ISSN:0975-9646

Integrity Verification and Availability Using PDP in Multi-Cloud

Vrushali K Gaikwad

Student,Dr.D.Y.Patil School of Engg and Technology Lohegaon, Pune, India.

***Abstract*— Nowadays in cloud environment a large data is produced. Many organizations are now migrating to cloud environment. Thus the providers are now delivering multi-cloud environment. But clients are worrying that their data should be correctly stored and maintained by providers without tampering it. As the providers provide enough security still there are many security issues occuring in cloud. A technique for verifying integrity of clients data is called Provable Data Possession (PDP). The security system is based on multi-prover , zero knowledge proof system, which helps to satisfy completeness, knowledge soundness, and zero-knowledge properties In this paper, we propose a PDP scheme in multi clouds to support service scalability and data migration.**

***Keywords*— Provable Data Possesion(PDP), Proofs Of Retrievability(POR), Cloud Service Provider(CSP), Hash Index Hierarchy, Homomorphic Verifiable Response, Third Party Auditor(TPA).**

I. INTRODUCTION

In recent years, cloud storage service has provided a comparably scalable , low-cost, position-independent platform for clients data. Cloud computing is based on open architectures and interfaces, thus it has the capability to provide a convinent environment to incorporate multiple internal or external cloud services together. Such a distributed cloud environment is known as a Multi-Cloud.Cloud computing is different from other information technology in three ways by,

A] Outsourced resources – This includes both hardware and software. On-site file server can provide a source for file handling, data storage, and information backup.

B] Pay-as-you-go – Cloud computing will require a basic start up fee followed by a monthly usage charge. User need to pay charge based on cloud time consumption and additional software features.

C] On-demand – In cloud computing, user pay for what they use for their own purpose.

For managing clients data various tools and technologies exists for multi-cloud, such as Platform VM Orchestrator, VMware vSphere, and Ovirt. These tools help to construct a distributed cloud storage platform (DCSP) to cloud providers. But many times such an important platform is vulnerable to security attacks, it will bring losses to the clients.

For example, the confidential data in an enterprise may be illegally accessed through a multi-cloud provided remote interface, or the data may be lost or tampered through the enterprise. Thus, it is neccessary for cloud service providers (CSPs) to provide security techniques for their storage services. Provable data possession (PDP) is a secure technique for a storage provider to prove the client about their data integrity without downloading data. The proof-checking without downloading makes it important for large-size files and folders to checkout the data have been tampered with or deleted without downloading the latest version of data. To achieve these goals, a verification framework for multi-cloud storage along with two fundamental techniques: hash index hierarchy (HIH) and homomorphic verifiable response (HVR) can be proposed.

Thus using above mentioned structure we can introduced an effective construction of PDP scheme. Thus it can be proved that this construction has completeness, knowledge soundness, and zero-knowledge properties. These properties ensure that PDP scheme can implement the security against data leakage attack and tag forgery attack.

A.Existing system

For multi cloud there exist various tools , such as Platform VM Orchestrator, VMware, vSphere, and Ovirt. To manage clients data the cloud providers uses these tools to construct a distributed cloud storage platform. If such an platform is vulnerable to security attacks, it would bring irretrievable losses to the clients. Thus, it is important for cloud service providers to provide security techniques for managing their storage services.

B.Proposed System

Researchers have proposed two basic approaches to check the availability and integrity of outsourced data in cloud storages, known as Provable Data Possession and Proofs of Retrievability .Ateniese et al. first proposed the PDP model for ensuring possession of files on untrusted storages. They proposed a lightweight PDP scheme based on cryptographic hash function and symmetric key encryption, but the servers can deceive the owners by using previous metadata or responses due to the lack of randomness in the challenges. We will proposed the PDP model for ensuring possession of files on untrusted storages and provided an RSA-based scheme for a static case that achieves the communication cost. The numbers of updates and challenges are limited and fixed in advance and users cannot perform block insertions anywhere.

II. LITERATURE SURVEY

In this section we are presenting the different methods which are previously used for Provable Data Possesion techinique.We discuss some limitations and advantages of these systems.

Researchers have proposed two basic approaches to verify availability and integrity of outsourced data in cloud storages, called Provable Data Possession (PDP) [2] and Proofs of Retrievability (POR) [3].They have addressed the problem in distributed cloud environments of provable data possession as per following aspects: high security, transparent verification, and high performance. B.Sotomayor[1], they present OpenNebula, an open source virtual infrastructure manager that can be used to deploy virtualized services on both a local pool of resources and on external IaaS cloud. Ateniese et al. [2] proposed the PDP model for ensuring possession of untrusted storages. They also proposed a publicly verifiable version, which allows anyone, 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 schemes 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.

In order to support dynamic data operations, Ateniese et al. developed a dynamic PDP solution called Scalable PDP [4]. They proposed a lightweight PDP scheme based on cryptographic hash function and symmetric key encryption, but the servers can deceive the owners by using previous metadata or responses due to the lack of randomness in the challenges. The numbers of updates and challenges are limited and fixed in advance and users cannot perform block insertions anywhere. Based on this work, Erway et al. [5] introduced two Dynamic PDP schemes with a hash function tree to realize O(log1) communication and computational costs for a -block file. The basic scheme, called DPDP-I, retains the drawback of Scalable PDP, and in the 'blockless' scheme, called DPDP-II.

Furthermore, these schemes are also not effective for a multi-cloud environment because verification path of the challenge block cannot be stored completely in a cloud. Juels and Kaliski [3] presented a POR scheme, which relies largely on preprocessing steps that the client conducts before sending a file to a CSP. Unfortunately, these operations prevent any efficient ex-tension for updating data.

