ISSN:0975-9646



# Integrity Verification in Multi-Cloud Storage Using Cooperative Provable Data Possession

Megha Patil , Prof. G.R.Rao

*Computer Engineering Department, BVDUCOE Pune-43(India)*

***Abstract*- Storage outsourcing in cloud computing is a rising trend which prompts a number of interesting security issues. Provable data possession (PDP) is a method for ensuring the integrity of data in storage outsourcing. This research addresses the construction of efficient PDP which called as CooperativePDP (CPDP) mechanism for distributed cloud storage to support data migration and scalability of service, which considers the existence of multiple cloud service providers to collaboratively store and maintain the clients' data. CooperativePDP (CPDP) mechanism is based on homomorphic verifiable response, hash index hierarchy for dynamic scalability, cryptographic encryption for security. Moreover, it proves the security of scheme based on multi-prover zero-knowledge proof system, which can satisfy knowledge soundness, completeness, and zero-knowledge properties. This research introduces lower computation and communication overheads in comparison with non-cooperative approaches.**

***Keywords*— Multiple Cloud, Cloud Storage Security, Cooperative Provable Data Possession, Homomorphic Variable Response, Zero knowledge**

## I. INTRODUCTION

Data storage on cloud is one of the well known services offered by *cloud computing.* Because of this service subscribers do not have to store their own data on local servers, where instead their data will be stored on the cloud service provider's servers. Cloud storage makes it possible for users to remotely store their data and enjoy the on-demand high quality cloud applications without the any burden of local hardware and software management, which boasts an array of advantages like unlimited storage capability, anywhere accessibility etc. Since Cloud computing environment is constructed on open architectures and interfaces; it has the potential to incorporate multiple internal and/or external cloud services together to provide high interoperability. This type of distributed cloud environment is called as a *multi-Cloud.* The proverb of not putting all your eggs in one basket applies in Multi-cloud too. A multi-cloud approach is one where an enterprise uses two or more cloud services, therefore reducing the risk of widespread data loss or outage due to a component failure in a single cloud computing environment.

Frequently, by using virtual infrastructure management (VIM) [1], a multi-cloud allows clients to easily access his/her resources remotely through interfaces such as Web services provided by Amazon EC2. There exist various tools and technologies for multi-cloud, such as VMware vSphere, Platform VM Orchestrator and Ovirt. These tools help cloud providers to create a distributed cloud storage platform (DCSP) for managing clients' data. But, if such an important platform is vulnerable to security attacks, it would bring irrevocable losses to the clients. For example, the secret data in an enterprise may be illegally accessed by using remote interfaces, or organization relevant data and archives are lost or tampered with when they are stored into an uncertain storage pool outside the enterprise.

One of the biggest issues with cloud data storage is that of data integrity verification at untrusted servers. Also, there exist various motivations like maintaining reputation for cloud service providers (CSP) to behave unfaithfully towards the cloud users.For example, the cloud service provider(CSP), which experiences Byzantine failures infrequently, may decide to hide the data errors from the clients for the benefit of their own like for maintaining their reputation or for saving money and storage space the service provider might neglect to keep or deliberately delete rarely accessed data files which belong to an ordinary client Therefore, it is crucial for cloud service providers (CSPs) to provide security techniques for managing their storage services.

Provable data possession (PDP) [2] (or proofs of retrievability (POR) [3]) is such a probabilistic proof technique for a storage provider to prove the integrity and ownership of clients' data without downloading data. Checking proof without downloading makes it especially important for large-size files and folders (typically including many clients' files) to check whether these data have been tampered with or deleted without downloading the latest version of data. Consequently, it is able to replace traditional hash and signature functions in storage outsourcing. Different PDP schemes have been recently proposed, such as Scalable PDP [4] and Dynamic PDP [5]. However, these schemes mainly focus on PDP issues at untrusted servers in a *single* cloud storage provider and are not suitable for a multi-cloud environment.

