RESEARCH ARTICLE OPEN ACCESS

Cooperative Provable Data Possession for Integrity Verification in Multi-Cloud Storage

Mrs.K.E.Eswari MCA.[1], M.Phil.,M.E., V.Thamayanthi[2]
Associate Professor[1], Research Scholar MCA[2],
Department of Computer Applications,
Nandha Engineering College,
Erode-52.

----------**********----------

Abstract

Provable data possessions is a technique for certifying the reliability of data in storage subcontracting. This project we address the structure of an competent PDP system for circulated cloud storage to support the scalability of service and data migration, in which we consider the existence of multiple cloud service providers to supportively store and maintain the clients' data. We exsistant a Cooperative PDP system based on holomorphic provable response and hash index hierarchy. We prove the refuge of our system based on multi-prover zero-knowledge proof system, which can satisfy completeness, knowledge soundness, and zero-knowledge properties. In addition, we expressive performance optimization mechanisms for our system, and in particular exsistant an efficient system for choosing optimal parameter values to minimize the computation costs of clients and storage service providers. Our experiments show that our solution introduces lower computation and communication costs in comparison with non-cooperative approaches.

Keywords— **Storage Refuge, Provable Data Possession, Interactive Protocol, Zero-knowledge, Multiple Cloud, Co-operative.**

----------**********----------

1. INTRODUCTION

In recent years, cloud storage service has become a faster profit growth point by providing a comparably low-cost, scalable, position-independent platform for clients' data. Since cloud computing environment is constructed based on open architectures and interfaces, it has the capability to integrate multiple internal and/or external cloud services together to provide high interoperability. We call such a circulated cloud environment as a multi-Cloud (or hybrid cloud). Often, by using virtual infrastructure management (VIM), 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 Platform VM Orchestrator, VMware vSphere, and Overt. These tools help cloud providers construct a circulated cloud storage platform (DCSP) for managing clients' data. However, if such an important platform is vulnerable to refuge attacks, it would bring irretrievable losses to the clients. For example, the confidential data in an enterprise may be illegally accessed through a remote interface provided by a multi-cloud, or relevant data and archives may be lost or tampered with when they are stored into an uncertain storage pool outside the enterprise.
Therefore, it is indispensable for cloud service providers (CSPs) to provide refuge techniques for managing their storage services.

Provable data possession (PDP) or (proofs of retrievability (POR) is such a probabilistic proof technique for a storage provider to prove the integrity and ownership of clients' data without

downloading data. Thus, it is able to replace traditional hash and signature functions in storage outsourcing. Various PDP systems have been recently proposed, such as Scalable PDP and Dynamic PDP. However, these systems mainly focus on PDP issues at untrusted servers in a *single* cloud storage provider and are not suitable for a multi-cloud environment.

Motivation: To provide a low-cost, scalable, locationindependent platform for managing clients' data, current cloud storage systems adopt several new circulated file systems, for example, Apache Hadoop Distribution File System (HDFS), Google File System (GFS), Amazon S3 File System, CloudStore etc. These file systems share some similar features: a single metadata server provides centralized management by a global namespace; files are split into blocks or chunks and stored on block servers; and the systems are comprised of interconnected clusters of block servers. Those features enable cloud service providers to store and process large amounts of data.

In summary, a verification system for data integrity in circulated storage environments should have the following features:

Usability aspect: A customer should utilize the integrity check in the way of collaboration services. The system should conceal the details of the storage to reduce the burden on clients;

Refuge aspect: The system should provide adequate refuge features to resist some existing attacks, such as data leakage attack and tag forgery attack;

Performance aspect: The system should have the lower communication and computation overheads than non-cooperative solution.

Related Works:To check the availability and integrity of outsourced data in cloud storages, researchers have proposed 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 and provided an RSA-based system for a static case that achieves the (1) communication cost. They also proposed a publicly provable version, which allows anyone, not just the owner, to challenge the server for data possession.

