International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 15
May 2018

Study on Deduplicatable Dynamic Proof of Storage for Multi-User Environments

P.Sunil Kumar[1], K.Mamatha[2] , N.Aparna[3] , P.Lavanya Kumari[4]
(Lavanya.Madina@Gmail.Com)
[1,2,3] Student From Department Of Cse Vignan Institute Of Technology And Science
[4] Assistant Professor From Department Of Cse Vignan Institute Of Technology And Science

ABSTRACT:

Dynamic Proof of Storage (PoS) could be a helpful scientific discipline primitive that allows a user to see the integrity of outsourced files and to with efficiency update the files in a very cloud server. Though researchers have planned several dynamic PoS schemes in single user environments, the matter in multi-user environments has not been investigated sufficiently. A sensible multi-user cloud storage system wants the secure client-side cross-user deduplication technique, that permits a user to skip the uploading method and procure the possession of the files now, once alternative house owners of an equivalent files have uploaded them to the cloud server. To the simplest of our data, none of the present dynamic PoS will support this system. during this paper, we have a tendency to introduce the conception of deduplicatable dynamic proof of storage associated propose an economical construction referred to as DeyPoS, to realize dynamic PoS and secure cross-user duplication, at the same time. Considering the challenges of structure diversity and personal tag generation, we have a tendency to exploit a unique tool referred to as Homomorphic genuine Tree (HAT). We have a tendency to prove the protection of our construction, and therefore the theoretical analysis and experimental results show that our construction is economical in follow.

KEYWORDS: Deduplication, Proof of ownership, Dynamic proof of storage, Cloud Computing.

I. INTRODUCTION Users ought to be convinced that the files keep within the server don't seem to be tampered. Ancient techniques for safeguarding knowledge integrity, like message authentication codes (MACs) and digital signatures need users to transfer all of the files from the cloud server for verification that incurs a significant communication value. These techniques don't seem to be appropriate for cloud storage services wherever users could check the integrity oftentimes, like each hour. Thus, researchers introduced Proof of Storage (PoS) for checking the integrity while not downloading files from the cloud server. What is more, users may need many dynamic operations, like modification, insertion, and deletion, to update their files, whereas maintaining the potential of PoS. Dynamic PoS is projected for such dynamic operations. In distinction with PoS, dynamic PoS employ structures, like the Merkle tree. Thus, once dynamic operations are dead, users regenerate tags (which are used for integrity checking, like MACs and signatures) for the updated blocks solely, rather than create for all blocks. To rised perceive the subsequent contents. We tend to gift additional details concerning PoS and dynamic PoS. In these schemes, every block of a file is hooked up a (cryptographic) tag that is employed for substantiating the integrity of that block. Once a champion desires to ascertain the integrity of a file, it every which way selects some block indexes of the file, and sends them to the cloud server. Consistent with these challenged indexes, the cloud server returns the corresponding blocks beside their tags. The champion checks the block integrity and index correctness. The previous are often directly bonded by cryptanalytic tags. a way to affect the latter is that the major distinction between PoS and dynamic PoS In most of the PoS schemes, the block index is "encoded" into its tag, which implies the champion will check the block integrity and index correctness at the same time. However, dynamic PoS cannot cypher the block indexes into tags, since the dynamic operations could modification several indexes of non-updated blocks that incurs reserve computation

and communication value. As an example, there's a file consisting of one thousand blocks, and a replacement block is inserted behind the second block of the file. Then, 998 block indexes of the first file are modified, which implies the

user should generate and send 999 tags for this update. Structures are introduced in dynamic PoSs to unravel this challenge. As a result, the tags are hooked up to the structure instead of the block indexes .However, dynamic PoS remains to be improved in an exceedingly multi-user atmosphere, because of the necessity of cross-user American state duplication on the client-side. This means that users will skip the uploading method and acquire the possession of files now, as long because the uploaded files exist already within the cloud server. As a result, the tags area unit connected to the structure rather than the block indexes .However, dynamic PoS remains to be improved in associate extremely multi-user atmosphere, due to the requirement of cross-user American state duplication on the client-side. This suggests that users can skip the uploading methodology and acquire the possession of files currently, as long as a result of the uploaded files exists already among the cloud server. This methodology can shrink house for storing for the cloud server, and save transmission metric for users. To the only of our information, there aren't any dynamic PoS that will support secure cross-user American state duplication.

II. LITRATURE SURVEY

1.COMPACT PROOFS OF RETRIEVABILITY Authors: Hovav Shacham Description: In this paper, built from BLS signatures and secure in the random oracle model, features a proof-ofretrievability protocol in which the client's query and server's response are both extremely short. This scheme allows public verifiability: anyone can act as a verifier, not just the file owner. Our second scheme, which builds on pseudorandom functions (PRFs) and is secure in the standard model, allows only private verification. It features a proof-of-retrievability protocol with an even shorter server's response than our first scheme, but the client's query is long. Both schemes rely on homomorphic properties to aggregate a proof into one small authenticator value.

