A Hybrid Cloud Approach For Secure Authorized De-Duplication

1Ch. Sudheer, 2A. Jyothi, 3A. Obulesh
1M.Tech, Anurag Group of Institutions, Hyderabad, Telangana, India
2Asst.Professor, Anurag Group of Institutions, Hyderabad, Telangana, India
3Asst.Professor, Anurag Group of Institutions, Hyderabad, Telangana, India

Abstract: Data de-duplication is one of important data compression techniques for eliminating duplicate copies of repeating data, and has been widely used in cloud storage to reduce the amount of storage space and save bandwidth. To protect the confidentiality of sensitive data while supporting de-duplication, the convergent encryption technique has been proposed to encrypt the data before outsourcing. To better protect data security, this paper makes the first attempt to formally address the problem of authorized data de-duplication. Different from traditional de-duplication systems, the differential privileges of users are further considered in duplicate check besides the data itself. We also present several new de-duplication constructions supporting authorized duplicate check in a hybrid cloud architecture. Security analysis demonstrates that our scheme is secure in terms of the definitions specified in the proposed security model. In this paper, we present a scheme that permits a more fine-grained trade-off. The intuition is that outsourced data may require different levels of protection, depending on how popular it is: content shared by many users, such as a popular song or video, arguably requires less protection than a personal document, the copy of a payslip or the draft of an unsolicited scientific paper. As more corporate and private users outsource their data to cloud storage providers, recent data breach incidents make end-to-end encryption an increasingly prominent requirement. Unfortunately, semantically secure encryption schemes render various cost-effective storage optimization techniques, such as data de-duplication, ineffective. We present a novel idea that differentiates data according to their popularity. Based on this idea, we design an encryption scheme that guarantees semantic security for unpopular data and provides weaker security and better storage and bandwidth benefits for popular data. This way, data de-duplication can be effective for popular data, whilst semantically secure encryption protects unpopular content. We show that our scheme is secure under the Symmetric External Decisional Diffie-Hellman Assumption in the random oracle model.

Keywords: Cloud, Encryption, de-duplication.

I. INTRODUCTION

Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet). The name comes from the common use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams. Cloud computing entrusts remote services with a user's data, software and computation. Cloud computing consists of hardware and software resources made available on the Internet as managed third-party services. [4] D. Song, E. Shi, I. Fischer, and U. Shankar These services typically provide access to advanced software applications and high-end networks of server computers.

1.1 Characteristics and Services Model

The silent characteristics of cloud computing based on the definitions provided by the National Institute of Standards and Terminology (NIST) are outlined below.

- **On-demand self-service**: A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service’s provider.
- **Broad network access**: Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, laptops, and PDAs).
- **Resource pooling**: The provider’s computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location-independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or data center). Examples of resources include storage, processing, memory, network bandwidth, and virtual machines.
- **Rapid elasticity**: Capabilities can be rapidly and elastically provisioned, in some cases automatically, to quickly scale out and rapidly released to quickly scale in. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time.
- **Measured service**: Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be managed, controlled, and reported providing transparency for both the provider and consumer of the utilized service.
1.2 Services Models

Cloud Computing comprises three different service models, namely Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS). The three service models or layer are completed by an end user layer that encapsulates the end user perspective on cloud services. The model is shown in figure below. If a cloud user accesses services on the infrastructure layer, for instance, she can run her own applications on the resources of a cloud infrastructure and remain responsible for the support, maintenance, and security of these applications herself[2]. If she accesses a service on the application layer, these tasks are normally taken care of by the cloud service provider.

![Fig.1: Structure of service models.](image)

1.3 Benefits of cloud computing

- **Achieve economies of scale** – increase volume output or productivity with fewer people. Your cost per unit, paper or product plummets.
- **Reduce spending on technology infrastructure.** Maintain easy access to your information with minimal upfront spending. Pay as you go (weekly, quarterly or yearly), based on demand.
- **Globalize your workforce on the cheap.** People worldwide can access the cloud, provided they have an Internet connection.
- **Streamline processes.** Get more work done in less time with less people.
- **Reduce capital costs.** There’s no need to spend big money on hardware, software or licensing fees.
- **Improve accessibility.** You have access anytime, anywhere, making your life so much easier
- **Monitor papers more effectively.** Stay within budget and ahead of completion cycle times.
- **Less personnel training is needed.** It takes fewer people to do more work on a cloud, with a minimal learning curve on hardware and software issues.
- **Minimize licensing new software.** Stretch and grow without the need to buy expensive software licenses or programs.
- **Improve flexibility.** You can change direction without serious “people” or “financial” issues at stake.