This property greatly extended application areas of PDP protocol due to the separation of data owners and the users. However, these systems are insecure against replay attacks in dynamic scenarios because of the dependencies on the index of blocks. Moreover, they do not fit for multi-cloud storage due to the loss of homomorphism property in the verification process.

Our Contributions: In this project, we address the problem of provable data possession in circulated cloud environments from the following aspects: *high refuge*, *transparent verification*, and *high performance*. To achieve these goals, we first propose a verification framework for multi-cloud storage along with two fundamental techniques: hash index hierarchy (HIH) and homomorphic provable response (HVR).

2. STRUCTURE AND TECHNIQUES

In this section, we exsistant our verification framework for multi-cloud storage and a formal definition of CPDP. We introduce two fundamental techniques for constructing our CPDP system: hash index hierarchy (HIH) on which the responses of the clients' challenges computed from multiple CSPs can be combined into a single response as the final result; and homomorphic provable response (HVR) which supports circulated cloud storage in a multi-cloud storage and implements an efficient construction of collisionresistant hash function, which can be viewed as a random oracle model in the verification protocol.

A. Verification Framework for Multi-Cloud

Although exsisting PDP systems offer a publicly accessible remote interface for checking and managing the great amount of data, the majority of existing PDP systems are incapable to satisfy the inherent requirements from multiple clouds in terms of communication and computation costs. To address this problem, we consider a multi-cloud storage service as illustrated in Figure 1. In this architecture, a data storage service involves three

different entities: Clients who have a large amount of data to be stored in multiple clouds and have the permissions to access and manipulate stored data; Cloud Service Providers (CSPs) who work together to provide data storage services and have enough storages and computation resources; and Trusted Third Party (TTP) who is trusted to store verification parameters and offer community query services for these parameters.

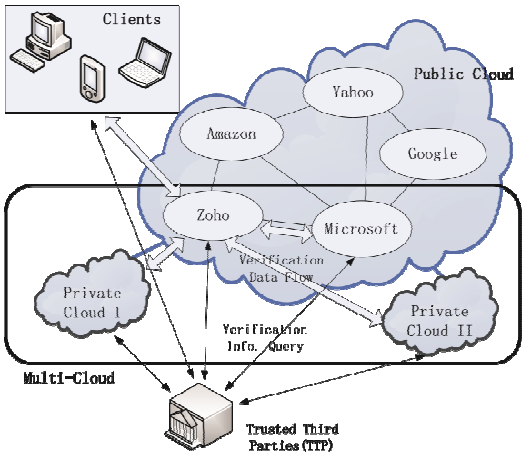

Fig. I. Verification architecture for data integrity.

In this architecture, we consider the existence of multiple CSPs to supportively store and maintain the clients' data. Moreover, a cooperative PDP is used to verify the integrity and availability of their stored data in all CSPs. The verification procedure is described as follows: Firstly, a client (data owner) uses the secret key to pre-process a file which consists of a collection of $n$blocks, generates a set of public verification information that is stored in TTP, transmits 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.

B. Definition of Cooperative PDP

In order to prove the integrity of data stored in a multi-cloud environment, we define a framework for CPDP based on interactive proof system (IPS) and multi-prover zero-knowledge proof system (MPZKPS), as follows:

Definition 1 (Cooperative-PDP): A cooperative provable data possessionis a collection of two algorithmsand an interactive proof system as follows:

1**:**takes a refuge parameter κas input, and returns a secret key or a public-secret key pair ;

2:takes as inputs a secret key sk, a file F, and a set of cloud storage providers and returns the triples, whereis the secret in tags,is a set of verification parameters $u$and an index hierarchy denotes a set of all tags, is the tag of the fraction of in a protocol of proof of data possession between CSPs and verifier (V).A trivial way to realize the CPDP is to check the data stored in each cloud one by one.

