International Journal of Modern Engineering Research (IJMER)
www.ijmer.com Vol. 3, Issue. 4, Jul. - Aug. 2013 pp-2317-2319 ISSN: 2249-6645

An Efficient PDP Scheme for Distributed Cloud Storage

Jimnisha Shaik[1], Syed Gulam Gouse[2]
[1]*M. Tech, Nimra College of Engineering & Technology, Vijayawada, A.P., India*
[2]*Professor, Dept.of CSE, Nimra College of Engineering & Technology, Vijayawada, A.P., India*

***ABSTRACT*:** *Cloud computing is the use of Internet for the tasks performed on the local computer, with the hardware and software demands maintained elsewhere. It represents a different way to architect and remotely manage various computing resources. Cloud is widely used everywhere owing to its convenience, be it in simple data analytic program or composite web or mobile applications. Cloud computing is being driven by many which includes Amazon, Google and Yahoo as well as traditional vendors including IBM, Microsoft and Intel. The data should be available in the cloud for it to be accessed by many users. There are four main types of cloud storage- Multi Cloud Storage or distributed cloud storage, Public Cloud Storage, Private Cloud Storage, Mobile Cloud Storage. Distributed cloud is a combination of public and private cloud storage where some critical data resides in the enterprise's private cloud while other data is stored and accessible from a public cloud storage provider. Provable data possession (PDP) is a technique for ensuring the integrity of data stored in storage outsourcing. In this paper, we propose a method called cooperative PDP (CPDP) scheme based on homomorphic verifiable response and hash index hierarchy.*

INDEX TERMS: *Cloud, Hash index, Integrity, PDP.*

I. INTRODUCTION

In recent years, the concept of parallel computing has emerged to solve the problems with a greater computational speed. It's operation is based upon the principle that larger problems can be reduced to a number of smaller ones, then which are solved parallely.Parallel computing can be implemented in several ways of computing like bit level, instruction level, task and data parallelism. Based on the level at which the hardware supports parallelism, it can be classified as- multi-core and multi-processor. Generally in a computer system a problem can be solved using a stream of instructions. Only one instruction is executed at a time, and then the others are executed. On the other hand in parallel computing, uses the multi-processing elements to solve a problem. This is accomplished by breaking the problem into various independent parts so that each processing element can execute its part of the algorithm simultaneously with others. There is another important concept which is responsible for the effective and efficient computation of our problem is, distributed system. Using high performance computers connected by using high speed communication links, it is possible to build a single system consisting of multiple computers and using it as a single consolidated system.

In a distributed system, the computers are not independent but are interconnected by a high-speed network. Here are a few requirements for a distributed system- Like reliability and security, consistency of replicated data, concurrent transactions, and fault tolerance. The major aim of constructing the distributed system is that its behavior should be transparent to the user. In a distributed memory architecture if we take into account each processor has its own local storage and all the processing is done locally. All systems are interconnected using a LAN.

In any kind of computer system which involves data storage and retrieval, availability is one of the major security issues to be concerned. Provable Data Possession (PDP) is such a technique which ensures data availability or proof of retrievability (POR).It's the proof which is provided for the storage provider, in order to prove the ownership and integrity of client's data without being downloading it. The proof-checking of the data without being downloading is very important, especially when it comes to very large sized data blocks. It is necessary because it is to be ensured that the data is not deleted or altered. PDP schemes are very useful when it comes to these kinds of issues. However, this scheme will be effective only for single cloud storage, but not for the distributed-cloud storage environment.

II. RELATED WORK

To check the availability and integrity of the outsourced data in cloud storages, researchers have proposed two basic approaches called Provable Data Possession (PDP) [1] and Proofs of Retrievability (POR) [2]. In [1], the authors first proposed the PDP model for ensuring possession of files on un trusted storages and provided an RSA-based scheme for a static case that achieves the O(1) communication cost. In order to support dynamic data operations, the authors developed a dynamic PDP solution called Scalable PDP [3]. They proposed a lightweight PDP scheme based on cryptographic hash function and the symmetric key encryption, but the servers can deceive the owners by using previous metadata or responses due to the lack of the randomness in the challenges.