2. A Dynamic Proof of Retrievability (PoR) Scheme with O(logn) Complexity Authors: Zhen Mo, Yian Zhou, Shigang Chen Description:In this paper, Cloud storage brings security concerns. One major concern is about the data integrity. In this paper, we extend the staticPoR scheme to dynamic scenario. We propose a new authentication data structure called Cloud Merkle B+ tree (CMBT). Compared with the existing dynamic PoR scheme, our worst case communication complexity is O(logn) instead of O(n).

3. Practical Dynamic Proofs of Retrievability Authors: Elaine Shi, Emil Stefano, Charalampos Papamanthou Description: In this paper,We propose a dynamic PoR scheme with constant client storage whose bandwidth cost is comparable to a Merkle hash tree, thus being very practical. Our construction outperforms the constructions of Stefanov et al. and Cash et al.both in theory and in practice. Specifically, for n outsourced blocks of bits each, writing a block requires +O(log n)bandwidth and O(log n) server computation (is the securityparameter). Audits are also very efficient, requiring + O(_2 log n) bandwidth. We also show how to make our scheme publicly verifiable, providing the first dynamic PoR scheme with such a property. We finally provide a very efficient implementation of our scheme.

4. Proofs of Ownership in Remote Storage Systems Authors: Shai Halevi, Danny Harnik, Benny Pinkas Description: In this work we put forward the notion of proof-of-ownership, by which a client can prove to a server that it has a copy of a file without actually sending it. This allows to counter attacks on file-deduplication systems where the attacker obtains a "short summary" of the file and uses it to fool the server into thinking that the attacker owns the entire file. 5. Dynamic Proofs of Retrievability for Coded Cloud Storage Systems Authors: Zhengwei Ren, Lina Wang, Qian Wang, Mingdi Xu Description: In

this paper, we proposed a new dynamic proof of retrievability scheme for coded cloud storage systems. Network coding and erasure codes are adopted to encode data blocks to achieve within-server and cross-server data redundancy, tolerating data corruptions and supporting communicationefficient data recovery. By using rb23Tree and an improved version of ASBB scheme our construction can support efficient data dynamics while defending against data replay attack and pollution attack. Security analysis and experimental evaluations demonstrated the practicality of our construction in coded cloud storage systems.

III. PROPOSED SYSTEM A. SYSTEM MODEL

The entire document should be in Times New Roman or Times font. Type 3 fonts must not be used. Other font types may be used if needed for special purposes. As shown in Fig. 1. for every file, original user is that the user World Health Organization uploaded the file to the cloud server, whereas ulterior user is that the user World Health Organization established the possession of the file however didn't truly transfer the file to the cloud server. There square measure 5 phases during a deduplicatable dynamic PoS system: pre-process, upload, deduplication, update, and proof of storage. B. PRE-PROCESS PART Users will transfer their native files. The cloud server decides whether or not these files ought to be uploaded. If the transfer method is granted, enter the transfer phase; otherwise, enter the deduplication part C. UPDATING Users will transfer their native files. The cloud server decides whether or not these files ought to be uploaded. If the transfer method is granted, enter the transfer phase; otherwise, enter the deduplication part.

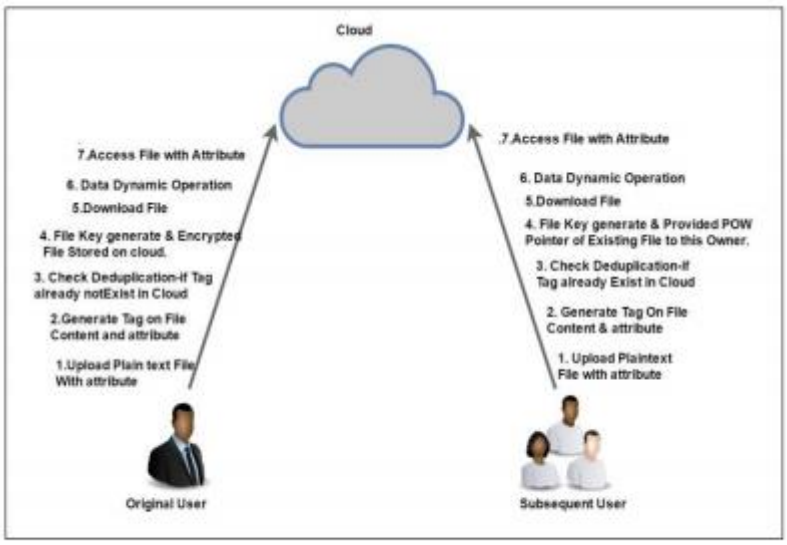

Fig 1: System Architecture

D. DEDUPLICATION CHECK The files to be uploaded exist already within the cloud server. the next users possess the files domestically and also the cloud server stores the structures of the files. ulterior users got to persuade the cloud server that they own the files while not uploading them to the cloud server. If these 3 phases (pre-process, upload, and deduplication) square measure dead just one occasion within the life cycle of a file from the angle of users. That is, these 3 phases seem only if users will transfer files. If these phases terminate unremarkably, i.e., users end transfering within the upload part, or they pass the verification within the deduplication part, we are saying that the users have the ownerships of the files.

