International Journal of Engineering and Technical Research (IJETR)
ISSN: 2321-0869 (O) 2454-4698 (P), Volume-5, Issue-2, June 2016

Data Auditing Using Proof of Retrievability and Recovery in Cloud Computing

Miss.Prachiti Karandikar, Prof. Dr.Pradeep K.Deshmukh

***Abstract*— Cloud computing is a way to increase capabilities without investing in infrastructure as well as licensing cost on new software. Users are storing data on cloud storage with the expectation that data should be accessible, consistent and correct. While using cloud storage data reliability and integrity are major problems that need to face. Cloud Service Provider (CSP) can modify or delete data which either takes large memory space or unused by client from a long time. Hence, some technique is needed for checking the data periodically for its integrity. To overcome this problem, proof of retrievability scheme is proposed here. It includes use of TPA along with CSP. TPA preprocesses the data and check data integrity reducing computational burden at client side. Also if the file is changed or deleted then recovery of original file is done with the help of distributed servers.**

***Index Terms*— Cloud computing, data security, data integrity, Proof of retrievability.**

I. INTRODUCTION

Cloud computing, is on-demand computing. Cloud computing is a useful resource where we can store all our data in order that some applications and software can get full advantages by making use of this technology without any server and regional hard disk for our data storage. Registration of user to the cloud server or to the third party which provide the cloud service is needed. Privacy of the information and security has to be considered. Day by day community bandwidth is growing .Due to this bandwidth and riskless but flexible network connections users can now use high pleasant offerings from information and program located at remote information centers.

There are many advantages of cloud over local storage. Cloud server provides facility to store user's data on a cloud. So customers can add their information on cloud and can access it without any additional burden of time, location, and cost.

Manuscript received.

Miss.Prachiti M. Karandikar, Computer Science and Engineering, Savitribai Phule Pune University/ Rajarshi Shahu College of Engineering/JSPM,Pune,India,8421595167,(e-mail:prachitimkarandikar@gmaill.com).

Dr.P.K.Deshmukh, Computer Science and Engineering, Savitribai Phule Pune University / Rajarshi Shahu College of Engineering / JSPM, Pune, India,9922266346 ,(e-mail:pkdeshmukh9@gmaill.com).

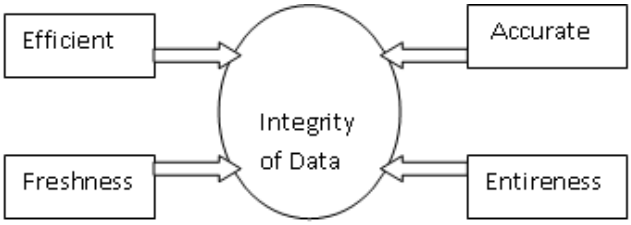

Fig.1.Advanced view of Integrity

Amazon, Microsoft, EC2, Google and are some famous cloud storage service providers that have attracted users to use cloud storage. Users are enjoying use of these services due to ease of access to their data which is hosted on another infrastructure.

With increased use of Internet technologies, the major obstacle is to preserve the originality of data. Difficulty of outsourced data may also be depend on the way by which the data owner find an effective solution to participate in frequent checking for integrity of data without the neighborhood reproduction of data documents.Fig.1 shows the advanced view of Integrity.

Users need to verify that their data remain as they stored on cloud. Because data stored on cloud can easily be lost or corrupted due to human errors and hardware and software failures. Also data can be changed or deleted by malicious cloud storage server. Traditionally entire data was retrieved from cloud and cryptographic techniques, hash values are used for integrity verification. Many schemes and researches have been done. Schemes fall into a) Private verifiability b) Public verifiability.

Private verifiability means data owner can verify integrity of their outsourced data. It provides higher efficiency but increases computational overhead at client side.

Public verifiability allows anyone to check the integrity of data. For this third party auditor is used which alleviates clients from performing lots of computation.

In cloud computing data is uploaded as well as updated by clients.

Many integrity verification schemes only focus on static data. Scheme proposed here provide public verifiability as well as dynamic data operation.

Framework contains four parties Clients, third party auditor (TPA), Cloud storage server (CSS) and Distributed Server (DS).

