing (SAC’11),pp.1550-1557.

[7]. K. Ren, C. Wang, and Q. Wang,(2012) “Security Challenges for the Public Cloud,” IEEE Internet Computing, vol. 16, no. 1, pp. 69-73.
[8]. D. Song, E. Shi, I. Fischer, and U. Shankar,(2012) “Cloud Data Protectionfor the Masses,” Computer, vol. 45, no. 1, pp. 39-45.
[9]. C. Wang, Q. Wang, K. Ren, and W. Lou,(2010) “Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing,” Proc. IEEE INFOCOM, pp. 525-533.
[10]. B. Wang, M. Li, S.S. Chow, and H. Li,(2013) “Computing EncryptedCloud Data Efficiently under Multiple Keys,” Proc. IEEE Conf. Comm. and Network Security (CNS ’13), pp. 90-99.
[11]. R. Rivest, A. Shamir, and L. Adleman,(1978) “A Method for ObtainingDigital Signatures and Public Key Cryptosystems,” Comm. ACM, vol. 21, no. 2, pp. 120-126.
[12]. Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou,(2009) “Enabling PublicVerifiability and Data Dynamic for Storage Security in Cloud Computing,” Proc. 14th European Conf. Research in Computer Security (ESORICS’09), pp. 355-370.