A Presumable Data Retentive Framework for Multi Cloud

S.L Sailaja[1], S.Anjanayya [2]

1M.Tech (CSE), SJCET- Kurnool (Dt), Affiliated to JNTUA University, Andhra Pradesh, INDIA.
2 Associate Professor, Department of CSE, SJCET- Kurnool (Dt), Affiliated to JNTUA University, Andhra Pradesh, INDIA

ABSTRACT

Cloud Storage or storage outsourcing is a rising trend where the cloud storage service provides a comparably low-cost, scalable, position-independent platform for storing client's data. This archival storage guarantees the authenticity of data on storage server, but does not guarantee for the tampering of data (modifications or deletions, to the data).Thus, the problem for proving the integrity of the data stored at the third party cloud service providers has increased its attention. This work defines a framework for the efficient construction of data possession at the cloud storage servers dynamically. This scheme is considered for a hybrid cloud in which multiple cloud storages servers are considered, based on a cryptographic technique Interactive Proof System-multi-proved Zero knowledge proof system. This system is based on homomorphic verifiable response from multiple cloud servers, and the stored data is indexed using collision resistant hash index hierarchy. This scheme satisfies the security properties of completeness, knowledge soundness, and zero-knowledge system. This work also presents an optimal method for minimizing the computation, communication costs of storage service providers.

KEYWORDS: *Data possession, data outsourcing, storage outsourcing, probabilistic possession.*

I. INTRODUCTION

As high-speed networks and ubiquitous Internet access become available in recent years, cloud storage has become a fastest profit growth point for many cloud storage service providers. As cloud computing environment is completely built on open architectures and open interfaces, it provides high interoperability by bringing multiple internal/external cloud services together. When coming to a distributed cloud environment, say multi-cloud or hybrid cloud, it has to allow many clients to easily access their resources remotely through web service interfaces. As the concept of cloud computing has become familiar, the need for opting of third party storage servers also has its significant growth instead of setting up its own infrastructure for any organization. Users just use services without being concerned about how computation is done and storage is managed. Storage Outsourcing can also be referred to as data outsourcing or outsourcing of data.

Data outsourcing means the owner of the data (client) moves his/her data to a third-party storage service provider(server) which is for a low fee, also faithful to the data owner. The significant features of data outsourcing include reduced cost for storage, maintenance, increased availability and transparency of the data. The main disadvantage of this kind of outsourcing is that the client /organization do not have any control over the data which has been uploaded. The data as well as control of the data will be in the hands of the third party which poses a question for maintaining the confidentiality of the data. Also the server does not guarantee whether the data has been tampered with or deleted. Also, given a large files for storage, the user has to be provided to validate the data without actually downloading the data. In this paper, we focus on designing a cloud storage system for ensuring the integrity of the data in storage out sourcing**.** In this work, a cloud storage system is considered as a large scale distributed storage system that consists of many independent storage servers.

II. EXISTING SYSTEM

To check for the integrity of outsourced data, researchers provides two basic approaches called Provable Data Possession (PDP) and Proofs of Retrievability (POR). In the first approach, they proposed the PDP model for ensuring possession of files on untrusted storages without downloading the actual data and provided an RSA-based scheme for a static case. It also includes a challenge for public verifiability, with this anyone, not just the owner can challenge the server for data possession.

This extends the application areas of PDP protocol by separating the data owners and the users. But this is insecure against replay attacks in dynamic scenarios because of dependencies in index of data blocks. Moreover, they are not fit for multi-cloud storage because there is no homomorphic property in the verification process. To overcome static file storage limits in PDP and to provide dynamic data operations in PDP the Scalable PDP model have been proposed. It is a lightweight PDP scheme based on Cryptographic hash function and symmetric key encryption. But due to the lack of randomness in the challenges the servers used to deceive the clients /owners by previous metadata responses. Another drawback in this model is block size and its length are fixed and hence the modification of blocks cannot be done anywhere. Based on this work two Dynamic PDP has been proposed. First is the basic scheme, called DPDP-I and Second is the, "block less" scheme, called DPDP-II. But these schemes are also not effective for a multi-cloud environment. Second approach is POR scheme; it describes the preprocessing steps that the clients should do before sending their file to a CSP. But not allow for updating the data efficiently. An improved version of this protocol called Compact POR has been proposed. This technique which uses homomorphic property to aggregate a proof into authenticator value but the solution is also static and could not prevent the leakage of data blocks in the verification process. The dynamic scheme with cost by integrating the Compact POR scheme and Merkle Hash Tree (MHT) into the DPDP has been proposed. Several POR schemes and models have been recently proposed by using RAID techniques. It introduced a distributed cryptographic system that allows a set of servers to solve the PDP problem. This is based on an integrity protected error correcting code (IP-ECC), which improves the security and efficiency of existing tools. Here, a file must be transformed into distinct segments with the same length, which are distributed across servers. It is more suitable for RAID rather than cloud storage.