II. RELATED WORK

In [1] new scheme is proposed to check the integrity of outsourced data. TPA is offered to scale down the computational burden of client. TPA does the task of auditing the data by challenging the CSS. Scheme provides public verifiability along with dynamic data operation. This PoR model provides safety against the reset attacks launched by

cloud storage server within the upload phase.TPA stores the tag of file to be uploaded and use these tags to check integrity.

In [2] authors defined a PDP model. It gives probabilistic proof that third party stored a file. User can access small blocks of file for producing the proof. Challenge and response method is used in this technique. Some constant amount of metadata of client's data is stored at client side. Locally stored metadata is used to verify proof which is given by server .Client gives challenge to server for proving possession and wait for response. Server then computes and sent proof to client. Metadata is used to check correctness of response. RSA based Homomorphic variable tags are used to achieve goal. PDP accesses random sets of blocks and samples servers' storage. Limitation of PDP is it gives only probabilistic proof not a deterministic proof. It cannot support dynamic data possession.

In [3] a new scheme known as proof of retrievability (POR) is proposed. Using this scheme, verifier (user) can determine that whether Prover (server) hacked his file or not. Scheme uses sentinels (called disguised blocks). Sentinels are hidden among usual file blocks for detecting data amendment by way of the server. Verifier challenges prover by specifying locations where sentinels are collected and asking to return associated value. Values are compared then to check integrity of data. In this approach single cryptographic key is computed and stored by verifier. Key is computed using keyed hash algorithm. Error resiliency of their system is improved due to error correction codes. This scheme increases larger storage requirement and computational overhead on prover.

In [4] authors proposed new technique to obtain PoR. Two schemes are proposed here. Pubic verifiability is implemented in first scheme. Here shortest query response of any POR is obtained which is secure in the random oracle model. Second scheme provides shortest response with private retrivability. It is secure in the standard model. Two homomorphic authenticators are used. First is based on PRF's and second based on BLS signature. Only one authentication value is allowed in both schemes. Here, erasure encoded file is broken up into n blocks by user. Each file block is accompanied by way of authenticators of equal size. Use of BLS signature give smaller sized proof as compared with RSA. It also accept higher error rate. But this scheme still works on static data only; dynamic data update is not supported.

In [5] PDP model is expanded. Verifiable updates on stored data are provided. It makes use of new variation of authenticated dictionaries. These dictionaries are centered on rank knowledge. Rank knowledge is used for organizing dictionary entries. To check the integrity of file blocks, authentication skip list is used. Untrusted server stores File F and its skip list. Root metadata is stored at client side. File f is divided into blocks. Client issues question atRank (i) to the server when he desires to verify integrity of block I. Server then computes tag T (i) as its proof and send to client. Clients compare proof given by server with stored metadata and check for integrity. Also to update the data client issue atRank (i) (for insertion) and atRank (i-1) (for deletion).It does not allow for public variability of the stored data.

Scheme proposed in Paper [6] provides provable security and desirable efficiency simultaneously. Two servers are used. Particularly one for auditing and another for data storing. Third party Auditor (TPA) is used for auditing purpose. TPA screens information stored in cloud storage as well as transactions between data owner and cloud storage server (CSS). Public verifiability is provided. All the Computation is done by server instead of client. This leads to reduction of computational overhead at client side. Security of this scheme is analyzed under variant of [2] which supports public verifiability. This is the game between challenger C (client) and storage server (adversary A) played to get proof of retrievability from Adversary A. If proof is valid for fraction of challenges, client can extract the file F.

In this paper [7], author tends to propose a secure cloud storage system supporting privacy- preserving open auditing. Author tend to any extend their result to modify the TPA to perform audits for various users simultaneously and with effectiveness. Inside and out security and performance analysis show the arranged schemes area unit indisputably secure and amazingly economical.

III. ISSUES

Implementations of all these aforementioned algorithms in Section II provide solutions that fulfill many requirements such as high scheme efficiency, stateless verification, and retrievability of data. Some provide public verifiability and some provide private verifiability. Another major concern is support of dynamic data operation. In cloud storage the outsourced information no longer only accessed however update by way of clients additionally. Regrettably, existing work ordinarily center of attention on static information documents. Dynamic data only addressed in [4]. Also all above work only provide information about file is corrupted or not. We propose a framework which provides public verifiability, dynamic data support along with recovery of corrupted file.