Provable data possession (PDP) is a technique for ensuring the integrity of data in storage outsourcing [10]. In this paper, we address the construction of an efficient PDP scheme for distributed cloud storage to support the scalability of service and data migration [7], in which we consider the existence of multiple cloud service providers to cooperatively store and maintain the clients’ data. We present a cooperative PDP (CPDP) scheme based on homomorphic verifiable response and hash index hierarchy. We prove the security of our scheme based on multi-prover zero-knowledge proof system, which can satisfy completeness, knowledge soundness, and zero-knowledge properties. In addition, we articulate performance optimization mechanisms for our scheme, and in particular present an efficient method for selecting optimal parameter values to minimize the computation costs of clients and storage service providers [8]. Our experiments show that our solution introduces lower computation and communication overheads in comparison with non-cooperative approaches.

The rest of the organized as follows: Section 2 gives literature, Section 3 describe the proposed method, Section 4 gives results and discussion and finally Section 5 conclude the paper.

**II. LITERATURE**

Literature survey is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy of company strength. Once these things are satisfied, ten next steps are to determine which operating system and language can be used for developing the tool. Once the programmers start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from book or from websites. Before building the system the above consideration taken into account for developing the proposed system.

The veritable response and hash index hierarchy, proposed a cooperative PDP scheme to support dynamic scalability on multiple storage servers. also showed that our scheme provided all security properties required by zero knowledge interactive proof system, so that it can resist various attacks even if it is deployed as a public audit service in clouds. Furthermore, optimized the probabilistic query and periodic verification to improve
the audit performance. Our experiments clearly demonstrated that our approaches only introduce a small amount of computation and communication overheads. Therefore, our solution can be treated as a new candidate for data integrity verification in outsourcing data storage systems. As part of future work, we would extend our work to explore more effective CPDP constructions. First, from our experiments we found that the performance of CPDP scheme, especially for large files, is affected by the bilinear mapping operations due to its high complexity. To solve this problem, RSA based constructions may be a better choice, but this is still a challenging task because the existing RSA based schemes have too many restrictions on the performance and security.

Next, from a practical point of view, we still need to address some issues about integrating our CPDP scheme smoothly with existing systems, for example, how to match index hash hierarchy with HDFS’s two-layer name space, how to match index structure with cluster-network model, and how to dynamically update the CPDP parameters according to HDFS’ specific requirements. Finally, it is still a challenging problem for the generation of tags with the length irrelevant to the size of data blocks. We would explore such an issue to provide the support of variable-length block verification.

2.1 Overview of Integrity Verification and Availability Techniques

Integrity verification should be made by clients to assure that their data has been properly stored and maintained in third party server. The availability and provability of data is provided by multi-cloud concept. The overview of various Provable Data Possession and Proof of Retrievability techniques are summarized as follows.

2.2 Space-Efficient Block Storage Integrity

It provide block-level integrity in encrypted storage systems [13] (i.e.), client detect the modification of data blocks in un trusted storage server. This neither change the block size nor the number of sectors accessed in modern storage systems. A trusted client component maintains state to authenticate blocks returned by the storage server for minimizing the size of state. Basic block integrity that exhibits a tradeoff between the level of security and the additional client’s storage overhead, and evaluations requires an average of only 0.01 bytes per 1024-byte block. Due to the length-preserving requirements for cryptographic blocks, it is not possible to add information to each block (e.g., a MAC) [14] to detect its modification. In I/O applications, it is undesirable to put these MACs in separate blocks also stored at the service, which would require the retrieval of two blocks (one of data, one of MACs) on the critical path of client read operation.

2.3 Provable Data possession

A cooperative PDP (CPDP) scheme without compromising data privacy based on modern cryptographic techniques, such as interactive proof system (IPS). Further introduce an effective construction of CPDP scheme using above-mentioned structure. Moreover, To give a security analysis of our CPDP scheme from the IPS model. Prove that this construction is a multi-prover zero-knowledge proof system (MP-ZKPS) which has completeness, knowledge soundness, and zero-knowledge properties. These properties ensure that CPDP scheme can implement the security against data leakage attack and tag forgery attack.