2.1 Drawbacks of Existing System

1) The existing schemes apply to the case of static, archival storage. Also dependent on the number of file blocks.
2) The numbers of updates, insertions, deletions are limited and are fixed
3) Insertions cannot be performed dynamically.
4) The existing schemes are vulnerable to two major security attacks: *Data leakage attack* and *Tag Forgery attack.*
5) Mainly applicable for a single cloud storage environment, but not for a distributed cloud storage environment.

III. PROPOSED SYSTEM

In this work, the architecture for a multi-cloud storage service can be viewed as three different entities: Clients, Cloud Service Providers (CSP's), Trusted Third party Administrator.

[1] **Clients**: End users who have large amounts of the data to be stored at third party storage servers. These users have permissions to access as well as manipulate the data.

[2] **CSP's**: CSP's provide data storage services and possess enough computation and storage services.

[3] **Trusted Third Party**: The third party administrator who is trusted to store verification parameters, also offer public query services for those parameters.

In this work, the verification procedure for data possession can be described as follows: Initially, the client uses secret key to pre-process the file as a collection of n-blocks. For the processed n-blocks, verification tags will be generated which are stored at the CSP's along with the file. At the same time, he generates the public verification information for the TTP's. The client can delete his/her own local copy of the data. The TTP acts independently and is reliable through the following functions:

[1] To setup and maintain the cryptosystem for the DPDP scheme.
[2] To generate and store owner's public key
[3] To store the public verification parameters to execute the verification protocol of the DPDP scheme.

3.1 Provable Data Possession Schemes

In order to address the problem of data integrity and authenticity in a distributed cloud environment. The following aspects are taken into consideration: High performance, scalability, automatic failure recovery, verification transparency, high security. For an efficient construction of this scheme, a significant cryptographic technique, Interactive Proof System with multi-prover Zero Knowledge Proof system is used for security

analysis. This model satisfies the security properties of completeness, soundness, zero-knowledge. The verification framework for the distributed cloud environment is implemented using two techniques. They are

[1] Hash Index Hierarchy
[2] Homomorphic verifiable response.

This framework provides security by resisting data leakage attacks and Tag forgery attacks. The framework can be defined as follows: Considering the security key and the tag for n-blocks of a file, the data possession scheme is given as

S= (KeyGen,TagGen,Proof).

The above said is a collection of two algorithms

- *(KeyGen,TagGen)*
- Interactive proof system, say*Proof*, can be explained as follows:

$Key(1\kappa)$**:** a security parameter κ as input, and returns a secret key sk or a public-secret key pair (pk, sk); $TagGen(sk, F,\mathcal{P})$: This function takes as inputs a secret key sk, a file F, and a set of cloud storage providers $\mathcal{P}$ $=\{Pk\}$, and returns the triples (ζ,ψ, σ), where $\zeta$is the secret in tags, $\psi = (u,\mathcal{H})$ is a set of verification parameters u and an index hierarchy $\mathcal{H}$ for F, $\sigma =\{\sigma(k)\}Pk\in\mathcal{P}$denotes a set of all tags, $\sigma(k)$ is the tag of the fraction $F(k)$ of F in Pk; $Pr(\mathcal{P}, V)$**:** a protocol of proof of data possession between all CSPs ($\mathcal{P} = \{Pk\}$) and a verifier (V), that is, $\langle\Sigma Pk\in(F(k), \sigma(k)) \longleftrightarrow V\rangle(pk, \psi)$= {1 $F = \{(k)\}$ is intact} {0 $F = \{F(k)\}$ is changed} ,where each Pk takes as input a file $F(k)$, a set of tags $\sigma(k)$, a public key pk, a set of public parameters ψ as the common input between $P$and V . At the end of each run of the protocol, V return as bit {0|1} denoting false and true. Where, $\Sigma Pk\in\mathcal{P}$denotes cooperative computing among all the CSP's in $Pk\in\mathcal{P}$. The co-operability among all the CSP's storing the same set of the data is verified one by one, i.e., $\wedge Pk\in\mathcal{P}\langle Pk(F(k), \sigma(k)) \longleftrightarrow V\rangle(pk, \psi)$, where $\wedge$ denotes the logical AND operations among the boolean outputs of all protocols $\langle Pk, V\rangle$on all the CSP's for all$Pk\in\mathcal{P}$.