IV. PROBLEM STATEMENT

No existing scheme can provide public verifiability and dynamic data operations simultaneously. To overcome this We propose a 'Proof of Retrievability (PoR)' construction which provides public verifiability, dynamic data operation and recovery for corrupted files.

V. IMPLEMENTATION DETAILS

This Framework contain three parties Clients, third party auditor (TPA), Cloud storage server (CSS), Distributed servers (DS)

A. System Model

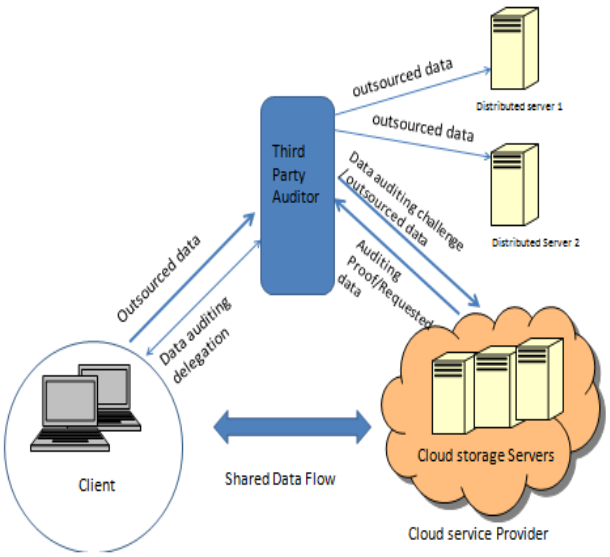

Fig. 2 System Architecture

Client: Client has large data files for outsourced on cloud. Also depend on cloud for maintenance and computation of data. Client can be either any organization or an individual.

Third party auditor (TPA): It is trusted third party which can expose the risk of cloud storage services on behalf of client. In this system TPA generates tag for data in file before uploading it to cloud storage server.

Cloud storage server(CSS):This entity is managed by Cloud service Provider (CSP).It has computation resource and storage space for maintenance of clients data.CSS have to provide integrity proof during integrity verification phase.

Distributed Servers (DS): These servers are used to store another copy of data stored on CSS. To recover the corrupted file, data backed up on DS is used

B. Security Model

When client thinks of checking originality of his data, he simply challenges the cloud storage server .Client can request to TPA integrity checking of any file block which is stored at cloud storage server.TPA forwards challenge to storage server and storage server response with proof in the form of hash values. TPA equates the received hash of file block form proof with hash values stored in his database. If the hash matches, it sends response to client, which show that the requested file stored on CSS is not corrupted otherwise it is corrupted. If the file is corrupted then TPA recovers file with the help of Distributed servers. TPA orders CSS for recovery of corrupted file using distributed servers. Distributed servers are used for backup of original files.

C. System Phases

1. Setup:

This is key generation phase.

Setup (security Parameter) $\rightarrow$ Key K

2. Upload : This is two step process

- Step 1: Client uploads file F to TPA.

Upload (K, F) $\xrightarrow{\text{Encryption}}$ (F*)

TPA generates tag using hash function and signs on it. Hash values are stored and receipt is sent back to client

- Step 2: TPA uploads file F to CSS

Upload (F*, t) $\longrightarrow$ (F*)

Where t=sig (h (F))

TPA sends file with tag to CSS.

3. INTEGRITY VERIFICATION

Client through TPA or TPA directly challenges CSS for integrity verification by sending query for file (which needs to be verified).CSS sends file as proof to TPA.TPA checks for integrity.

$$P(F^*, t) \xrightarrow{V(t,)} \begin{cases} 1 \text{ if } F^* \text{ passes verification} \\ \text{if } F^* \text{ fails verification} \end{cases}$$

4. UPDATE

Data modification, data insertion and data deletion is done.

