Efficient Method Based on Blockchain Ensuring Data Integrity Auditing with Deduplication in Cloud

Mohamed El Ghazouani[1], My Ahmed El Kiram[1], Latifa Er-Rajy[1], Yassine El Khanboubi[2]

[1] Cadi Ayyad University, Marrakesh (Morocco)
[2] Hassan II University, Casablanca (Morocco)

Received 30 April 2020 | Accepted 20 July 2020 | Published 6 August 2020

Abstract

With the rapid development of cloud storage, more and more cloud clients can store and access their data anytime, from anywhere and using any device. Data deduplication may be considered an excellent choice to ensure data storage efficiency. Although cloud technology offers many advantages for storage service, it also introduces security challenges, especially with regards to data integrity, which is one of the most critical elements in any system. A data owner should thus enable data integrity auditing mechanisms. Much research has recently been undertaken to deal with these issues. In this paper, we propose a novel blockchain-based method, which can preserve cloud data integrity checking with data deduplication. In our method, a mediator performs data deduplication on the client side, which permits a reduction in the amount of outsourced data and a decrease in the computation time and the bandwidth used between the enterprise and the cloud service provider. This method supports private and public auditability. Our method also ensures the confidentiality of a client's data against auditors during the auditing process.

Keywords

Auditing, Blockchain, Cloud Computing, Deduplication, Integrity.

DOI: 10.9781/ijimai.2020.08.001

I. Introduction

CLOUD computing is defined by the National Institute of Standards and Technology (NIST) as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable Computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" [1]. Cloud computing provides a number of opportunities, such as enabling services to be used without any understanding of their infrastructure, and that data and services are stored remotely but are accessible from anywhere. This way of remote storage is the most important cloud service because it allows cloud users to store their data from local storage systems to the cloud. According to the NIST's classification, the four major patterns of cloud deployment are the private cloud, community cloud, public cloud, and hybrid cloud [1]. Cloud service models are classified as software as a service (SaaS), platform as a service (PaaS) or infrastructure as a service (IaaS).

Once the data is stored on the cloud service platform, the data owners lose control over it. Although this technology offers many advantages, it also introduces new security challenges, especially those related to the integrity of the data. Data integrity is one of the most critical elements in any system. To ensure the integrity of outsourced data, a data owner should enable auditing mechanisms. Auditing is a process of analysis and verification, performed by an internal or external auditor, with the aim of identifying the security vulnerabilities of a system. In our paper, we use the auditing process to check the data integrity of the outsourced data.

* Corresponding author.

E-mail addresses: mohamed.elghazouani63@gmail.com (M. El Ghazouani), kiram@uca.ac.ma (M. A. El Kiram), errajy.latifa@gmail.com (L. Er-Rajy), elkhanboubi.yassine@gmail.com (Y. El Khanboubi).

The second important requirement when storing user data is storage efficiency. Data deduplication is the best choice for ensuring data storage efficiency. Data deduplication (also called intelligent compression or single-instance storage) eliminates redundant data and keeps just one copy of each file before the transfer of the data to be saved in the cloud server (deduplication on the client side is called source-based deduplication), or after it is transferred (deduplication on the server side is also called target-based deduplication). This technique means that multiple copies of the same data are not stored, which allows a reduction in data volumes and thereby reduces storage overhead.

Several cloud data integrity auditing protocols have been proposed in the last few years. In these protocols, the basic system model describes the various entities and their participation in the system, and the threat model highlights threats to an owner's data.

A. System Model: Private and Public Auditing

Several schemes are based on a private auditing system, which means that the data owner who audits the integrity of his data. In this type of auditing system, there are just two entities; the data owner and the cloud storage service (CSS):

- Data owner: the proprietor of the data; he is dependent on the cloud service provider for the proper maintenance of the data.
- Cloud Storage Server (CSS): the cloud service provider, who provides space to store an owner's data.

Fig. 1 shows a private auditing system. This system model provides the data owner with authority only to interact with the CSS to audit

data integrity and conduct data structure operations on outsourced data, whilst the readers only have the authority to read data.

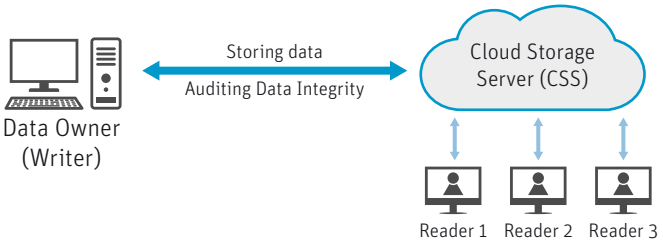

Fig. 1. Basic private auditing system.

Public auditing allows a third party to audit data integrity rather than the data owner. There are three entities in this type of system: the data owner, cloud storage server and a third party auditor (TPA). The TPA has the ability to access the services afforded by the CSS, and therefore, the data owner requests them to check the integrity of their data.

Fig. 2 shows a public auditing system. This system model provides authority only to a TPA to interact with the CSS to audit data integrity. The TPA can significantly alleviate the auditing costs of users.