To improve the system performance with respect to our scheme, To analyze the performance of probabilistic queries for detecting abnormal situations. This probabilistic method also has an inherent benefit in reducing computation and communication overheads. Probable Data Possession at untrusted Stores Giuseppe Ateniese [8] et all introduce a model which based on provable data possession (PDP) [10]. This is used for Verifying that server is processing the original data without retrieving it. In this model probabilistic proof of possession is generated by sampling random sets of blocks from the server. This helps to reduces I/O cost.

Fig. 2: Provable Data Possession at Untrusted Stores.

As shown in Fig 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. DP model for remote data checking supports large data sets in widely-distributed storage systems. A key component of this mechanism is the homomorphic verifiable tags. As shown in Fig client maintains a constant amount of metadata to verify the proof.

2.4 Privacy Preserving Public Auditing

C.Cong Q Wang And D.Boneh, c.centry[1] [11] and[12] proposed Privacy Preserving Public Auditing technique. In this technique public auditing allows TPA along with user to check the integrity of the outsourced data stored on a cloud & Privacy Preserving allows TPA to do auditing without requesting data. Here TPA can audit the data by maintaining cloud data privacy. They have used the homomorphic linear authenticator and random masking to guarantee that the TPA would not learn any knowledge about the data content stored on the cloud
server during the efficient auditing process, which not only eliminates the burden of cloud user from the tedious and possibly expensive auditing task, but also prevent the users from fear of the outsourced data leakage.

2.5 LT Codes-based Secure and Reliable Cloud Storage Service

Ning Cao[5][16] et all explore the problem of secure and cloud storage with the efficiency consideration of both data repair and data retrieval, and design a LT codes based cloud storage service (LTCS). LTCS provides efficient data retrieval for data users by utilizing the fast Belief Propagation decoding algorithm, and releases the data owner from the burden of being online by enabling public data integrity check and employing exact repair. LTCS is much faster data retrieval than the erasure codes based solutions. It introduces less storage cost, much faster data retrieval, and comparable communication cost comparing to network coding-based storage services.

2.6 Privacy-Preserving Public Auditing for Shared Data in the Cloud

Boyang Wang et [22]all proposed Oruta, the first privacy preserving public auditing mechanism for shared data in the cloud in . They have used ring signatures to construct homomorphic authenticators, so the TPA is able to audit the integrity of shared data, without retrieving the entire data. They have used HARS and its properties for constructing Oruta.

III. PROPOSED METHOD

To support distributed cloud storage, illustrate a representative architecture used in our cooperative PDP scheme as shown in Figure Our architecture has a hierarchy structure which resembles a natural representation of file storage

Step1:
- Express Layer: offers an abstract representation of the stored resources.
- Service Layer: offers and manages cloud storage services
- Storage Layer: realizes data storage on many physical devices.

Step2:
We make use of this simple hierarchy to organize data blocks from multiple CSP services into a large size file by shading their differences among these cloud storage systems.

Step 3:
For example, In Figure the resources in Express Layer are split and stored into three CSPs, that are indicated by different colors, in Service Layer. In turn, each CSP fragments and stores the assigned data into the storage servers in Storage Layer.

Step 4
In turn, each CSP fragments and stores the assigned data into the storage servers in Storage Layer. In storage layer, we define a common fragment structure that provides probabilistic verification of data integrity for outsourced storage.

Step 5
An instance of this structure is shown in storage layer of Figure an outsourced file $F$ is split into $n$ blocks $\{m1,2,\ldots,nn\}$, and each block $mi$ is split into sectors $\{mi,1,mi,2,\ldots,mi,s\}$.

Step 6

The fragment structure consists of $n$ block-tag pair $(mi, \sigma i)$, where $\sigma i$ is a signature tag of block $mi$ generated by a set of secrets $\tau = (\tau 1, \tau 2,\ldots)$. $Pk \in P$

$P(k,F(k),\sigma(k)) \leftrightarrow V)

$(pk, \psi) = \{1F = \{(k)\}$ is intact

$0 = \{(k)\}$ is changed\} (1)

where each $Pk$ takes as input a file $F(k)$ and a set of tags $\sigma(k)$, and a public key $pk$ and a set of public parameters $\psi$ are the common input between $P$ and $V$. A trivial way to realize the CPDP is to check the data stored in each cloud one by one, i.e., \$P \in P\ (P(k,F(k),\sigma(k)) \leftrightarrow V) \psi$, Where $\Lambda$ denotes the logical AND operations among the Boolean outputs of all protocols $(Pk, V)$. In order to check the data integrity, the fragment structure implements probabilistic verification as follows given a random chosen challenge (or query) $Q = \{(i, vi)\}i\in RI$, where $I$ is a subset of the block indices and $vi$ is a random coefficient.
3.1 System Model
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) [17] 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 Fig. 1. Alice and Bob share a data file in the cloud, and a public verifier audits shared data integrity with existing mechanisms.

