Ensuring Data Integrity in Cloud Storage: A Verifiable Approach with Outsourced Audits and Enhanced Performance

[1]K Vivek Reddy and [2]C. Shoba Bindu
[1]Research Scholar, Department of CSE, JNTU Anantapur, Anantapuramu, A.P, India
[2]Professor, Department of CSE, JNTU Anantapur, Anantapuramu, A.P, India
vivekjntuphd@gmail.com,shobhabindhu@gmail.com

Abstract: The utilization of cloud storage has garnered significant attention from diverse users. Nevertheless, the separation of data management and ownership introduces potential security challenges, making it difficult to uphold data integrity. This study endeavours to conduct a thorough examination of the security issues inherent in cloud storage. Alongside establishing the verifiability of data updates, we present a novel method for conducting data integrity audits through outsourcing. This proposed approach caters to the requisites of both data confidentiality and verifiable data updates, eliminating the dependence on third-party entities. The escalating quantity of subfiles in cloud storage has a logarithmic impact on computational complexity. Subsequently, we have developed a prototype implementation that will undergo testing to assess its performance and underscore the advantages of our proposed methodology. The experiments conducted on this approach revealed its superior practicality and efficiency compared to preceding solutions.

Keywords: Data Update, Data Auditing, Integrity Verification, Cloud data center.

1. Introduction

The swift advancement of networks and computers has triggered an exponential surge in digital data volume. As per an investigative report, the average stored digital data per person in 2020 reached approximately 5200 GB. However, resource limitations pose a challenge for tenants attempting to retain such extensive data volumes [1].Managing vast amounts of data becomes an arduous task for tenants with constrained resources. Fortunately, cloud storage offers a solution to this predicament. Through its services, tenants can effortlessly store and manage data, thereby reducing computational costs and conserving memory space [2].

Cloud storage's numerous advantages have led many tenants to opt for this solution. A Cisco report highlighted that, in 2020, there were over 3.6 billion consumers of Internet connections [3], with approximately 55% of Internet users adopting cloud storage.vCloud storage empowers tenants to oversee their outsourced data without physical possession, eliminating the need for hands-on control and restricting certain operations over the data.

However, the reliability of cloud storage services remains a concern. Apart from occasional failures to execute commands as instructed by tenants, providers grapple with issues pertaining to data integrity and confidentiality. These challenges may hinder public trust and utilization of these services [4].

Numerous solutions [5-9] have been proposed to ensure the availability and integrity of outsourced data, yet certain unresolved issues persist. One such challenge involves the utilization of Proof of Recoverable (PoR) or Verifiable Data (PDP) technology. The complexity of data contributes to a linear growth in computing complexity as the number of

outsourced files increases, negatively impacting solution efficiency. The escalating count of outsourced files further compounds the efficiency dilemma. Additionally, the security of outsourced data remains a pressing concern. Many previous works entrust security to a third party, introducing an element of risk due to the third party's potentially precarious reputation.

The vulnerability of third parties to adversarial attacks raises the specter of private information leakage. Despite the array of proposed solutions, the lack of universality and practicality hinders their adoption in large-scale outsourcing projects. Overcoming these challenges is crucial for the broader implementation of such solutions in practical scenarios [10].

Upon scrutinizing existing solutions, it becomes evident that lingering issues persist within them. Furthermore, the dearth of effective solutions capable of simultaneously addressing data integrity and dynamic updates without dependency on a third party is conspicuous. This paper endeavors to fill this void by proposing a novel solution.The proposed approach advocates the utilization of the Merkle sum hash tree as the foundational framework for crafting an efficient data integrity auditing system. Notably, this method offers the capability to manage updates autonomously, eliminating the necessity for reliance on a third party. Through this innovative approach, the paper aims to provide a robust solution to the challenges identified in prior methodologies.

2. Literature Survey

Over the years, numerous studies on data integrity auditing have been conducted in both industrial and academic sectors. In [11], the authors introduced a prototype for a data integrity auditing model employing random sampling. Subsequently, they developed two PDP schemes that were more efficient and secure. The authors [12] implemented a remote data possession check, incorporated algebraic signatures into her design. While her scheme exhibited improved communication overhead, it did not effectively prevent replay attacks. In a separate study, In [13], the authors introduced a remote cloud data integrity auditing system adept at thwarting replay attacks and deletion attacks. The researchers delved into the challenge of maintaining multiple copies of data within a cloud data center.

In [14], the authors proposed a new data integrity auditing model, ID-DPDP, capable of simultaneously checking all copies of a data file. Additionally, they employed bilinear pairings to construct a framework for multicloud storage. In [15], to enhance efficiency, the authors implemented a PDP protocol utilizing a hash function.

In [16], the authors presented a novel security protocol, named SIBAS, designed to safeguard the integrity of a data file over cloud data. This protocol successfully implemented a reliable key management system through Shamir's threshold protocol. In [17], the authors introduced a concept enabling users to remove the hardware token without storing private keys.

