Remote Integrity and Proof of Retrievability and Recovery in Cloud Computing

Geetanjali Sharma[1], Shubhangi Kale[2], Ruchira Hagawane[3], Sayali Dabhadkar[4], Shamal Mundkar[5]
[1, 2, 3, 4, 5] Department of Information Technology
[1, 2, 3, 4, 5] D. Y. Patil College of Engineering, Akurdi

Abstract- *Cloud computing is turning into very popular. Users are deciding on cloud as repository for his or her statistics. The statistics in the cloud ought to be reachable, correct, consistent and excessive first-class. While considering cloud as storage records security and integrity of stored statistics is burning trouble. When customers keep their records on cloud there may be a danger of amendment or updation of records. Many researchers had worked and proposed algorithms to clear up this trouble. This survey paper specializes in core techniques of proof of garage (POS) that are Proof of records Possession (PDP) and Proof of Retrievability (PoR). Both the techniques are used to make certain the cloud consumer about integrity of records garage on cloud. OPoR, any other distributed garage plan consisting of a allotted storage server and a TPA is proposed here. TPA is concept to be semilegitimate. Specifically, we bear in mind the undertaking of allowing the TPA, for the cloud customers, to pre-manner the facts earlier than moving to the distributed storage server and later confirming the statistics uprightness. OPoR outsources the overpowering calculation of the label generation to the cloud review server and takes out the contribution of client in the examining and inside the preprocessing ranges. Besides, we make stronger the Proof of Retrievabiliy (PoR) model to bolster dynamic facts operations, and similarly assure safety towards reset assaults dispatched by using the allotted garage server in the transfer level.*

Keywords- cloud computing, facts safety, records integrity, evidence of information ownership, proof of retrievability.

I. INTRODUCTION

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. 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 users 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. 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 Users need to verify that their data remain as they stored on cloud. Data saved on cloud can quite with ease be misplaced or corrupted as a result of human errors and hardware and software failures. Also data can be changed or deleted by malicious cloud storage server.

The main objective of this work is to find out whether the users outsourced data is original or whether it is affected by some malicious intruder. For this auditing is performed with the help of hash values. To reduce the computational burden of making hash values and integrity verification at client side, TPA(Third Party Auditor) is introduced. Also public verifiability and dyanamic data operation are SYNOPSIS provided. PoR model is the first to support dynamic update operations and security against reset attack in a verification scheme. The robustness against reset attack ensures that a malicious storage server can never gain any advantage of passing the verification of an incorrectly stored file by resetting the client (or the audit server) in the upload phase. Also recovery of deleted file is done by TPA. All the process is transparent to user.AES algorithm is used for encryption of file. And SHA1 is used for hashing purpose. Use of these algorithms improves security of file.

II. LITERATURE REVIEW

[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.

[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 clients 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.

[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.

[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 PRFs 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.

III. PROPOSED SYSTEM

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

Client :

Client has tremendous information files for outsourced on cloud. Also depend upon cloud for preservation and computation of information. Client may also be both any group or individual person.

Third Party auditor (TPA):

It is trusted third party which can expose the hazard of cloud storage services on behalf of Client. On this system TPA generates tag for data in file earlier than uploading it to cloud storage server.

Cloud storage server (CSS) :

This entity is managed by way of Cloud servicer provider (CSP).It has computation useful resource and cupboard space for maintenance of clients data.CSS have got to provide integrity proof throughout integrity verification segment.

Distributed Servers (DS) :

These servers are used to save another reproduction of data saved on CSS. To recuperate the corrupted file, information backed up on DS is used.