III. IMPLEMENTATION

Mathematical Model

This proposed work uses below function as follows.

1. *KeyGen()*
2. *TagBlock()*
3. *GenProof()*
4. CheckProof()

In this model, Symbol C is denote set component which used in this framework.

C={pk,sk,m,Tm,F,chal,Σ,V}

Where,

pk = public key, sk = secret key, m = file block,

Tm = metadata, F = collection of blocks ,

Chal = challenge, Σ = verification metadata corresponding to blocks in F, V= proof of possession

A PDP system can be constructed from a PDP scheme in two Phases, *Setup and Challenge*

A.Setup:-

Pk and sk runs by client i.e KeyGen

TagBlok(): Input={pk,F, Σ=(Tm1,…….,Tmn)}

Output={F,Σ}

B.Challenge:-

Client sends challenge chal to Server S.

Genproof():-Input={pk,F,chal,Σ)

Output={F,Σ}

CheckProof():-Input={pk,sk,chal,V}

Output={success,failure}

Following polynomial-time algorithms will be used :-

KeyGen(1^k) → (pk,sk)

PrepareUpdate(sk,pk,F,info,Mc) → {e(F),e(info),e(M)}

PerformUpdate(pk,Fi−1,Mi−1,e(F),e(info),e(M)) → {Fi,Mi,M′c,PM′c}

VerifyUpdate(sk,pk,F,info,Mc,M′c,PM′c) → {accept,reject}

Challenge(sk,pk,Mc) → {c}

Prove(pk,Fi,Mi,c) → {P}

Verify(sk,pk,Mc,c,P) → {accept,reject}

IV. SYSTEM ARCHITECTURE

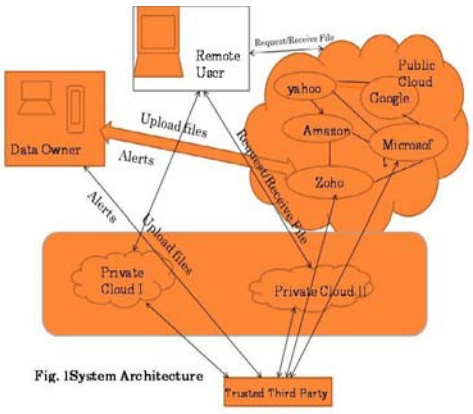

Fig. 1System Architecture

In this architecture, we consider the existence of multiple CSPs to cooperatively store and maintain the clients' data. Moreover, a 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. Then neither assume that CSP is trust to guarantee the security of the stored data, nor assume that data owner has the ability to collect the evidence of the CSP's fault after errors have been found. To achieve this goal, a TTP server is constructed as a core trust base on the cloud for the sake of security. We assume the TTP is reliable and independent through the following functions to setup and maintain the CPDP cryptosystem; to generate and store data owner's public key; and to store the public parameters used to execute the veriffcation protocol in the CPDP scheme. The TTP is not directly involved in the CPDP scheme in order to reduce the complexity of cryptosystem.

V. PDP

A PDP is a collection of two algorithms (Key Gen, Tag Gen) and interactive proof system Proof.

Key Gen: It takes a security parameter as an input and returns a secret key as output.

Tag Gen: It takes a secret key, file and set of cloud storage providers as input and returns triples.

Proof: It is a protocol of proof of data possession between the CSP's and veriffer.

Let H = Hk be a family of hash functions where k : {0 1}^{1} index by k ε K. This algorithm has a benefit in breaking the collision resistance of H.Collision-Resistance H: In this a hash family H(t, ε) collision resistant if no t-Time adversary has advantage atleast ε in breaking collision of H. First the KeyGen algorithm is run in this scheme to obtain the public or the private keyfor users. Then TagGen is generated by the clients for the outsourced data.

VI. HASH INDEX HIERARCHY FOR CPDP

Three layers are used to illustrate the relationships among the blocks for stored resources.They are as follows:

1.Express Layer: it shows representation of stored resources.

2.Service Layer : it offers and manages cloud storage and services and

3.Storage Layer : realizes data storage on physical devices

VII. HOMOMORPHIC VERIFIABLE RESPONSE FOR PDP

A homomorphism is a map f : P Q between two groups such that f (g1+ g2) = f(g1) x f(g2) for all g1,g2 ε P, where + denotes the operation in P and x denotes the operation in Q. Homomorphic verifiable 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 distributed cloud storage environment.

VIII. RESULT

In our PDP scheme, the client's communication overhead is not changed and the interaction among CSPs needs c−1 times constant-size communication overheads, where c is the number of CSPs in multi clouds. Thus, the amount of communication overheads is not increased. Further, we evaluated the performance of our PDP scheme in terms of computational overhead. For comparison, our experiments will executed as follows: a fixed-size file is used to generate the tags and prove data possession under the different number of sectors s. Then, the computational overheads of tag generation are created. The results shows that the overheads are reduced as the values of s are increased.

IX. CONCLUSION

Cloud is designed to provide a secure service to the external users. To fulfill their needs the resources should be highly and easily available. In this paper, it gives overview about cloud availability and various integrity verification techniques. Thus, the construction of an efficient PDP scheme for distributed cloud storage can be consturcted. Based on Homomorphic verifiable response and hash index hierarchy, a PDP scheme can be proposed to support dynamic scalability on multiple storage servers. The scheme provided shows that all security properties required by zero-knowledge interactive proof system.

REFERENCES

[1] 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.

[2] 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.1422, 2009.

[3] 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.598609.

[4] A. Juels and B. S. K. Jr., Pors: proofs of retrievability for large _les, in ACM Conference on Computer and Communications Security, P. Ning, S. D. C. di Vimercati, and P.F. Syverson, Eds. ACM, 2007, pp. 584597.