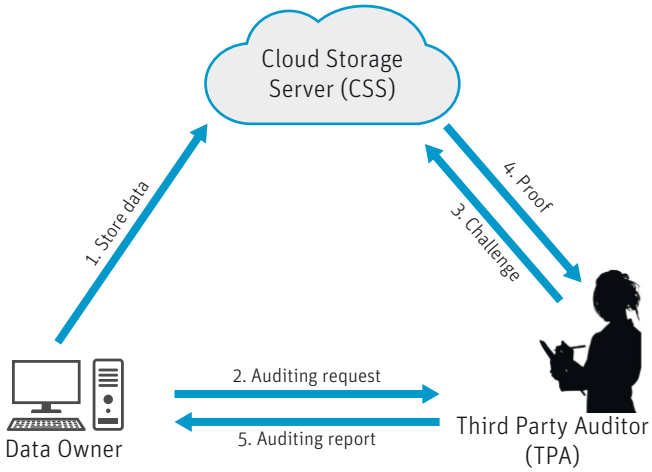

Fig. 2. Basic public auditing system.

B. Threat Models

A data owner assumes that a TPA is a reliable and honest entity that will verify the integrity of their data, but the TPA may be curious about that data. The TPA could thus be a threat for the data owner. To ensure the correct storage of the owner's data with the CSS, a privacy protection mechanism, which guarantees a TPA cannot access the owner's data, will thus be necessary.

A CSP also cannot be fully trusted; it can pose a threat to an owner's data. In order to save space, a CSP may remove data that is rarely accessed without any notification to the data owner. The outsourced data may be tampered with or even re-outsourced without notice by malicious a CSP. A CSP can also apply the wrong changes to an owner's data owing to system failure, management errors or other reasons, and hide these mistakes to protect their image. Undiscovered strangers may also be able to intrude on the cloud server and contaminate or erase an owner's data. When data is stored on a CSS and to respond to a TPA's query, CSS can use an authentic pair of data blocks as a substitute for the queried data blocks just to pass out the audit. The CSS can also retrieve the previously stored results of data that has been challenged simply to generate proof of data possession rather than to query the owner's data.

C. Our Method Goals

Motivated by data integrity and deduplication, we propose a new method for storage and auditing in cloud computing, based on the blockchain data structure. The proposed method achieves the following functions:

- **Confidentiality**: ensures the confidentiality of the owner's data against the TPA during the auditing process.
- **Batch auditing**: ensures that a mediator or TPA (depending on the auditing type) performs multiple auditing tasks, in a simultaneously way, received from different users.
- **Client-side deduplication** (storage efficiency): allows the mediator to eliminate duplicated files and file-blocks before sending the data to the cloud.
- **Private auditing**: allows only the mediator to verify the correctness of the data stored in the cloud.
- **Public auditing**: allows the TPA to check the correctness of the data stored in the cloud.
- **Data integrity**: ensures that the CSP cannot cheat and pass the auditing process without having stored the data intact.
- **Lightweight**: provides the model with low communication and computational overheads.

In this paper, we propose a new method that ensures both efficient storage based on data deduplication on the client side, and preserves data integrity auditing using blockchain technology in a cloud computing environment. The structure of this paper as follows: Section II outlines the various related works. Section III presents the different concepts used in our proposed method. Section IV provides a detailed description of our proposed method. Section V includes security analysis and performance evaluation. Finally, a conclusion is presented in Section VI.

II. Related Work

Many auditing protocols have been established to ensure the correctness of data stored in the cloud. Ateniese et al. [2] proposed a provable data possession (PDP) scheme. In this model, the third party auditor was allowed to statically check the correctness of the outsourced data without retrieving the data. The main goal of this model is to check that the server has the original data. Another improved version of this protocol is the E-PDP [3], which is 185 times faster compared to the first protocol. Proof of retrievability (POR) is another variation of PDP, proposed by Juels and Kaliski[4]. The main drawback of the above protocols is that they do not allow dynamic data auditing.

Erway et al. [5] proposed the concept of dynamic provable data possession (DPDP). According to this scheme, a data owner is allowed to modify the stored data. The main drawback of this scheme is that it cannot support public auditing. Wang et al. [6] resolved the above two problems by applying a Merkle Hash Tree (MHT) and presenting a public and dynamic auditing scheme, however, this scheme involves more computational costs during the updating and auditing phases.

Liu et al. [7] expanded MHT to rank-based MHT (R-MHT) with efficient verifiable fine-grained updates. Zhang and Blanton [8] improved the MHT scheme to include a balanced update tree. To minimize computation and communication costs, Zhu et al. [9] presented a new auditing scheme known as index-hash table-based public auditing (IHT-PA), however, it is inefficient for dynamic updating operations. Tian et al. [10] introduced a new scheme based on a dynamic hash table (DHT), which supports public and dynamic auditing. This scheme achieves better performance in the updating phases. Tang and Zhang [11] proposed a verifiable data possession (PVDP), which allows both private and public verifiability simultaneously, to check the integrity of

client files stored on a cloud server without downloading all those files. Li, Tan and Jia [12] proposed a simple efficient auditing scheme for checking the integrity of data stored in the cloud. This scheme supports dynamic operations and batch auditing.