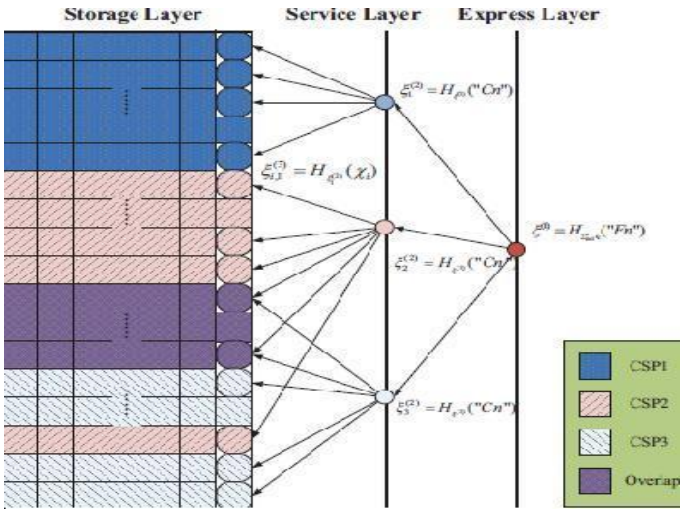
IV . SYSTEM MODEL

4.1 Hash Index Hierarchy

The architecture for this data possession scheme is a hierarchial model of file storage. This structure consists of three different levels which expresses the relationship among different storage resources:

1) **Express Layer**: an abstract representation of the stored resources.
2) **Service Layer**: manages and offers cloud storage.
3) **Storage Layer**: realizes data storage on many physical devices.

The storage layer defines a common fragment structure, a data structure that maintains a set of block-tag pairs. Given a file of blocks (m_1, m2,m3,…m_n), the respective block-tag pair will be generated as $(mi, \sigma i)$, where σi is a signature tag of block mi, generated by a set of secrets $\tau = (\tau1, \tau2, \cdots, \tau s)$. To check for the data integrity, the fragment structure implements the probabilistic verification as follows: Given a random probabilistic query $Q = \{(i , vi)\}i\in RI$, where I is a subset of the block indices and vi is a random Coefficient, the constant-size response is derived as $(\mu1, \mu2, \cdots, \mu s, \sigma')$, where μi comes from all $\{mk,i, vk\}k\in I$ and σ' is from all $\{\sigma k, vk\}k\in I$ To support virtualization of storage services, a simple Index-hash table $\chi = \{\chi i\}$ is introduced to record the changes of file blocks as well as to generate the hash value of each block in the verification process. The structure of index-hash table χ is similar to that of the structure of file block allocation table in file systems. This index-hash table consists of parameter values of serial number, block number, version number, and random integer. All the records in the index table are made unique to prevent forgery of data blocks and tags. Using this index records $\{\chi i\}$, this scheme also support dynamic data operations.



4.2 Homomorphic Verifiable Response

A homomorphism is a map function $:\mathbb{P} \rightarrow \mathbb{Q}$ between two groups such that $f(g1 \oplus g2) = f(g1) \otimes f(g2)$ for all $g1$, $g2 \in \mathbb{P}$, where $\oplus$ denotes the operation in $\mathbb{P}$ and $\otimes$ denotes the operation in $\mathbb{Q}$. This notation has been used to define Homomorphic Verifiable Tags (HVTs). Given two values σi and σj for two messagesmi and mj, anyone can combine them into a valueσ' corresponding to the sum of the messages mi +mj. A response is called homomorphic verifiable response in a Data Possession protocol, if given two responses θi and θj for two queries or challenges on Qi and Qj from two CSPs, then an efficient algorithm combines them into a response θ corresponding to the sum of the challenges $Qi\cup$. This Homomorphic verifiable response technique not only reduces the communication bandwidth, but also conceals the location of outsourced data, i.e., from which CSP the data is being accessed.

4.3 Working Model

This protocol can be described as follows: The manager who is a client first runs algorithm $KeyGen$ to obtain the public/private key pairs for CSPs as well as public users. The clients generate the tags of outsourced data by using $TagGen$. The protocol $Proof$ is performed by a 5-move interactive proof protocol between a verifier (client) and more than one CSP, in which CSPs need not to interact with each other during the verification process, an organizer (TTP) is used to organize and manage all CSPs.