In [18], the authors utilized hyperledger fabric to devise a secure data integrity protocol. In a separate study, In [19], the authors proposed a blockchain-based verification system for public validation. The researchers highlighted the system's capability to record verification process results into time-sensitive transactions.

In [20], the authors developed a certificate-based auditing protocol that could be proven using the random oracle model. Despite its features, their scheme necessitated a

constant computational overhead to generate a validation tag for a block, and they were unable to implement a robust data dynamic auditing system.

In order to achieve data dynamic update and data integrity auditing, In [21], the authors introduced a new dynamic PDP protocol. This method used hash values and tags to guarantee the integrity of the data. A remote DPDP scheme was then developed in [22], which can be used for maintaining the integrity of the data while keeping the client's storage overhead low.

In [23], the authors presented a prototype of a dynamic cloud data platform that can support full-scale data dynamics. In order to improve its efficiency, In [24], the authors used a flexlist framework to design a DPDP scheme that can provide multiple updates at the same time. In [25], the authors studied the issue of data integrity auditing in cloud data centers. They came up with a solution that allows them to maintain the integrity of their data while keeping the cost of storage low.

The researchers [26] replaced the modular design of the protocol with vector dot products to improve its efficiency. In [27], the authors introduced a dynamic data auditing framework that can be used to perform fully dynamic operations.

In [28], the authors proposed a dynamic proof of data replication and possession protocol that uses an IBMT balanced tree. The researchers noted that their proposed solution could save a lot of resources, but it required the introduction of a TPA to fully verify the integrity of the data in the cloud.

3. Preliminaries

Merkle sum hash tree is implemented by [29], which is originated from the merkle hash tree [30]. The Merkle sum hash tree is used verify the integrity of the any subset in the group which is same as merkle hash tree. Although each leaf of MSHT can manage various data blocks, it only saves one at a time. Figure 1 shows the data blocks which are managed under the lead is given as input to the hash value. In Figure 1, R_n represents the root node which contains the all the data blocks, S_R represents the signature which is generated using the secure signature protocol to the hash value R_n.

In MSHT, the data integrity auditing is achieved with the information of auxiliary validation φ. The φ contains the information of the child nodes on the path and data blocks hash values [31]. For instance, to verify the data integrity of the $f_{3,1}$, the φ_3 is given as ((r_{14}, 1), (r_{21}, 2)). Then, the new R_n' is computed by the verifier by utilizing the φ_3. R_n and R_n' is compared, if R_n' is equal to the R_n, then the verifier believes that $f_{3,1}$is not tampered.

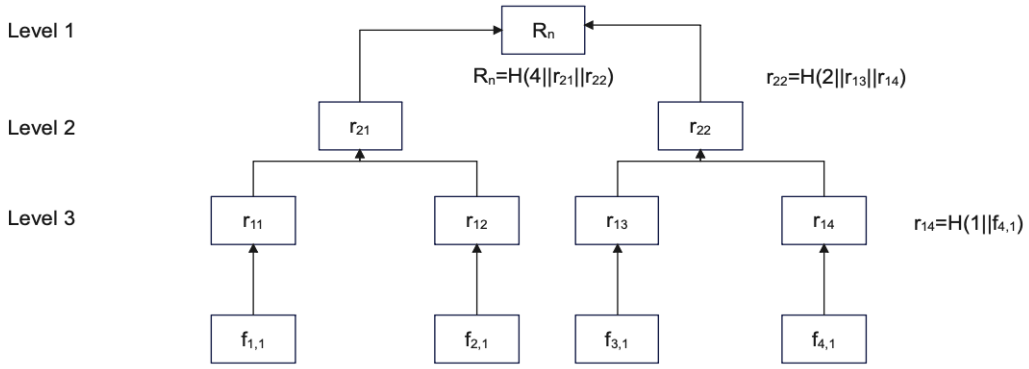

Figure 1: Structure of Merkle Sum Hash Tree

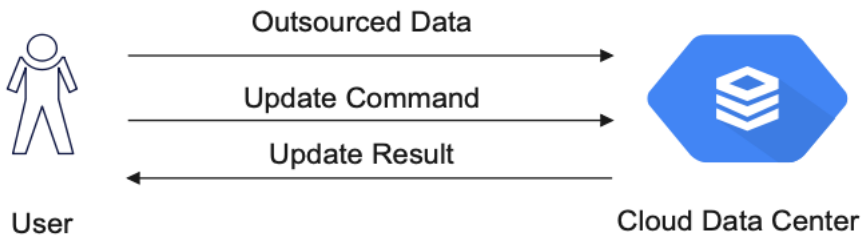

Figure 2: Structure of the Proposed Scheme

4. Problem Statements

4.1 Proposed Structure:This paper aims to analyse the issue of high efficiency when it comes to supporting reliable data updates in a cloud storage system. The system framework used is composed of two participants: a cloud data center and a trusted authority which is shown in Figure 2.