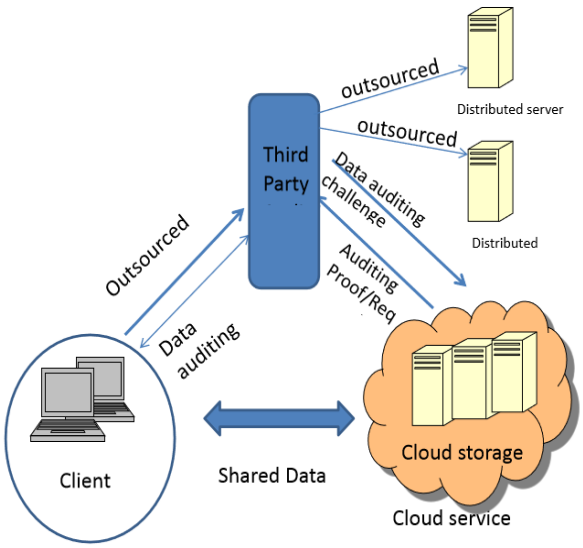

Figure 1. Proposed System

Client challenges the cloud storage server to ensure integrity of data. Client can request to TPA integrity checking

of any file block which is stored at cloud storage server. TPA first computes and then stores the hash values of blocks. 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 from proof with hash values stored in his database. If the hash matches, then file is original otherwise its corrupted. If the file is corrupted then TPA recovers corrupted file from distributed server on which backup is taken.

The main propose OPoR, a new PoR scheme with two independent cloud servers. Particularly, one server is for auditing and the other for storage of data. The cloud audit server is not required to have high storage capacity. Different from the previous work with auditing server and storage server, the user is relieved from the computation of the tags for files, which is moved and outsourced to the cloud audit server. Furthermore, the cloud audit server also plays the role of auditing for the files remotely stored in the cloud storage server. A strengthened security model is develop by considering the reset attack against the storage server in the upload phase of an integrity verification scheme. It is the first PoR model that takes reset attack into account for cloud storage system. Present an efficient verification scheme for ensuring remote data integrity in cloud storage. The proposed scheme is proved secure against reset attacks in the strengthened security model while supporting efficient public verifiability and dynamic data operations simultaneously.

IV. IMPLEMENTATION

Cloud Computing moves the application software and databases to the centralized large data centers, where the management of the data and services may not be fully trustworthy. In this work, we study the problem of ensuring the integrity of data storage in Cloud Computing. To reduce the computational cost at user side during the integrity verification of their data, the notion of public verifiability has been proposed. However, the challenge is that the computational burden is too huge for the users with resource-constrained devices to compute the public authentication tags of file blocks. To tackle the challenge, we propose OPoR, a new cloud storage scheme involving a cloud storage server and a cloud audit server, where the latter is assumed to be semi-honest. In particular, we consider the task of allowing the cloud audit server, on behalf of the cloud users, to pre-process the data before uploading to the cloud storage server and later verifying the data integrity. OPoR outsources the heavy computation of the tag generation to the cloud audit server and eliminates the involvement of user in the auditing and in the preprocessing phases. Furthermore, we strengthen the Proof of Retrievabiliy (PoR) model to support dynamic data operations, as well as ensure security against reset attacks launched by the cloud storage server in the upload phase. Client challenges the cloud storage server to ensure integrity of data. Client can request to TPA integrity checking of any file block which is stored at cloud storage server. TPA first computes and then stores the hash values of blocks. 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 from proof with hash values stored in his database. If the hash matches, then file is original otherwise its corrupted. If the file is corrupted then TPA recovers corrupted file from distributed server on which backup is taken. This Framework contain three parties Clients, third party auditor(TPA),Cloud storage server(CSS),Distributed servers(DS) Client: Client has tremendous information files for outsourced on cloud. Also depend upon cloud for preservation and computation of information. Client may also be both any group or indivisual person. Third Party auditor (TPA): it is trusted third party which can expose the hazard of cloud storage services on behalf of Client. On this system TPA generates tag for data in file earlier than uploading it to cloud storage server. Cloud storage server(CSS):This entity is managed by way of Cloud servicer provider (CSP).It has computation useful resource and cupboard space for maintenance of clients data.CSS have got to provide integrity proof throughout integrity verification segment. Distributed Servers(DS):These servers are used to save another reproduction of data saved on CSS. To recuperate the corrupted file, information backed up on DS is used. We present an efficient verification scheme for ensuring remote data integrity in cloud storage. The proposed scheme is proved secure against reset attacks in the strengthened security model while supporting efficient public verifiability and dynamic data operations simultaneously proposed a dynamic version of the prior PDP scheme. However, the system imposes a priori bound on the number of queries and do not support fully dynamic data operation.Dynamic data storage in distributed scenario, and the proposed challenge-response protocol can both determine the data correctness and locate possible errors.

