AN EFFICIENT PROVABLE MULTICOPY DYNAMIC DATA POSSESSION IN CLOUD COMPUTING SYSTEMS

S.JENIFER VAISLA[1], S.K.SARAVANAN [2]

[1] *Student, Department of Computer Applications, Valliammai Engineering College, Tamil Nadu, India*
[2] *Assistant Professor, Department of Computer Applications, Valliammai Engineering College, Tamil Nadu, India*

---***---

Abstract - *An efficient provable multicopy dynamic data possession in cloud computing systems deals with stored data in dynamic way to server. Multicopy Means, one data to be copied in multiple server. The user to upload the data in server with automatically single data to take multiple copies then that copies are stored in multiple server. Uploading the data in multi-server is to avoid the data loss from Hacking and server crash. In this paper, we introduce a new technique is Map based provable multicopy dynamic data that has the features like uses multicopy of data, File security and preventing data corruption. In this paper, there are three polynomial algorithms for protecting the data - Keygen, Copygen and Taggen. AES algorithm is used for data security. The authorized user to seamlessly access the file copies stored by the CSP. In Existing system Single copy of dynamic data and DES Algorithm is used for the above process.*

Key Words: Provable data possession (PDP), storage security, Cloud Service Provider(CSP), Cloud Computing, Dynamic Data, Data Integrity, Multi-copy

1. INTRODUCTION

Cloud Service Provider (CSP) is allows store more data on private computer system. The data storage infrastructure to store and retrieve data and it store unlimited amount of data. This is from of cloud computing that provides virtualized computing resources over the internet. This model is third-party provider hosts hardware, software, server, storage and other infrastructure component on behalf of its users. The customers pay on a per-use basis, typically by the hour, week or month. Some provider also charge customers based on the amount of virtual machine space they use.

PDP is technique for validating remote data integrity checking is a crucial technology in cloud computing. The two provably-secure PDP schemes that are more efficient than previous solutions, even when compared with schemes that achieve weaker guarantees. In particular, the overhead at the server is low (or even constant), as opposed to linear in the size of the data. Experiments using our implementation verify the practicality of PDP and reveal that the performance of PDP is bounded by disk I/O and not by cryptographic computation.

In remote data integrity checking protocols, the client can challenge the server about the integrity of a certain data file, and the server generates responses proving that it has access to the complete and uncorrupted data. The basic requirements are that the client does not need to access the complete original data file when performing the verification of data integrity, and that the client should be able to verify integrity for an unlimited number of times.

Juels et al describe a "proof of retrievability" (PoR) model and give a more rigorous proof of their scheme. In this model, spot-checking and error-correcting codes are used to ensure both "possession" and "retrievability" of data files on archive service systems. Specifically, some special blocks called "sentinels" are randomly embedded into the data file F for detection purpose and F is further encrypted to protect the positions of these special blocks. However, like [11], the number of queries a client can perform is also a fixed priori and the introduction of pre-computed "sentinels" prevents the development of realizing dynamic data updates. In addition, public verifiability is not supported in their scheme. Although schemes with private verifiability can achieve higher scheme efficiency, public verifiability allows anyone, not just the client (data owner), to challenge the cloud server for correctness of data storage while keeping no private information.

The MB-PMDDP of a database system is its structure described in a formal language. This schema specifies, based on the database administrator's knowledge of possible applications, the fact that can enter the database, or those of interest to the possible end-user. A model of this "theory" closely corresponds to a database, which can be seen at any instant of time as a mathematical object.

2. IMPLEMENTATION

We propose a map-based provable multi-copy dynamic data possession (MB-PMDDP) scheme. This scheme provides an adequate guarantee that the CSP stores all copies that are agreed upon in the service contract. Moreover, the scheme supports outsourcing of dynamic data, i.e., it supports block-level operations such as block modification, insertion, deletion, and append. The authorized users, who have the right to access the owner's file, can seamlessly access the copies received from the CSP. A thorough comparison of MB-

PMDDP with a reference scheme, which one can obtain by extending existing PDP models for dynamic single-copy data is given. We also report our implementation and experiments using Amazon cloud platform. The security of our scheme against colluding servers is discussed. The Figure 2.1 shows the system architecture of the proposed system.