E. USERS PROOF OF STORGE Users solely possess a little constant size information domestically and that they need to examine whether or not the files square measure dependably hold on

within the cloud server while not downloading them. The files might not be uploaded by these users however they pass the deduplication part and prove that they need the ownerships of the files. Note that, the update part and also the proof of storage part will be dead multiple times within the life cycle of a file. Once the possession is verified, the users will randomly enter the update part and also the proof of storage part while not keeping the first files domestically.

IV. CALCULATION

1]User Module:- - New User - Give Attributes or Privilege When User register e. g. Student or Staff etc. -User login in system -user Upload file in system. -User select privilege or attribute first e.g. student or staff -Browse Text File to Upload and click on Upload button and generates tag file for it. - If tag exist in server database then file is deduplicated & print message - file already exist, then give proof of ownership pointer to this user of existing file for accessing & this user is also owner of that existing file. -If tag not exist in server database then file is unique then encrypt file and stored on cloud folder in drive. -User also can download file from cloud. -user shows all file that his own uploaded i.e. unique file & deduplicated file - click on download link to download that file 2] Access File -user shows all files for his attribute uploaded by owner of file. -click on download link to download that file 3] Subsequent User This user are those user who upload files on cloud and if file they upload on cloud is duplicate or already existing on cloud then they become subsequent user of file. They get ownership over that file and they can access that file. V.

CONCLUSION We planned the great necessities in multi-user cloud storage systems and introduced the model of deduplicatable dynamic PoS. we had develop a unique tool known as HAT that is Associate in Nursing economical genuine structure. Supported HAT, we had planned the primary sensible deduplicatable dynamic PoS theme known as DeyPoS and evidenced its security within the random oracle model. The theoretical and experimental results show that our DeyPoS implementation is economical, particularly once the file size and therefore the range of the challenged blocks area unit giant

REFRENCES

[1] H. Shacham and B. Waters, “Compact Proofs of Retrievability,”Journal of Cryptology, vol. 26, no. 3, pp. 442–483, 2013.

[2] Z. Mo, Y. Zhou, and S. Chen, “A dynamic proof of retrievability (PoR) scheme with o(logn) complexity,” in Proc. of ICC, pp. 912–916, 2012.

[3] E. Shi, E. Stefanov, and C. Papamanthou, “Practical dynamic proofs of retrievability,” in Proc. of CCS, pp. 325–336, 2013.

[4] C. Erway, A. K¨upc ¨u, C. Papamanthou, and R. Tamassia, “Dynamic provable data possession,” in Proc. Of CCS, pp. 213–222, 2009.

[5] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg, “Proofs of ownership in remote storage systems,” in Proc. of CCS, pp. 491–500, 2011.

[6] Z. Ren, L. Wang, Q. Wang, and M. Xu, “Dynamic Proofs of Retrievabilityfor Coded Cloud Storage Systems,” IEEE Transactionson Services Computing, vol. PP, no. 99, pp. 1–1, 2015.

[7] R. Tamassia, “Authenticated Data Structures,” in Proc. of ESA,pp. 2–5, 2003.

[8] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, “Enabling public verifiability and data dynamics for storage security in cloudcomputing,” in Proc. of ESORICS, pp. 355–370, 2009.

[9] F. Armknecht, J.-M. Bohli, G. O. Karame, Z. Liu, and C. A. Reuter, “Outsourced proofs of retrievability,” in Proc. of CCS, pp. 831– 843,2014.

[10] J. Douceur, A. Adya, W. Bolosky, P. Simon, and M. Theimer,“Reclaiming space from duplicate

files in a serverless distributedfile system," in Proc. of ICDCS, pp. 617–624, 2002.

[11] A. Juels and B. S. Kaliski, Jr., "PORs: Proofs of retrievability forlarge files," in Proc. of CCS, pp. 584–597, 2007.

[12] H. Shacham and B. Waters, "Compact proofs of retrievability," inProc. of ASIACRYPT, pp. 90–107, 2008.

[13] Y. Dodis, S. Vadhan, and D. Wichs, "Proofs of retrievability via hardness amplification," in Proc. of TCC, pp. 109–127, 2009.

[14] K. D. Bowers, A. Juels, and A. Oprea, "HAIL: A high-availabilityand integrity layer for cloud storage," in Proc. of CCS, pp. 187 198, 2009.