After updation also, CSS have to pass integrity verification

$$P(Fu^*, tu) \xrightarrow{V(tu,\ update)} \begin{cases} t \text{ if } Fu^* \text{ passes updation} \\ \text{if } Fu^* \text{ fails updation} \end{cases}$$

5. RECOVERY

If File fails verification test, it means that file is corrupted so client can go for recovery to get his original file. Client requests TPA to recover his file and TPA messages to DS for recovery.

Request (t) $\xrightarrow{Reco\ (F^*, t)}$ Recovered file Fr*

Again verification is done on recovered file.

D. Algorithm

- AES Encryption

The AES algorithm is a symmetric block cipher used to Encrypt (encipher) and decrypt (decipher) information. It uses the same key for encrypting and decrypting, so both the sender and receiver must know and use of same secret key. AES as well as most encryption algorithms is reversible. The AES algorithm operates on bytes, which makes it simpler to implement and explain. AES is an iterated block cipher means that the same operations are performed many times on a fixed number of bytes.

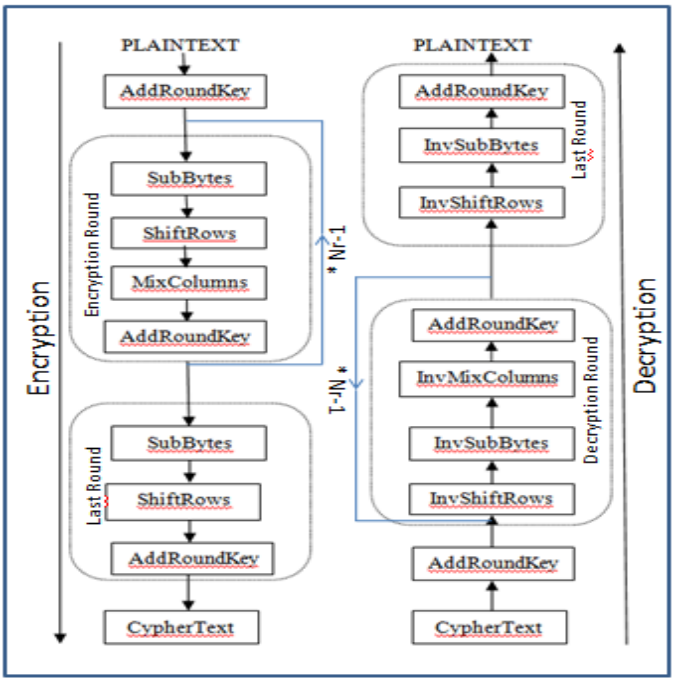

Fig. 3 AES Encryption and Decryption

- SHA1

SHA1 is a message digest algorithm .It takes as a input a Message and produces as output 160-bit hash value. .
SHA1 algorithm is 6-step process

i. Padding of „1000…“,
ii. Appending message length,
iii. Preparing 80 process functions,
iv. 80 constant,
v. Preparing 5 word buffers,
vi. Processing input in 512 blocks.

Both the transmitter and intended receiver of a message in Computing and verifying a Digital signature uses the SHA1. It is computationally not feasible to find a message which corresponds to a given message digest when we use SHA1. Also task of finding two different messages which produce the same message digest is hard. Any change to a message result in a different message digest, and the signature will fail to verify.

- SPEKE

SPEKE stops “man in the middle attack” by the integration of the password. An invader who is superior to recite and alter all messages between m and N can’t discover the shared key K and can’t make deduce more than one password in each interaction with a party who knows that.

All messages between M and N can’t discovered shared key K And can’t deduce more than one password in each interaction with a party who knows that.

1. M and N selects appropriately safe and large Prime Pr. Also chooses the hash function H().
2. M and N settle on a share password ∏.
3. M and N both build G=H (∏2) mod Pr.
4. M picks a secret random integer A, then sends N,

“Ga mod Pr”.

5. N picks a secret random integer B, then sends M,

“Gb mod Pr”.

6. M and N each terminate if their received values are not in the range [2, Pr-2] to avoid small subgroup confinement attack
7. M computes K= (G a mod Pr) mod Pr.
8. M computes K= (G b mod Pr) mod Pr.

