|| Volume 5 || Issue 9 || September 2020 || ISSN (Online) 2456-0774
INTERNATIONAL JOURNAL OF ADVANCE SCIENTIFIC RESEARCH
AND ENGINEERING TRENDS

MULTI-KEYWORD SEARCH OVER PROOF OF STORAGE FOR ENCRYPTED CLOUD DATA

Nilesh Pinjarkar[1], Prof. Rahul Gaikwad[2]

Department of Computer Engineering, Godavari College of Engineering, Jalgaon [1,2]

nilesh.pinjarkar81@gmail.com

***Abstract*— Cloud computing has become a well-known way to deal with oversee individual information for the financial investment funds and the board adaptability in ongoing year. Notwithstanding, the delicate information must be scrambled previously moving operations to cloud workers for the thought of security, which makes some customary information usage capacities, for example, the plaintext keyword search, incomprehensible. To tackle this issue, we present a multi-keyword ranked search conspire over scrambled cloud information supporting dynamic activities productively. Our plan uses the vector space model with cosine likeness measure to accomplish a multi-keyword ranked search. In any case, conventional arrangements need to endure high computational costs. Our plan can uphold dynamic activity appropriately and successfully, which implies that the refreshing expense of our plan is lower than other plans. We present our essential plan first, which is secure under the known ciphertext model. At that point, the improved plan is introduced later to ensure security much under the realized foundation model. The probes this present reality informational index show that the exhibitions of our proposed plans are palatable.**

Keywords: *Cloud computing, dynamic searchable encryption, multi-keyword ranked search.*

I INTRODUCTION

Cloud computing has been generally acknowledged and deployed in our day by day life because of the incredible benefits that it brings about, for example, decreasing infrastructure costs, providing high versatility and accessibility. More and more people depend on cloud storage administrations to lessen their local storage burden. Namely, data is outsourced to the cloud server and can be gotten to on demand later. Meanwhile, how to ensure the security and integrity of the outsourced data without keeping a local copy for data owners is an basic concern to address. One of the main solutions is to apply proofs of storage (POS) that is also alluded to proofs of retrievability (POR) [1] or proofs of data possession (PDP) [2], in which the integrity of data stored in cloud server can be confirmed without having to download every one of the data. The essential thought is dividing the whole data document into numerous blocks, each of which is utilized to generate a homomorphic verifiable tag (HVT) sent to the cloud server together with the data document. Afterward, the verifier chooses a lot of data blocks instead of the whole document to review the outsourced data from the cloud server (prover) with the assistance of those HVTs, which can significantly decrease the communication overheads.

Since the first POR and PDP plans are presented in 2007, there have been lots of efforts devoted to constructing proofs of storage plans with more advanced highlights such as open key unquestionable status [3], data dynamics [4], [5] (for example modifying/inserting/deleting data blocks), various cloud servers [6] and data sharing [7]. We focus on the initial two highlights—open unquestionable status and support of data dynamics. With regards to the former, we observe that most of existing freely verifiable POS plans employ expensive operations (for example group exponentiation) to generate HVTs for data blocks. Consequently, it is prohibitively expensive to generate HVTs for medium or huge size data records. For instance, one of the most popular POS plans, proposed by Wang et al. [8], accomplishes throughput of data pre-processing at speed 17.2KB/s with an Intel Core 2 1.86 GHz workstation CPU, which means it will take about 17 hours to generate HVTs for a 1GB document. Even if the client has a CPU with 8 cores, despite everything it requires more than 2 hours' substantial computation. Such amount of substantial computation isn't appropriate for a laptop, not to mention tablet computer or advanced

mobile phone. Open evidence of POS enables any outsider to confirm the integrity of data in cloud storage, which significantly eliminates the burden from data owner. Nevertheless, in practice, it isn't attractive to allow anyone to review the data at their will, and instead, delegation of the auditing task must be in a controlled and organized manner. Otherwise, the following two extraordinary cases may happen: (1) some data documents could draw in too much attention from open, and are evaluated unnecessarily too frequently by the general population, which may really bring about appropriated denial of administration assault against the cloud storage server; (2) on the contrary, some unpopular data documents might be reviewed by the open too infrequently, so that the possible data loss event may be distinguished and
alarmed to the data owner too late and no compelling countermeasure can be done to decrease the harm. Instead, the data owner could assign the auditing undertaking to some semi-trusted outsider auditor, and this auditor is completely responsible to review the data stored in cloud storage for the data owner, in a controlled way, with proper frequency. We call such a select auditor as Owner-Delegated-Auditor or ODA for short. In certifiable applications, ODA could be another server that provides free or paid auditing administration to many cloud clients.