Yu et al. [13] proposed an identity-based auditing scheme for checking the integrity of cloud data, but Xu et al. revealed that this scheme is vulnerable to data recovery attack. They thus presented a secure and efficient identity-based public auditing scheme using the RSA algorithm for cloud storage [14]. Lee et al. [15] presented a new data integrity check scheme for remotely acquired and stored stream data.

Yuan and Yu [16] proposed a public and constant cost storage integrity auditing scheme with secure deduplication (PCAD). Zhang et al. [17] introduced a fast asymmetric extremum content-defined chunking algorithm for data deduplication in backup storage systems. Gaetani et al. [18] proposed a blockchain-based database to ensure data integrity in cloud computing environments. El Khanboubi and Hanoune [19] proposed a new scheme exploiting blockchains to improve data upload and storage in the cloud.

III. Concepts Used in our Proposed Method

In our proposed method, we introduce the use of five concepts. Blockchain [20] is able to effectively ensure the integrity and authenticity of the exchanged data, and especially auditability by providing a private layer where cloud data is treated and stored in less time. The security of blockchain technology is enforced in a distributed and public way rather than relying on a central party to do so, as is the case for other databases. The blockchain has appeared as a fascinating technology which offers compelling and imperious properties about data integrity. A Merkle Hash Tree [21] is a binary tree that represents the data structure used in blockchain technology. A Third Party Auditor has the ability to check the data possession of the Cloud. The data deduplication technique eliminates redundant data and stores just one copy of each file or file-blocks (chunks) in order to minimize both network traffic and storage space. A mediator performs data deduplication to eliminate duplicated files and file-blocks.

A. Third-Party Auditor

A TPA, who has considerable computation and communication ability, is delegated by the cloud user to check the data possessed by the cloud. TPA is a semi-trusted entity with the expertise and ability to check the correctness of data on behalf of the data owner. The data owner, who employs the TPA to verify the integrity of their data, is alleviated from the burden of expensive auditing operations. Although the data owner has confidence in the TPA's data checking, they can be also a threat to the data owner. One of the most important issues in the data audit process is thus preventing data leakage and preserving the privacy of data.

B. Deduplication Technique

In this paper, we explore the technique of deduplication in the server where many thin clients are connected. For instance, many users from an enterprise-x may intend to outsource a large quantity of data to the cloud and many of these files or file-blocks are duplicates. It is therefore necessary to find and remove duplication within the data. Thus, we decided to use a mediator with the ability to manage client-side data deduplication. By transferring only a single copy of duplicate data, a deduplication system optimizes storage and bandwidth efficiency in cloud storage servers. Accordingly, client-side deduplication implies low communication and computation costs between the client and the CSP, and saves storage space. A file can be divided into many file-blocks (chunks) that can be part of many files. Chunking is an essential step achieving data deduplication, which permits a reduction in the storage space and alleviates the outgoing network traffic when uploading data to the cloud storage server.

Chunking is a challenging technique in the deduplication process, but it can be performed within several algorithms [22]: File-Level Chunking (FLC), Fixed-Size Chunking (FSC), Variable-Size Chunking (VSC), Content-Aware Chunking (CAC). In our proposal, we use a CAC algorithm, where the file is divided based on its content which improves the file-blocks reuse probability, unlike an FSC that splits a file into equally sized file-blocks which reduces the probability of using the same file-block in other files. Consequently, the CAC algorithm outperforms the FSC algorithm in terms of deduplication efficiency and has been extensively used in various storage systems.

C. Mediator

To reduce computational operations among users and to perform data deduplication using a central node in enterprise-x, we decided to use the concept of a mediator. A mediator manages the deduplication process internally in the server, so there is no security issue. The mediator has the ability to manage the storage of a user's data, and even to check the integrity of this data. The mediator has two tasks to perform:

- A client side deduplication to eliminate duplicated files and file-blocks before storing the data in the cloud. Accordingly, the quantity of stored data and the bandwidth used between the client and the Cloud server are both reduced.
- Check the integrity of a user's data stored in the cloud in the case of private auditing, where TPA is absent. It can be seen as an internal auditor, with the proviso that the mediator should have considerable expertise and ability to verify the correctness of the stored data.

IV. Description of the Proposed Method

Blockchain technology brings us many reliable and convenient services, such as preserving the integrity of data, however there are several security issues and challenges behind this innovative technique that should be overcome [23]. In our proposed method, each block in the blockchain database will only store the Merkle root, and the information of the file and the hash of the previous block. The files are not stored in the blockchain, rather they are stored in the CSP servers.

The mediator is trusted and allowed to see the content of the files and their hashes. It computes the file-blocks' hashes and the Merkle root, and then it compares them with a local database of Merkle roots and hashes stored in the previous operations in order to identify duplicated files/file-blocks. The TPA is semi-trusted and allowed to verify the integrity of the files, but it is prohibited from access to the content of the files. The CSP is semi-trusted and allowed to see the content of data, but it is obliged to follow the steps needed for the auditing process.

