International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 04 | April - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

A Blockchain-Based Multi-Cloud Storage Data Auditing Scheme to Locate Faults

SIVANRAJA A
Reg No: 95222226002, II$_{nd}$Year M. TECH
Department Of CSE, PSN College Of Engineering & Technology, Melathediyoor, Tirunelveli
Sivanraja554@gmail.com

Dr.VARGHEESE
Associate Professor: Department Of CSE
PSN College Of Engineering & Technology, Melathediyoor, Tirunelveli.

Abstract—**The advent of network storage services has revolutionized user experiences worldwide, offering unparalleled convenience, cost-effectiveness, and robust availability. However, relying solely on a single service provider entails inherent risks. To address this, sophisticated multi-cloud storage systems have emerged, aiming to mitigate data corruption risks. Nevertheless, ensuring data integrity in such systems necessitates robust auditing schemes. Traditional approaches often depend on centralized entities like third-party auditors (TPAs) or cloud service organizers, leaving users vulnerable to malicious actors during disputes. In response, we propose a blockchain-based multi-cloud storage auditing framework. By leveraging blockchain technology, we establish an immutable ledger to record interactions among users, service providers, and organizers, serving as indisputable evidence. Smart contracts are employed to detect and resolve service disputes, compelling untrusted organizers to accurately identify malicious providers. Additionally, we utilize blockchain networks and homomorphic verifiable tags to facilitate low-cost batch verification without the need for TPAs. Theoretical analyses and empirical evaluations demonstrate the scheme's efficacy and cost-effectiveness in multi-cloud environments, promising enhanced data integrity and dispute resolution mechanisms.**

Keywords—Multi-cloud storage, blockchain, data auditing, dispute arbitration

I. INTRODUCTION

In recent years, network computing technologies such as cloud computing [1] and transparent computing [2] have significantly enhanced user experiences by providing robust storage and processing capabilities [3]. Through these innovations, users can seamlessly offload their data to cloud storage servers, eliminating the need for local infrastructure maintenance. Major companies like Microsoft, IBM, and Amazon have pioneered their own network storage services, further expanding the accessibility of these technologies.

However, entrusting data to network computing services introduces inherent security risks for users [4]. When users opt for network storage services, they relinquish ownership of their data to the cloud service provider (CSP). This separation between data owner and physical controller creates vulnerabilities, as the integrity and availability of data become solely dependent on the CSP. Regrettably, CSPs are not infallible; they may inadvertently corrupt or maliciously delete data to alleviate storage burdens. To mitigate these risks, many users opt for a diversified approach, leveraging multiple network storage services to collectively store and manage their data [5].Despite the advantages of distributed network storage strategies in enhancing data security and efficiency, concerns persist regarding user confidence in the security of outsourced data within multi-cloud systems [6]. Addressing these concerns necessitates the development of effective measures to reassure users about the security and integrity of their data across diverse cloud environments.

In addressing the aforementioned issue, the academic community has made significant strides. Ateniese et al. [7] introduced the concept of provable data possession (PDP) in 2007, offering a practical solution employing spot-checking techniques to probabilistically verify data integrity stored by users on servers [8]. This advancement allows users to ensure data integrity without the need to retain the original data locally, thereby confirming its correct maintenance. While subsequent data auditing schemes have emerged [9], most focus solely on single CSP integrity, proving inefficient in multi-cloud environments where each CSP must be individually checked.

To enhance the efficiency of data auditing in multi-cloud storage, numerous schemes supporting batch auditing have been proposed [10], [11], [12]. However, deploying these schemes poses several challenges. Firstly, many rely on a trusted third-party auditor (TPA), which proves elusive and

introduces single-point failures and performance bottlenecks when trusted by multiple CSPs. Secondly, in multi-cloud storage, a cloud service organizer typically manages all CSPs, with companies like IBM and Alibaba offering such multi-cloud management services. Nonetheless, existing schemes assume the organizer's honesty and reliability, overlooking the potential for errors [13]. Lastly, enabling batch data audits necessitates aggregating responses from all CSPs, complicating users' ability to accurately identify and prove malicious CSPs post-data corruption discovery.

The emergence of blockchain technology [14] presents a solution to challenges in data auditing. Various blockchain-based auditing schemes [15], [16], [17] leverage its decentralization and traceability to enhance trust in multi-cloud storage. Some schemes store TPA audit logs on the blockchain for monitoring [18], [19], while others utilize smart contracts for auditing [20]. Additionally, blockchain is employed to prevent collusion between malicious TPAs and CSPs [21], reducing reliance on centralized TPAs.