2.1. PDP and POR

To restore security assurances eroded by cloud environments, researchers have proposed two basic approaches to client verification of file availability and integrity. The cryptographic community has proposed tools called proofs of retrievability (PORs) and proofs of data possession (PDPs).PDP scheme checks that a remote cloud server retains a file, which consists of a collection of n blocks. The data owner processes the data file to generate some metadata to store it locally. The file is then sent to the server, and the owner delete the local copy of the file. The owner verifies the possession of file in a challenge response protocol. A POR is a challenge response protocol that enables a prove (cloud-storage provider) to demonstrate to a verifier (client) that a file F is retrievable, i.e., recoverable without any loss or corruption. The benefit of a POR over simple transmission of F is efficiency. The response can be highly compact (tens of bytes), and the verifier can complete the proof using a small fraction of F.As a standalone tool for testing file retrievability against a single server, though, a POR is of limited value. Detecting that a file is corrupted is not helpful if the file is irretrievable and the client has no recourse. Thus PORs are mainly useful in environments where F is distributed across multiple systems, such as independent storage services. In such environments, F is stored in redundant form across multiple servers.

2.2. Privacy-Preserving PDP Schemes

The data owner first encrypts the file, Sends both the encrypted file along with the encryption key to the remote server. Moreover, the data owner sends the encrypted file along with a key-commitment that fixes a value for the key without revealing the key to the TPA. The primary purpose of this scheme is to ensure that the remote server correctly possesses the client's data along with the encryption key, and to prevent any information leakage to the TPA which is responsible for the auditing task. Thus, clients especially with constrained computing resources and capabilities can resort to external audit party to check the integrity of outsourced data, and this third party auditing process should bring in no new vulnerabilities towards the privacy of client's data. In addition to the auditing task of the TPA, it has another primary task which is extraction of digital contents.

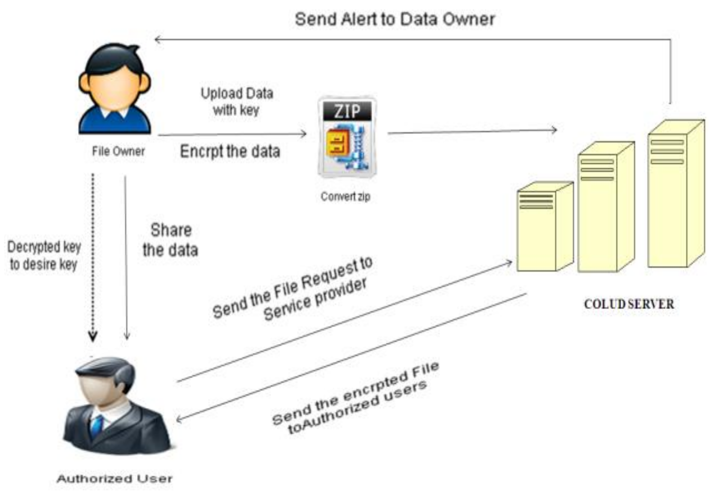

Fig 2.1 System Architecture

2.3. System Components

A. Data Owner

That can be an organization or an individual originally possessing sensitive data to be store in the cloud.

B. CSP

Who manages Cloud Servers (CSs) and provide paid storage space on its infrastructure to stores files.

C. Authorized Users

Authorized users, these users only to download the file from the others users. These users to send the file request to admin and file owner.

D. Registration

The user can store the file into the cloud storage only if he/she is a registered owner of this web application. The registration can be made as either free or a paid registration depending on the organization's requirement.

E. Copies Generation

File copies are created by Data Owner side. Propose system allows a user to stores all copies of a file in a storage system. Each copy of file can be produced at the time and it will store into storage system with tag of each file copies.

F. File Division

User's file is divided into data blocks of different sizes for improving the efficiency of storage and as well as to improve security of file.

G. File Upload

File is uploaded on the cloud storage with the help of CSP. Files store on cloud storage infrastructure which is location independent. While we upload file ultimately CSP store all copies of file which is agreed on service.

H. View Files

File content can be viewed in original format by Data Owner and Authorizer User but file content is viewed in encrypted format by CSP and Verifier.

I. File Modify

File can be modified only by the File Owner. Modification will be done by inserting, appending, editing or changing the data.

J. File Deletion

The Uploaded file can be deleted by the File Owner or CSP.

K. File Download

Only the verified Files can be downloaded by the File Owner and Authorized User. But they don't know which copy of file is downloaded.

L. File Verification

Public key shared by data owner for the verification process. File cannot be downloaded by the verifier side at the time of verification process.

M. File Storage

File is stored in the cloud is an Encrypted format using the private key which is generated by data owner. File copies can be stored in multiple servers with unique copy.

N. Verifier

Verifier is one of the users in this application. Verifier is used to verify the copies of file that are stored into cloud storage. Verifier randomly checks the integrity of all file copies by sending challenge to the CSP.