Users: Due to the limited storage capacities of the user, he or she can't manage and save massive amounts of data on their own. Instead, they prefer to use the cloud to store their data. In the future, the User plans to use a few new data blocks to update the existing outsourced data set.The user would like to make sure that the data collected and stored in the cloud is secure. This can be done through an audit of the data update process.

Cloud Data Center: The cloud data center utilizes the Internet to connect various devices and resources, such as computational devices and disks, and establish a resource pool that enables the tenant to utilize efficient storage services. Since the outsourcing data is handled by the

cloud data center, it performs updates according to the user's commands. It also returns relevant proofs to show that the data has been updated.

4.2 Security challenges:

The following security challenges need to be solved in the proposed model.

Data Corruption: One of the main reasons why outsourcing data is prone to contamination is due to various factors: Data loss can be caused by various factors, such as hardware failure, software failure, and the cloud manager's erroneous operations. Moreover, the data center might be removing some of the outsourced data blocks to save on storage overhead. A hacker could also target the data center and cause damage.

Privacy Leakage: In most cases, the data collected from outsourced sources contains the personal information of a user. When the cloud storage manager discovers that there is something about the user's data that is private, they might be curious to see what it is. On the other hand, an attacker might be able to access the data to dig a deeper understanding of the user's privacy. This is a severe issue that should be considered.

Improper Data Update:To update the outsourced data, a cloud data center must perform various calculations and protocols. This process can involve running expensive computational resources and adding more storage capacity. The cloud data center might not be able to perform the user's command properly.

5. Proposed Method

The goal of this paper is to create a high-efficiency auditing scheme that can guarantee the confidentiality and integrity of data stored in cloud storage. The following phases explains the procedure of the proposed method.

5.1 Initialization

This phase involves the user registration with public and private key generations.

User Registration: Before a user can use the cloud data center's storage services, they have to become a real-world customer of the cloud providers. This process involves completing the registration and authentication process with the cloud. Once the user has become a valid customer, they can then use the storage services.

Key Generation: The cloud generates the public and private key (K_{pc}, K_{sc}) using the elliptic curve digital signature algorithm. The cloud distributes the public key K_{pc} and holds the private key K_{sc}. In the same way, user computes the public key and private key (K_{pu}, K_{sc}). The user shares the public key K_{pu} and holds the private key K_{sc}. The user also selects the identifier I_f for the file F.

5.2 Data Pre-processing:

The data pre-processing includes the encryption of the data along with the segmentation of the ciphertext.

Data Encryption: In general, the data which is outsourced contains the privacy data. Hence the user which is uploading the data in to the cloud needs to encrypt the data. To clearly understand, the user calculates key $K= H(ID||I_f||K_{st})$, where H(.) represents the secure has function. In the next step,the user performs the encryption $c=enc_k(C)$.

Segmentation of Ciphertext: the user divide the cipher text c in to number of subfiles $c= (c_1, c_2 \ldots c_n)$. In the next step, each subfile is divided in to s number of data blocks $c_{i1},c_{i2} \ldots c_{is}$, where i-1,2…n. As a result, the outsourced data obtained by the user is given as,

$$c = (c_1, c_2 \ldots . c_n = \{c_{i,j}\} \; where \; 1 \leq i \leq n, 1 \leq j \leq s$$

5.3 Data Outsourcing

In this module, The MSHT structure is used for outsourced data set and after construction of MSHT, the complete tree is outsourced to the cloud storge.

Tree construction: The user construct the MSHT with the help of data blocks which was received. This MSHT contains the leaf nodes $N_1,N_2 \ldots N_n$. Each lead node saves the respective subfile f_i. Along with that, the user gets the root node R_n and computes the signature S_R, where $S_R=S_{Ksu}(R_n)$. Then the user uploads complete MSHT to the cloud along with S_R.

File Storage: The cloud will audit the correctness of the MSHT, once the S_R,MSHT and outsource data set f are received form the user. The cloud reconstructs the MSHT by utilizing the f and constructs the new hash value R_n'. Then, the cloud verifies the correctness of the S_R by comparing the R_n and R_n'. Finally the cloud returns the success message to the user as a result of the data storage.

Data Deletion: After the cloud successfully stores the outsourced file f, the user deletes all the copies of the file and the data set to save the local storage capacity. The user stores the root R_n in the local disks.

5.4 Integrity Auditing

When a user transfers their outsourced data to the cloud, they must regularly perform data integrity audits to make sure that the information is secure. The details of this process are outlined below.

Information retrieval : The user randomly choose an integer i where $1 \leq i \leq n$. Later the user retrieves the file $f_i=(f_{i,1},f_{i,2} \ldots f_{i,s})$ and its associated auxiliary data φ_i from the cloud.