In [4], the authors introduced two Dynamic PDP schemes with a hash function tree to realize O(log n) communication and computational costs for a n-block file. The basic scheme, called DPDP-I, retains the drawback of Scalable PDP method, and in the 'blockless' scheme, called DPDPII. In [2], the authors presented a POR scheme, which relies largely on preprocessing steps that the client conducts before sending a file to a CSP. Several POR schemes and models have been recently proposed including in [5][6]. In [5], the authors introduced a distributed cryptographic system that allows a set of servers to solve the PDP problem. This system is based on the integrity protected error correcting code (IP-ECC), which improves the security and efficiency of the existing tools, like POR.

III. PROPOSED METHOD

In this section, we introduce the principles of our cooperative provable data possession for distributed clouds, including the main technique, model, fragment structure, index hierarchy, and the architecture to support our scheme.

A. ***Homomorphic Verifiable Response:*** A homomorphism is the map f : P → Q between two groups such that f(g1+g2) = f(g1)×f(g2) for all g1, g2 ∈ P, where + denotes the operation in P and × denotes the operation in Q. This notation is used to define the Homomorphic Verifiable Tags (HVTs): Given two values σi and σj for two message mi and mj , anyone can combine them into a value σ′ corresponding to the sum of the message mi + mj.

B. ***Cooperative PDP:*** A cooperative provable data possession(CPDP) scheme S′ is a collection of two algorithms and an interactive proof system, S′ = (K, T, P).

KeyGen(1^k): It takes a security parameter k as the input, and returns a secret key sk or a public-secret key pair (pk, sk);
TagGen(sk, F,P): It takes as inputs a secret key "sk", a file
F, and a set of cloud storage providers P = {Pk}, and returns the triples (ζ ,ψ, σ), where ζ is the secret of tags, ψ = (u,H) is a set of verification parameters u and an index hierarchy
H for F, σ = {σ(k)}Pk∈ P denotes a set of all tags, σ(k)is the tags of the fraction F (k) of F in Pk.

Proof(P, V): It is a protocol of proof of the data possession between the CSPs (P = {Pk }) and a verifier (V), that is,(∑Pk∈ P Pk(F(k), σ(k)), V) (pk, ψ), where each Pk takes as input a file F(k) and a set of tags σ(k), and a public key pk and a set of public parameters ψ is the common input between P and V . At the end of the protocol run, " V" returns a bit {0|1} denoting false and true where, $\sum_{P_K \in p}$ denotes the collaborative computing in Pk ∈ P.

C. ***Fragment Structure of Cooperative PDP:*** We propose a fragment structure of CPDP scheme as shown in the Figure 1, which has following characters:

- A file is split into n × s sectors and each block (or s sectors) corresponds to a tag, so that the storage of signature tags can be reduced with the order of s.
- The verifier can check the integrity of the file by random sampling approach, which is a matter of the utmost importance for large or huge files.
- This structure relies on the homomorphic properties to aggregate the data and tags into a constant size response, which minimizes network communication overheads.