C. Hash Index Hierarchy for CPDP

To support circulated cloud storage, we illustrate a resistantative architecture used in our cooperative PDP system. Our architecture has a hierarchy structure which resembles a natural resistantation of file storage. They are described as follows:

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

In storage layer, we define a common fragment structure that provides probabilistic verification of data integrity for outsourced storage.Each CSP fragments and stores the assigned data into the storage servers in Storage Layer.This architecture also provides special functions for data storage and management.

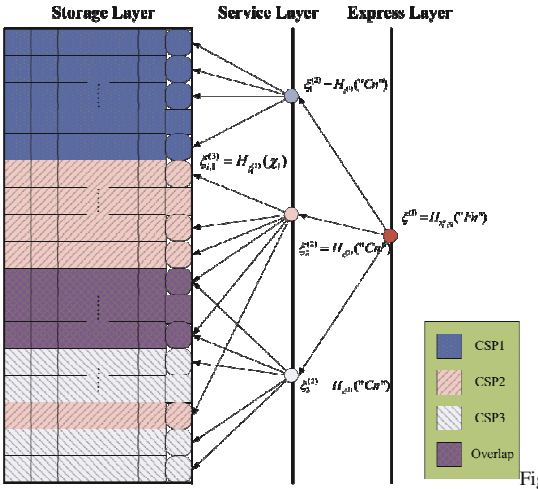

Fig.2.Index -hash hierarchy of CPDP model.

Given a collision-resistant hash function (·), we make use of this architecture to construct a Hash Index Hierarchy $\mathcal{H}$(viewed as a random oracle), which is used to replace the common hash function in prior PDP systems.

As a virtualization approach, we introduce a simple index-hash table χ= {χ_i} 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 $\chi$is similar to the structure of file block allocation table in file systems.

The index-hash table consists of serial number, block number, version number, random integer, and so on. Different from the common index table, we assure that all records in our index table differ from one another to prevent forgery of data blocks and tags.

a. *Homomorphic Provable Response for CPDP*

Homomorphic provable response is the key technique of CPDP because it not only reduces the communication bandwidth, but also conceals the location of outsourced data in the circulated cloud storage environment.

When provable data possession is considered as a challenge-response protocol, we extend this notation to the concept of Homomorphic Provable Responses (HVR), which is used to integrate multiple responses from the different CSPs in CPDP system as follows:

Definition 2 (Homomorphic Provable Response): A response is called homomorphic provable response in a PDP protocol, if given two responses $\theta_i$and $\theta_j$for two challenges $Q_i$and $Q_j$from two CSPs, there exists an efficient algorithm to combine them into a response θcorresponding to the sum of the challenges. Homomorphic provable response is the key technique of CPDP because it not only reduces the communication bandwidth, but also conceals the location of outsourced data in the circulated cloud storage environment.

D. COOPERATIVE PDP SYSTEM

In this section, we propose a CPDP system for multicloud system based on the above-mentioned structure and techniques. This system is constructed on collision-resistant hash, bilinear map group, aggregation algorithm, and homomorphic responses.

a. *Notations and Preliminaries*

Let $\mathbb{H}$= {H_k} be a family of hash functions: {0, 1} $^n\longrightarrow$ {0, 1} *index by $k\in\mathcal{K}$. So that, we have the following definition.

Definition 3 (Collision-Resistant Hash): A hash family $\mathbb{H}$is (t,)-collision-resistant if no t-time adversary has advantage at least ϵin breaking collisionresistance of $\mathbb{H}$..

Definition 4 (Bilinear Map Group System): A bilinear map group system is a tuple $\mathbb{S}$= $\langle p,,,\rangle$composed of the objects as described above.

b. *Our CPDP System*

In our system the manager first runs algorithm $KeyGen$to obtain the public/private key pairs for CSPs and users. Then, the clients generate the tags of outsourced data by using $TagGen$. Anytime, the protocol $Proof$is performed by a 5-move interactive.