Rebuilding of Merkle Root: The user utilizes the φ_i and f_i to rebuild the root. Later, the user construct the new hash value R_n^*markle root which is different from the R_n. If $R_n=R_n^*$ holds then the user trust that file $f_i=(f_{i,1},f_{i,2} \ldots f_{i,s})$ is not corrupted. If $R_n=R_n^*$ not holds, the user trust that file $f_i=(f_{i,1},f_{i,2} \ldots f_{i,s})$ is corrupted.

5.5 Data updating

In order to update the data files, the user will use a few new data blocks. These blocks will replace the old ones. The detailed steps in this process are also outlined below.

Command Generation: let us consider that the user wants to update the data block $f_{g,k}$in the cloud which is stored under the lead node r_g, where $1\leq g\leq n$ and $1\leq k\leq S$. Along with that, the user retrieves the subfile $f_g=(f_{g,1},f_{g,2}\ldots f_{g,s})$ from the cloud and its associated auxiliary data φ_k. In the meantime, the user develops the $Sig_T=Sign_{Ksu}(upd||g||k||T)$. The user finally uploads the updated data command U_c along with new data block $f_{g,k}^*$ to the cloud.

Data Update: Once the U_c and $f_{g,k}^*$ is received to the cloud, it verifies the correctness of the U_c by verifying the signature Sig_T. If the Sig_T is not correct, then the cloud sends the failure message. If the Sig_T is correct, then the cloud updates the data block $f_{g,k}$with the $f_{g,k}^*$. In the meantime, cloud obtains the new MSHT* which is shown in figure 3. As anext step, the cloud forwards the new root R_n^* and $Sig_r=Sign_{Ksc}(R_n)$ to the user.

Result verification: The user checks the data update result by rebuilding the Merkle root after receiving the new root R_n^* and $Sig_r=Sign_{Ksc}(R_n)$ by the cloud.

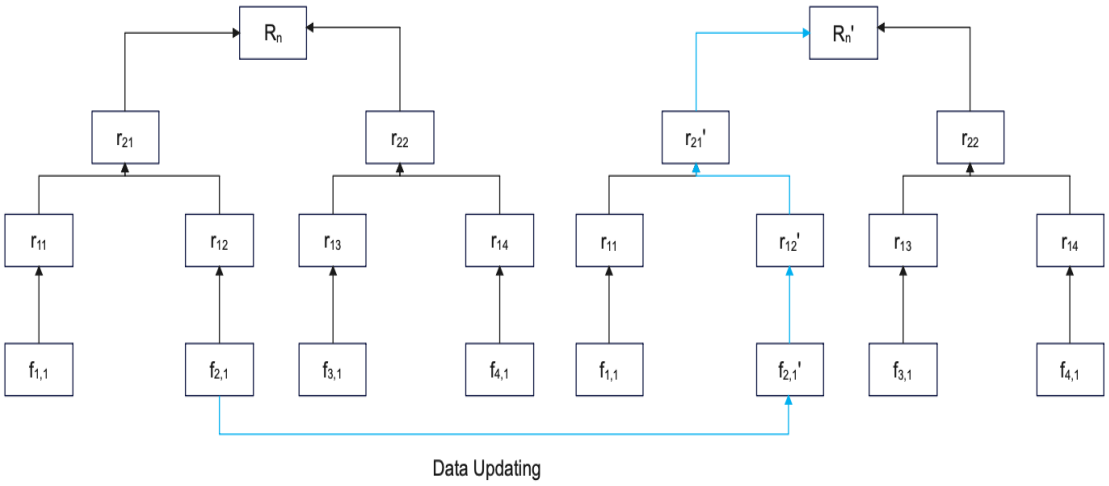

Figure 3: Data Updating Procedure

6. Experiment Analysis

The prototype system is used to test the computational capabilities of the proposed solution. We then perform a comprehensive evaluation of the primary computations on a desktop having the configuration of Intel core i7 processor with 16GB of RAM and 3.7 GHz processor. The simulation experiments are carried out to test the various cryptographic algorithms that are related to the proposed solution. The main libraries used for these tests are the cryptography and open socket layer.For the security of our proposed solution, we use SHA 1 as the hash function, while AES is the encryption algorithm. For the signature algorithm, we use ECDSA. For simplicity, we assume that the system's various subfiles are classified into data blocks.

6.1 Data Pre-processing:

This phase serves to complement the processes of data encryption and segmentation of cipher text. The duration for data pre-processing is typically associated with both the size of the outsourced file and the number of sub-files. In our experiment, we have designated 2500 sub-files. Subsequently, the scale of the outsourced files is increased to 20 MB, and we calculate the time spent on pre-processing the data. As the volume of outsourced data grows, the time dedicated to pre-processing linearly increases. The growth rate of proposed model is lower than that of the PVDT scheme [16], moreover, it demands less time compared to the PVDT scheme when considering the size of the external file. Specifically, the proposed scheme for this experiment requires approximately 55.6 milliseconds to process 20 MB of data, whereas the PVDT scheme incurred a cost of 65.4 milliseconds. Despite the user being responsible for executing the data pre-processing phase, the method can also be executed offline, as it is specific to a given outsourced file. In conclusion, the proposed scheme exhibits greater efficiency in the data pre-processing stage.

