Security Safeguarding Delegatable Proof Storage with Information Dynamics in Cloud Storage

[*]Masrath Begum, [$]Dr.Mohammed Abdul Waheed, [#]Shambhavi Tapsale

[*]Assistant Professor, [#]Reader, GNDEC Bidar, India.

[$]Associate Professor, VTU CPGS, Kalaburgi, Karnataka, India,

[*]masrath456@gmail.com, [$]dr.mawaheed@gmail.com, [#]shambhavitapsale67@gmail.com

Abstract: Recently the cloud storage has achieved huge popularity. The confidential data is stacked on the cloud servers and thus the issue arises with the integrity and security of the data. In order to tackle this issue a lot of research has been done and most important among all of them is a scheme called Proof of Storage (POS). The POS scheme lets the third party to validate the integrity of the data stacked on the cloud servers. The problem allied with the POS technique was that these techniques adopted very costly and slow operations for generating tags which authenticate the data. So here we are introducing the advanced version of POS scheme called Delegatable Proof of Storage (DPOS). This scheme is efficient just like the Private POS scheme and also promotes the third party auditor, which validates the data integrity on the part of data owner. When the DPOS scheme is compared with POS system it accelerates the tag generation process, also makes sure that in any aspect it do not compromise the efficiency. The system also promotes dynamic operations and brings down the computation for data update to O(log n) and it requires only constant cost for communication.

Keywords —Third Party Auditor, Message Authentication Code, Data Dynamics, Proof of Storage, Owner Delegated Auditor.

I. INTRODUCTION

Cloud computing presents a great list of benefits like no cost for infrastructure, offering scalability and availability. And probably this is the reason why people are shifting to the cloud storage for loading their confidential data onto the remote servers so that their burden of local storage is minimized. People tend to store their private data onto the cloud without bothering to keep a backup on their local devices and data can be retrieved on request later on. So the honesty and privacy of stored files has to be assured by the cloud service provider. Earlier the owner of the data had the burden of computing integrity of data and for this purpose he needed to download the complete file and then examine it for its honesty. But with the advent of Proof of Storage (POS) [1], this overhead was eradicated. The principal behind the POS scheme is data file which is being uploaded on the cloud server is diverged into number of blocks and from each of the block homomorphic verifiable tag is generated, then the tag along with data file would be sent to server of the cloud. Now the verifier chooses few blocks of data instead of complete file in order to audit the file which was loaded on the server of the cloud (prover) using HVTs, which brings down the overhead caused by the communication.

Since POS scheme was invented in 2007 a lot of work has been done in order to add improved features to it. We are concentrating on the two concepts – public verifiability in [3], and data dynamics in [4]. Most of the previous POS schemes used methods which were expensive enough in generating the HVTs for block of data. This means it would be more expensive to generate HVT for the files which are large in size. Example of this can be POS scheme discovered by Wang et al. [5], takes about 17 hours in order to create HVTs for file with size 1GB with 8 cores CPU. And this huge calculation is not suitable for smart phones or laptop etc.

Public verifiability means that any third party can validate the honesty of the data stacked on the cloud server, which definitely eradicates the load of computation from the owner of data. When adopted it is not a good idea to permit anybody to validate the data whenever they wish to, because of two main reasons: (1) Files which are popular would get audited by people very repeatedly, without any need. This can end up with denial of service attack. (2) On other hand the data files that do not drive the attention from people would get audited scarcely, so in case of attack on integrity of data, the forgery would not be noticed by the data owner and it would be too late to take any countermeasures to fix the problem. We overcome this issue we are delegating this work of auditing to the some semi-trusted third party. Now this semi-trusted third party would take total responsibility of auditing the data stacked on the server and this will be done in some commanded manner on the behalf of owner of data. This third party auditor is a server which offers the services which can be paid or in some cases it can also be

 DOI : 10.35291/2454-9150.2019.0304

free. We name such kind of auditor as Owner delegated Auditor (ODA).

