International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 03 | Mar-2016 www.irjet.net p-ISSN: 2395-0072

Competent Demonstrable Data Possession for Integrity Verification in Multi-Cloud Storage

Dinesh Kumar K, Karuppuchamy V

[1] *PG Student, Dept. of I.T, Muthayammal Engineering College, Rasipuram, Tamilnadu, India*
[2] *Assistant Professor, Dept. of C.S.E, Muthayammal Engineering College, Rasipuram, Tamilnadu, India*

Abstract - *Provable data possession (PDP) is a technique which is for ensuring the integrity of data in storage outsourcing. In this paper, we address the construction of an efficient PDP scheme for distributed cloud storage which supports the scalability of service and data migration, in which we consider the existence of multiple cloud service providers to cooperatively store and maintain the clients' data. We present a cooperative PDP (CPDP) scheme which was based on homomorphic verifiable response and hash index hierarchy. We prove the security of our scheme based on the multi-proven zero-knowledge proof system, which can satisfy things such as completeness, knowledge soundness, and zero-knowledge properties. In addition, we articulate the performance optimization mechanisms for our scheme, and in particular present an efficient method for selecting the optimal parameter values to minimize computation costs of clients and storage service providers. Our experiments show that our solution introduces lower computation and communication overheads in comparison with other non co-operative approaches.*

Key Words: **Cloud computing, data replication, outsourcing data storage, dynamic environment.**

1. INTRODUCTION

In recent years, the cloud storage service has become one of the faster profit growth point by providing a comparably low-cost, scalable, position-independent platform for clients' data. Since the cloud computing environment is constructed based on the open architectures and the interfaces, it has the capability to incorporate multiple internal and/or external cloud services together to provide high interoperability. We call this distributed cloud environment as a multi-Cloud. By using virtual infrastructure management (VIM), a multi-cloud allows clients to easily access their resources remotely through interfaces such as Web services provided by Amazon EC2. There exist various types of tools and technologies for multi-cloud, such as Platform VM Orchestrator, VMware vSphere, and Ovirt. These are the tools which helps cloud providers construct a distributed cloud storage platform (DCSP) for managing clients' data. If such an important platform is vulnerable to security attacks, it would bring several losses to the clients. For example, the confidential data in an enterprise or company may be illegally accessed through a remote interface provided by a multi-cloud, or relevant data and archives may be lost/damaged or tampered with when they are stored into an uncertain storage pool outside the enterprise. Therefore, it is indispensable for CSPs (cloud service providers) to provide security techniques for managing their storage services. PDP (Provable data possession) is such a probabilistic proof technique for a storage provider to prove the integrity and ownership of clients' data without downloading data. The proof-checking without downloading the data makes it especially important for large size files and folders which include many clients' files to check whether these data have been tampered/damaged with or deleted without downloading the latest version of data. Thus, it is able to replace traditional hash and signature functions in storage outsourcing. Various provable data possession schemes have been recently proposed, such as Scalable provable data possession (SPDP) and Dynamic provable data possession (DPDP). However, these schemes mainly focus on provable data possession issues at un-trusted servers in a *single* cloud storage provider and it is not suitable for a multi-cloud environment.

2. PROPOSED ALGORITHM

2.1. Design Considerations

- Efficient provable data possession (PDP) scheme is used for distributed cloud storage to support the scalability of service and the data migration.
- CPDP (Cooperative Provable Data Possession) scheme based on the homomorphic verifiable response and the hash index hierarchy.
- To minimize the computation costs of clients and storage service providers.
- Articulates the performance optimization mechanisms.
- Providing some security based on multi-proven zero-knowledge proof system, which may satisfy the completeness, knowledge soundness, and zero-knowledge properties.