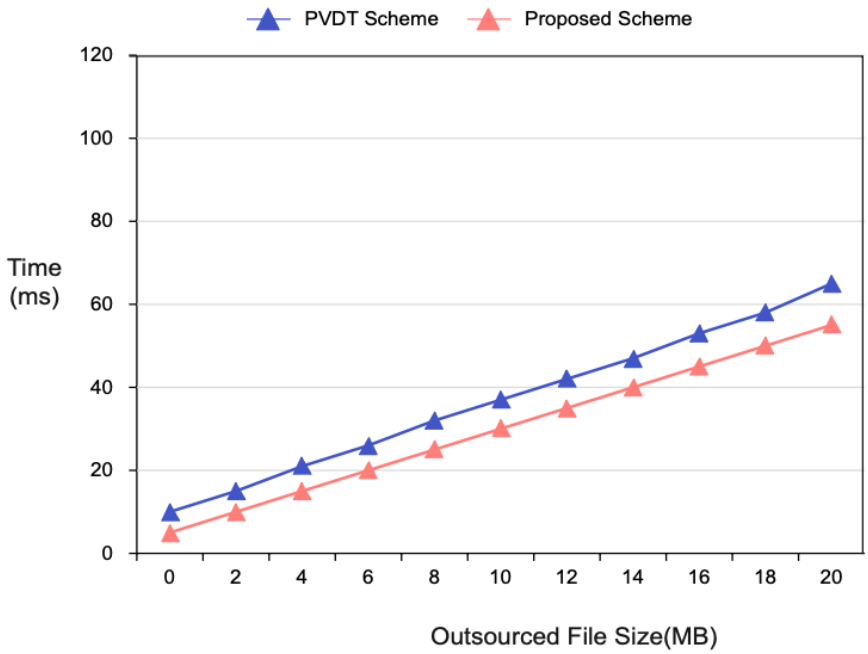

Figure 4: Data Pre-processing Time Overhead

6.2 Data Outsourcing:

The phase of outsourcing data involves establishing a framework for managing an outsourced file and transmitting it to the cloud. The time required for data outsourcing is influenced by the quantity of data blocks. Initially, 500 data blocks are allocated, and subsequently, this number is increased from 1000 to 10,000. The resulting running time overhead is illustrated in Figure 5. The graph in Figure 5 demonstrates that, during the data outsourcing phase, the running time for all three solutions increases as more data blocks are added. Notably, our designed solution exhibits a lower growth rate compared to the other two alternatives. At 5000 data blocks, our solution's running time is nearly 4.9 milliseconds, whereas the PVDT [16] and MRA [32] requires around 32.8 milliseconds. In contrast, the time overhead for the MRA is approximately 40.3 milliseconds. Consequently, the proposed method would necessitate less time to outsource an equivalent number of data blocks. In

terms of efficiency during the data outsourcing phase, proposed solution surpasses the PVDT [16] and MRA [32].