Three different network entities can be identified as follows: Client module: an entity that has large data files to be stored in the cloud and relies on the cloud for data maintenance and computation, can be either individual consumers or organizations. Cloud Storage Server (CSS) module: an entity, which is managed by Cloud Service Provider (CSP), has significant storage space and computation resource to maintain clients data. The CSS is required to provide integrity proof to the clients or cloud audit server during the integrity checking phase. Cloud Audit Server (CAS) module: a TPA, which has expertise and capabilities that clients do not have, is trusted to assess and expose risk of

cloud storage services on behalf of the clients upon request. In this system, the cloud audit server also generates all the tags of the files for the users before uploading to the cloud storage server. The basic goal of PoR model is to achieve proof of retrievability. Informally, this property ensures that if an adversary can generate valid integrity proofs of any file F for a non-negligible fraction of challenges, we can construct a PPT machine to extract F with overwhelming probability. It is formally defined by the following game between a challenger C and an adversary A, where C plays the role of the audit server (the client) and A plays the role of the storage server: Setup Phase: The challenger C runs the Setup algorithm to generate its key pair (pk, sk), and forwards pk to the adversary A. Upload Phase: C initiates an empty table called Rlist. A can adaptively query an upload oracle with reset capability as follows: Upload: When a query on a file F and a state index i comes, C checks if there is an entry (i, ri) in the R-list. If the answer is yes, C overwrites ri onto its random tape; otherwise, C inserts (i, ri) into R-list where ri is the content on its random tape. Then C runs (F, t) Upload(sk, F; ri), and returns the stored file F and the file tag t. Here Upload(; ri) denotes an execution of the upload algorithm using randomness ri. Challenge Phase: A can adaptively make the following two kinds of oracle queries: IntegrityVerify: When a query on a file tag t comes, C runs the integrity verification protocol Integrity VerifyA C(pk, t) with A. Update: When a query on a file tag t and a data operation request update comes, C runs the update protocol UpdateA C(sk, t, update) with A.

V. PROPOSED METHODOLOGY

RC6

RC6 proper has a block size of 128 bits and supports key sizes of 128, 192, and 256 bits, but, like RC5, it may be parameterised to support a wide variety of word-lengths, key sizes, and number of rounds. RC6 is very similar to RC5 in structure, using data-dependent rotations, modular addition, and XOR operations; in fact, RC6 could be viewed as interweaving two parallel RC5 encryption processes, although RC6 does use an extra multiplication operation not present in RC5 in order to make the rotation dependent on every bit in a word, and not just the least significant few bits.

"'Encryption Procedure:"'

```
B = B + S[0]
D = D + S[1]
 for i = 1 to r
 do
 {
 t = (B*(2B + 1)) XOR lg w
 u = (D*(2D + 1)) XOR lg w
 A = ((A t) XOR u) + S[2i]
 C = ((C u) XOR t) + S[2i + 1]
 (A, B, C, D) = (B, C, D, A)
}
 A = A + S[2r + 2]
 C = C + S[2r + 3]
```

"'Decryption Procedure:"'

```
C = C - S[2r + 3]
A = A - S[2r + 2]
for i = r downto 1
do
(A, B, C, D) = (D, A, B, C)
 u = (D*(2D + 1)) XOR lg w
 t = (B*(2B + 1)) XOR lg w
 C = ((C - S[2i + 1]) XOR t) u
 A = ((A - S[2i]) XOR u) t
D = D - S[1]
 B = B - S[0]
```

VI. RESULT ANALYSIS

Outputs / Snap shots of the results 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. given below shows that 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 efficient. Shows that system require small amount of extra time with extra feature of recovery.