2.2. Description of the Proposed Algorithm

2.2.1. DEPSKY Algorithm

The DEPSKY model system or the algorithm is a storage cloud-of-clouds that overcomes the limitations of the individual clouds by using an some efficient set of Byzantine quorum system protocols, cryptography, secret sharing, erasure codes and the diversity that comes from using several clouds. The DEPSKY protocols at most require two communication round-trips for each operation and can store only approximately half of the data in each cloud for the typical case.

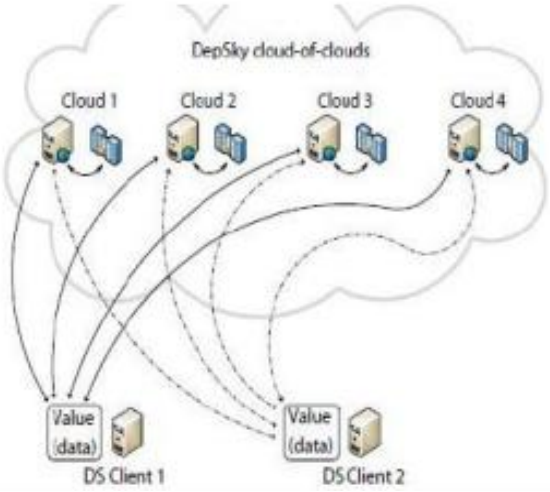

Fig -1: Depsky Model

Data and software process algorithm steps executed by the cloud customers to add the privacy structure of enforcement to the software and data before transferring them to the cloud. Privacy feedback algorithm describe essential components that should has to be considered and planned through when the designing privacy aware cloud service .The main aim of this algorithm is to inform user about various privacy mechanism applied on their data and make them aware of risks.

2.2.2. AES Algorithm

Advanced Encryption Standard is a block cipher with a block length of 128 bits. AES allows three different key lengths: 128, 192, or 256 bits. Most of our discussion will assume that the key length is 128 bits. Encryption is consists of 10 rounds of processing for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys. Except for the last round in each case, all other rounds will be identical. Each round of the processing includes only one single-byte based substitution algorithm, a row-wise permutation algorithm, a column-wise mixing algorithm, and the addition of the round key. The order which these four steps are executed is different for encryption and decryption. Therefore, the first four bytes of a 128-bit input block occupy the first column in the 4 × 4 matrix. The next four bytes occupy the second column, and so on. The 4×4 matrix is referred to as the state array in AES. AES also has the notion of a word. A word consists of 4 bytes that is 32 bits. Therefore, each column of the state array is a word, as is each row. Each round of processing works on the input state array and gives an output state array. The output state array was given by the last round which is rearranged into a 128-bit output block. Unlike Data Encryption Standard, the decryption algorithm differs substantially from the encryption algorithm. Although, the same steps are used in encryption and decryption, the order in which the steps are carried out is different, as mentioned earlier. The nature of substitutions and permutations in Advanced Encryption Standard (AES) allows for a fast software implementation of the algorithm.