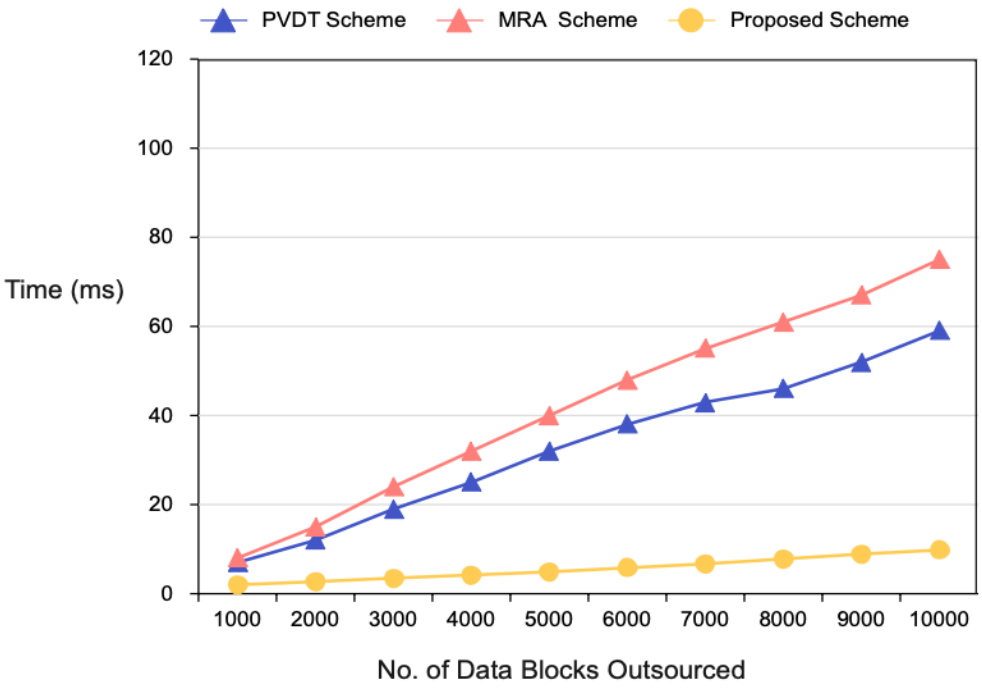

Figure 5: Data Outsourcing Time Overhead

6.3 Integrity Auditing:

The process of integrity checking ensures the proper functioning of the data center, aiming to guarantee the security and availability of data. The volume of data outsourced for integrity checking is directly linked to the runtime of the process. For instance, the number of outsourced data blocks has been increased from 1000 to 10,000, as depicted in Figure 6, which illustrates the associated costs of integrity checking. In this figure, the time costs for the three distinct solutions employed in integrity checking are presented. Notably, the growth rate of the designed solution is lower than that of the other two alternatives, attributed to its logarithmic increase in running time costs as the number of outsourced data blocks rises. Conversely, the running time costs for the other two solutions exhibit a linear growth pattern. The designed solution requires the least amount of time for the process, whereas the previous solution necessitates the most. For instance, with 5000 outsourced data blocks, the running time for the designed solution is approximately 1.9 milliseconds, compared to 31.8 milliseconds for PVDC [16] and 39.8 milliseconds for MRA [32]. In terms of efficiency, the proposed method outperforms the existing two schemes.

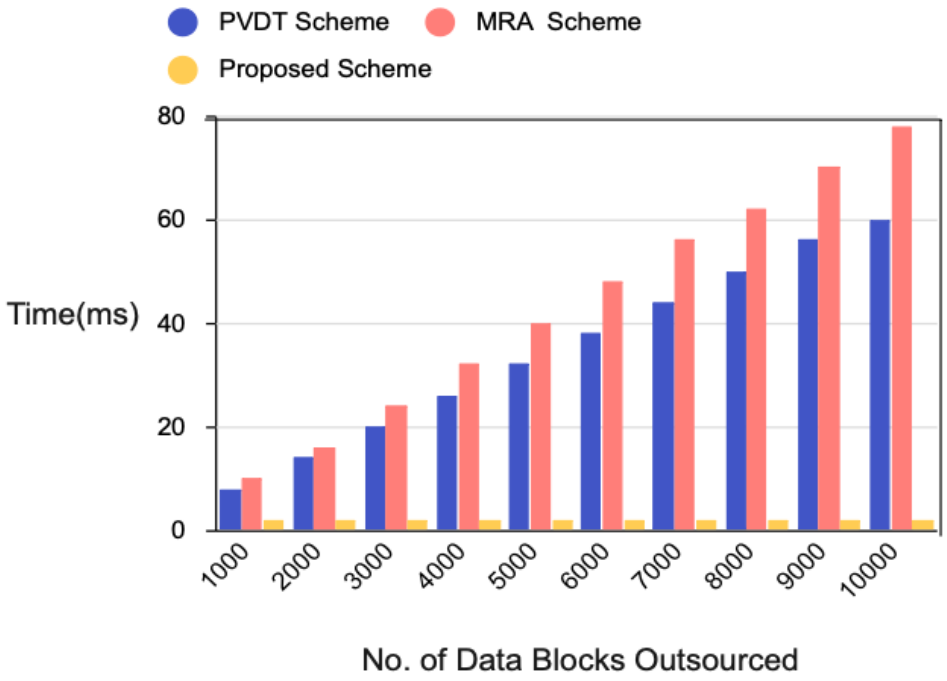

Figure 6: Integrity Auditing Time Overhead

6.4 Data Update:

The data updating phase involves the substitution of old data blocks with new ones, facilitating the refresh of outsourced data. The running time associated with data updating is primarily influenced by the number of outsourced subfiles. To meet the demand, the number of subfiles has to be increased, ranging from 1000 to 10,000, as illustrated in Figure 7, depicting the running time and cost of both the designed and previous solutions during the data updating phase. The growth rate of the designed solution is lower than that of the previous one, and its cost is also lower. Additionally, the proposed solution incurs a reduced time overhead. For instance, with 5000 subfiles, the proposed solution requires a time overhead of approximately 3.2 milliseconds, while the previous solution's running time was around 54.3 milliseconds. Consequently, it can be inferred that the proposed solution significantly outpaces the MRA [32] in terms of speed during the data updating process.