The second element we think about in a POS plot is supporting dynamic operations, in which data owners may demand to modify, insert, or erase data blocks in the wake of outsourcing its original data to a cloud server. This is an ideal property when designing new POS plans. Upon generating a HVT, the block index information I must be a piece of the inputs. This is to prevent a cloud server using the equivalent HVT for different blocks while still passing the verification. As a consequence, if a new block $m*$ is inserted after the I-th block mi , then the indices of all the following blocks after mi must be changed accordingly and all the corresponding HVTs need to be recomputed, which is unreasonable if the number of blocks is gigantic (for a 1GB document, in the event that we set one data block be 4KB, then the number of data block is about $2^{18} \approx 1$ million). To avoid this problem, we can manage the indices instead of the HVTs upon data updating. There have been a number of scientists working on techniques supporting dynamics for POS plans, which results into two main classes of solutions: index table-based and tree-based methods. The former employs an index table to manage the block indices which significantly diminishes the communication cost however taking O(n) computation for every datum update, while the last one uses tree-based structures, for example, the Merkle Hash Tree and the rank-based authenticated skip list that need only O(log n) computation however brings in extra O(log n) communication overhead for block auditing, where n is the number of data blocks.

II LITERATURE SURVEY

Since the first POR and PDP plans are presented in 2007, there have been lots of efforts devoted to constructing proofs of storage plans with more advanced highlights such as open key unquestionable status [3], data dynamics [4], [5] (for example modifying/inserting/deleting data blocks), various cloud servers [6] and data sharing [7]. We focus on the initial two highlights—open unquestionable status and support of data dynamics. With regards to the former, we observe that most of existing freely verifiable POS plans employ expensive operations (for example group exponentiation) to generate HVTs for data blocks. Consequently, it is prohibitively expensive to generate HVTs for medium or huge size data records. For instance, one of the most popular POS plans, proposed by Wang et al. [8], accomplishes throughput of data pre-processing at speed 17.2KB/s with an Intel Core 2 1.86 GHz workstation CPU, which means it will take about 17 hours to generate HVTs for a 1GB document. Even if the client has a CPU with 8 cores, despite everything it requires more than 2 hours' substantial computation. Such amount of substantial computation isn't appropriate for a laptop, not to mention tablet computer or advanced mobile phone. Open evidence of POS enables any outsider to confirm the integrity of data in cloud storage, which significantly eliminates the burden from data owner. Nevertheless, in practice, it isn't attractive to allow anyone to review the data at their will, and instead, delegation of the auditing task must be in a controlled and organized manner. Otherwise, the following two extraordinary cases may happen: (1) some data documents could draw in too much attention from open, and are evaluated unnecessarily too frequently by the general population, which may really bring about appropriated denial of administration assault against the cloud storage server; (2) on the contrary, some unpopular data documents might be reviewed by the open too

infrequently, so that the possible data loss event may be distinguished and alarmed to the data owner too late and no compelling countermeasure can be done to decrease the harm. Instead, the data owner could assign the auditing undertaking to some semi-trusted outsider auditor, and this auditor is completely responsible to review the data stored in cloud storage for the data owner, in a controlled way, with proper frequency. We call such a select auditor as Owner-Delegated-Auditor or ODA for short.[8] In certifiable applications, ODA could be another server that provides free or paid auditing administration to many cloud clients.

The second element we think about in a POS plot is supporting dynamic operations, in which data owners may demand to modify, insert, or erase data blocks in the wake of outsourcing its original data to a cloud server. This is an ideal property when designing new POS plans. Upon generating a HVT, the block index information I must be a piece of the inputs. This is to prevent a cloud server using the equivalent HVT for different blocks while still passing the verification. As a consequence, if a new block m∗ is inserted after the I-th block mi , then the indices of all the following blocks after mi must be changed accordingly and all the corresponding HVTs need to be recomputed, which is unreasonable if the number of blocks is gigantic (for a 1GB document, in the event that we set one data block be 4KB, then the number of data block is about 2 18 ≈ 1 million). To avoid this problem, we can manage the indices instead of the HVTs upon data updating. There have been a number of scientists working on techniques supporting dynamics for POS plans, which results into two main classes of solutions: index table-based and tree-based methods. The former employs an index table to manage the block indices which significantly diminishes the communication cost however taking O(n) computation for every datum update, while the last one uses tree-based structures, for example, the Merkle Hash Tree and the rank-based authenticated skip list that need only O(log n) computation however brings in extra O(log n) communication overhead for block auditing, where n is the number of data blocks.