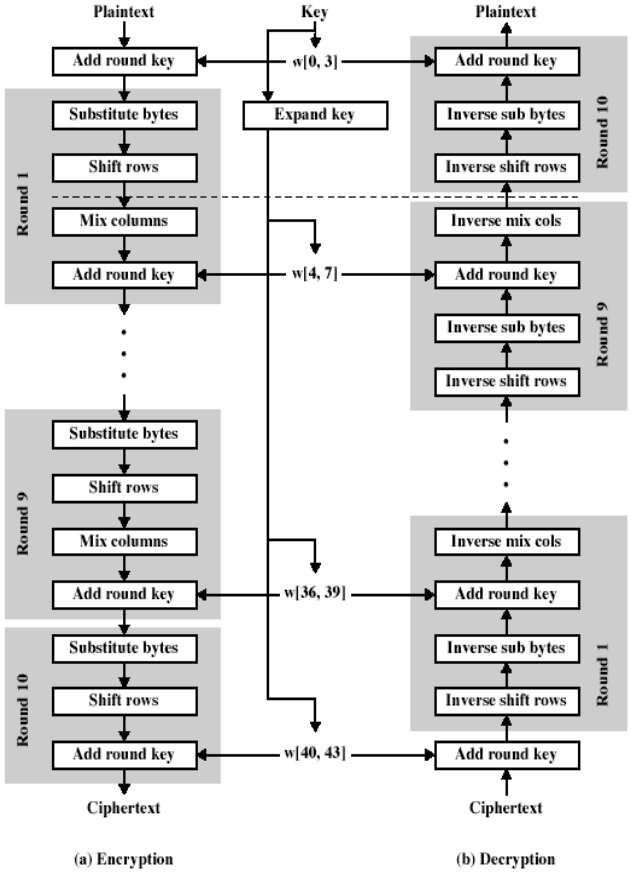

Fig -2: AES Encryption and Decryption

Before looking into details, we can see several comments about overall AES structure:

1. Advanced Encryption Standard (AES) cipher is not a Feistel structure.
2. The key which is provided as input is expanded into an array of 44 words (32-bits each), distinct words (128 bits) serve as a round key for each round; these are indicated in Fig -2.

3. 4 different stages are used, 1 permutation and 3 of substitution:
- Substitute bytes – Uses an S-box, which is used to perform a byte-to-byte substitution of the block
- Shift rows – A simple permutation
- Mix columns – A substitution that makes use of arithmetic over GF (2^8).
- Add round key – A simple bitwise XOR of the current block with the portion of the expanded key
4. The structure is very simple. Fig -3 shows the structure of a full encryption round.

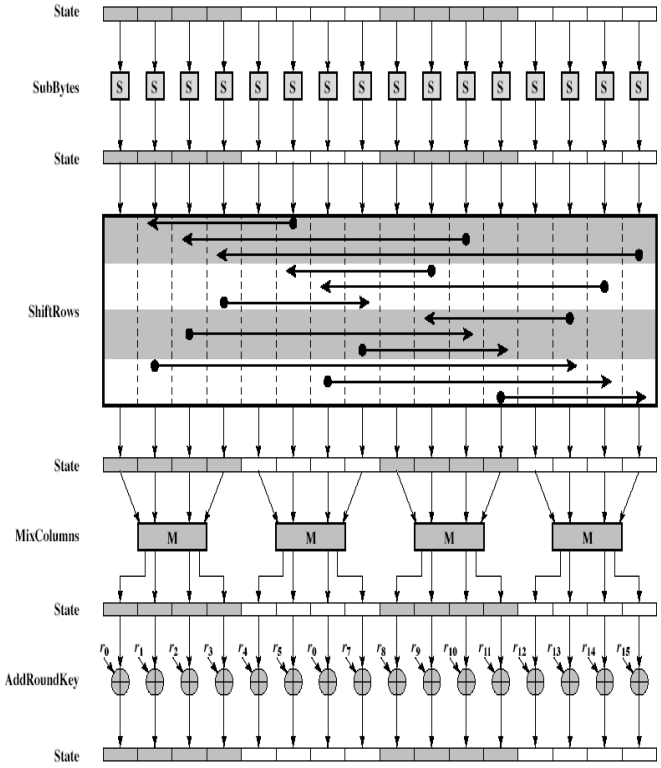

Fig -3: AES Encryption Round

5. Only Add Round Key stage uses the key. Any other stage is reversible without knowledge of the key.
6. The Add Round Key is a form of Vernam cipher and by this it would not be formidable. The other 3 stages together provide confusion, diffusion, and nonlinearity, but by this would not provide security because they do not use the key.
7. Each stage is easily reversible.
8. Decryption uses the same keys but in reverse order. Decryption is not identical to encryption.
9. At each horizontal point, the dashed line in Fig -2, is the State that is same for both encryption and decryption.
10. The final round of both encryption and decryption consists of 3 stages; it is the consequence of the particular structure of AES.