## II. RELATED WORK

Security in cloud is indispensible. To check the availability and integrity of outsourced data in cloud storages, researchers have suggested two basic approaches called Provable Data Possession (PDP) [2] and Proofs of Retrievability (POR) [3]. Ateniese et al. [2] first proposed the PDP model for ensuring possession of files on untrusted

storages without retrieving it. Client maintains constant amount of metadata to verify proof.

This PDP approach has also provided an RSA-based scheme for a static case that achieves the $O$ (1) communication cost. They also suggested a publicly verifiable version, which allows client (data owner) as well as anyone other than owner, to challenge the server for data possession. This property has made immense impact on application areas of PDP protocol due to the separation of data owners and the users. However, these strategies are insecure against replay attacks in dynamic scenarios. Moreover, they do not fit for multi-cloud storage due to the loss of homomorphism property in the verification process.

Ateniese et al. developed a dynamic PDP solution called Scalable PDP [4]. This highly efficient and provably secure PDP technique is based entirely on symmetric key cryptography without requiring any bulk encryption. This PDP technique allows outsourcing of dynamic data, i.e. it supports operations, such as deletion, block modification and append, However, since it is based upon symmetric key cryptography, it is unsuitable for public (third-party) verification. Also, servers can deceive the owners by using previous metadata or responses due to the lack of randomness in the challenges. The numbers of challenges and updates are limited and fixed in advance and users cannot perform block insertions anywhere.

Based on previous work, Erway et al. [5] proposed two Dynamic PDP schemes with a hash function tree to realize (log $n$) communication and computational costs for a $n$-block file. The basic scheme, called DPDP-I, keeps the drawback of Scalable PDP, and in the 'blockless' scheme, called DPDPII, the data blocks can be leaked by the response of a challenge, .However, these schemes are also not effective for a multi-cloud environment because the verification path of the challenge block cannot be stored completely in a cloud [8].

Juels and Kaliski [3] presented a POR scheme, which depends largely on preprocessing steps that the client conducts before sending a file to a CSP. Unfortunately, these actions prevent any efficient extension for updating data. Shacham and Waters [6] introduced an improved version of this protocol called Compact POR. This protocol uses homomorphic property to aggregate a proof into (1) authenticator value and ($t$) computation cost for $t$ challenge blocks, but their solution could not prevent the leakage of data blocks in the verification process because of its static nature.

Wang et al. [7] presented a dynamic scheme with (log $n$) cost by integrating the Compact POR scheme and Merkle Hash Tree (MHT) into the DPDP. Several POR schemes and models have been recently proposed including [9], [10]. In [9] Bowers *et al.* introduced a distributed cryptographic system that allows a set of servers to solve the PDP problem. This structure is based on an integrity-protected error correcting code (IP-ECC), which upgrades the security and efficiency of existing tools. However, a file must be transformed into distinct segments with the same length, which are distributed across servers. Therefore, this system is more suitable for RAID rather than cloud storage.

## III. VERIFICATION FRAMEWORK OVERVIEW

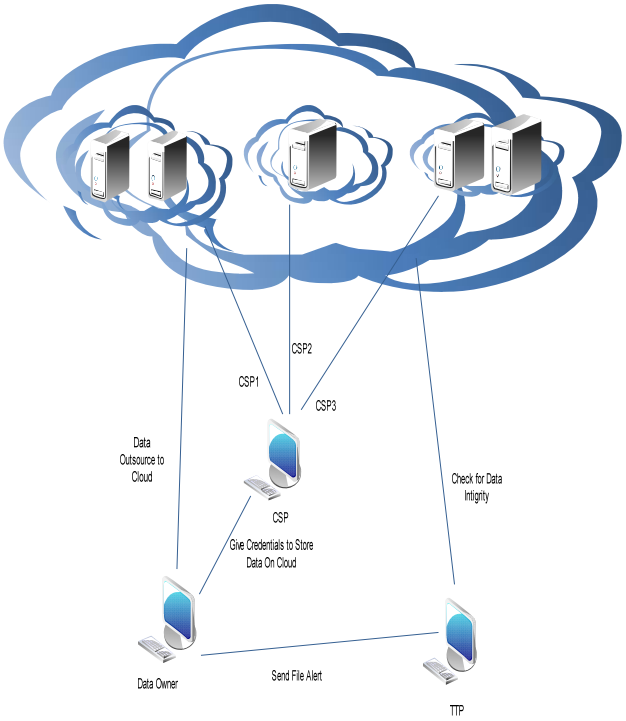


*Fig. 1 System Architecture*

This system architecture involves three different entities: Clients who have a large amount of data to be stored in multi-cloud and have the permissions to access and manipulate stored data. Cloud Service Providers (CSPs) who work together to provide data storage services have enough storages and computation resources. Trusted Third Party (TTP) is trusted to store verification parameters for integrity checking and offer public query services for these parameters.