- **Comparison Among Different Mechanisms**

  Our system model includes the cloud server, a group of users and Oruta stands for “One Ring to Rule Them All.” [18] a public verifier, 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.

3.2 Threat Model

3.2.1 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. 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 sign 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.

3.2 Technique (ORUTA)

To enable the TPA efficiently and securely verify shared data for a group of users, Oruta should be designed to achieve following properties:

- **Public Auditing**: The third party auditor is able to verify the integrity of shared data for a group of users without retrieving the entire data.

- **Correctness**: The third party auditor is able to correctly detect whether there is any corrupted block in shared data.

- **Unforgeability**: Only a user in the group can generate valid verification information on shared data.

- **Identity Privacy**: During auditing, the TPA cannot distinguish the identity of the signer on each block in shared data.

1.3 Possible Alternative Approaches

To preserve the identity of the signer on each block during public auditing, one possible alternative approach is to ask all the users of the group to share a global private key. Then, every user is able to sign blocks with this global private key[20]. However, once one user of the group is compromised or leaving the group, a new global private key must be generated and securely shared among the rest of the group, which clearly introduces huge overhead to users in terms of key management and key distribution. While in our solution, each user in the rest of the group can still utilize its own private key for computing verification metadata without generating or sharing any new secret keys.

Another possible approach to achieve identity privacy is to add a trusted proxy between a group of users and the cloud in the system model. More concretely, each member’s data is collected, signed, and uploaded to the cloud by this trusted proxy, then a public verifier can only verify and learn that it is the proxy signs the data, but cannot learn the identities of group members. Yet, the security of this method is threatened by the single point failure of the proxy. Besides, sometimes, not all the group members would like to trust the same proxy for generating signatures and uploading data on their behalf. Utilizing group signatures is also an alternative option to preserve identity privacy.

Unfortunately, as shown in our recent work how to design an efficient public auditing mechanism based on group signatures remains open.2 Trusted Computing offers another possible alternative approach to achieve the design objectives of our mechanism. Specifically, by utilizing direct anonymous attestation which is adopted by the Trusted Computing Group as the anonymous method for remote authentication in trusted platform module, users are able to preserve their identity privacy on shared data from a public verifier. The main problem with this approach is that it requires all the users using designed hardware, and needs the cloud provider to
move all the existing cloud services to the trusted computing environment, which would be costly and impractical.

3.4 Ring Signatures

The concept of ring signatures was first proposed by Rivets in 2001[6][21]R. Rivest, A. Shamir, and L. Adelman. 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 (referred to as BGLS in this paper) is constructed on bilinear maps. We will extend this ring signature scheme to construct our public auditing mechanism.

3.5 New Ring Signature Scheme (Overview)

As 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. However, traditional ring signatures cannot be directly used into public auditing mechanisms [19], because these ring signature schemes do not support lockless verifiability. Without block less 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. Therefore, 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 block less 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 2001, a ring signature scheme was proposed by Rivest, Shamir, Tauman.

The signature scheme convinces a verifier that a document has been signed by one of n independent signers.

• A signer can connect the head and tail of the series of values by using own secret key.

• A verifier computes series of values from the message and members’ public keys, and checks that a signature has a ring structure.

• Anyone cannot distinguish a part of the signature which is used secret key.

3.6 Homomorphic Authenticators

Homomorphic authenticators (also called homomorphic verifiable tags) are basic tools to construct public auditing mechanisms Besides un forgeability (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[9] H. Shacham and B. Waters, “Block less 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.

3.7 RSA with Holomorphic Encryption

Homomorphic Encryption systems are used to perform operations on encrypted data without knowing the private key (without decryption), B. Chen, R. Curtmola[13], the client is the only holder of the secret key. Homomorphic verifiable response is the key technique of CPDP because it not only reduces the communication bandwidth. In this technique we are using RSA.