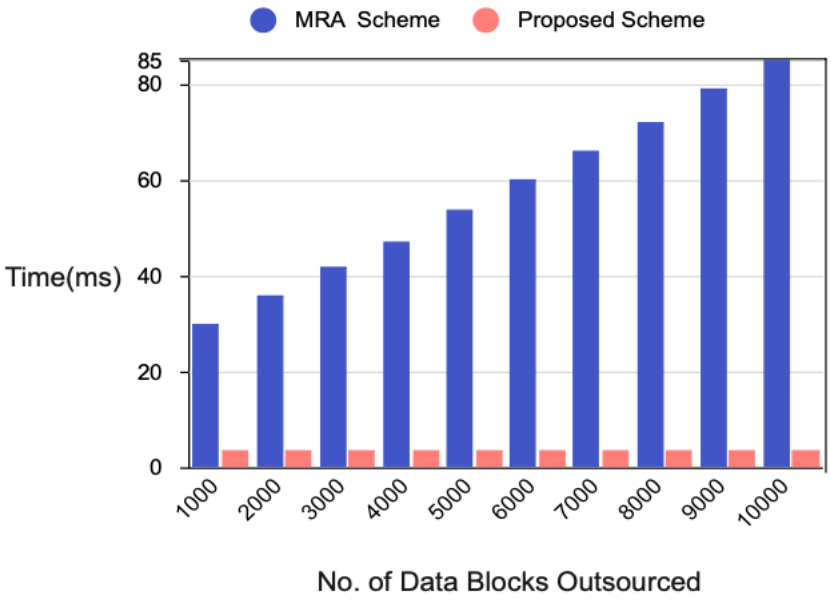

Figure 6: Data Updating Time Overhead

7. Conclusion

This research investigates the challenge of outsourcing the data integrity checking process within a cloud computing environment. Utilizing MSHT, we have developed an efficient method capable of executing block-based updates and restoring local backups within the cloud data center. The proposed approach allows users to conduct data integrity checks and availability audits, ensuring proper management of the cloud data center. Additionally, it enables the updating of old blocks with new ones. Due to the inherent lack of trust in the cloud data center, users can perform an audit of the update process. Security analysis of our proposed scheme indicates its capability to fulfil the design goal of reducing dependence on a third party. Results from experiments and performance comparisons demonstrate the superior efficiency of our method. This proposed scheme addresses the challenge of data integrity auditing by offering a reliable replacement for old data blocks. In certain scenarios, tenants may wish to add new data blocks while removing old ones. Future research will focus on designing a data auditing scheme that supports dynamic operations and ensures data integrity. Moreover, this method is versatile enough to handle various tasks such as inserting and updating verifiable data.

References

[1] C. Yang, F. Zhao, X. Tao, and Y. Wang, “Publicly verifiable outsourced data migration scheme supporting efficient integrity checking,” Journal of Network and Computer Applications, vol. 192, 2021.

[2] T. Wang, J. Zhou, X. Chen, G. Wang, A. Liu, and Y. Liu, “A three-layer privacy preserving cloud storage scheme based on computational intelligence in fog computing,” IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, no. 1, pp. 3–12, 2018.

[3] K. Xue, W. Chen, W. Li, J. Hong, and P. Hong, “Combining data owner-side and cloud-side access control for encrypted cloud storage,” IEEE Transactions on Information Forensics and Security, vol. 13, no. 8, pp. 2062–2074, 2018.

[4] C. Yang, X. Tao, F. Zhao, and Y. Wang, "Secure data transfer and deletion from counting bloom filter in cloud computing," Chinese Journal of Electronics, vol. 29, no. 2, pp. 273–280, 2020.

[5] J Maha Lakshmi, Krishna Prasad K & Viswanath G, "Enhancing Cloud Security: A Blockchain-Based Verification Framework for Multi-Cloud Virtual Machine Images," Frontiers in Health Informatics, vol. 13, no. 3, pp. 9535–9549, 2025.

[6] C.Yang,X.Chen,andX.Yang,"Blockchain-basedpubliclyverifiable data deletion scheme for cloud storage," Journal of Net- work and Computer Applications, vol. 103, pp. 185–193, 2018.

[7] S. Subashini and V. Kavitha, "A survey on security issues in service delivery models of cloud computing," Journal of Net- work and Computer Applications, vol. 34, no. 1, pp. 1–11, 2011.

[8] C.Yang,Y.Liu,F.Zhao,andS.Zhang,"Provabledatadeletion from efficient data integrity auditing and insertion in cloud storage," Computer Standards & Interfaces, vol. 82, 2022.

[9] P. Wei, D. Wang, Y. Zhao, S. K. S. Tyagi, and N. Kumar, "Blockchain data-based cloud data integrity protection mech- anism," Future Generation Computer Systems, vol. 102, pp. 902–911, 2020.

[10] B. Li, Q. He, F. Chen, H. Jin, Y. Xiang, and Y. Yang, "Auditing cache data integrity in the edge computing environment," IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 5, pp. 1210–1223, 2021.

[11] Y. Yu, Y. Li, B. Yang, W. Susilo, G. Yang, and J. Bai, "Attribute- based cloud data integrity auditing for secure outsourced stor- age," IEEE Transactions on Emerging Topics in Computing, vol. 8, no. 2, pp. 377–390, 2020.