Table -1: AES Parameters

Key Size (words/bytes/bits)	4/16/128	6/24/192	8/32/256
Plaintext block size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Number of Rounds	10	12	14
Round key size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Expanded key size (words/bytes)	44/176	52/208	60/240

The number of AES parameters depends on the key length (Table-1). We assume the key length of 128 bits.

3. SYSTEM ORGANIZATION

3.1. System Architecture

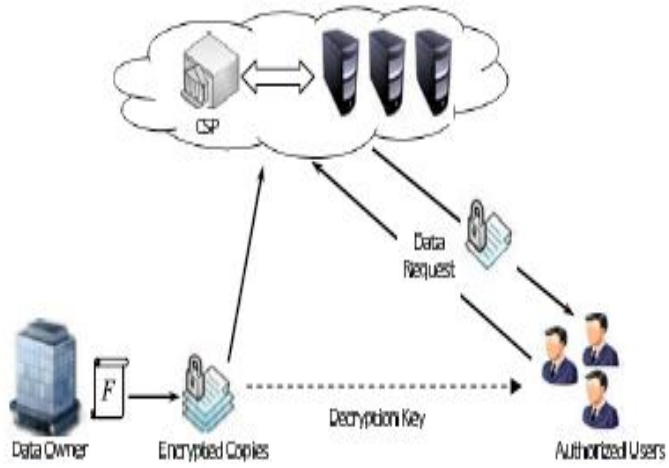

Fig -4: System Architecture

3.2. Methodologies Used

3.2.1. Homomorphic Token Technique

In the cloud data storage system, users will store their data in the cloud and no longer uses the data locally. Thus, the availability and correctness of the data files being stored on the distributed cloud servers must be guaranteed. The key issue is to detect any unauthorized data modification and corruption effectively, possibly due to server compromise and/or random Byzantine failures. In the distributed case when such inconsistencies are successfully detected, to find which server the data error lies is also the great significance, it is always be the first step to fast recover the storage errors and/or identify the potential threats of external attack. To address these

problems, our main scheme for ensuring cloud data storage is presented in homomorphic token technique. The first part is devoted to a review of basic tools from coding theory that was needed in our method for file distribution across cloud servers. Then, the homomorphic token is introduced. In the token computation function, we are considering a family of universal hash function, chosen to protect the homomorphic properties, which can be perfectly integrated with the verification of erasure-coded data. It is shown that how to derive a challenge response rule for identifying misbehaving servers as well as verifying the storage correctness. The procedure for file retrieval and error recovery is based on the erasure correcting code. Finally, we describe how to extend our method to a third party auditing with only slight modification of the main design.

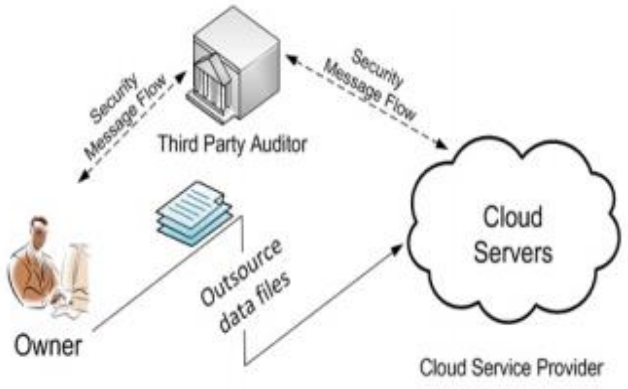

Fig -5: Towered secure storage multi-cloud computing architecture

3.2.2. File Division Technique

To achieve secure storage and access on outsource data files in the cloud we use the technique of multiple division to protect the data files. Proposed model has enabled us to store data easily and securely. In this method, we are dividing the data into multiple parts. When the number of parts increases the security of data also increase which makes it is difficult for intruder to check all file to match the content. We are not focusing on any particular technique for multiple division, we can use any basic method to divide file and to reduce the cost.