This protocol can be described as follows: 1) the organizer initiates the protocol and sends a commitment to the verifier; 2) the verifier returns a challenge set of random index-coefficient pairs $Q$to the organizer; 3) the organizer relays them into

each P_i in $\mathcal{P}$ according to the exact position of each data block; 4) each P_i returns its response of challenge to the organizer; and 5) the organizer synthesizes a final response from received responses and sends it to the verifier.

In contrast to a single CSP environment, our system differs from the common PDP system in two aspects:

1) Tag aggregation algorithm: In stage of commitment, the organizer generates a random $\gamma \in_R \mathbb{Z}_p$ and returns its commitment H_1^{γ} to the verifier. This assures that the verifier and CSPs do not obtain the value of γ. Therefore, our approach guarantees only the organizer can compute the final σ' by using γ and σ_k' received from CSPs.

2) Homomorphic responses: Because of the homomorphic property, the responses computed from CSPs in a multi-cloud can be combined into a single final response as follows: given a set of $\theta_k = (\pi_k, \sigma_k', \mu_k, \eta_k)$

4 REFUGE ANALYSIS

We give a brief refuge analysis of our CPDP construction. This construction is directly derived from multi-prover zero-knowledge proof system (MPZKPS), which satisfies following properties for a given assertion L:

1) *Completeness*: whenever $x \in L$, there exists a strategy for the provers that convinces the verifier that this is the case;

2) *Soundness*: whenever $x \notin L$, whatever strategy the provers employ, they will not convince the verifier that $x \in L$;

3) *Zero-knowledge*: no cheating verifier can learn anything other than the veracity of the statement.

A. Collision resistant for index-hash hierarchy

In our CPDP system, the collision resistant of indexhash hierarchy is the basis and prerequisite for the refuge of whole system, which is described as being secure in the *random oracle model*. A successful hash collision can still be used to produce a forged tag when the same hash value is reused multiple times, e.g., a legitimate client modifies the data or repeats to insert and delete data blocks of outsourced data.

Theorem 1 (Collision Resistant): The index-hash hierarchy in CPDP system is collision resistant, even if the client generates $\sqrt{2p \cdot \ln \frac{1}{1-\varepsilon}}$ documents with the same file name and cloud name, and the client repeats $\sqrt{2^{L+1} \cdot \ln \frac{1}{1-\varepsilon}}$ times to modify, insert and delete data blocks, where the collision probability is at least ε, $\tau_i \in \mathbb{Z}_p$, and $|R_i| = L$ for $R_i \in \chi_i$.

B. Completeness property of verification

In our system, the completeness property implies public verifiability property, which allows anyone, not just the client (data owner), to challenge the cloud server for *data integrity* and *data ownership* without the need for any secret information. First, for every available data-tag pair $(F, \Phi) \in Ta(sk, F)$ and a random challenge $Q = \{(i, v_i)\}_{i \in I}$, the verification protocol should be completed with success probability according to the Equation, that is,

$$\Pr\left[\left\langle \sum_{P_k \subset \mathcal{P}} P_k(F^{(k)}, \sigma^{(k)}) \leftrightarrow O \leftrightarrow V \right\rangle (pk, \psi) = 1\right] = 1 .$$

In this process, anyone can obtain the owner's public key $pk = (g, h, H_1 = h^{\alpha}, H_2 = h^{\beta})$ and the corresponding file parameter $\psi = (u, \xi^{(1)}, \chi)$ from TTP to execute the verification protocol, hence this is a public provable protocol. Moreover, for different owners, the secrets α and β hidden in their public key pk are also different, determining that a success verification can only be implemented by the real owner's public key.

C. Zero-knowledge property of verification

CPDP construction is in essence a Multi-Prover Zero-knowledge Proof (MP-ZKP) system [11], which can be considered as an extension of the notion of an interactive proof system (IPS). Roughly speaking, in the scenario of MP-ZKP, a polynomial-time bounded verifier interacts with several provers whose computational powers are

unlimited. According to a *Simulator* model, in which every cheating verifier has a simulator that can produce a transcript that "looks like" an interaction between a honest prover and a cheating verifier, we can prove our CPDP construction has Zero-knowledge property.