RSA Algorithm

Step 1: RSA is a block cipher in which the plain text and cipher text are integers between 0 and n-1 for some n. encryption and decryption are of the following form period for some plain text block M and cipher block c. 

\[ c = M^e \mod n \]

\[ M = c^d \mod n = (M^e)^d \mod n = (M^e)^d \mod n. \]

\[ e \cdot d \equiv 1 \mod \phi(n) \]

\[ \phi(n) = (p-1)(q-1) \]

\[ p \cdot q = n \]

Step 2: 

\[ M = c^e \mod n \]

\[ M = (M^e)^d \mod n \] both sender and receiver knows the values of d. This is public key encryption algorithm with a public key of \[ K_U = \{e, n\} \] and private key of \[ K_R = \{d, n\} \].

Step 3: It is possible to find values of e,d,n such that M ed mod n=M for all M<n. It is relatively easy to calculate e,d,n for all values of M<n. it is infeasible to determine d given e and n.

![Fig. 4: RSA Algorithm.](image)

JOSEPH’S JOURNAL OF MULTIDISCIPLINARY STUDIES (JJMDS)
St.Joseph’s Degree & PG College, King Koti, Hyderabad
Step 5: Suppose we have two ciphers $C_1$ and $C_2$ such that

$$C_1 = m_1e \mod n$$
$$C_2 = m_2e \mod n$$

$$C_1.C_2 = m_1em_2e \mod n = (m_1m_2)e \mod n$$

RSA cryptosystem realize the properties of the multiplicative Homomorphic encryption, but it still has a lack of security, because if we assume that two ciphers $C_1$, $C_2$ corresponding respectively to the messages $m_1$, $m_2$, so

$$C_1 = m_1e \mod n$$
$$C_2 = m_2e \mod n$$

Step 6: The client sends the pair ($C_1$, $C_2$) to the Cloud server, the server will perform the calculations requested by the client and sends the encrypted result ($C_1 \times C_2$) to the client.

IV. RESULTS AND DISCUSSION

The paper proposing a method consist of 6 modules to accomplish the task.

- **Module1 (Cloud User)**

The Cloud User who have a large amount of data to be stored in multiple clouds and have the permissions to access and manipulate stored data. The User’s Data is converted into data blocks The data block is uploaded to the cloud. The TPA views the data blocks and Uploaded in multi cloud. The user can update the uploaded data. If the user wants to download their files, the data’s in multi cloud is integrated and downloaded.

- **Module2 (Owner Registration)**

In this module an owner has to upload its files in a cloud server, he/she should register first. Then only he/she can be able to do it. For that he needs to fill the details in the registration form. These details are maintained in a database.

- **Module 3(Owner Login)**

In this module, any of the above mentioned person have to login, they should login by giving their email id and password.

- **Module4 (Third Party Auditor Registration)**

In this module if a third party auditor TPA (maintainer of clouds) wants to do some cloud offer, they should register first. Here we are doing like, this system allows only three cloud service providers.

- **Module 5(Data Sharing)**

Consider how to audit the integrity of shared data in the cloud with static groups. It means the group is pre-defined before shared data is created in the cloud and the membership of users in the group is not changed during data sharing. The original user is responsible for deciding who is able to share her data before outsourcing data to the cloud. Another interesting problem is how to audit the integrity of shared data in the cloud with dynamic groups a new user can be added into the group and an existing group member can be revoked during data sharing while still preserving identity privacy

- **Module6 (Third Party Auditor)**

Trusted Third Party (TPP) who is trusted to store verification parameters and often publics query services for these parameters. In our system the Trusted Third Party, view the user data blocks and uploaded to the distributed cloud. In distributed cloud environment each cloud has user data blocks. If any modification tried by cloud owner a alert is send to the Trusted Third Party.

4.1 System Architecture

The paper “Oruta-Privacy preserving public Auditing for Shared Data in the Cloud” is based on the following architecture.
V. CONCLUSION

In this propose oruta, a privacy-preserving public auditing mechanism for shared data and de-duplication of 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 the best of our knowledge, designing an efficient public auditing mechanism with the capabilities of preserving identity privacy and supporting traceability is still open. Another problem for our future work is how to prove data freshness (prove the cloud possesses the latest version of shared data) while still preserving identity privacy.

REFERENCES