O. Integrity Verification

Verifier randomly send a challenge to the CSP to check integrity and consistency of file copies then CSP send proof of that challenge and finally verifier check is it correct or not without downloading of files copies.

2.4 Key Generation

Let ˆe : K1 × K2 → KT be a bilinear map and g a generator of G2. The data owner runs the KeyGen algorithm to generate a private key $x \in KP$ and a public key $y = kx \in K2$.

2.5 Initializing a cipher object

A cipher object obtained via get instance must be initialized for one of three modes, which are defined as final integer constants in the cipher class. The modes can be referenced by their symbolic names, which are shown below along with a decryption of the purpose of each mode:

ENCRYPT_MODE

Encryption of data.

DECRYPT_MODE

Decryption of data.

WRAP_MODE

Wrapping a java.security.key into bytes so that the key can be securely transported.

2.6 Using Encryption

This section takes the user through the process of generating a key, creating and initializing a cipher object, encrypting a file, and then decrypting it. Throughout this example, we use the Advanced Encryption Standard (AES).

Generating a key

To create an AES key, we have to instantiate a KeyGenerator for AES. We do not specify a provider, because we do not care about a particular AES key generation implementation, since we do not initialize the KeyGenerator, a system-provided source of randomness and a default keysize will be used to create the AES key:

KeyGenerator keygen=keyGenerator.getInstance ("AES");
SeccretKey AESKey=Keygen.generateKey ();

After the key has been generated, the same KeyGenerator object can be re-used to create further keys.

The cipher class

The cipher class provides the functionality of a cryptographic cipher used for encryption and decryption. Encryption is the process of taking data (called cleartext) and a key, and producing data (ciphertext) meaningless to a third-party who does not know the key. Decryption is the inverse process: that of taking ciphertext and a key and producing cleartext.

Creating a cipher

The next step is to create a Cipher instance. To do this, we use one of the getInstance factory methods of the Cipher class. We must specify the name of the requested transformation, which includes the following components, separated by slashes (/);

- The algorithm name
- The mode (optional)
- The padding scheme (optional)

We create an AES cipher in Electronic Codebook mode, with PKCS5-style padding. We do not specify a provider, because we do not care about a particular implementation of the requested transformation. The standard algorithm name for AES is "AES", the standard name for the Electronic Codebook mode is "ECB", and the standard name for PKCS5-style padding is "PKCS5Padding":

Cipher aesCipher; //Create the cipher
aesCipher=Cipher.getInstance ("AES/ECB/PKCS Padding");

2.7 Result

In existing system DES algorithm is used .In the proposed system AES algorithm, a secure algorithm is used for data security. It is used to generate the key its most secure 256-bit key length. A symmetric key encryption can be extremely secure and encrypting and decrypting symmetric key data is relatively easy to do, giving you very good reading and writing performance. In fact, many solid state drives, which are typically extremely fast, use symmetric key encryption internally to store data and they are still faster than unencrypted traditional hard drives.

3. METHODOLOGY

A. Existing System

In Fig 3.1 existing system the uploaded data are stored in Single copy way. Then authorized users send the file directly to the file owner, this is not the correct way and service provider may be access files illegally. So automatically data security loss.

Existing System Disadvantages:

- It uses Single copy.
- Service Provider may be Hack the data and authorized users not send file request to file owner.

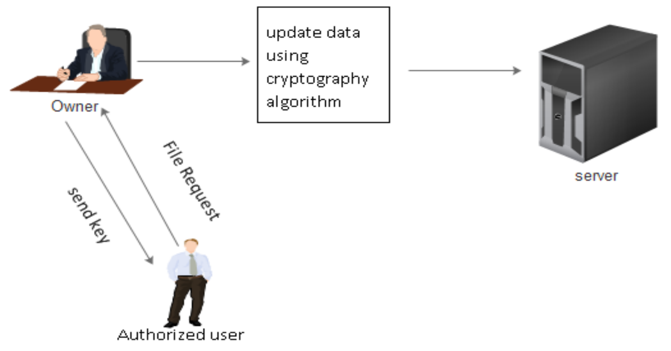

Fig 3.1 Exsisting system Architecture

B. Proposed System

In proposed system uploaded data are stored in multiple server (Multi copy).In this system one scheme and three algorithms were used. They are KeyGen, CopyGen, and TagGen. If user upload the data, automatically prepare three copies then stores that data in three servers for security and to avoid server overload. Those copies are encrypted so that cloud service provider or any others can't hack the data. When user uploads the data, servers automatically convert it to zip format. So servers reduce the file size automatically. User shares the file to authorized user. Then authorized user send the file request to cloud server again, server send the encrypted data to authorized use and authorized user get the decrypt key from data owner. In this system AES algorithm is used for data security.

