RESEARCH ARTICLE OPEN ACCESS

A Survey on Proof of Retrievability and its Techniques

Reshma A. Hegde[1], Madhura Prakash[2]
1(*Dept of ISE, BNM Institute of Technology, Bengaluru*)
2 (*Dept of ISE, BNM Institute of Technology, Bengaluru*)

Abstract:

Cloud storage permits users to store their data in a remote server to eliminate overpriced local storage and management costs, then access data anytime, anyplace. Variety of solutions are projected to tackle the integrity of stored cloud data and its retrievability. One amongst the most important consideration with data storage in cloud today is data integrity verification of untrusted cloud. A very important approach to attain this goal of data integrity verification is Proof Of Retrievability, in which the storage server will assure verifier via a concise proof that a client's file is obtainable. In this work, the study on static and dynamic approaches of Proof Of Retrievability and numerous techniques are discussed.

Keywords **— Cloud storage, proof of retrievability, integrity, data dynamics**

I. INTRODUCTION

In Cloud computing users can store their data into the cloud so as to enjoy the on-demand prime quality applications and services from a shared group of computing resources. Cloud computing makes these benefits a lot of appealing than ever, it also adds challenging security hazards towards client's outsourced cloud data. It's of critical importance to clients to possess strong sturdy proof that they actually get the service they pay. The clients need to test that their stored cloud data is not deleted or being altered with time. Hence the data possessor must be convinced that their data is stored safely in the cloud. Numerous trends in computing systems are opened up today which relates to new forms of cloud data outsourcing.

Users are mitigated from the burden of data maintenance and data storage by data outsourcing. Outsourcing data brings with it several benefits. But correlated with it are many dangers involved. Clients need to check the data time to time for data correctness. PoR (Proof Of Retrievability) is the scheme for data integrity correctness. Proof Of Retrievability is a challenge-response protocol which allows the cloud storage provider to prove to the client that the stored File F is accessible without any modification or loss.[1] In this paper, different schemes which supports Proof of Retrievability is proposed. The paper provides various static and dynamic approaches for Proof Of Retrievability.

II. PROOF OF RETRIEVABILITY(POR)

Proof of Retrievability (POR) is a solid challenge response proof given to a client by the prover that the stored file in the cloud is is safe and integral that the user can fully recover it.The main advantage of Proof of Retrievability over other schemes is efficiency. The reply can be compact and using small portion of file F, the verifier can complete the correctness of the file F. If the file is irretrievable, sensing whether the file is corrupted is not helpful. Thus Proof of Retrievability is mainly deployed in environment, where file is allocated across several systems. The file F is stored in multiple servers in redundant form. Static and dynamic PoR schemes are discussed below in detail.[1]

A. Static Schemes

1) Basic Scheme: This scheme doesn't involve encoding of the entire data, only a small bits of data are encrypted which reduces the burden on client side. And also storage overhead on client is minimized. This scheme minimizes the size of proof of data integrity and also reduces utilization of network bandwidth. [2]

Pros: Minimizes the storage and computational overhead on both server and client side.

Cons: The quantity of queries that can be asked is mounted apriori. But this range is quite massive and might be sufficient if the amount of information storage is little. It is a challenge to extend the quantity of queries with this scheme.

2) POR for Large Files: In this scheme the prover stores solely a single cryptographic key. Here the verifier stores only a single cryptographic key—irrespective of the dimension and range of the files. This method requires that the prover access solely a tiny portion of a (large) file F within the course of a POR. The part of F "touched" by the verifier is actually independent of the length of F and actually include few hundreds of blocks. This retrievability scheme encrypts F and arbitrarily embeds a group of randomly-valued blocks called sentinels. [3] The employment of encryption here makes the sentinels identical from file blocks.

The prover is challenged by verifier by specifying the location of a set of sentinals and the prover is asked to return the sentinel values. If the prover has changed or deleted considerable part of F then with high chance it'll even have restrained a number of sentinels. It is so unlikely to reply

properly to the verifier. To safeguard against corruption by the prover of a tiny portion of F, we can also adopt error-correcting codes.

Pros: Makes sure retrievability and possession of files on service systems.

Cons: Difficult particularly when the data to be encrypted is huge. Because of inserted sentinals and error correcting codes there is a huge overhead on server side. There also requires large storage space on the prover side.

3)Compact POR: In this scheme two short, homomorphic authenticator are being employed. The first one, makes Proof of Retrievability scheme more secure within the standard model based on Pseudorandom Functions. The second makes the Proof of Retrievability scheme secure within the random oracle model based on BLS(Boneh-Lynn-Shacham) signatures.[4]

Pros: It needs less communication overhead and boundless range of queries can be raised.

Cons: Used just for static information.

4) 2-Phase Protocol: This "spot-checking" method is used in this challenge-response protocol. In every challenge, a division of file blocks is sampled to detect any adversarial behavior. This sampled results is calculated which is returned back to the client. The returned results are computed using extra information which were rooted into the file at the time of encoding.[5]

Pros: Storage overhead on client side is minimized. This scheme tolerates maximum error rates and is proved efficient and secure under adversarial behaviour.

Cons : Used solely for static data.

***5) HAIL*:** High Availability and Integrity Layer for Cloud Storage(HAIL) manages integrity and convenience across a set of servers or independent server storage services. It makes use of Proof of Retrievability(PoR) as building blocks in which storage assets will be tested and reallocated once the failures are found.[6]It does in a way that transcends the fundamental single-server design of Proof Of Retrievability and exploits within cross server and within server redundancy. It depends on one trustworthy verifier that can be a client or a service functioning on behalf of a client which interacts with server to check the correctness of stored file in the cloud.

