PPAKBS: Privacy-Preserving Public Auditing for Key generation Based Cloud Storage using Batch Signature

Mr. R. Nallakumar, M.E., (Ph.d)
Department of computer science and engineering
Anna University Regional Centre
Coimbatore, Tamilnadu

N Keerthika
Department of computer science and engineering
Anna University Regional Centre
Coimbatore, Tamilnadu
E-mail: keerthika163@gmail.com

Abstract: To protect privacy data in cloud storage beside frauds, adding imperfection tolerance to cloud storage jointly with data truthfulness checking and malfunction reparation becomes critical. In this paper, we propose a novel public auditing framework for the public key generation-code-based cloud storage. To solve the auditing difficulties of unsuccessful authenticators in the lack of data possessors, we introduce a batch signature process, which is confidential to stimulate the authenticators, into the conventional public auditing framework model. Moreover, the proposed system designs a novel public truthfull authenticator, which is generated by a combine of private keys and can be regenerated using biased keys. Extensive security examination shows that proposed method is verifiable secure under random batch model and experimental evaluation indicates that our scheme is highly proficient and can be possibly integrated into the key generation -code-based cloud storage.

Keywords: About six key words separated by commas (Minimum 4 key words)

I. INTRODUCTION

Cloud storage is now gaining popularity because it offers a flexible on-demand data outsourcing service with appealing benefits: relief of the burden for storage management, universal data access with location independence, and avoidance of capital expenditure on hardware, software, and personal maintenance, etc., [1]. Nevertheless, this new paradigm of data hosting service also brings new security threats towards users’ data, thus making individuals or enterprisers still feel hesitant.

While Cloud Computing makes these advantages more appealing than ever, it also brings new and challenging security threats towards users’ outsourced data. Since cloud service providers (CSP) are separate administrative entities, data outsourcing is actually relinquishing user’s ultimate control over the fate of their data. As a result, the correctness of the data in the cloud is being put at risk due to the following reasons. First of all, although the infrastructures under the cloud are much more powerful and reliable than personal computing devices, they are still facing the broad range of both internal and external threats for data integrity. Examples of outages and security breaches of noteworthy cloud services appear from time to time.

The integrity of data in cloud storage, however, is subject to skepticism and scrutiny, as data stored in an untrusted cloud can easily be lost or corrupted, due to hardware failures and human errors. To protect the integrity of cloud data, it is best to perform public auditing by introducing a third party auditor (TPA), who offers its auditing service with more powerful computation and communication abilities than regular users.

A privacy-preserving public auditing system for data storage security in cloud computing in this the homomorphic linear authenticator and random masking to guarantee that the third party auditor (TPA) would not learn any knowledge about the data content stored on the cloud server during the efficient auditing process. It not only eliminates the burden of cloud user from the tedious and possibly expensive auditing task, but also alleviates the users’ fear of their outsourced data leakage.

Many mechanisms dealing with the integrity of outsourced data without a local copy have been proposed under different system and security models up to now. The most significant work among these studies are the PDP (provable data possession) model and POR (proof of retrievability) model, which were originally proposed for the single-server scenario by Ateniese et al. [2] and Juels and Kaliski [3], respectively. Considering that files are usually striped and redundantly stored across multi-servers or multi-clouds, [4] explore integrity verification schemes suitable for such multi-servers or multi-clouds setting with different redundancy schemes, such as replication, erasure codes, and, more recently, regenerating codes.

II. RELATED WORK

In [2] authors introduced 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. To presented two provably-secure PDP schemes that are more efficient than previous solutions, even when compared with schemes that achieve weaker guarantees.

In [3] authors defined and explore proofs of retrievability (PORs). A POR scheme enables an archive or back-up service (prover) to produce a concise proof that a user (verifier) can retrieve a target file F, that is, that the archive retains and reliably transmits file data sufficient for the user to recover F in its entirety. A POR may be viewed as a kind of cryptographic proof of knowledge (POK), but one specially designed to handle a large file (or bitstring) F. To explore POR protocols here in which the communication costs, number of memory accesses for the prover, and storage requirements of the user (verifier) are small parameters essentially independent of the length of F.