Despite these advancements, accurate dispute arbitration and resistance to malicious organizers remain unresolved. To address these challenges, we propose a blockchain-based multi-cloud storage auditing scheme. Homomorphic verifiable tags facilitate batch verification of data integrity, minimizing overhead. The scheme involves users and CSPs jointly generating integrity metadata, with the organizer aggregating CSP responses. Interaction records are stored on the blockchain, enabling smart contract-based dispute detection and arbitration. If data corruption occurs, the smart contract prompts the organizer to identify malicious CSPs, leveraging blockchain's tamper-resistance. In summary, the four contributions of the proposed scheme are as follows:

- To propose a secure and accurate data auditing scheme for multi-cloud storage services.
- By introducing blockchain technology, this scheme achieves a public batch data integrity audit in multi-cloud scenarios without the need for centralized TPA.
- To find the malicious one among all cloud service providers and resolve data possession disputes.

The objective is to create a robust and effective system capable of precisely identifying faults within multi-cloud storage environments. This proposed scheme leverages blockchain technology to pinpoint and address faults effectively, enhancing the reliability of data storage systems. By implementing this solution, organizations can efficiently detect and resolve issues within their multi-cloud setups, ensuring data integrity and mitigating risks effectively.

II. LITERATURE SURVEY

In order to ensure the integrity of outsourced data, Juels et al. [22] first presented the ‘‘Proofs of Retrievability’’ (POR) mechanism, but this scheme does not consider public audit, and data owner must bear a heavy audit burden. To support public audit, Ateniese et al. [23] presented the ‘‘Provable Data Possession’’ (PDP) mechanism based on RSA signature, which introduced an independent TPA to verify the integrity of outsourced data on behalf of users, greatly reducing unnecessary overhead. Since then, many public audit schemes based on homomorphic signature technology [24], [25] have been proposed successively. However, in many practical applications, data owners want specific users to check files in cloud storage. In view of this, a PDP protocol with designated verifier was proposed by Ren et al. [26], but this scheme cannot resist replay attack. In order to overcome this shortcoming, a new designated verifier audit scheme was constructed by Yan et al. [27].

However, the above schemes cannot support the dynamic update of data. To support the dynamic update of data, Erway et al. [28] presented a full dynamic data integrity audit scheme by introduced rank-based authentication skip list. Wang et al. [29] proposed a data integrity verification scheme by introduced merkle hash tree (MHT). Zhu et al. [30] constructed another public auditing scheme based on index hash table (IHT). However, the above three schemes generate a lot of computation and communication cost during the update and verification process. To solve this problem, a new structure called dynamic hash table (DHT) was proposed by Tian et al. [31], and used this structure update the cloud data with high efficiency. However, the scheme does not consider the confidentiality of data. Therefore, Hwang et al. [32] constructed a public audit scheme that supports data confidentiality and data dynamic operations.

The rise of blockchain technology has made it possible to solve the centralization problem of data auditing schemes. Blockchain is an immutable distributed ledger that records the state of all entities in the blockchain network [26], [27]. With the assistance of the decentralization, non-repudiation and traceability features of the blockchain, many blockchain-based data auditing schemes have been proposed. By using the blockchain to construct a random challenge message, Xue et al. [28] achieved a fair identity-based public auditing scheme. Thus, preventing untrusted TPAs and storage service providers from colluding and forging the data integrity proofs. To supervise the semi-trusted TPAs, Zhang et al. [29] ask them to publish their audit logs on the blockchain. A similar blockchain-based data auditing scheme is also designed by Zhang et al. [17]. Liu et al. [16] advance a blockchain-based PDP scheme applicable to the IoT environment. Xu et al. [20] proposed an arbitrable data auditing scheme that uses smart contracts to fairly resolve data integrity disputes. Although these schemes solve the centralization problem of traditional approaches, they are only suitable for singlecloud environments. When these schemes are extended to multi-cloud scenarios, due to the lack of batch auditing methods, the auditing process brings a lot of computing and communication overheads.