[12] C. Yang, X. Tao, S. Wang, and F. Zhao, "Data integrity check- ing supporting reliable data migration in cloud storage," in International Conference on Wireless Algorithms, Systems, and Applications, pp. 615–626, Springer, 2020.

[13] Y.Li,Y.Yu,G.Min,W.Susilo,J.Ni,andK.K.R.Choo,"Fuzzy identity-based data integrity auditing for reliable cloud storage systems," IEEE Transactions on Dependable and Secure Com- puting, vol. 16, no. 1, pp. 72–83, 2019.

[14] G. Cui, Q. He, B. Li et al., "Efficient verification of edge data integrity in edge computing environment," IEEE Transactions on Services Computing, 2022.

[15] J. Lin, W. Yu, N. Zhang, X. Yang, and L. Ge, "Data integrity attacks against dynamic route guidance in transportation- based cyber-physical systems: modeling, analysis, and defense," IEEE Transactions on Vehicular Technology, vol. 67, no. 9, pp. 8738–8753, 2018.

[16] C. Yang, J. Wang, X. Tao, and X. Chen, "Publicly verifiable data transfer and deletion scheme for cloud storage," International Journal of Distributed Sensor Networks, vol. 15, 458 pages, 2019.

[17] G. Ateniese, R. Burns, R. Curtmola et al., "Provable data pos- session at untrusted stores," in in the 14th ACM conference on Computer and communications security, pp. 598–609, Alex- andria, Virginia, USA, 2007.

[18] L. Chen, "Using algebraic signatures to check data possession in cloud storage," Future Generation Computer Systems, vol. 29, no. 7, pp. 1709–1715, 2013.

[19] Y.Yu,Y.Zhang,J.Ni,M.H.Au,L.Chen,andH.Liu,"Remote data possession checking with enhanced security for cloud storage," Future Generation Computer Systems, vol. 52, pp. 77–85, 2015.

[20] G.Viswanath and P Venkata Krishna, "Hybrid encryption framework for securing big data storage in multi-cloud environment," Evolutionary intelligence, vol. 14, no. 2, pp. 691-698, 2021, DOI:10.1007/s12065-020-00404-w .

[21] H.Wang,"Identity-baseddistributedprovabledatapossession in multicloud storage," IEEE Transactions on Services Computing, vol. 8, no. 2, pp. 328–340, 2015.

[22] J. Chang, B. Shao, Y. Ji, and G. Bian, "Efficient identity-based provable multi-copy data possession in multi-cloud storage, revisited," IEEE Communications Letters, vol. 24, no. 12, pp. 2723–2727, 2020.

[23] Y. Fan, X. Lin, G. Tan, Y. Zhang, W. Dong, and J. Lui, "One secure data integrity verification scheme for cloud storage," Future Generation Computer Systems, vol. 96, pp. 376–385, 2019.

[24] W. Shen, J. Qin, J. Yu, R. Hao, J. Hu, and J. Ma, "Data integrity auditing without private key storage for secure cloud storage," IEEE Transactions on Cloud Computing, vol. 9, no. 4, pp. 1408–1421, 2021.

[25] N. Lu, Y. Zhang, W. Shi, S. Kumari, and K.-K. R. Choo, "A secure and scalable data integrity auditing scheme based on hyperledger fabric," Computers & Security, vol. 92, 2020.

[26] Y. Zhang, C. Xu, X. Lin, and X. Shen, "Blockchain-based public integrity verification for cloud storage against procrastinating auditors," IEEE Transactions on Cloud Computing, vol. 9, no. 3, pp. 923–937, 2021.

[27] Y. Li and F. Zhang, "An efficient certificate-based data integrity auditing protocol for cloud-assisted WBANs," IEEE Inter- net of Things Journal, 2021.

[28] F.Liu,D.Gu,andH.Lu,"Animproveddynamicprovabledata possession model," in in 2011 IEEE International Conference on Cloud Computing and Intelligence Systems, pp. 290–295, Beijing, China, 2011.

[29] M. Miao, J. Ma, X. Huang, and Q. Wang, "Efficient verifi- able databases with insertion/deletion operations from dele- gating polynomial functions," IEEE Transactions on Information Forensics and Security, vol. 13, no. 2, pp. 511–520, 2018.

[30] L. Rao, H. Zhang, and T. Tu, "Dynamic outsourced auditing services for cloud storage based on batch-leaves- authenticated Merkle hash tree," IEEE Transactions on Services Computing, vol. 13, no. 3, pp. 451–463, 2020.

[31] C. Yang, Y. Liu, and X. Tao, "Assure deletion supporting dynamic insertion for outsourced data in cloud computing," International Journal of Distributed Sensor Networks, vol. 16, no. 9, 2020.

[32] H. Yu, Z. Yang, M. Waqas et al., "Efficient dynamic multi- replica auditing for the cloud with geographic location," Future Generation Computer Systems, vol. 125, pp. 285–298, 2021.