Theorem 2 (Zero-Knowledge Property): The verification protocol $Proof(\mathcal{P}, V)$ in CPDP system is a computational zero-knowledge system under a simulator model, that is, for every probabilistic polynomial-time interactive machine V^*, there exists a probabilistic polynomial-time algorithm $\mathcal{S}^*$such that the ensembles $View(\langle_{Pk}\mathcal{P}P_k(F^{(k)}, \sigma^{(k)}) \leftrightarrow O \leftrightarrow V^*\rangle(pk, \psi))$ and $\mathcal{S}^*(pk, \psi_{\sum})$ are computationally indistinguishable.

Theorem 3 (Knowledge Soundness Property): Our system has (t, ϵ') knowledge soundness in random oracle and rewindable knowledge extractor model assuming the computational Diffie-Hellman (CDH) assumption holds in the group $\mathbb{G}$for $\epsilon' \geq \epsilon$.

5 PERFORMANCE EVALUATION

In this section, to detect abnormality in a lowoverhead and timely manner, we analyze and optimize the performance of CPDP system based on the above system from two aspects: evaluation of probabilistic queries and optimization of length of blocks. To validate the effects of system, we introduce a prototype of CPDP-based audit system and exsistant the experimental results.

A. *Performance Analysis for CPDP System*

We use $[E]$ to denote the computation cost of an exponent operation in $\mathbb{G}$, namely, g^x, where $x$is a positive integer in $\mathbb{Z}_p$and $g \in \mathbb{G}$or $\mathbb{G}_T$. We neglect the computation cost of algebraic operations.The most complex operation is the computation of a bilinear map $(\cdot,\cdot)$ between two elliptic points.

TABLE 1
Comparison of computation overheads between our CPDP system and non-cooperative (trivial) system.

	CPDP System	Trivial System
KeyGen	$3[E]$	$2[E]$
TagGen	$(2n+s)[E]$	$(2n+s)[E]$
Proof($\mathcal{P}$)	$c[E] + (t+cs+1)[E]$	$c[E] + (t+cs-c)[E]$
Proof(V)	$3[E] + (t+s)[E]$	$3c[E] + (t+cs)[E]$

Then, we analyze the storage and communication costs of our system. We define the bilinear pairing takes the form $e : E(\mathbb{F}_{p^m}) \times E(\mathbb{F}_{p^{km}}) \rightarrow \mathbb{F}^*_{p^{km}}$, where $p$is a prime, $m$is a positive integer, and $k$is the embedding degree (or refuge multiplier). In this case, we utilize an asymmetric pairing : $\mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$to replace the symmetric pairing in the original systems.

Further, our system has better performance compared with non-cooperative approach due to the total of computation overheads decrease $3(c-1)$ times bilinear map operations, where $c$is the number of clouds in a multicloud. The reason is that, before the responses are sent to the verifier from $c$clouds, the organizer has aggregate these responses into a response by using aggregation algorithm, so the verifier only need to verify this response once to obtain the final result.

TABLE 2
Comparison of communication overheads between our CPDP and non-cooperative (trivial) system.

	CPDP System	Trivial System
Commitment	l_2	cl_2
Challenge1	$2tl_0$	$2tl_0$
Challenge2	$2tl_0/c$	
Response1	$0 + 2\ 1 + l_T$	$(sl_0 + l_1 + l_T)c$
Response2	$sl0 + l1 + l_T$	

Without loss of generality, let the refuge parameter κ be 80 bits, we need the elliptic curve domain parameters over $\mathbb{F}_p$ with $|p| = 160$ bits and $m = 1$ in our experiments. This means that the length of integer is $l_0 = 2\kappa$ in $\mathbb{Z}_p$. Similarly, we have $l_1 = 4\kappa$ in $\mathbb{G}_1$, $l_2 = 24\kappa$ in $\mathbb{G}_2$, and $l_T = 24\kappa$ in $\mathbb{G}_T$ for the embedding degree $k = 6$. The storage overhead of a file with $(f) = 1M$-bytes is $stor(f) = n \cdot s \cdot l_0 + n \cdot l_1 = 1.04M$-bytes for $n = 10^3$ and $s = 50$. The storage overhead of its index table χ is $n \cdot l_0 = 20K$-bytes.