Another feature on which we are focusing to add it to POS scheme is dynamic operation particularly update operation where data owner can easily update data block once it is sent to the cloud server. Earlier when the data block was updated at i-th position then the HVTs of the following blocks needed to be re-created. This sounds impractical when dealing with large file with large number of blocks. In order to manage the issue we can consider the indices rather than HVTs once the block of data is updated. Here we are utilizing markle Hash Tree which is tree based structure and rank based authentication skip list that requires just O(log n) calculation, this leads to overhead due to communication of O(log n), here n is number of blocks.

To overcome the concern associated with the POS system, we introduce a upgraded version known as the Delegatable Proof Of Storage (DPOS) [6]. This scheme like publically verifiable POS lets third party to audit the data and is also efficient enough like verifiable scheme which is private. We also provide the update block feature with reduced computation of O(log n) and concurrently it needs constant cost for communication. Scheme also safeguards the privacy of data against ODA. Also we show that our scheme is very efficient for creating HVTs which we call Message Authentication Code (MAC).

II. OUTLINE OF DPOS SCHEME

The DPOS scheme split up the data file into n number of blocks ($block_n$). And these blocks are then encrypted to safeguard the privacy, then we are utilizing homomorphic tag authentication function to create Message Authentication tags (MAC_n) for each of the block, this function do not have any kind of expensive working. The size of MAC is 2/n-fraction of complete data file, where value of n can be any integer which is positive. Also secret key (Sk) will also be created based on the content of the file. Then the (MAC_n) and the (Sk) will be delegated to the auditor also to the cloud server. Now ODA can verify the honesty of the file by inspecting the consistency between the ($M.AC_n$) stored in ODA and (MAC_n) stacked in the cloud storage server by utilizing the secret key. Owner of the data file is also given authority to check the honesty of the data blocks by again validating consistency between the (MAC_i) stored in the owners side and cloud side using the same secret key (Sk).

In order to lessen the cost for communication and to increase the privacy we are incorporating our new tag function with previous techniques. We are also personalizing the technique introduced by Kete et al. [7], to minimize the size of proof to O(1) from O(m). Also we are upgrading Okamoto's technique so that leakage of the information to the ODA while the auditing process is prevented.

A. Contribution of DPOS scheme

- The experimental results prove that efficiency provided by the DPOS scheme is as good as private POS scheme. The observed throughput for creating the tags is 26MB/s. At the same time the scheme gives authority to the third party auditor to audit the file just like the public verifiable scheme.
- This scheme also allows the dynamic operations particularly block update. The calculation involved in update process is put down to O(log n) with steady cost for communication is required to validate the block.
- This scheme works great under Billinear Strong Diffie-Hellman Assumptions.
- The output of the experiment gives proof of the better performance of DPOS when in comparison with early developed schemes. In particular the speed for creating tags is very high.

III. RELATED WORK

Since last few decades a lot of experimentation has been done which looks for auditing the honesty of the data stored at remote servers. Ateniese et al. [2], introduced PDP scheme which was first scheme with public verifiability. This technique utilized RSA- based techniques for generating homomorphic tags and it used few number of blocks of data to validate the honesty of data, rather than downloading complete file. As the data is sampled the complexity of communication gets minimized. The drawback with the scheme was it leaked the data to the third party auditor while auditing process. Meanwhile Juels and Kaliski [1] proposed the better model known as POR which used the techniques like spot-checking and also code for correcting errors to justify the retrievability and honesty of the data stacked remotely. Their scheme had a control over number of auditing tasks and it did not let public verifiability. Shacham and Waters [3], innovated POR scheme which is public verifiable and builds verification tags from the BLS signature. Later Hao at el. [12], proposed the scheme called two privacy preserving auditing by public and they did not apply any masking technique. In all these techniques the owner of the data had a burden of computation for generating the tags particularly when the file is huge. This makes the impractical use of this technique in mobile devices.