- RECOVERY

If file corruption have been detected then

1. TPA requests original file from one of the distributed server
2. Applies hashing and stores hash values.
3. TPA sends recovered data to cloud storage server.
4. CSS stores the new recovered file.

E. Mathematical Model

Let W is the Whole System Consist of W= {I, P, O, S, F}
Where,

I – input, P-Procedure, S-Success, F-Failure

- **Input (I):**

I= {F}

Where F is Collection of files to be stored on cloud server

F= {f1, f2, FN}

- **Procedure (P):**

P={S, Up, $H^{T}_{n,}$ H^{C}_{n} , Ud,R}

Where,

1. Setup (Key generation)

S= {K1, K2, .Kn} Where, K is the set of keys generated.

Setup (security Parameter) ⟶ Key K

2. Upload

Up= {Up1, Up2, Upn} Where Up is the set of files uploaded to cloud storage.

Ek (F, K) ⟶ UPI

F is encrypted and stored to cloud server.

3.Integrity verification using hash values

SHA1(F*) ⟶ H

- $H^T = \{ H^T_1, H^T_2, \ldots\ldots\ldots\ldots, H^T_n \}$

where H^T_n is set of hash values stored at TPA.

- $H^c = \{ H^c_1, H^c_1, \ldots\ldots\ldots\ldots, H^C_1 \}$

where H^C_n is set of hash values stored at cloud storage server.

Equal (H^T_i, H^c_i) = } 1 if F* passes verification
0 if F* fails verification

4. Update

Ud= {Ud1, Ud2…Udn} where Ud is the set of files to be updated

5. Recovery

Request (t) $\xrightarrow{Recover(F*,t)}$ Recovered file Fr*

VI. Experimental Setup

The system is built using Java framework (version jdk1.8) on Windows platform. The eclipse is used as a development tool. The system doesn't require any specific hardware to run; any standard machine is capable of running the application. The system analysis is carried out on datasets consisting of files.

VII. Results and Discussion

In this section we will provide a thorough experimental evaluation of the construction proposed. Existing system only provides information about whether the outsourced file is corrupted or not. This system provides recovery for deleted file.

Fig. 4 depicts the comparison of time required for tag generation and verification. Verification time is less than tag generation time because tag generation is required for whole file but for verification, comparison of some part of file is sufficient.

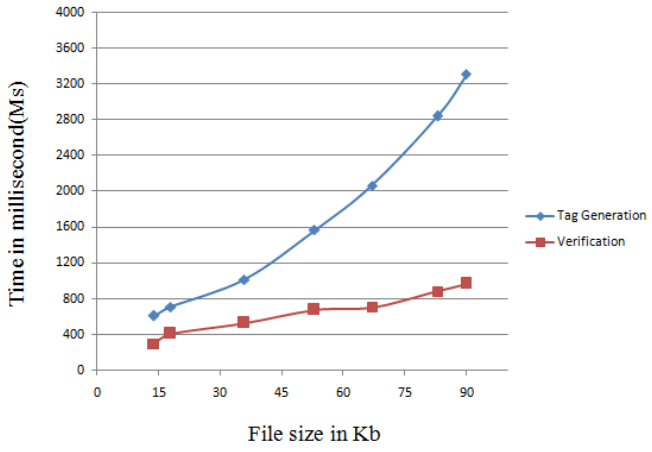

Fig. 4. Time for tag generation and verification

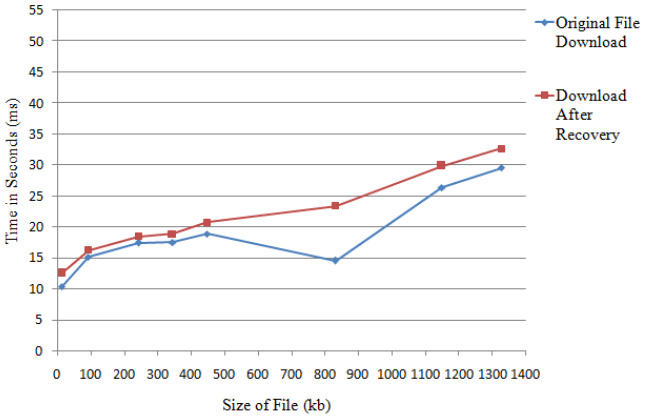

Fig. 5.Time cost for original file download and download with recovery.