B. Probabilistic Verification

We recall the probabilistic verification of common PDP system (which only involves one CSP), in which the verification process achieves the detection of CSP server misbehavior in a random sampling mode in order to reduce the workload on the server. Assume the CSP modifies e blocks out of the n-block file, that is, the probability of disrupted blocks is $\rho_b = \frac{e}{n}$. Let t be the number of queried blocks for a challenge in the verification protocol.

$$P(\rho_b, t) \geq 1 - (\frac{n-e}{n})^t = 1 - (1-\rho_b)^t,$$

Where, (ρ_b, t) denotes that the probability P is a function over ρ_b and t. Hence, the number of queried blocks is $t \approx \frac{\log(1-P)}{\log(1-\rho_b)} \approx \frac{P \cdot n}{e}$ for a sufficiently large n and $t \ll n$.[3] This means that the number of queried blocks t is directly proportional to the total number of file blocks n for the constant P and e. Therefore, for a uniform random verification in a PDP system with *fragment structure*, given a file with $sz = n \cdot s$ sectors and the probability of sector corruption ρ, the detection probability of verification protocol has $P \geq 1 - (1 - \rho)^{\cdot \omega}$, where ω denotes the sampling probability in the verification protocol.

Another advantage of probabilistic verification based on random sampling is that it is easy to identify the tampering or forging data blocks or tags. The identification function is obvious: when the verification fails, we can choose the partial set of challenge indexes as a new challenge set, and continue to execute theverificationprotocol. The above search process can be repeatedly executed until the bad block is found. The complexity of such a search process is $(\log n)$.

C. Parameter Optimization

In the fragment structure, the number of sectors per block s is an important parameter to affect the performance of storage services and audit services. Hence, we propose an optimization algorithm for the value of s in this section. Our results show that the optimal value can not only minimize the computation and communication overheads, but also reduce the size of extra storage, which is required to store the verification tags in CSPs.

For instance, we assume a multi-cloud storage involves three CSPs $\mathcal{P} = \{P_{1,2}, P_3\}$ and the probability of sector corruption is a constant value $\{\rho_1, \rho_2, \rho_3\} = \{0.01, 0.02, 0.001\}$. We set the detection probability P with the range from 0.8 to 1, e.g., $P = \{0.8, 0.85, 0.9, 0.95, 0.99, \text{and } 0.999\}$. For a file, the proportion of data blocks is 50%, 30%, and 20% in three CSPs, respectively, that is, $r_1 = 0.5$, $r_2 = 0.3$, and $r_3 = 0.2$.The computational cost of CSPs can be simplified to $t + 3s + 9$. Then, we can observe the computational cost under different s and P.When s is less than the optimal value, the computational cost decreases evidently with the increase of s, and then it raises when s is more than the optimal value.

We choose the optimal value of s on the basis of practical settings and system requisition. For NTFS format, we suggest that the value of s is 200 and the size of block is 4KBytes, which is the same as the default size of cluster when the file size is less than 16TB in NTFS. In this case, the value of s ensures that the extra storage doesn't exceed 1% in storage servers.To validate the effectiveness and efficiency of our proposed approach for audit services, we have implemented a prototype of an audit system.The elliptic curve utilized in the experiment is a MNT curve, with base field size of 160 bits and the embedding degree 6.