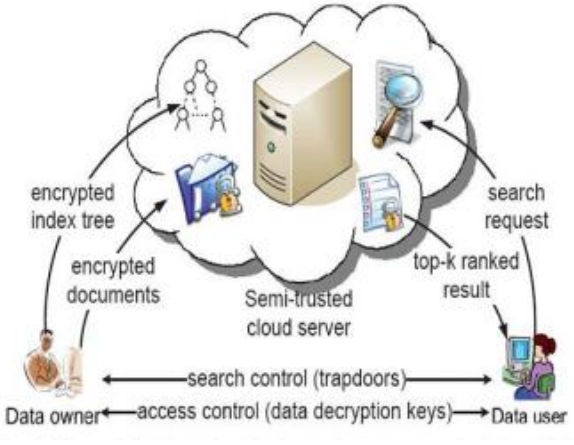
III PROPOSED METHODOLOGY

Arrangement stage The data owner runs the key generating algorithm KeyGen(1λ) for only once, to generate the ace key pair (pk, sk) and the verification key pair (vpk, vsk). For each input data document, the data owner may choose to apply some error deletion code [46] on this document, and runs the tagging algorithm Tag over the (deletion encoded) record, to generate authentication tags {(σi , ti)} and document parameter ParamF . Toward the end of arrangement stage, the data owner sends the (deletion encoded) record F, all authentication tags {(σi , ti)}, document parameter ParamF , and open keys (pk, vpk) to the cloud storage server.

The data owner also chooses a restrictive outsider auditor, called Owner-Delegated-Auditor (ODA, for short), and delegates the verification key pair (vpk, vsk) and record parameter ParamF to the ODA. From that point forward, the data owner may keep only keys (pk, sk, vpk, vsk) and record parameter ParamF in local storage, and erase everything else from local storage.

Proof stage The proof stage consists of numerous proof sessions. In each proof session, the ODA, who runs algorithm V, interacts with the cloud storage server, who runs algorithm P, to review the integrity of data owner's record, in the interest of the data owner. Therefore, ODA is also called verifier and cloud storage server is also called prover.[20]

Revoke stage In the revoke stage, the data owner downloads all tags {ti} from cloud storage server, revokes the current verification key pair, and generates a new verification key pair and new tags . The data owner also chooses a new ODA, and representatives the new



We plan to protect data integrity and security of data owner's document. The data owner is completely trusted, and the cloud storage server and ODA are semi-confided in different sense: (1) The cloud storage server is confided in data security (We expect the server needs to get to plaintext to provide additional administrations to the data owner), and isn't trusted in maintaining data integrity (for

example the server may erase some infrequently gotten to data for economic benefits, or cover up the data corruption events brought about by server disappointments or assaults to maintain reputation). (2) Before he/she is revoked, the ODA is confided in performing the appointed auditing task and protecting his/her verification mystery key safely, however isn't confided in data security. A revoked ODA could be potentially malicious and might surrender his/her verification mystery key to the cloud storage server. We accept that all communication among the data owner, the cloud storage server and ODA is by means of some protected channel (for example channel security and integrity are protected). Framing assault among these three gatherings can be managed existing technique and is out of scope of this paper.

IV ALGORITHM

A Delegatable Proofs of Storage (DPOS) scheme consists of three algorithms (KeyGen, Tag, UpdVK)

1.KeyGen : Generate the public master keys and public verification keys.

2. Tag : Provide unique file identifier authentication tags.

3. UpdVK : This update the public master key and public verification key with new authentication tags.

B) AES

Advance Encryption standard are used for symetric encryption.It Encrypt data of DataOwner . this secret key is send by DataOwner to DataUser securely.

c) SHA

1) DataOwner send Encrypted data to Owner Delegated Auditor.

2) Owner Delegated Auditor Take Outsource Encrypted data from cloud

3) Take both the data Hash value

4) If Hash value equal then data same

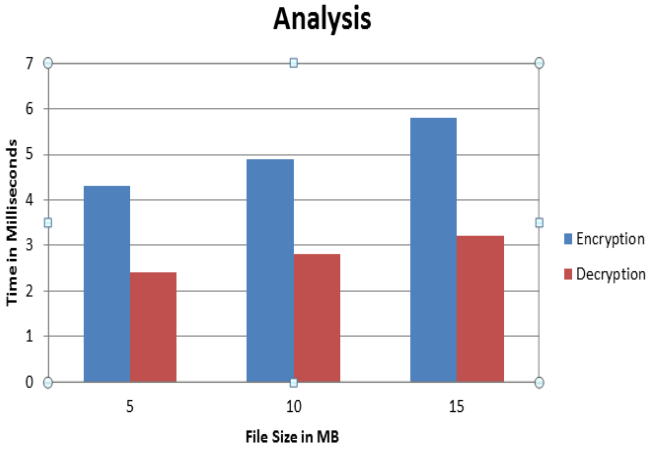
5) If value are not equal then data are change

6) DO update or change its Encrypted data.