Each file gives rise to a Merkle hash tree. The Merkle hash tree allows a digest to be made of all the file-blocks linked to that block. File-blocks are not stored in the Merkle Tree, rather their hashes are stored in each node. If a small bit is changed in any file-block, there will certainly be a significant difference between the bit patterns of the resulting Merkle roots. Each Merkle root, generated from the hashes of file-blocks (leaf nodes) corresponding to a file, is stored in a new block in the blockchain with other information describing the file. The Merkle root is fundamental because it relies on the hashes of all underlying file-blocks. It therefore allows efficient and secure verification of data content.

In our proposed method, we use blockchain technology, where information for a file is stored in a block. Each block contains the user ID (U_{id}), the file ID (F_{id}), version number v, timestamp t, the number

of file-blocks N, the Merkle root n_0 and the hash of the previous block in the chain. One block B in the blockchain may correspond to the file F of the user U. The following block C may correspond to file L of the user J. However, the entire tree and the file-blocks, of a file, stored in the CSP database may correspond to one or more blocks in the chain, and thus to one or more users. The lengths in bytes of the different records in a block are: U_{id} 8, F_{id} 8, v 4, t 4, N 8, n_0 32, *HashPrev* 32. Fig. 3 displays the information stored in each block of the blockchain.

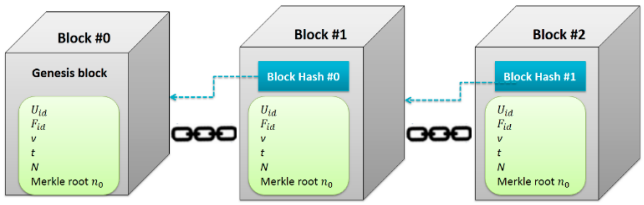

Fig. 3. Blockchain structure used in our proposal.

We decided to use MD5 hashing algorithm that generates a unique 32 chars string. Although the SHA-3 algorithms are more secure as compared to MD5, this latter is better in terms of speed and the hash string length produced is smaller than any other hashing algorithm. According to Yu Sasaki and Kazumaro Aoki [24], even though there have been many powerful collision attacks on MD5, the preimage resistance of MD5 has not been broken yet. Nevertheless, any other hashing algorithm may be applicable.

A. System Model for Private Auditing

In this mode of auditing, the mediator should have the ability to perform the auditing process.

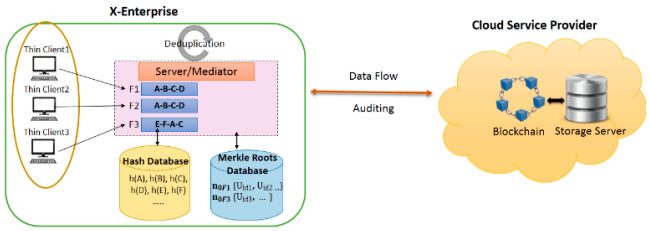

Fig. 4. System model for private auditing.

According to some statistics, more than 75% of the outsourced data in the cloud is not unique [25], and so the manipulation of deduplication could greatly reduce storage cost and the required space to store large data. The use of this technique in this system model thus ensures the maximum use of available storage space through the elimination of redundant data, and the amount of outsourced data and the bandwidth used between the enterprise and the CSP are also both reduced.

The main advantage of this system model is that the audit tasks are performed by an internal entity, which is the mediator, which implies low communication and computation costs. The mediator could therefore perform multiple auditing tasks simultaneously, and received from different users.

B. System Model for Public Auditing

In this mode of auditing, a third party auditor checks the integrity of the data stored in the cloud.

The auditing tasks are delegated to a competent external entity, the TPA, which implies more communication and computation costs. The auditing request could be sent directly by users to the TPA. We also use, in this system model, a technique of deduplication that reduces both storage cost and the bandwidth used between the enterprise and the CSP. A TPA could also perform multiple auditing tasks simultaneously, as received from different users.

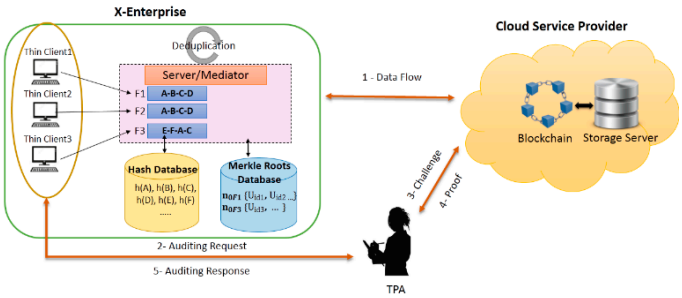

Fig. 5. System model for public auditing.