Figure .5. depicts the time required for "original file download" and 'download after doing recovery'. Experiments shows that our system requires bit extra time for 'download after recovery' than original file download. This feature improves fault tolerance of the system within near about same time.

('Original file download' means time required for normal download of file when file is available at cloud server. 'Download after recovery' means time required to download file when it is deleted from cloud server and recovered using copy on another distributed server.)

VIII. Conclusion

An integrity verification scheme is proposed here which gives an idea of asking proof of retrievability for cloud storage. Also feature for recovery of corrupted data is introduced.

Dynamic data operations are provided for user to update the data on cloud easily instead of uploading whole file again. Here a third party auditor is presented for the purpose of preprocessing, uploading the data on cloud storage server and recover the corrupted data behalf of client. The third party auditor performs the data integrity verification of the outsourced data upon the client's request. Use of TPA scales down the computational burden for tag generation on client. Result analysis conclude that 'To Recover and download' doesn't require much more time than "download of original file" so this feature improves fault tolerance of the system within near about same time.

Acknowledgment

The authors would like to thank the researchers as well as publishers for making their resources available and teachers of RSCOE, Computer Engineering for their guidance. We are also thankful to the reviewer for their valuable suggestions. We also thank the college authorities for providing the required infrastructure and support.

REFERENCES

[1] J.Li,X.Tan.XChen and D.S.Wong,"OPoR : Enabling proof of retrievability in cloud computing with Resource constrained Devices," IEEE transactions on cloud computing on volume:XX No:2015

[2] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in CCS '07: Proceedings of the 14th ACM conference on Computer and communications security. New York, NY, USA: ACM, 2007, pp. 598–609.

[3] A. Juels and B. S. K. Jr., "Pors: proofs of retrievability for large files," in CCS '07: Proceedings of the 14th ACM conference on Computer and communications security. New York, NY, USA:ACM, 2007, pp. 584–597.

[4] H. Shacham and B. Waters, "Compact proofs of retrievability,"in ASIACRYPT '08: Proceedings of the 14th International Conference on the Theory and Application of Cryptology and Information Security. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 90–107.

[5] C.Erway,A.Kupcu,C.Papamanthou, and R.Tamassia," Dynamic provable data possession",cryptography e print archive,Report2008,432,2008/432,2008,http://eprint.iacr.org/ .

[6] J.Li,X.Tan.XChen and D.S.Wong,"An efficient proof of retrievability with public auditing in cloud computing,"in/NCoS,2013,pp,93-98"

[7] C.Wang,Q.Wang,K.Ren,and W.Lou,"Privacy preserving public auditing for data storage security in cloud computing,"in INFOCOM,2010 Proceedings IEEE. IEEE,2010 pp.1-9.

[8] E.-C.Chang and J.Xu,"Remote integrity check with dishonest servers,"in preceedings of ESORICS 2008 ,volume 5283 of LNCS.springer-verlag,2008,pp. 223 -237

[9] Q.Zheng and S.Xu,"Fair and dynamic proofs of retrievability," in CODASPY,2011,pp.237-248

[10] C.Wang,Q.wang and K.Ren,"Ensuring data storage security in cloud computing,"in preceedings of IWQoS 2009,Charleston,USA 2009

About Author

Ms.Prachiti M. Karandikar.

Ms.Prachiti Karandikar is currently pursuing ME degree in computer science technology from Rajarshi shahu college of engineering,Pune. Completed B.E degree in information Technology from Bharati Vidyapeeth College of Engineering,Kolhapur under the Shivaji University, Kolhapur in Jun 2013. She has teaching experience of one year. Her area of interest is Cloud Computing,SQL.

Prof.Dr. Pradeep K. Deshmukh.

Prof. Dr. Pradeep K. Deshmukh received M.E, PhD in Computer Sci. and Engg. He is currently working as Professor in computer engg. department, JSPM's RSCOE, Pune. He has 24 years of teaching experience. His area of interests is Cloud Computing, Network Security and operating system.