Yang at el. [8], invented the POR scheme which did support public verifiability and also the cost involved in computation was constant. Apon et al. [10] formulated the scheme used in [9], showed a function which was simple and need few number of AND gates and few hours. With all the early publically verifiable schemes the owner of the data

 DOI : 10.35291/2454-9150.2019.0304

had a lot of computation burden when it comes to creating the tags. Thus this was not practical technique with large files with large number of blocks.

IV. MODEL OF THE SYSTEM

Delegatable Proof of Storage (DPOS) works by implementing three different algorithms (KeyGen, Tag, Update) and two other algorithms which helps in interaction (P, V). These algorithms are explained below

- KeyGen(1^{λ}) → (Sk): When the parameter is given for the security (1^{λ}) this random algorithm for generating key creates a secret key (Sk). We use the symmetric key generation algorithm called Advanced Encryption Standard (AES).
- Tag(sk, F) → ($Block_n$, MAC_i): If input to the tag generation algorithm is given as secret key (Sk) and file containing data F, the algorithm creates n number of blocks and MAC for each of the block.
- Update($Block_n$, Fname) → (Block', MAC'): when update algorithm is given n-th block as input along with file name it generates new block and new MAC.
- (P(MAC_i) V(Sk, MAC_i) → Accept or Reject: It outputs a bit either Accept or Reject when the prover algorithm P communicate with verifier algorithm V. (MAC_i) is given as input to prover and to the verifier input is given as (Sk, MAC_i).

The architecture of the DPOS scheme is depicted in Figure 1. DPOS system works with three modules – data owner, auditor and cloud server. These three modules run (keyGen, Tag, Update, (P, V)) algorithms in three phases for implementation. The first phase will be executed only one time in the beginning, where as the phase for the proof and update may execute large number of times

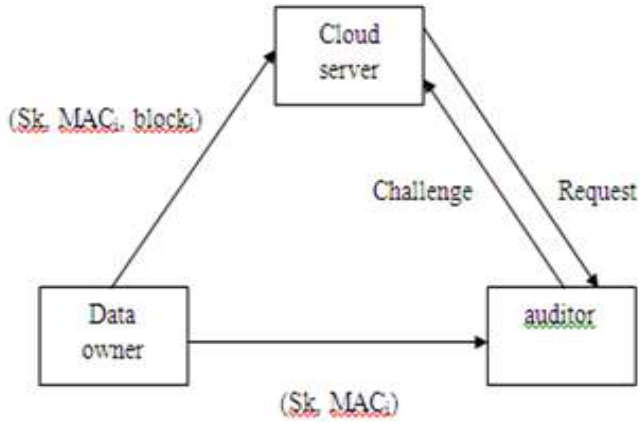

Figure 1: Architecture of DPOS

Setup Phase: At first the owner of the data executes the algorithm for only one time which generates the secret key (Sk) called KeyGen(1^{λ}) and encrypts the file using the (Sk). Now over this encoded file the tag generation algorithm is run, which creates the n number of blocks ($Block_n$) and the MAC for each of the n-blocks. In the end owner of the data delegates the encoded blocks, tags for authentication (MAC_n) and the secret key (Sk) towards the remote cloud server. He also transmits securely the secret key (Sk) and (MAC_i) to the auditor. Now the data owner deletes the original file and encoded file from his local storage and just keeps the secret key and (MAC_n).

Proof Phase: There can be any number of sessions which consists of proofs for honesty of data. Proof session begins when the ODA attempts verify the honesty of the data on the behalf of the owner of data by running the V algorithm, and tries to communicate with the cloud server which runs the P algorithm. So the sever for storage on cloud is known as Prover and the ODA is known as Verifier.