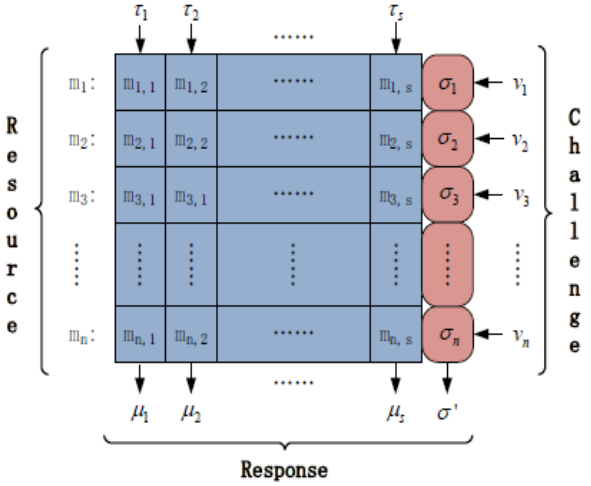

Figure 1: The fragment structure of Cooperative PDP model

The above structure, considered as a common representation for some existing schemes in [1][7], can be converted to MAC-based, ECC or RSA schemes. By using BLS signatures and the random oracle model, it is easy to design a practical CPDP scheme with the shortest homomorphic verifiable responses for public verifiability. This structure also creates favorable conditions for the architecture of the CSPs.

D. ***Hash Index Hierarchy:*** The Architecture for data storage in distributed clouds is shown in Figure 2. This architecture is based on a hierarchical structure with three layers to represent the relationship among all blocks for stored resources. Three layers can be described as follows:

- First Layer (Express Layer): It offers an abstract representation of the stored resources
- Second Layer (Service Layer): It promptly offers and manages cloud storage services
- Third Layer (Storage Layer): It directly realizes data storage on many physical devices.

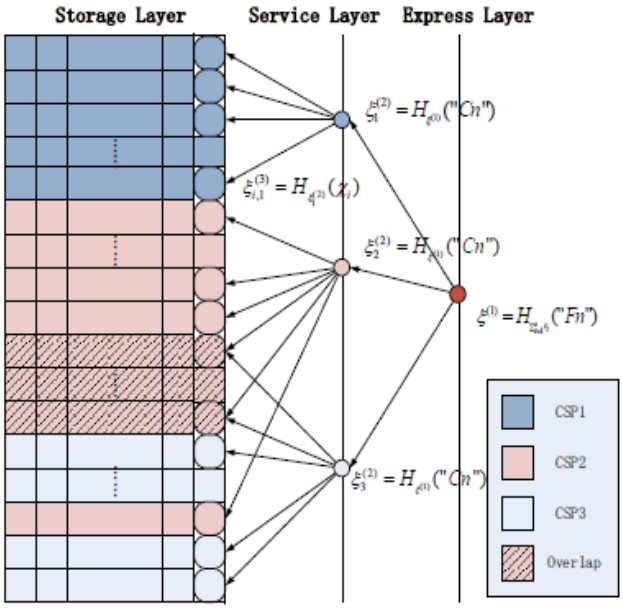

Figure 2: The architecture of Cooperative PDP model

This architecture naturally accommodates the hierarchical representation of the file systems. We make use of a simple hierarchy to organize the multiple CSP services, which involve private clouds or public clouds, by shading the differences between these clouds. As shown in Figure 2, the resources in the Express Layer are split and stored into three CSPs in the Service Layer. In turn, each CSP fragments and stores the assigned data into the storage servers at the Storage Layer. We distinguish different CSPs by different colors, and the denotation of the Storage Layer is the same as in Figure 1. Moreover, we follow the logical order of data blocks to organize Storage Layer. This architecture could provide some special functions for the data storage and management, e.g., there may exist an overlap among data blocks (as shown in dashed line) and discontinuous blocks (as shown on a non continuous color).

IV. CONCLUSION

With the techniques such as homomorphism verifiable response and hash index hierarchy, cooperative provable data possession (CPDP) concept has been achieved and hence integrity and availability is verified .The zero- knowledge proof system is used and hence increases the security so it can be used widely in public cloud services thereby increasing their performance. By this approach the computation time and as well as cost is reduced. Our system can be used as a new method for data integrity verification in out sourcing data storage on distributed cloud environment.

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]. 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.

[3]. 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.

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

[5]. 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.

[6]. 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.

[7]. H. Shacham and B. Waters. Compact proofs of retrievability. In ASIACRYPT, pages 90–107, 2008.