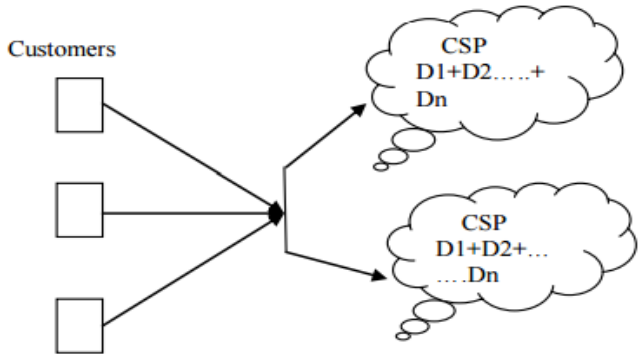

Fig -6: File Division Technique

To illustrate this mechanism we use example there are three clients represented as a C1 and C2 and they store this data in two different cloud service providers which was represented as CSP1 and CSP2. The clients can store there data in the CSP which will be divided in the form of multi division technique in the client side. The multiple data stored in a redundant form into both the cloud service provider.

3.2.3. Cooperative Provable Data Possession

CPDP (Cooperative PDP) schemes adopting zero-knowledge property and three-layered index hierarchy, respectively. In particular efficient scheme for selecting the number of optimal sectors and in each block to reduce or minimize the computation costs of clients and storage service providers. CPDP (Cooperative PDP) scheme is used without compromising the data privacy based on modern cryptographic methods.

3. CONCLUSIONS

Based on homomorphic verifiable response and hash index hierarchy, we proposed a CPDP scheme which support dynamic scalability on multiple storage servers. Also showed that this method provides all the security properties required by zero knowledge interactive proof system, so that it can manage various attacks even if it is given as a public auditing service in clouds. Furthermore, it optimized the periodic verification and probabilistic query to improve the audit performance. These experiments clearly demonstrated the approaches which only introduce a small amount of communication and computation overheads. Therefore, the solution can be treated as a new candidate for data integrity verification in outsourcing data storage systems. As part of future work, we would extend our work to explore more effective CPDP constructions. 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 an issue to provide the support of variable-length block verification.

REFERENCES

[1] Ayad F.Barsoum and M. Anwar Hasan, "Provable Multicopy Dynamic Data Possession in Cloud Computing Systems," DOI 10.1109/TIFS.2014.2384391, IEEE Transactions On Information Forensics And Security, Vol. 10, No. 3, March 2015.

[2] Z. Hao, S. Zhong and N. Yu "A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability", *IEEE Trans. Knowl. Data Eng.*, vol. 23, no. 9, pp.1432 -1437 2011

[3] C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in *Proc. 16th ACM Conf. Comput. Commun. Secur. (CCS)*, New York, NY, USA, 2009, pp. 213–222.

[4] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in *Proc. 4th Int. Conf. Secur. Privacy Commun. Netw. (SecureComm)*, New York, NY, USA, 2008, Art. ID 9.

[5] F. Sebé, J. Domingo-Ferrer, A. Martinez-Balleste, Y. Deswarte, and J.-J. Quisquater, "Efficient remote data possession checking in critical information infrastructures," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 8, pp. 1034–1038, Aug. 2008.

Books:

1. "Software Engineering Concepts" - Richard Fairley, Tata McGraw Hill Ltd, Forth Edition.
2. "Java 2 Complete Reference" - Herbert Schildt, Tata McGraw Hill Ltd, Forth Edition.
3. "JSP 2.0 The Complete Reference" - Patric**,** Tata McGraw Hill Ltd, Second Edition.
4. "The Complete Reference J2EE" - Herbert Schildt, Tata McGraw Hill Ltd, Fifth Edition.

Web References:

1. www.jspin.com
2. www.javaworld.com
3. www.sun.java.com
4. www.wrox.com