Update Phase: Here the possessor of the data downloads particular encrypted block from the cloud server and then using secret key (Sk) he Decrypts the block and then updates the block. And then encrypts that block again and generates the new MAC and delegates it to the cloud server and the audited

V. DYNAMIC DPOS SCHEME

When the one block of file is updated, new MAC has to be recreated for that corresponding block and need to be delivered to the cloud server. This can affect other MAC too but to avoid this issue we are managing indices rather than managing MAC on updating data. Every block is associated with index which is unique for each of them. We are introducing the binary tree structure which is known as index management tree (IMT) and also the idea of AVL tree [11], is borrowed where it is not mandatory to create new MAC for blocks other than update block. This absolutely minimizes the complexity associated with calculation and communication.

In DPOS system tag generation is different when compared to the basic DPOS and we use three different algorithms for allowing update operation are UpdateRequest, UpdateIndex and UpdateConform.

A. Generating tags:

Proposed DPOS scheme follow same method for generating MAC just like basic DPOS and initialization of the IMT also has to be done by owner of the data. Update data phase: Basically three algorithms are used to update the data and they are Updaterequest, UpdateIndex and UpdateConform. The data owner runs the UpdateRequest(B,i) algorithm which takes data block to be updated and the position of that corresponding block, which creates the update request (UR) and this request will be delegated for the update operation to the ODA and the cloud server and this updates the IMT with URI.

 DOI : 10.35291/2454-9150.2019.0304

The ODA runs this UpdateIndex (URI)→ {null} algorithm. UpdateConform (URI)→ {success, fail}: the blocks of the file and HVT of that block will get updated by sever in the cloud when it accepts the request from owner of data.

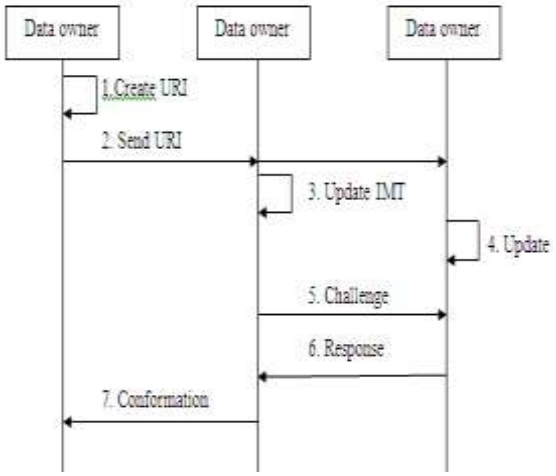

Figure 2: The workflow of update operation

VI. PERFORMANCE ANALYSIS

For the purpose of comparison we have evaluated our scheme and few other public verifiable and private verifiable schemes for performance analysis.

A. Pre-processing time of data:

Here we are determining the time required for creating the HVTs i.e. MAC in our case. The pre-processing time does not comprise of time for loading and storing the data on the disk. Time taken by the private POS scheme is 1.71seconds and DPOS needs 3.99 seconds and other public verifiable schemes needs 1373 seconds. So the throughput we observe in experiments is 26MB/s and the private POS works with speed of 56MB/s. This delay in preprocessing time is due to the time required for encrypting the blocks. DPOS scheme is 200 times and faster than [5] and [8]. So scheme is more appropriate for the mobile devices when compared with early systems.

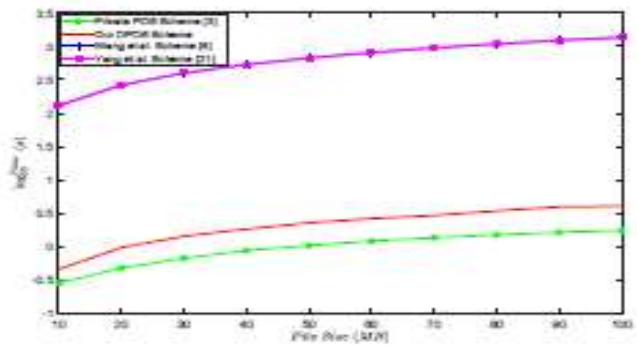

Figure 3: Pre-processing time comparison

B. Communication cost:

Cost required for communication involves the server, auditor functions like challenging and giving proof. Earlier the cost increased linearly with number of blocks, but our scheme involves cost which remains constant as we are using polynomial commitment scheme. As seen in the Figure 4 the cost for communication remains constant for DPOS scheme.