This architecture Fig.1 has considered the existence of multiple CSPs to cooperatively store and maintain the data outsourced by client. A cooperative PDP is used to verify the integrity and availability of their stored data in all CSPs. As Data Owner Cannot fully trust to the CSP so here we will use trusted third party for security of outsourced data. This system will also make use of back up servers.

The verification method is described as follows: Firstly, a client (data owner) uses the secret key to pre-process a file which consists of a group of $n$ blocks, produces a set of public verification information that is stored in TTP, transfers the file and some verification tags to CSPs, and may delete its local copy; Then, by using a verification protocol, the clients can issue a challenge for one CSP to check the integrity and availability of outsourced data with respect to public information stored in TTP.

## IV. COOPERTIVE PROVABLE DATA POSSESSION SHCEME

This work addresses the construction of an efficient PDP scheme for distributed cloud storage to support data migration and scalability of service, in which we consider the existence of multiple cloud service providers to cooperatively store and maintain the clients' data. It presents a *cooperative* PDP (CPDP) scheme based on homomorphic verifiable response and hash index hierarchy. Multi-prover zero-knowledge proof system is used to prove the security of this scheme, which can satisfy knowledge soundness, completeness and zero-knowledge properties.

### *A. Hash index hierarchy:*

To support distributed cloud storage, architecture used in cooperative PDP scheme as shown in fig. 2. Our structure has a hierarchy structure which resembles a natural representation of file storage. This structure consists of three layers to represent relationships among all blocks for stored resources. This hierarchy structure and layers are described as follows:

*1) Express Layer*: This layer offers an abstract representation of the stored resources;

*2) Service Layer*: This layer offers and manages cloud storage services; and

*3) Storage Layer*: This layer represents data storage on many physical devices.

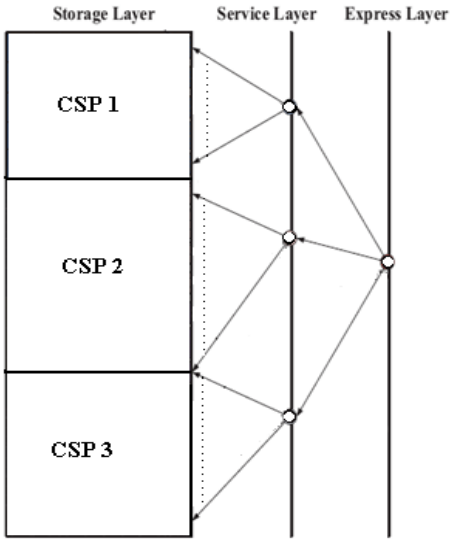


*Fig.2 Hash index hierarchy.*

This hierarchy used to organize data blocks from multiple CSP services into a large size file by shading their differences among these cloud storage systems. In Figure the resource in Express Layer are split and stored into three CSPs that are indicated by different colors are shown in Service Layer. After that each CSP fragments and stores the assigned data into the storage servers in Storage Layer. It also makes use of colors to distinguish different CSPs. Moreover, it follows the logical order of the data blocks to organize the Storage Layer.

### *B. Homomorphic Verifiable Response:*

Homomorphic Verifiable Responses (HVR), which is used to integrate multiple responses from the different CSPs in CPDP scheme. If given two responses $\theta i$ and $\theta j$ for two challenges $Qi$ and $Qj$ from two CSPs, there exist an efficient algorithm to combine them into a response $\theta$ corresponding to the sum of the challenges $Qi \cup Qj$ then a response is called homomorphic verifiable response in a PDP protocol. Homomorphic verifiable response is the key technique of CPDP because it not only reduces the communication bandwidth, but also hides the location of outsourced data in the distributed cloud storage environment.