Several data auditing schemes have emerged to efficiently verify data integrity in multi-cloud storage environments, garnering attention from researchers. Zhu et al. [10] introduced the first protocol for data integrity audit in multi-cloud storage, employing homomorphic verifiable tags to enable collaboration among multiple network storage providers for generating integrity proofs during bulk verification. He et al. [6] leveraged recoverable coding technology to develop a multi-cloud storage PDP protocol,

facilitating auditor batch verification and rapid identification of erroneous data blocks. Wang et al. [11] implemented an identity-based PDP protocol within a multi-cloud system. Li et al. proposed an identity-based public data auditing scheme tailored for scenarios involving multi-cloud environments and multiple copies. However, these schemes rely on trusted third-party auditors (TPAs), posing inherent risks. Moreover, the aggregation of CSPs' proofs for efficient batch audits complicates the accurate identification of malicious CSPs. Consequently, there remains a pressing need for a more secure and precise data auditing scheme for multi-cloud storage to safeguard users' outsourced data.

III. Proposed Methodology

3.1 System Model

The proposed multi-clouds storage data auditing scheme is established on the blockchain. There are five kinds of entities in the system: user, cloud service provider, organizer, system manager and key generation center. The definitions of them are as follows.

- User (U) is a client of network storage services. U will store his data in multiple CSPs and hire an organizer to manage these CSPs. He will also check the integrity of outsourced data.
- Cloud Service Provider (CSP) provides network storage services for users, but users' outsourced data may be damaged by the CSP.
- Organizer (O) is a special cloud service provider that manages the interaction between U and CSPs in the multi-cloud storage environment.
- System Manager is a representative of blockchain nodes and is used to deploy blockchains and smart contracts. He will be offline after the system is deployed.
- Key Generation Center (KGC) is an authority and responsible for secretly generating security parameters.

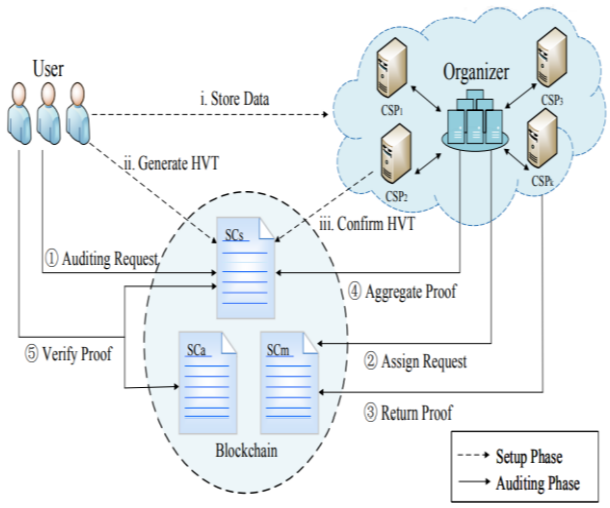

Fig. 1: The blockchain-based multi-cloud storage data auditing scheme

In addition, there are many blockchain nodes in the system. Similar to the same entities in other blockchain systems, they obtain benefits by maintaining the blockchain. They are the security guarantee for the correct operation of the blockchain and smart contracts. Since they are not involved in the data audit process, we treat them as part of the infrastructure and omit them from the approach.

The proposed auditing scheme is mainly composed of four phases: the initialization phase, the setup phase, the auditing phase and the dispute arbitration phase. In the initialization phase, KGC generates security parameters and the system manager deploys the blockchain-based system. During the setup phase, U outsources data to CSPs and sets HVTs together. With the help of O and the blockchain, U uses the challenge-response model to verify the integrity of the outsourced data during the auditing phase. Finally, if there is a service dispute, the smart contract will arbitrate the dispute fairly.

3.2 Threat Model

In the threat model, we define the possible behavior of each entity and consider the possible attacks of our scheme. We assume that all network storage service providers, whether CSPs or O, are not trusted. A malicious or careless CSP may lose data and try to hide it to maintain its reputation or delete data that users rarely use to reduce their storage burden. An untrusted O may have incorrectly executed user commands or returned incorrect information to users. In order to get some compensation, a malevolent U may also frame honest CSPs or O by applying for arbitration. But in any case, none of them will turn a blind eye to the other part's malicious behaviors which may harm their own interests. We also believe that system manager and KGC are trusted. The behaviors of the system manager are public and agreed by blockchain nodes. Since they will not participate in the data audit process, trust in them will not affect the security.

3.3 Methods

In this section, we describe the proposed blockchain-based multi-cloud storage data auditing scheme.