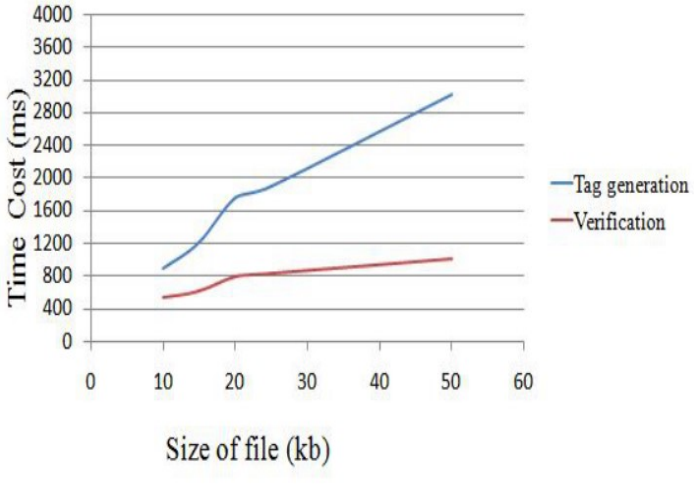

Figure 2. Performance Measure

Table 1. Result

Data in KB	time complexity for decrypt [Proposed system] in second	time complexity for decrypt [Exisitng system] in second
128	0.51	1
256	0.63	1.2
384	0.76	1.42
512	0.91	1.6
640	1	1.79

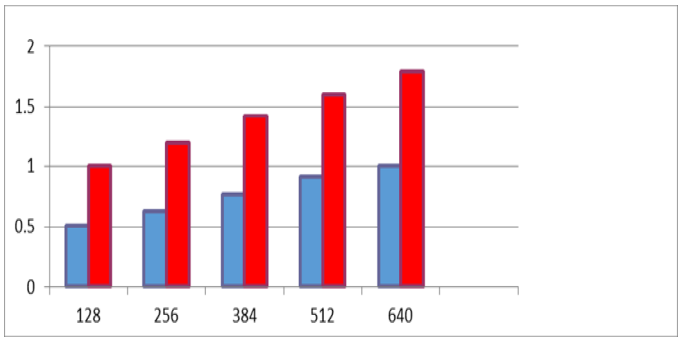

Figure 3. Result Analysis

VII. CONCLUSION

In this paper, a new proof of retrievability for cloud storage is introduced, in which a trustworthy audit server is introduced to preprocess and upload the data on behalf of the clients. In OPoR, the computation overhead for tag generation on the client side is reduced significantly. The cloud audit server also performs the data integrity verification or updating the outsourced data upon the clients request. Besides, we construct another new PoR scheme proven secure under a PoR model with enhanced security against reset attack in the upload phase. The scheme also supports public verifiability and dynamic data operation simultaneously. There are several interesting topics to do along this research line. For instance, we can (1) reduce the trust on the cloud audit server for more generic applications, (2) strengthen the security model against reset attacks in the data integrity verification protocol, and (3) find more efficient constructions requiring for less storage and communication cost. We leave the study of these problems as our future work.

VIII. ACKNOWLEDGMENT

We take this opportunity to thank Mr. A. J. Patankar, the Head of the Department (Information Technology) and Mr. K. D. Bamane the project coordinator and our project guide Mrs. Geetanjali Sharma, for their valuable guidance and for providing all the necessary facilities, which were indispensable in the completion of this project report. We are thankful to all the staff members of the Department of Information Technology of D. Y. Patil College of Engineering, Akurdi for their valuable time, support, comments, suggestions and persuasion. We would also like to thank the Institute for providing the required facilities, internet access and important books.

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.