Pros: Robust assurance of file-intactness: Low overhead, Robust adversarial model: Direct communication between client and server.

Cons: Used just for static information.

B. Dynamic POR

1) Data Correctness: The Data Correctness scheme involves encoding small bit of data per block instead of encrypting the entire block of information. This reduces the burden on the client side and reduces bandwidth necessities. Hence this type is more suitable for resource constrained devices. In this, the third party auditor solely needs to store one cryptographical key and two functions that generate random sequences ,no matter the dimensions of the file F. In this type , the third party auditor adds some metadata to the file and then stores them in the cloud. Hence during verifying he uses this metadata to verify the correctness of the data. It supports dynamic updation , insertion, deletion of the data. Hence the scheme is a dynamic approach for data verification.[7]

Pros: Lowers the burden on client side as the work of verification and other processes is done by the third party auditor.

Cons: Verifying many files from multiple clients at the same time isn't attainable

2) Public Auditability: There are two types of auditing schemes, public and private. Although private auditing does a higher scheme potency, it is overhead on client side. In pubic auditing a Third Party Auditor(TPA) having a private key can efficiently audit the data to verify the integrity of cloud storage data while keeping no personal information of the clients. Hence this scheme supports stateless verification requiring no state information. Thus public auditability permits the clients to delegate the verification to third party auditors reducing the overhead.

This scheme is designed to support two necessary goals i,e data dynamics and public auditability simultaneously. To support data dynamics, Markle Hash Tree(MHT) model proof is improvised. The model carries out block tag authentication[8].This also overcomes the drawback of data correctness by allowing multiple client files auditing simultaneously. To support multiple auditing, Bilinear Aggregate Signature technique is used. This scheme is economical and provably secure.

Pros: Public auditability for integrity of stored data, Dynamic updation support.

Cons: Efficiency of the scheme remains unpredictable.

3) A Dynamic PoR Scheme with O(logn) Complexity : This scheme has three stages

Pre-process stage: The client generates metadata of the information pre processed, before uploading the file to the server. Then the client will outsource the file to the server and solely keeps the metadata[10].

Verification stage: The client sporadically checks the data for correctness. The client inquires the server randomly to supply the proof. It verifies the proof with the metadata so that it can sight any file corruption. Hence data integrity is checked with high likelihood.

Update stage: The client requests the server to update the file. After every update, the server will prove to the client that it is appropriately handled

Pros: It is attainable to sight corruption with high likelihood even if the cloud service provider tries to hide them. Also this scheme supports dynamic updates. The worst case performance when compared to alternate scheme is O(log n) complexity

Cons: Less resourceful.

4) E POR: Efficient Proofs Of Retrievability (E PoR) is an economical and secure retrievability scheme. In this type, single data block consists of s clusters, and for every verification subsets of one block are retrieved.[11]. The computation value is O(s) cluster exponentiations on the prover side while O(`) group multiplication/addition/PRF on verifier side. Also the storage overhead is 1=s of the information size and communication value is O(1) bits per verification. It is tried that this projected retrievability scheme is secure under a robust Diffie Hellman Algorithm.

Pros: This scheme is more economical and secure. It solely requires steady number of communication bits per verification.

III. Conclusion

This study provides a consolidated report of all the techniques of the Proof of Retrievability schemes which are Static and Dynamic in single and hybrid clouds.

REFERENCES

[1]K. Zeng, "Publicly verifiable remote data integrity," in ICICS, 2008, pp. 419–434.

[2] Sravan Kumar R, Ashutosh Saxena, Data Integrity Proofs in Cloud Storage, 978-1-4244-8953-4/11/@ 2011 IEEE.

[3] Ari Juels and Burton S. Kaliski, PORs: Proofs of Retrievability for Large Files, CCS '07 Proceedings of the 14th ACM conference on Computer and communications security, 978-159593-703-2,USA

[4] Hovav Shacham1 and Brent Waters, Compact Proofs of Retrievability, J. Pieprzyk (Ed.): ASIACRYPT 2008, LNCS 5350, pp. 90–107, 2008 International Association for Cryptologic Research 2008.

[5] Kevin D. Bowers, Ari Juels, Alina Oprea, Proofs of Retrievability: Theory and Implementation, CCSW'09, Journal of Systems and Software, v.85 n.5, p.1083-1095, May, 2012.

[6] Kevin D. Bowers, Ari Juels, Alina Oprea, HAIL: A High-Availability and Integrity Layer for Cloud Storage, ACM 978-160558-784-4 /09/11, USA.

[7] Malathi.M, Murugesan. T, A Scheme for Checking Data Correctness in the Cloud, 2012 International Conference on Information and Network Technology (ICINT 2012) IPCSIT vol. 37 (2012).

[8] Qian Wang, Cong Wang, Kui Ren, Wenjing Lou, and Jin Li, Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing, IEEE Trans. Parallel Distrib. Systems.

[9] Qian Wang, Cong Wang, Kui Ren, Wenjing Lou, and Jin Li, Enabling Public Verifiability and Data Dynamic for Storage Security in Cloud Computing, IEEE Trans. Parallel Distrib. Systems.

[10] Zhen Mo Yian Zhou Shigang Chen , A Dynamic Proof of Retrievability (PoR) Scheme with O(logn) Complexity, IEEE Trans. Parallel Distrib. Systems.

[11] Jia Xu and Ee-Chien Chang, Towards Efficient Proofs of Retrievability in Cloud Storage, IEEE Trans. Parallel Distrib. Systems.