In order to achieve accurate audit without TPA in multi-cloud storage and resist malicious organizers, we propose a data auditing scheme based on blockchain. Before outsourcing data to CSPs, U and CSPs jointly generate HVT of data. Both parties confirm that their HVT is consistent with the help of blockchain. During the data auditing process, U generates a challenge nonce and requires CSPs to respond. After all CSPs calculate the respond based on challenged data blocks, O will aggregate the results into one integrity proof and send it to U through the blockchain for audit. When there is a data integrity dispute, the smart contract can judge whether the dispute exists based on the records on the blockchain, thereby preventing the framing behavior of malicious U. If the smart contract determines that there is a problem with the service provider side, it will ask O to find the malicious CSP within the specified time, otherwise it will consider O to be malicious. At this point, the rational O must honestly find out the malicious entity (including himself) and cannot frame other CSPs, because the framed CSP can prove to the smart

contract that he is honest through the interaction records on the blockchain. The proposed scheme is shown in Fig. 1.

3.3.1Initialization Phase

During the initialization phase, the system manager deploys the blockchain platform and provides a smart contract template for users to generate their own smart contracts. The system manager also requests KGC to generate system parameters. Then, the KGC selects two large primes p and q to generate the RSA modulus N = pq, and let g be the generator of QRN . N and g will be returned to the system manager and published on the blockchain. After the blockchain is deployed, U, O, CSP and other blockchain nodes join to the blockchain. U negotiates with O and CSPs to specify some details of the storage service such as the service price, the compensation plan for potential service dispute and the frequency of checking the data integrity. Finally, O deploys the storage service verification smart contract (SCs), arbitration smart contract (SCa) and the multi-cloud management smart contract (SCm) on the blockchain according to the negotiation result.

3.3.2 Setup Phase

In the setup phase, U divides its data into n data blocks and stores them in m CSPs. U and CPSs then jointly generate the HVT T ag(bi) as the integrity metadata for each data block bi through the blockchain network. The specific operations in the setup phase are as follows.

- **Step i**: [*U* sends data blocks].
  In order to decrease the verification overhead and adapt to the distributed environment of multi-cloud storage, *U* divides its data *F* into n data blocks, $F = (b_1, \ldots, b_i, \ldots, b_n)$. Each data block bi can be seen as a sufficiently large integer. Then *U* sends b_i to *O* through a secure off-chain channel. After receiving b_i, *O* distributes these data blocks to each CSP according to the convention in the negotiation result.
- **Step ii:** [*U* generates HVTs].
  For every b_i , *U* calculates the HVT $Tag(b_i) = g^{b_i} \bmod N$ as the integrity metadata. It is noticeable that the size of $Tag(b_i)$ is much smaller than b_i. Therefore *U* can check data integrity using $Tag(b_i)$ with minimal storage overhead. Finally, *U* publishes each $Tag(b_i)$ on the SCs through a blockchain transaction.
- **Step iii**: [*CSPs* confirm HVTs].
  For each b_i' received from O, CSPs also calculate $Tag(b_i') = g^{b_i'} \bmod N$. They then compare the calculation results with the $Tag(b_i)$ that U uploaded to the blockchain in step ii. If the results of the comparison are consistent, it means that CSPs received the correct bi and they will confirm $Tag(b_i)$ in the SCs. Otherwise, it indicates that there is a problem with the data block. At this time, CSPs will refuse to confirm $Tag(b_i)$ and will terminate the network storage service. Since U did not delete their local data, the failure in the setup phase has not affected the data integrity. Finally, if the service is implemented smoothly, U will delete its data saved locally after all the HVTs are confirmed by the corresponding CSP.

3.3.3. Auditing Phase

In the auditing phase, *U* checks the multi-cloud storage service to ensure that his data is securely stored on the *CSPs*. To reduce the overhead of this phase, *U* verifies partial data blocks in batch to probabilistically check the integrity of whole outsourced data. For example, if *U* divided his data into 10,000 data blocks in the previous phase, and evenly stored them in 10 *CSPs*. If a *CSP* damaged 1% of the data blocks, then *U* can detect this malicious behavior with a probability greater than 99% by randomly auditing 460 blocks. Specifically, *U* generates a challenge nonce *chal* and randomly selects some data blocks. The related *CSPs* then compute an integrity *proof* as a response. *O* aggregates all the into a value proof and returns it to *U*. The specific operations in the auditing phase are as follows.

Step 1: [*U* sends auditing request].