Proposed System Advantages:

- Multicopy Data reduce access time and communication cost for user.
- If one copy is corrupted it will be redirected to another server and the file can be downloaded.
- Convert Zip format upload the data.
- In this system used AES algorithm. It is most secure 256-bit key length.

C. Algorithms

(PK, SK) ← KeyGen(). This algorithm is run by the data owner to generate a public key PK and a private key SK. The private key SK is kept secret by the owner, while PK is publicly known.

$\tilde{E}$← CopyGen (CNi , E)1≤i≤n. This algorithm is run by the data owner. It takes as input a copy number CNi and a file F, and generates n copies $\tilde{E}$= {$\tilde{E}$i} 1≤i≤n. The owner sends the copies $\tilde{E}$ to the CSP to be stored on cloud servers.

Φ← TagGen (SK, $\tilde{E}$). This algorithm is run by the data owner. It takes as input the private key SK and the file copies $\tilde{E}$, and outputs tags/authenticators set Φ, which is an ordered collection of tags for the data blocks. The owner sends Φ to the CSP to be stored along with the copies $\tilde{E}$.

4. SUMMARY AND CONCLUDING REMARKS

In the existing system the data's were send though email. But the user may not know whether he/she got the mail he/she only know when logged on to their mail-id, by connecting with the internet. In this work, we have studied the problem of creating multiple copies of dynamic data file and verifying those copies stored on untrusted cloud servers. In proposed system, the data is split into multiple copies with secured key attached to each copy. So that the data will be more secured and the space and cost will be reduced.

The proposed scheme is the first to address multiple copies of dynamic data. The communication between the authorized users and the CSP is measured in our system, where the authorized users can effortlessly access a data copy received from the CSP using a single secret key shared with the data owner. Furthermore, the proposed scheme supports public verifiability, allows arbitrary number of auditing, and allows possession-free verification where the verifier has the capability to verify the data integrity even though they neither possesses nor retrieves the file blocks from the server.

The corrupted data copy can be rebuilt even from a complete damage using duplicated copies on other servers. Through algorithms, we have shown that the proposed system is probably safe. As a future enhancement, the alert message can be send to the user to know about the mail in their inbox through mobile phone. More server can be used for security propose and RSA algorithm can also be used.

REFERENCES

[1] Ayad F. Barsoum and M. Anwar Hasan, " Provable Multicopy Dynamic Data Possession in Cloud computing systems", in IEEE Transactions On Information Forensics And Security, Vol. 10, No. 3, March 2015

[2] G. Ateniese et al., "Provable data possession at untrusted stores," in Proc. 14th ACM Conf. Comput. Commun. Secur. (CCS), New York, NY, USA, 2007, pp. 598–609.

[3] K. Zeng, "Publicly verifiable remote data integrity," in Proc. 10th Int. Conf. Inf. Commun. Secur. (ICICS), 2008, pp. 419–434.

[4] Y. Deswarte, J.-J. Quisquater, and A. Saïdane, "Remote integrity checking," in Proc. 6th Working Conf. Integr. Internal Control Inf. Syst. (IICIS), 2003, pp. 1–11.

[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.

[6] P. Golle, S. Jarecki, and I. Mironov, "Cryptographic primitives enforce communication and storage complexity," in Proc. 6th Int. Conf. Finan-cial Cryptograph. (FC), Berlin, Germany, 2003, pp. 120–135.

[7] M. A. Shah, M. Baker, J. C. Mogul, and R. Swaminathan, "Auditing to keep online storage services honest," in Proc. 11th USENIX Workshop Hot Topics Oper. Syst. (HOTOS), Berkeley, CA, USA, 2007, pp. 1–6.

[8] M. A. Shah, R. Swaminathan, and M. Baker, "Privacy-preserving audit and extraction of digital contents," IACR Cryptology ePrint Archive, Tech. Rep. 2008/186, 2008.

[9] E. Mykletun, M. Narasimha, and G. Tsudik, "Authentication and integrity in outsourced databases," ACM Trans. Storage, vol. 2, no. 2, pp. 107–138, 2006.

[10] 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.

BIOGRAPHIES

Ms. Jenifer Vaisla is a Student Pursuing MCA course in Valliammai Engineering College. She is a talented, dedicated and hardworking student.

Mr. S.K.Saravanan is an Assistant Professor, in Department of Computer Application, Valliammai Engineering College. He has 16 years of teaching experience in Engineering College.