C. Storage Phase

As shown in our model (Fig. 4 and Fig. 5), when users intend to store their files on the cloud server, the mediator checks the existence of the entirety of each file or some of its file-blocks in the cloud storage server. The mediator therefore initiates data information: U_{id}, F_{id}, v, t, N corresponding to the file. Afterward, the mediator divides the file into N file-blocks using a CAC algorithm $(fb_0, fb_1, \ldots fb_{N-1})$ where $N=2^d$ and d is the depth of the Merkle tree, then he calculates its hashes $h(fb_i)$ with a secure hash function where $0 \leq i \leq N-1$, in order to compute the Merkle root n_0. Thereafter, he computes the Merkle root n_0 (*for* i *in* $N-2....0$: $n_i = h(n_{2i+1} \| n_{2i+2})$) of the file. After that, the mediator compares it with a local database of Merkle roots stored in the previous storage operations in order to identify duplicated files.

1. Case 1: The File Has Never Been Stored

If the generated Merkle root n_0 does not resemble any roots, the mediator keeps the root locally in the Merkle roots database with the U_{id} of the user in order to use them in the next storage operations. After that, the mediator performs file-blocks level deduplication by comparing the generated hashes with those located in the hash database (comparing the file-blocks' hashes will take less time than comparing file-blocks). There are two cases:

a) Case 1.1: Storing All File-Blocks

If the mediator did not find any identical file-block hashes, he stores the hashes of all file-blocks in the hash database, then outsources all file-blocks to the cloud storage server. A new block is then created in the blockchain, this block contains the file's information and the Merkle root that correspond to the file. After that, the CSP stores the entire tree with the file-blocks corresponding to that file. The concerned user thereafter maintains a pointer to the block that corresponds to his file. So that, each user preserves a file ID list; this list contains pointers, each pointer points directly to the particular block in the blockchain corresponding to that file.

b) Case 1.2: Storing Some File-Blocks

If the mediator finds some identical hashes, he ignores them and stores the other file-blocks hashes (unduplicated hashes) in the hash database. After that, he stores the unduplicated file-blocks in the cloud storage server. A new block is then created in the blockchain, this block contains the file's information and the Merkle root that correspond to the file. After that, the CSP stores the entire tree with the uploaded file-blocks corresponding to that file. Hereafter, the concerned user maintains a pointer to the block that corresponds to his file. In this case, the mediator does not need to upload all the file-blocks because some of them have been stored previously by the same or other users, so, the mediator ignores the duplicated file-blocks, which reduces disk utilization.

2. Case 2: The File Has Already Been Stored

a) Case 2.1: The Same User Tries to Store the Same File

If the generated Merkle root n_0 resembles a root which has already been registered in the Merkle roots database, and if the current U_{id} already exists in the user's ID list related to this Merkle root, the mediator informs the file owner that he has already stored this file in a previous storage operation. In this case, the mediator does not need to send the file to the CSP to be stored, which reduces the disk's utilization.

b) Case 2.2: Another User Tries to Store the Same File

If the generated Merkle root n_0 resembles a root which has already been registered in the Merkle roots database, and if the current U_{id} does not exist in the users ID list related to this root, the mediator adds the new U_{id} to the user's ID list corresponding to this Merkle root, and then stores n_0 in a new block in the blockchain with the file information. In this case, the mediator does not need to send the file to the CSP to be stored, because it has been stored by another user in a previous storage operation. Finally, the user maintains a pointer to the block that corresponds to his file.

D. Execution Flow for Storage and Auditing Files

The execution flow for the storage and auditing of files is shown in the Fig. 6.

E. Auditing Phase

For the auditing process, we follow the technique of verification used in [26]. So to verify the integrity of a file:

- The mediator/TPA (depending on auditing type) computes the generator seed $r = h^P(n_0)$ where leaves are divided into P chunks.
- After that, the mediator/TPA derives leaf numbers in each P chunk as: for j in 0 ... P – 1: $l_j = G(r, j)$ with G some cryptographic pseudo-random number generator (PRNG).
- Then, the mediator/TPA sends the leaf numbers $\{lj\}$ to the CSP.
- The CSP provides the appropriate sibling information to the mediator/TPA, which allows the mediator/TPA to compute the new Merkle root n'_0.
- The mediator/TPA verifies whether $n_0 = n'_0$ or not.
- The mediator/TPA then calculates the new generator seed $r' = h^P(n'_0)$.
- The mediator/TPA deduces the leaf numbers $l'_j = G(r', j)$.
- Hereupon, the mediator/TPA checks whether $l'_j = l_j$ for each j *in* $0 \ldots P - 1$, and if they match, then the file is verified.
- Finally, the mediator/TPA informs the cloud client of the results.