Firstly, *U* selects a large enough integer r secretly to compute a challenge nonce . *U* then sends an auditing request $req =< I, A >$, in which *I*is a set of the data block numbers that *U* wants to check, and A is a set of coefficients corresponding to the challenged data block.

Step 2: [*O* assigns auditing request].

After receiving the auditing request from the blockchain, *O* finds the location of in the set *I* and notifies the corresponding CSP_s to response chal through SCm. Specifically, *O* generates a subset Ik of I for CSP_k, which contains all the data blocks stored in CSP_k in *I*. *O* also generates a coefficient subset Ak of A for CSP_k. Then O sends the auditing request req_k the CSP_k via a blockchain transaction.

Step 3: [CSP_s response auditing request].

For a CSP_k, if *U* asks to verify its stored data blocks and the corresponding coefficient subset $ACSP_k$ generates an integrity proof $proof_k = \prod_{i=1}^{j} c$ and calls the method in to return as a response. This response will trigger calculating $\sigma_k = \prod_{i=1}^{j} Ta$ so that *U* can verify whether is right in batch. In addition, if CSP_k does not respond in a limited time, it will be considered that CSPk violates the agreement. The blockchain will record the default and conduct the arbitration in the dispute arbitration phase.

Step 4: [*O* aggregates responses].

After all CSPs have returned their integrity proofs through the blockchain, *O* aggregates all to generate an aggregate value $pro \bmod N$. Finally, *O*returns *Hash(proof)* to *U* through the blockchain. Here, The role of

the hash function is to reduce communication overhead and blockchain storage overhead. Meanwhile, *O* aggregates all to generate an aggregate value σ Both Hash(proof) and σ are saved on the blockchain.

Step 5: [*U* batch verification].

Finally, *U* receives σ and Hash(proof) from the blockchain, and verifies the proofs of *CSPs* in a batch way according to the following equation.

$$Hash(proof) = Hash(\sigma^{r} mod N)$$

If the equation holds, *U* considers that this verification is correct. Otherwise it indicates that the *CSP* or *O* may be malicious, and the data stored in multiple *CSPs* may be damaged. Then U will publish its r on the blockchain to apply for arbitration through the *SCa*.

3.3.4. Dispute Arbitration Phase

If the response of *O* does not pass the verification, *U* will apply for arbitration in SCa and the scheme will enter the dispute arbitration phase. In this phase, SCa checks if there is a malicious behavior according to the interactive record recorded on the blockchain. The specific dispute arbitration process is as follows.

Firstly, *U* uploads to the blockchain to trigger the arbitration smart contract SCa. Since *U* may upload a wrong to frame an honest CSP, SCa calculates g r 0 and compares the result with chal to judge the validity of . If chc then revealed by *U* can be considered correct. Otherwise, SCa considers that *U* publishes a fake r and determines *U* is malicious.

If *U* passes the above check, SCa further determines if there is a malicious service provider according to the following equation.

$$Hash(proof) = Hash(\sigma^{r'} mod N)$$

Specifically, SCa calculates and compares the result with Hash(proof) that O published on the blockchain. If holds, CSPs can be considered to have made the correct response and the arbitration applied by U is unreasonable, so will think U is malicious and punish it. Otherwise, determines that there is a malicious service provider. There are two types of behaviors that can lead to a service dispute at this time: CSP corrupts data blocks or O error calculates proof. Then askes O to find out the reason for the error and publish the result on the blockchain. The specific process.

Similarly, O will find the reason according to prookk and . For each CSPk, O verifies its correctness by checking whether proofk meets the following equation.

If does not satisfy the equation, it means that the integrity of data blocks stored by $CSP_k$is destroyed and O will expose this result on the blockchain. will also verify proofk to confirm this result. Finally, if all responses of $CSP_s$are correct, it means that the error occurs in Step 4 where O wrongly aggregates the responses. O will admit this error on the blockchain and make compensation. In addition, a malicious O may frame an honest CSP_k to shrink from the responsibility. But as O can not find the evidence to prove that CSP_k is wrong, will detect that CSP_k is innocent. Therefore O is forced to admit its calculation error on the blockchain. In the end, a malicious U, O or *CSPk* will be punished according to the negotiated clauses.

In addition, if the *CSP* or O does not respond in time during the service verification phase, U will consider the situation as service unavailable and apply to the for arbitration. Then the will check the blockchain. And if it does not find a response from *CSP* or O, can determine that it is malicious. Finally, the arbitration result will be published on the blockchain by O or , and the violator will be punished.