In [4] authors illustrated a provably-secure scheme that allows a client that stores t replicas of a file in a storage system to verify through a challenge-response protocol that (1) each unique replica can be produced at the time of the challenge and that (2) the storage system uses t times the storage required to store a single replica. MR-PDP extends previous work on data possession proofs for a single copy of a file in a client/server storage system. Using MR-PDP to store t replicas is computationally much more efficient than using a single-replica PDP scheme to store t separate, unrelated files (e.g., by encrypting each file separately prior to storing it).

In [5] authors introduced a HAIL (High-Availability and Integrity Layer), a distributed cryptographic system that allows a set of servers to prove to a client that a stored file is intact and retrievable. HAIL strengthens, formally unifies, and streamlines distinct approaches from the cryptographic and distributed-systems communities. Proofs in HAIL are efficiently computable by servers and highly compact typically tens or hundreds of bytes, irrespective of file size. HAIL cryptographically verifies and reactively reallocates file shares. It is robust against an active, mobile adversary, i.e., one that may progressively corrupt the full set of servers. We propose a strong, formal adversarial model for HAIL, and rigorous analysis and parameter choices.

In [6] authors proposed a novel efficient Distributed Multiple Replicas Data Possession Checking (DMRDPC) scheme to tackle new challenges. The goal is to improve efficiency by finding an optimal spanning tree to define the partial order of scheduling multiple replicas data possession checking. But since the bandwidths have geographical diversity on the different replica links and the bandwidths between two replicas are asymmetric, to must resolve the problem of finding an Optimal Spanning Tree in a Complete Bidirectional Directed Graph, which we call the FOSTCBDG problem.

In [7] authors discussed Remote Data Checking (RDC) technique by which clients can establish that data outsourced at untrusted servers remains intact over time. RDC is useful as a prevention tool, allowing clients to periodically check if data has been damaged, and as a repair tool whenever damage has been detected. Initially proposed in the context of a single server, RDC was later extended to verify data integrity in distributed storage systems that rely on replication and on erasure coding to store data redundantly at multiple servers. Recently, a technique was proposed to add redundancy based on network coding, which offers interesting tradeoffs because of its remarkably low communication overhead to repair corrupt servers.

In [8] authors designed and implement a practical data integrity protection (DIP) scheme for a specific regenerating code, while preserving its intrinsic properties of fault tolerance and repair-traffic saving. Our DIP scheme is designed under a mobile Byzantine adversarial model, and enables a client to feasibly verify the integrity of random subsets of outsourced data against general or malicious corruptions. It works under the simple assumption of thin-cloud storage and allows different parameters to be fine-tuned for a performance-security trade-off. To implement and evaluate the overhead of our DIP scheme in a real cloud storage testbed under different parameter choices. To further analyze the security strengths of our DIP scheme via mathematical models.

### III. PROPOSED METHODOLOGY

The proposed methodology presents a public auditing system and discusses two basic schemes and their demerits. Then we present our main scheme and show how to extent our main scheme to support batch auditing for the third party auditor (TPA) upon delegations from multiple users. Finally, the proposed methodology discusses how to generalize our privacy-preserving public auditing scheme and its support of data dynamics.
A. Homomorphic Ring Authenticable Signatures (HRAS)

Homomorphic ring authenticators are employed to store blocks of knowledge which will have distinctive properties: correctness, block less verifiability, unforgeability, non plasticity and identity privacy. HARS contains three process: Key Generation (KG), Ring Signature (RS) and Ring Verification (RV). In KG, each user in the group generates her public key and private key. In RS, a user in the group is able to sign a block with her private key and all the group members’ public keys. A verifier is allowed to check whether a given block is signed by a group member in RV.

Key Generation: Here Users can generate their own public/private key pairs.