1) The organizer (CSP) initiates the protocol and sends a commitment to the verifier (Client).
2) The verifier returns a challenge set of random index-coefficient pair'sQ to the organizer.
3) The TTP relays them into each Pi in $\mathcal{P}$ according to the exact position of each data block.
4) Each Pi returns its response of challenge to the organizer.
5) The organizer synthesizes a final response from received responses from all the CSP's and sends it to the verifier.

In the above said process, this scheme would guarantee that the verifier accesses files without knowing on which CSPs or in what geographical locations their files reside.

4.4 Advantages of Proposed System

1) Since these schemes are built based on collision-resistance signatures and random oracle model, these schemes have short query and response from public verifiability.
2) In distributed cloud storage, a file is divide into $n \times s$ sectors for which all the S sectors are represented with a single tag, there by reducing the storage for all the tags for S sectors.
3) As this scheme rely upon homomorphic properties, for aggregation of data and tags, into a constant size response, thereby minimizing the network communication bandwidth.
4) The hierarchial structure for storage provides virtualization for storage thereby concealing the location of the storage servers.

V. SECURITY ANALYSIS

This scheme satisfies the security properties of Completeness, Soundness, and Zero-knowledge

1) ***Completeness*** – To challenge the data integrity and ownership, the secret information is required with the verification protocol. The <Data, Tag> pair taken a (F,σ)ε TagGen(Sk,F), a random challenge is generated given as Q – (I,V_i) where I ε I can be checked for the success probability.

2) ***Soundness*** - The verifier is infeasible to accept any responses. This property can be used as a stricter notion for the unforgetability of the tags in order to avoid the ownership cheating. Hence, using this property, no intruder or server can tamper the data or forge the data tags. This property can be used efficiently to resist Tag Forgery attacks.

3) ***Zero –Knowledge –*** This property preserves the privacy of data blocks and respective tags to resist data leakage attacks, by introducing a random integer $\lambda_{j,k}$ for a response $\mu_{j,k}$, the random integer $\lambda_{j,k}$ is not known to the intruders.

VI. CONCLUSION

In this paper, a cloud storage system that consists of multiple cloud servers is considered. This scheme uses collision resistant hash index hierarchy model to store the data at multiple sites in an indexed manner. The response from the multiple cloud servers is a homomorphic verifiable response from multiple cloud servers. This work is completely based on multi-prover zero knowledge system, also satisfies all the security properties, thereby minimizing the communication overheads and computational costs for storage providers.

REFERENCES

[1]. G. Ateniese, R. C. Burns, R. Curtmola, J. Herring, L. Kissner, Z. N. J. Peterson, and D. X. Song, "Provable data possession at untrusted stores," in *ACM Conference on Computer and Communications Security*, P. Ning, S. D. C. di Vimercati, and P. F. Syverson, Eds. ACM, 2007, pp. 598–609.

[2]. G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in *Proceedings of the 4th international conference on Security and privacy in communication networks, SecureComm*, 2008, pp. 1–10.

[3]. C. C. Erway, A. K¨upc¸ ¨u, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in *ACM Conference on Computer and Communications Security*, E. Al-Shaer, S. Jha, and A. D. Keromytis, Eds. ACM, 2009, pp. 213–222.

[4]. H. Shacham and B. Waters, "Compact proofs of retrievability," in*ASIACRYPT*, ser. Lecture Notes in Computer Science, J. Pieprzyk, Ed., vol. 5350. Springer, 2008, pp. 90–107.

[5]. Y. Zhu, H. Hu, G.-J. Ahn, Y. Han, and S. Chen, "Collaborative integrity verification in hybrid clouds," in *IEEE Conference on the 7th International Conference on Collaborative Computing: Networking,Applications and Worksharing, CollaborateCom, Orlando, Florida, USA, October 15-18*, 2011, pp. 197–206.

[6]. K.D.Bowers , A.Juels and A.Opera, "Hail: a high-availability and integrity layer for cloud storage," in ACM Conference on *computer and communications security*, E.Al-shaer, S.Jha, and A.D.Keromytis,Eds. ACM, 2009, pp.187-198.

[7]. Q. Wang, C.Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in *ESORICS*, ser. Lecture Notes in Computer Science, M. Backes and P. Ning, Eds., vol. 5789. Springer, 2009, pp. 355–370.