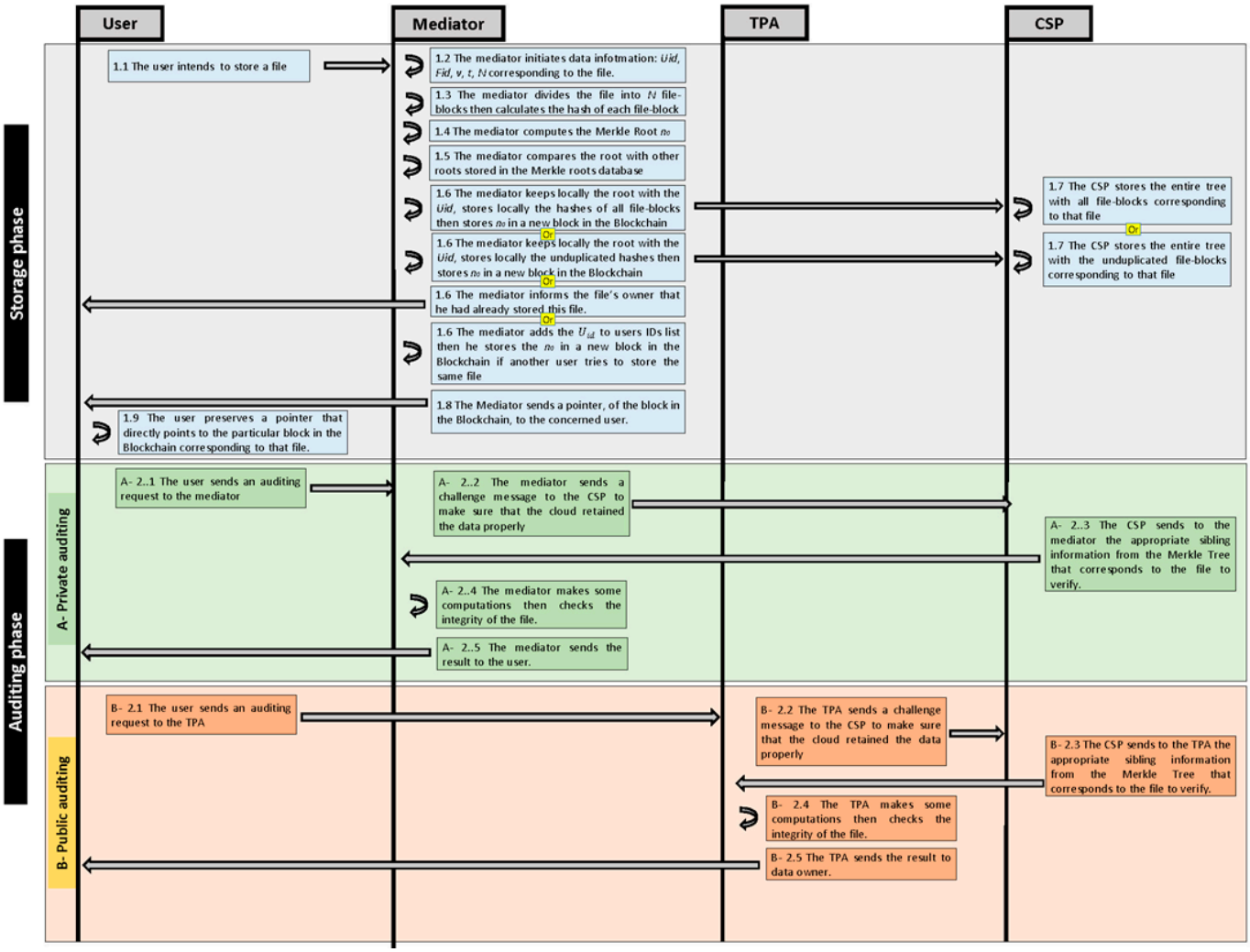

Fig. 6. Execution flow for storage and auditing of files.

V. Security Analysis and Performance Evaluation

Several data integrity auditing schemes have implemented data deduplication in the cloud server side. This way of working has high computational costs. Other schemes follow a fixed-size chunks method which is simple and extremely fast, but this approach suffers from low deduplication efficiency. In our method, deduplication is performed on the client side by the mediator and using a content-aware chunking algorithm. Instead of saving three or four copies of the same file/chunk in the cloud, deduplication allows the elimination of all the redundant data and stores only one copy of the file/chunk that belongs to one or multiple cloud users. This technique decreases communication, computation and storage costs.

Note that in our proposal, we indicate the use of two types of auditing, private auditing and public auditing, to verify the correctness of the data stored in the cloud. It depends on the need of the enterprise; if the mediator has the ability to verify the accuracy of the data, in this case, he manipulates a private auditing, or he prefers the use of an external auditor to verify their data (public auditing).

Owing to the technique that we used during the auditing process, the TPA will have no idea about the owner's data, which implies that the confidentiality of the data is ensured against auditors.

The mediator or TPA (depending on the auditing type) could perform multiple auditing tasks simultaneously, received from different users. In a case where the mediator/TPA receives several auditing requests for the same file from different users, it may be ineffective to handle them as individual tasks rather than to batch them together and perform only one audit task by interacting with the CSP to check the data integrity. After that, it replies all concerned users by the auditing result. The deduplication technique is thus not only efficient for data storage because it reduces storage cost and the required space to store a large data, but it is also efficient for the auditing process where multiple users want to verify the same file while reducing the communication and computation cost between auditor and CSP.

The use of blockchain in our proposal is mainly applicable in scenarios where the data history is very important. This method is practical for real application scenarios, such as in the justice domain where judgments must be stored and must not be modified. It can also be useful for real estate agents, to register property titles for example, where it is forbidden to modify this type of data. It could be also used for storing medical records or even collecting taxes.