IV. Results and Discussion

First evaluate the time costs of the off-chain operation to evaluate the off-chain computational overhead of our scheme. Specifically, test the time costs of CSPs generate proofs, O aggregates proofs and U checks proof under the different number of data blocks. The experimental results are shown in Fig. 4.1. According to the theoretical analysis, when the total number of data blocks is 10,000, if 1% of them are damaged, then U checks 460 blocks to have a 99% probability of discovering the malicious behavior of CSP. In the experiment, when the challenged block number is 460, the time cost of CSP is about 5 seconds, U is 17 ms and O is 14 ms respectively. In addition, the time cost of CSP increases with the increase in the number of challenged data blocks, while the time cost of U and O remains stable. This experimental result is in line with our expectations. Based on the homomorphism of the HVT, U only needs to check proofs in batch. And the computational overhead of O is only related to the number of CSPs. Therefore, U and O have relatively small computational overhead in our scheme, and the computational overhead of CSP is acceptable.

We also tested the time consumption of the approach to generate HVTs. We found that the average time cost to generate HVTs increases almost linearly as the number of total blocks grows, and for every 100 increments, the time cost increases by about 1.2 seconds. This computational overhead is lower than most existing data auditing approaches. In addition, we compare the time costs of our scheme with the previous blockchain-based multi-cloud storage data auditing scheme at different accuracy rates. The experimental results are shown in Fig. 4.2. Although the cost of our scheme in aggregation proof is higher than that of literature, the total time cost of our scheme is better than their scheme. Moreover, users' requirements for audit accuracy have a great impact on the cost of the program. When auditing 10,000 data blocks, the 97% accuracy rate will be approximately 70% faster than the 99% accuracy rate.

Table1. Time costs of each operation

		100	200	300	400	500
Generate HVT	(Existing)	1.7	3.6	5.4	6.9	7.3
	(Proposed)	1.5	3	5	6	7
Generate Proof	(Existing)	2.1	3	4.9	5.8	7
	(Proposed)	1	2.8	4.6	5.3	6.7