A. CPDP for Integrity Audit Services

Based on our CPDP system, we introduce an audit system architecture for outsourced data in multiple clouds by replacing the TTP with a third party auditor (TPA). In this architecture, this architecture can be constructed into a visualization

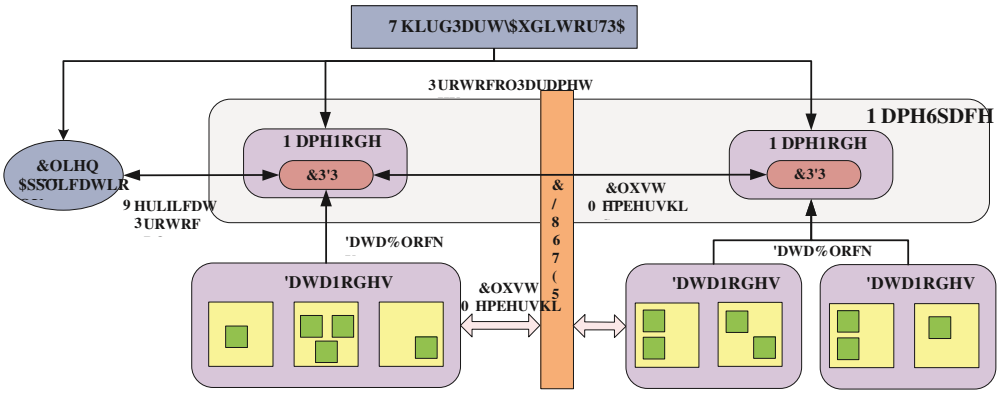

Fig. 4. Applying CPDP system in Hadoop circulated file system (HDFS).

infrastructure of cloud-based storage service [1]. We show an example of applying our CPDP system in Hadoop circulated file system (HDFS), which a circulated, scalable, and portable file system. HDFS' architecture is composed of NameNode and DataNode, where NameNode maps a file name to a set of indexes of blocks and DataNode indeed stores data blocks. To support our CPDP system, the index-hash hierarchy and the metadata of NameNode should be integrated together to provide an enquiry service for the hash value or index-hash record.

Firstly, we quantify the performance of our audit system under different parameters, such as file size sampling ratio w, sector number per block s, and so on. Our analysis shows that the value of ? should grow with the increase of order to reduce computation and communication costs. Thus, our experiments were carried out as follows: the stored files were chosen from 10KB to 10MB; the sector numbers were changed from 20 to 250 in terms of file sizes; and the sampling ratios were changed from 10% to 50%. These results dictate that the computation and communication costs (including I/O costs) grow with the increase of file size and sampling ratio.

6 CONCLUSIONS

In this paper, we exsistanted the construction of an efficient PDP system for circulated cloud storage.

Based on homomorphic provable response and hash index hierarchy, we have proposed a cooperative PDP system to support dynamic scalability on multiple storage servers.

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 system, especially for large documents, is

affected by the bilinear mapping operations due to its complexity. To solve this problem, RSAbased constructions may be a better choice, but this is still a challenging task because the existing RSAbased systems have too many restrictions on the performance and refuge [2]. Next, from a practical point of view, we still need to address some issues about integrating our CPDP system smoothly with existing systems, for example, how to match indexhash 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 a issue to provide the support of variable-length block verification.

ACKNOWLEDGMENTS

The work of Y. Zhu and M. Yu was supported by the National Natural Science Foundation of China (Project No.61170264 and No.10990011). This work of Gail-J.Ahn and Hongxin Hu was partially supported by the grants from US National Science Foundation (NSFIIS-0900970 and NSF-CNS-0831360) and Department of Energy (DE-SC0004308).

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 Refuge, P. Ning, S. D. C. di Vimercati, and P. F. Syverson, Eds. ACM, 2007, pp. 598–609.k*

[3] *A. Juels and B. S. K. Jr., "Pors: proofs of retrievability for large documents," in ACM Conference on Computer and Communications Refuge, 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 Refuge and privacy in communication netowrks, SecureComm, 2008, pp. 1–10.*

[5] *C. C. Erway, A. Ku¨pc¸u¨, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in ACM Conference on Computer and Communications Refuge, 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 refuge 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 Refuge, 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, Collaboration, Orlando, Florida, USA, October 15-18, 2011, pp. 197–206.*