### *C. Security Analysis*:

Multi-prover zero-knowledge proof system is directly used for security, which satisfies following properties:

*1) Collision resistant for index-hash hierarchy:* The index-hash hierarchy in CPDP scheme is collision resistant, even if the client generates files with the same file name and cloud name collision doesn't occur there.

*2) Completeness property of verification:* In this scheme, the completeness property implies public verifiability property. Due to this property allows client as well as anyone other than client (data owner) can challenge the cloud server for data integrity and data ownership without the need for any secret information.

*3) Zero-knowledge property of verification:* This paper makes use of the zero-knowledge property to preserve the privacy of data blocks and signature tags. Initially, randomness is adopted into the CSPs' responses in order to resist the data leakage attacks.

*4) Knowledge soundness of verification:*The soundness means that it is infeasible to fool the verifier to accept false statements. Often, the soundness can also be considered as a stricter notion of unforge ability for file tags to avoid cheating the ownership. This denotes that the CSPs, even if collusion is tried, cannot be tampered with the data or forge the data tags if the soundness property holds. Thus CPDP scheme can resist the tag forgery attacks to avoid cheating the CSPs' ownership.

## V. CONCLUSION AND FUTURE WORK

In this research, we presented the construction of an efficient PDP scheme for distributed cloud storage. Based on hash index hierarchy and homomorphic verifiable response, we have proposed a cooperative PDP scheme to support dynamic scalability on multiple storage servers. This paper also showed that our scheme provided all security properties required by zeroknowledge interactive proof system.

We would extend our work to explore more effective CPDP constructions as future work. As we are using multi-cloud, so there are multiple cloud service provider's for multiple clouds. As we want to store block in each cloud so the request has to go from each Cloud Service Provider, so to reduce the complexity we can use the Centralized Cloud Service Provider. Therefore, every request is managed by centralized Cloud Service Provider. During uploading and

downloading User has to answer the Security Question. Security Questions and Answers are provided by user during the registration phase. So during Uploading/Downloading operation If user is normal then he can answer that security questions if he/she is intruder then he/she cannot answer that questions. Thus, using this we can provide more Security. Also, we can provide the Security to uploaded data and the digest by using the encryption algorithm.

## REFERENCES


[1] B. Sotomayor, R. S. Montero, I. M. Llorente, and I. T. Foster,"Virtual infrastructure management in private and hybrid Clouds," *IEEE Internet Computing*, vol. 13, no. 5, pp. 14–22, 2009.

[2] 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.

[3] A. Juels and B. S. K. Jr., "Pors: proofs of retrievability for Large files," in *ACMConference on Computer and Communications Security*, P. Ning, S. D. C. di Vimercati, and P. F. Syverson, Eds. ACM, 2007, pp. 584–597.

[4] 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 netowrks, SecureComm*, 2008, pp. 1–10.

[5] 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.

[6] 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.

[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.

[8] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Dynamic audit services for integrity verification of outsourced Storages in clouds," in *SAC*, W. C. Chu, W. E. Wong, M. J. Palakal, and C.-C. Hung, Eds. ACM, 2011, pp. 1550–1557.

[9] K. D. Bowers, A. Juels, and A. Oprea, "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.

[10] Y. Dodis, S. P. Vadhan, and D. Wichs, "Proofs of retrievability via hardness amplification," in *TCC*, ser. Lecture Notes in Computer Science, O. Reingold, Ed., vol. 5444. Springer, 2009, pp. 109–127.

[11] L. Fortnow, J. Rompel, and M. Sipser, "On the power of multiprover Interactive protocols," in *Theoretical Computer Science*, 1988, pp. 156–161.

[12] 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.

[13] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A berkeley view of cloud computing," EECS Department, University of California, Berkeley, Tech. Rep., Feb 2009.

[14] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology (CRYPTO'2001)*, vol. 2139 of LNCS, 2001, pp. 213–229.