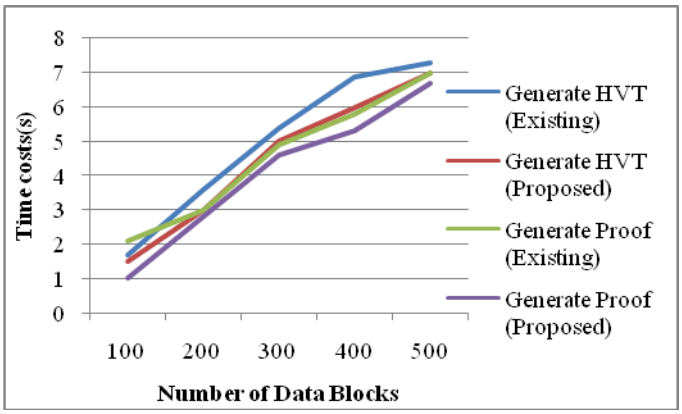

Fig. 4.1: Time costs of each operation

Table2: Time costs of the audit at 99% accuracy

	1000	2000	3000	4000	5000
Existing	1.8	2	3.1	4.2	4.6
Proposed	1	1.7	2.6	3.5	4

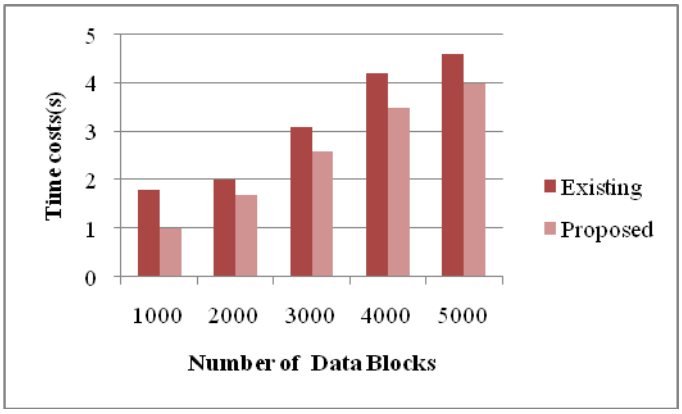

Fig. 4.2: Time costs of the audit at 99% accuracy

V. Conclusion

This project introduces a blockchain-based data auditing scheme aimed at identifying malicious service providers within multi-cloud storage services. Utilizing homomorphic verifiable tags, users can efficiently batch-verify their data across multiple cloud service providers at a reduced cost. The blockchain autonomously records interactions between users, organizers, and cloud service providers during data auditing. In case of disputes, smart contracts utilize these records to pinpoint malicious providers, eliminating the need for a trusted third-party auditor (TPA). Leveraging the credibility and traceability of blockchain, the scheme compels untrusted organizers to truthfully report provider misconduct. A prototype system demonstrates the effectiveness and affordability of the proposed auditing scheme in multi-cloud environments.Future iterations of the project consider optimizing storage overhead by potentially migrating some interaction records from the blockchain to IPFS (InterPlanetary File System). Additionally, the integration of sidechain techniques aims to offload computations from the main blockchain, further reducing computational overhead. Furthermore, the project explores designing an auditing mechanism to monitor the number of backups within the multi-cloud storage service..

References

[1] A. Michael, F. Armando, G. Rean, D. J. Anthony, K. Randy, K. Andy, L. Gunho, P. David, R. Ariel, S. Ion, and Z. And, Matei, “A view of cloud computing,” ACM Commun., vol. 53, no. 4, 2010.

[2] J. Ren, H. Guo, C. Xu, and Y. Zhang, “Serving at the edge: A scalable IoT architecture based on transparent computing,” IEEE Netw., vol. 31, no. 5, pp. 96–105, 2017.

[3] L. Da Xu, W. He, and S. Li, “Internet of things in industries: A survey,” IEEE Trans. Ind. Informat., vol. 10, no. 4, pp. 2233–2243, 2014.

[4] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, “Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds,” in Proc. 16th ACM Conf. Comput. Commun. Security, 2009, pp. 199–212.

[5] H. Wang, P. Shi, and Y. Zhang, “Jointcloud: A cross-cloud cooperation architecture for integrated internet service customization,” in Proc. 2017 IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS), 2017, pp. 1846–1855.

[6] K. He, C. Huang, J. Wang, H. Zhou, X. Chen, Y. Lu, L. Zhang, and B. Wang, “An efficient public batch auditing protocol for data security in multi-cloud storage,” in Proc. 8th ChinaGrid Annu. Conf., 2013, pp. 51–56.

[7] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, “Provable data possession at untrusted stores,” in Proc. 14th ACM Conf. Comput. Commun. Security, 2007, pp. 598–609.

[8] R. S. Kumar and A. Saxena, “Data integrity proofs in cloud storage,” in Proc. 3rd Int. Conf. Commun. Netw. (COMSNETS), 2011, pp. 1–4.

[9] M. Sookhak, A. Gani, H. Talebian, A. Akhunzada, S. U. Khan, R. Buyya, and A. Y. Zomaya, "Remote data auditing in cloud computing environments: a survey, taxonomy, and open issues," ACM Comput. Surv., vol. 47, no. 4, pp. 1–34, 2015.

[10] Y. Zhu, H. Hu, G.-J. Ahn, and M. Yu, "Cooperative provable data possession for integrity verification in multicloud storage," IEEE Trans. Parallel Distrib. Syst., vol. 23, no. 12, pp. 2231–2244, 2012.

[11] H. Wang, "Identity-based distributed provable data possession in multicloud storage," IEEE Trans. Serv. Comput., vol. 8, no. 2, pp. 328–340, 2014.

[12] S. Shin and T. Kwon, "A survey of public provable data possession schemes with batch verification in cloud storage," J. Internet Serv. Inf. Security (JISIS), vol. 5, no. 3, pp. 37–47, 2015.

[13] H. Wang and Y. Zhang, "On the knowledge soundness of a cooperative provable data possession scheme in multicloud storage," IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 1, pp. 264–267, 2013.

[14] S. Nakamoto et al., "Bitcoin: A peer-to-peer electronic cash system," Consulted, pp. 1–9, 2008.

[15] Y. Xu, C. Zhang, G. Wang, Z. Qin, and Q. Zeng, "A blockchainenableddeduplicatable data auditing mechanism for network storage services," IEEE Trans. Emerg. Top. Comput., pp. 1–12, 2020, doi: 10.1109/TETC.2020.3005610.

[16] B. Liu, X. L. Yu, S. Chen, X. Xu, and L. Zhu, "Blockchain based data integrity service framework for IoT data," in Proc. IEEE Int. Conf. Web Services (ICWS), 2017, pp. 468–475.