To demonstrate that our proposed method is efficient, we performed experiments through an application developed using Java and PHP languages, on a computer with an Ubuntu 17 OS running on an Intel CoreTM i3 CPU with a 2.27 GHz clock and 4 GB RAM. The remote storage was implemented using MySQL. The size of each file is increased by 200 MB. We used large files to show the usefulness of thededuplication technique. Fig. 7 shows a plot of computation time in seconds against input size in MB.

We can see that the computation time for an unduplicated file is greater than the computation time for a duplicated file, which shows that the computation time and the bandwidth, used between the enterprise and the CSP, are both reduced thanks to the deduplication technique.

Several systems perform a public data integrity auditing where the responsibility of data integrity verification is delegated to a third party auditor as in [2]-[4],[6]-[10],[14],[15]. Other schemes manipulate a private auditing where the data owner checks the integrity of the externalized data as in [5]. However, in some cases, it is not practically feasible for the data owner to verify the data integrity all the time. Hence, our approach supports both public and private auditability property.

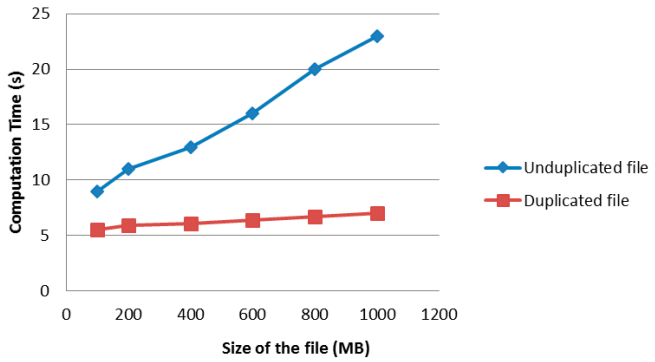

Fig. 7. Comparison of the time computed in storage operations for duplicated and unduplicated files of different sizes.

Some researches mention the use of data deduplication technique to improve storage systems as in [16], [17]. Besides, the decentralization and security characteristics of blockchain technology have attracted many researchers to propose various schemes exploiting blockchain-based databases to ensure data integrity and improve data storage in cloud computing environments as in [18], [19]. The value-added of our method over these methods is that our proposal permits both data deduplication, which guarantees the storage efficiency, and data integrity auditing that verifies the correctness of the outsourced data.

In our system, the deduplication is performed at the file level or block level on the client side. Hashes of files or blocks are computed and stored, and if the hash for any new file or block is found to be present in the stored hashes, then the copy is removed, which permits to eliminate duplicated files and file-blocks before storing the data in the cloud. Accordingly, the quantity of stored data and the bandwidth used between the client and the Cloud server are both reduced. The main benefit of manipulating deduplication in our system is that storage efficiency is increased and network efficiency is enhanced.

As with any system, it is necessary to have a fault tolerance property that enables the system to continue its operation properly in the event of the failure of any of its components.

VI. Conclusions

In this paper, we demonstrated the feasibility of using deduplication and blockchain technologies to manage storage and auditing processes for cloud data.

We proposed the concept of the mediator, who performs client-side data deduplication to identify and remove duplicated files and file-blocks using a content-aware chunking algorithm, which reduces the computation costs on the user's side, communication costs in the channel and storage costs in CSP.

Our method relies on blockchain to perform the auditing process in a transparent and lightweight way. The main goal of using blockchain is that TPA can check the integrity of the outsourced data without gaining any knowledge of the user's data. Consequently, we can consider the use of blockchain as a new model for trust.

References

[1] P. Mell and T. Grance, "The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology," *Nist Spec. Publ.*, vol. 145, p. 7, 2011. https://doi.org/10.1136/emj.2010.096966.

[2] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson *et al.*, "Provable data possession at untrusted stores," *Proc. 14th ACM Conf. Comput. Commun. Secur. - CCS '07*, p. 598, 2007. https://doi.org/10.1145/1315245.1315318.

[3] G. Ateniese, R. Burns, R. Curtmola, J. Herring, O. Khan, L. Kissner *et al.*, "Remote data checking using provable data possession," *ACM Trans. Inf. Syst. Secur.*, vol. 14, no. 1, pp. 1–34, 2011. https://doi.org/10.1145/1952982.1952994.

[4] A. . Juels and B. S. Kaliski Jr., "Pors: Proofs of retrievability for large files," *Proc. ACM Conf. Comput. Commun. Secur.*, pp. 584–597, 2007. https://doi.org/10.1145/1315245.1315317.

[5] C. C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, "Dynamic Provable Data Possession," *ACM Trans. Inf. Syst. Secur.*, vol. 17, no. 4, pp. 1–29, 2015. https://doi.org/10.1145/2699909.

[6] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 5, pp. 847–859, 2011. https://doi.org/10.1109/TPDS.2010.183.

[7] L. Chang, J. Chen, L. T. Yang, X. Zhang, C. Yang, R. Ranjan *et al.*, "Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 9, pp. 2234–2244, 2014. https://doi.org/10.1109/TPDS.2013.191.

[8] Y. Zhang and M. Blanton, "Efficient Dynamic Provable Possession of Remote Data via Update Trees," *ACM Trans. Storage*, vol. 12, no. 2, pp. 1–45, 2016. https://doi.org/10.1145/2747877.