For a user \( u_i \) in the group \( U \), she randomly picks \( x_i \in Z_p \) and computes \( w_i = g^x_i \in G_2 \). Then, user \( u_i \)’s public key is \( pk_i = w_i \) and her private key is \( sk_i = x_i \).

Ring Signature: Here User must reckon the ring signatures on the blocks in shared information by mistreatment private key and cluster members’ public keys.

Given all the \( d \) users’ public keys \( (pk_1, \ldots, pk_d) = (w_1, \ldots, w_d) \), a block \( m \in Z_p \), the identifier of this block \( id \) and the private key \( s_k \), for some \( s \), user us randomly chooses \( a_i \in Z_p \) for all \( i \neq s \), where \( i \in [1, d] \), and let \( \sigma_i = g^a_i \). Then, computes

\[
\beta = H_1(id) \theta^m \in G_1 \quad (1) \\
\sigma_x = \left( \frac{\beta}{\psi(\Pi_{i \neq s} w_i^{d_i})} \right)^{1/sx} \in G_1 \quad (2)
\]

Ring Verification: The general public leading role audits the integrity of shared information by corroboratory the proof verification.

Given all the \( d \) users’ public keys \( (pk_1, \ldots, pk_d) = (w_1, \ldots, w_d) \), a block \( m \), an identifier \( id \) and a ring signature \( \sigma = (\sigma_1, \ldots, \sigma_d) \), a verifier first computes \( \beta = H_1(id) \theta^m \in G_1 \), and then checks,

\[
e(\beta, g_2)A = \prod_{i=1}^{d} e(\sigma_i, w_i) \quad (3)
\]

B. Privacy-Preserving Public Auditing Reduce Signature Storage

The Privacy preserving auditing reduce signature is an important issue should consider in the construction of auditing

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 means a \( p \)-bit block requires a \( d \times p \)-bit ring signature, which forces users to spend a huge amount of space on storing ring signatures. It is very frustrating for users, because cloud service providers, such as Amazon, will charge users based on the storage space they used. To reduce the storage for ring signatures and still allow the TPA to audit shared data efficiently, we exploit an aggregated approach.

C. Dynamic Cloud distributions

To enable dynamic cloud operation of each user in the group to easily modify data in the cloud and share the latest version of data with the rest of the group, privacy preserving auditing should also support dynamic operations on shared data. An 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 HRA5), traditional methods, which only use the index of a block as its identifier, are not suitable for supporting dynamic operations on shared data. 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 to re-compute the signatures of these blocks, even though the content of these blocks are not modified.

D. Secure Batch Auditing

The secure batch auditing usage of public auditing in the cloud, the TPA may receive amount of auditing requests from different users in a very short time. Unfortunately, allowing the TPA to verify the integrity of shared data for these users in several separate auditing tasks would be very inefficient. Therefore, with the properties of bilinear maps, to further extend privacy is to support batch auditing, which can improve the efficiency of verification on multiple auditing tasks. More concretely, assume there are \( B \) auditing tasks need to be operated, the shared data in all the \( B \) auditing tasks are denoted as \( M_1, \ldots, M_B \) and the number of users sharing data \( M_b \) is described as \( db \), where \( 1 < b < B \). To efficiently audit these shared data for different users in a single auditing task, the TPA sends an auditing message as \( \{(j, y_j)\}_{j=1}^{b} \) to the cloud server. After receiving the auditing message, the cloud server generates an auditing proof \( \{ \lambda_b, \mu_b, \Phi_b, \{id_b\}_{j=1}^{b} \} \) for each
shared data Mb as we presented in ProofGen, where \(1 < b < B,\)
\(1 < l < k.\)

IV. CONCLUSION

The proposed system presents a privacy preserving public auditing mechanism for shared data in the cloud. The system utilize a ring signatures to construct homomorphic authenticators, so the third party authenticator is able to audit the integrity of shared data, yet cannot distinguish who is the signer on each block, which can achieve identity privacy. To better appropriate for the dynamic distribution code, to design our authenticator based on the batch signature. This authenticator can be efficiently generated by the data owner simultaneously with the encoding procedure.

REFERENCES