[17] Y. Zhang, R. H. Deng, X. Liu, and D. Zheng, "Blockchain based efficient and robust fair payment for outsourcing services in cloud computing," Inform. Sci., vol. 462, pp. 262–277, 2018.

[18] F. Xiang, W. Huaimin, S. Peichang, F. Yingwei, and W. Yijie, "Jcledger: A blockchain based distributed ledger for jointcloud computing," in Proc. 2017 IEEE 37th Int. Conf. Distrib. Comput. Syst. Workshops (ICDCSW), 2017, pp. 289–293.

[19] 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 Trans. Cloud Comput., pp. 1–15, 2019, doi: 110.1109/TCC.2019.2921553.

[20] Y. Xu, J. Ren, Y. Zhang, C. Zhang, B. Shen, and Y. Zhang, "Blockchain empowered arbitrable data auditing scheme for network storage as a service," IEEE Trans. Serv. Comput., vol. 13, no. 2, pp. 289–300, 2020.

[21] X. Yang, X. Pei, M. Wang, T. Li, and C. Wang, "Multi-replica and multi-cloud data public audit scheme based on blockchain," IEEE Access, vol. 8, pp. 144 809–144 822, 2020.

[22] A. Juels and B. S. Kaliski, ''PORs: Proofs of retrievability for large files,'' in Proc. 14th ACM Conf. Comput. Commun. Secur., 2007, pp. 583–597.

[23] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, ''Provable data possession at untrusted stores,'' in Proc. 14th ACM Conf. Comput. Commun. Secur., 2007, pp. 598–609.

[24] Y. Zhang, C. Xu, X. Liang, H. Li, Y. Mu, and X. Zhang, ''Efficient public verification of data integrity for cloud storage systems from indistinguishability obfuscation,'' IEEE Trans. Inf. Forensics Security, vol. 12, no. 3, pp. 676–688, Mar. 2017.

[25] S. G. Worku, C. Xu, J. Zhao, and X. He, ''Secure and efficient privacypreserving public auditing scheme for cloud storage,'' Comput. Electr. Eng., vol. 40, no. 5, pp. 1703–1713, Jul. 2014.

[26] Y. Ren, J. Xu, J. Wang, and J.-U. Kim, ''Designated-verifier provable data possession in public cloud storage,'' Int. J. Secur. Appl., vol. 7, no. 6, pp. 11–20, Nov. 2013.

[27] H. Yan, J. Li, and Y. Zhang, ''Remote data checking with a designated verifier in cloud storage,'' IEEE Syst. J., vol. 14, no. 2, pp. 1788–1797, Jun. 2020.

[28] C. Erway, A. Kupccu, C. Papamathou, and R. Tamassia, ''Dynamic provable data possession,'' in Proc. 16th ACM Conf. Comput. Commun. Secur., 2009, pp. 213–222.

[29] Q. Wang, C. Wang, J. Li, K. Ren, and W. J. Lou, ''Enabling public verifiability and data dynamics for storage security in cloud computing,'' in Proc. Eur. Symp. Res. Comput. Secur. Berlin, Germany: Springer, 2009, pp. 355–370.

[30] Y. Zhu, G.-J. Ahn, H. Hu, S. S. Yau, H. G. An, and C.-J. Hu, ''Dynamic audit services for outsourced storages in clouds,'' IEEE Trans. Services Comput., vol. 6, no. 2, pp. 227–238, Apr. 2013.

[31] H. Tian, Y. Chen, C.-C. Chang, H. Jiang, Y. Huang, Y. Chen, and J. Liu, ''Dynamic-hash-table based public auditing for secure cloud storage,'' IEEE Trans. Services Comput., vol. 10, no. 5, pp. 701–714, Sep. 2017.

[32] M. S. Hwang, T. H. Sun, and C. C. Lee, ''Achieving dynamic data guarantee and data confidentiality of public auditing in cloud storage service,'' J. Circuits, Syst. Comput., vol. 26, no. 5, pp. 175–190, 2017.

[33] B. Wang, B. Li, and H. Li, ''Panda: Public auditing for shared data with efficient user revocation in the cloud,'' IEEE Trans. Services Comput., vol. 8, no. 1, pp. 92–106, Jan. 2015.

[34] Y. Luo, M. Xu, K. Huang, D. Wang, and S. Fu, ''Efficient auditing for shared data in the cloud with secure user revocation and computations outsourcing,'' Comput. Secur., vol. 73, pp. 492–506, Mar. 2018.