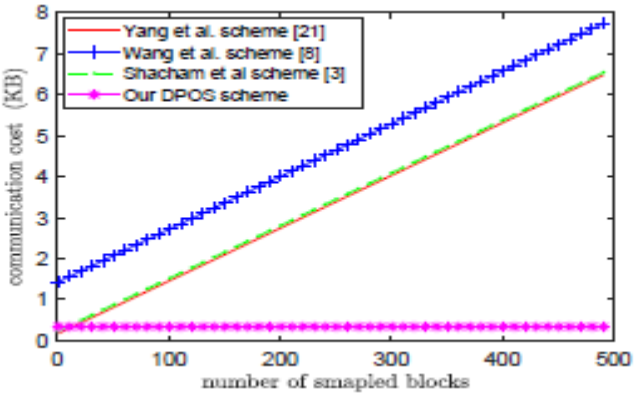

Figure 4: communication cost comparison

VII. CONCLUSION

The proposed DPOS scheme safeguards the privacy by encrypting the data. System allows the third party known as ODA to validate the integrity of the data file just like public verifiability scheme and at the same time DPOS system is as efficient as private POS scheme particularly for creating MAC. Scheme makes sure that data is not leaked to the auditor during the process of auditing. Finally, scheme supports the dynamic operation specially updating using AVL-tree.

REFERENCE

[1]Juels and J. Burton S. Kaliski, "PORs: Proofs of retrievability for large files," *in Proceedings of the 14th ACM Conference on Computer and Communications Security*, CCS 2007, pp. 584–597, ACM, 2007

[2]G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," *in Proceedings of the 14th ACM Conference on Computer and Communications Security*, CCS 2007, pp. 598–609, ACM.

[3]H. Shacham and B. Waters, "Compact proofs of retrievability," *in Advances in Cryptology* - ASIACRYPT 2008, vol. 5350 of LNCS, pp. 90–107, Springer 2008.

[4]C. Erway, A. K¨upc¸ ¨u, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," *in Proceedings of the 16th ACM Conference on Computer and Communications Security*, CCS 2009, pp. 213– 222, ACM, 2009.

[5]C. Wang, S. S. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy preserving public auditing for secure cloud storage," *IEEE Transactions on Computers*, TC 2013, vol. 62, no. 2, pp. 362–375, 2013.

[6]J. Xu, A. Yang, J. Zhou, and D. S. Wong, "Lightweight delegatable proofs of storage," *in Proceedings of 21st European Symposium on Research in Computer Security*, ESORICS 2016, pp. 324–343, Springer International Publishing, 2016.

 DOI : 10.35291/2454-9150.2019.0304

[7]I. G. Aniket Kate, Gregory M. Zaverucha, "Constant-Size Commitments to Polynomials and Their Applications," *in Advances in Cryptology* - ASIACRYPT 2010, pp. 177–194.

[8]K. Yang and X. Jia, "An efficient and secure dynamic auditing protocol for data storage in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, pp. 1717–1726, Sept 2013.

[9]S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters, "Candidate indistinguishability obfuscation and functional encryption for all circuits*," in FOCS'13–54th Annual Symposium on Foundations of Computer Science*, pp. 40–49, Oct 2013.

[10] D. Apon, Y. Huang, J. Katz, and A. J. Malozem of, "Implementing cryptographic program obfuscation." IACR Cryptology ePrint Archive, 2014.

[11] G. Adelson-Velsky and E. Landis, "An algorithm for the organization of information," *in Proceedings of the USSR Academy of Sciences, pp. 1259–1263, 1962*.

[12] Z. Hao, S. Zhong, and N. Yu, "A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability," IEEE Transactions on Knowledge and Data Engineering, TKDE 2011, vol. 23, no. 9, pp. 1432–1437, 2011.

 DOI : 10.35291/2454-9150.2019.0304