V RESULT AND ANALYSIS

In evaluation, here consider two factors, Time required for encryption and time required for decryption. Also, be consider size of the file. Different size of data, there is varying time I encryption and decryption as shown in the table. As compare to small size data files, single data which have large size required less time. In this system Symmetric cryptography use both the side same secret key. Time Complexity is less in this system.

File Size(MB)	Encryption	Decryption
5	4.3	2.4
10	4.9	2.8
15	5.8	3.2



VI CONCLUSIONS

We proposed a novel POS conspire which is lightweight and protection preserving. On one side, the proposed plan is as efficient as private key POS plot, particularly very efficient in authentication tag generation. On the other side, the proposed plan supports outsider auditor and can revoke an auditor whenever, close to the functionality of freely verifiable POS conspire. Compared to existing freely verifiable POS plots, ours improves the authentication tag generation speed by hundreds of times. Our conspire also prevents data spillage to the auditor during the auditing process. Finally, we designed a new AVL-tree based completely dynamic mechanism for our POS plot. The experimental outcomes confirmed the performance efficiency of our plan.

ACKNOWLEGMENT

I would like to thank my project guide ”Prof. Rahul Gaikwad” who always being with presence and constant, constructive criticism to made this paper. I would also like to thank all the staff of Computer Department for their valuable guidance, suggestions and support through the paper work, who has given co-operation for the project with personal attention. At the last I thankful to my friends, colleagues for the inspirational help provided to me through a paper work.

REFERENCES

[1] A. 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. Kupc¸ ¨ 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. C. Erway, A. Kupc¸ ¨ u, C. Papamanthou, and R. Tamassia, "Dy- ¨ namic provable data possession," ACM Transactions on Information and System Security, vol. 17, pp. 15:1–15:29, April 2015.

[6] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, "MR-PDP: Multiple-replica provable data possession," in Proceedings of the 28th International Conference on Distributed Computing Systems, ICDCS 2008, pp. 411–420, IEEE, 2008.

[7] B. Wang, B. Li, and H. Li, "Oruta: Privacy-preserving public auditing for shared data in the cloud," in Proceedings of 5th International Conference on Cloud Computing, Cloud 2012, pp. 295–302, IEEE, 2012.

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

[9] 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.

[10] G. Ateniese, S. Kamara, and J. Katz, "Proofs of storage from homomorphic identification protocols," in Advances in Cryptology - ASIACRYPT 2009, vol. 5912 of LNCS, pp. 319–333, Springer, 2009.

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

[12] T. Okamoto, "Provably secure and practical identification schemes and corresponding signature schemes," in CRYPTO '92: Annual International Cryptology Conference on Advances in Cryptology, pp. 31– 53.

[13] J. Alwen, Y. Dodis, and D. Wichs, "Leakage-Resilient Public-Key Cryptography in the Bounded-Retrieval Model," in CRYPTO '09: Annual International Cryptology Conference on Advances in Cryptology, pp. 36–54, 2009.

[14] G. Ateniese, R. Burns, R. Curtmola, J. Herring, O. Khan, L. Kissner, Z. Peterson, and D. Song, "Remote data checking using provable data possession," ACM Transaction on Information and System Security, TISSEC 2011, vol. 14, no. 1, pp. 12:1–12:34, 2011.

[15] H. Shacham and B. Waters, "Compact proofs of retrievability," Journal of Cryptology, JOC 2013, vol. 26, no. 3, pp. 442–483, 2013.

[16] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in The 14th European Symposium on Research in Computer Security, ESORICS 2009, vol. 5789 of LNCS, pp. 355–370, Springer, 2009.

[17] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," IEEE Transactions on Parallel and Distributed Systems, TPDS 2011, vol. 22, no. 5, pp. 847–859, 2011.

[18] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security

in cloud computing," in Proceedings of the 29th Conference on Computer Communications, INFOCOM 2010, pp. 525–533, IEEE, 2010.

[19] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Dynamic audit services for integrity verification of outsourced storages in clouds," in Proceedings of the 2011 ACM Symposium on Applied Computing, SAC 2011, pp. 1550–1557, ACM, 2011.

[20] 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.