[9] 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. Serv. Comput.*, vol. 6, no. 2, pp. 227–238, 2013. https://doi.org/10.1109/TSC.2011.51.

[10] T. Hui, Y. Chen, C. Chen Chang, H. Jiang, Y. Huang, Y. Chen *et al.*, "Dynamic-Hash-Table Based Public Auditing for Secure Cloud Storage," *IEEE Trans. Serv. Comput.*, vol. 10, no. 5, pp. 701–714, 2017. https://doi.org/10.1109/TSC.2015.2512589.

[11] C. ming Tang and X. jun Zhang, "A new publicly verifiable data possession on remote storage," *J. Supercomput.*, vol. 75, pp. 77–91, 2019. https://doi.org/10.1007/s11227-015-1556-z.

[12] A. Li, S. Tan, and Y. Jia, "A method for achieving provable data integrity in cloud computing," *J. Supercomput.*, vol. 75, pp. 92–108, 2019. https://doi.org/10.1007/s11227-015-1598-2.

[13] Y. Yong, L. Xue, M. H. Au, W. Susilo, J. Ni, Y. Zhang *et al.*, "Cloud data integrity checking with an identity-based auditing mechanism from RSA," *Futur. Gener. Comput. Syst.*, vol. 62, pp. 85–91, 2016. https://doi.org/10.1016/j.future.2016.02.003.

[14] Z. Xu, L. Wu, M. K. Khan, K. K. R. Choo, and D. He, "A secure and efficient public auditing scheme using RSA algorithm for cloud storage," *J. Supercomput.*, vol. 73, no. 12, pp. 5285–5309, 2017. https://doi.org/10.1007/s11227-017-2085-8.

[15] K. M. Lee, K. M. Lee, and S. H. Lee, "Remote data integrity check for remotely acquired and stored stream data," *J. Supercomput.*, vol. 74, no. 3, pp. 1182–1201, 2018. https://doi.org/10.1007/s11227-017-2117-4.

[16] J. Yuan and S. Yu, "Secure and constant cost public cloud storage auditing with deduplication," in *2013 IEEE Conference on Communications and Network Security, CNS 2013*, 2013, pp. 145–153. https://doi.org/10.1109/CNS.2013.6682702.

[17] Z. Yucheng, D. Feng, H. Jiang, W. Xia, M. Fu, F. Huang *et al.*, "A Fast Asymmetric Extremum Content Defined Chunking Algorithm for Data Deduplication in Backup Storage Systems," *IEEE Trans. Comput.*, vol. 66, no. 2, pp. 199–211, 2017. https://doi.org/10.1109/TC.2016.2595565.

[18] E. Gaetani, L. Aniello, R. Baldoni, F. Lombardi, A. Margheri, and V. Sassone, "Blockchain-based database to ensure data integrity in cloud computing environments," *CEUR Workshop Proc.*, vol. 1816, pp. 146–155, 2017.

[19] Y. El Khanboubi and M. Hanoune, "Exploiting Blockchains to improve Data Upload and Storage in the Cloud," vol. 11, no. 3, pp. 357–364, 2019.

[20] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," *Www. Bitcoin.Org*, p. 9, 2008. https://doi.org/10.1007/s10838-008-9062-0.

[21] R. C. Merkle, "I NFORMAT I ON SYSTEMS LABORATORY By," 1979.

[22] Siva Sankar K. Venish A., "Study of Chunking Algorithm in Data Deduplication," *Proc. Int. Conf. Soft Comput. Syst. Adv. Intell. Syst. Comput.*, vol. 398, pp. 319–329, 2016. https://doi.org/10.1007/978-81-322-2674-1.

[23] I. C. Lin and T. C. Liao, "A survey of blockchain security issues and challenges," *Int. J. Netw. Secur.*, vol. 19, no. 5, pp. 653–659, 2017. https://doi.org/10.6633/IJNS.201709.19(5).01.

[24] Sasaki Y., Aoki K. (2009) "Finding Preimages in Full MD5 Faster Than Exhaustive Search" *In: Joux A. (eds) Advances in Cryptology - EUROCRYPT 2009. Lecture Notes in Computer Science,* vol 5479. Springer, Berlin, Heidelberg.

[25] J. Gantz and D. Reinsel, "The Digital Universe Decade – Are You Ready?," *Idc*, vol. 2009, no. May, p. 16, 2010.

[26] F. Coelho, "An (almost) constant-effort solution-verification proof-of-work protocol based on Merkle trees," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5023 LNCS, pp. 80–93, 2008. https://doi.org/10.1007/978-3-540-68164-9_6.

Mohamed El Ghazouani

He is a Doctor at the Computer Science Department of Cadi Ayyad University in Marrakesh, Morocco. He received his Master's degree in Information Systems Engineering from the same university in 2015. His research interests are Computer Science, Cloud Computing Security, Big Data and Blockchains.

My Ahmed El Kiram

He is a full Professor of Computer Science at the Department of Computer Science, Faculty of Science Semlalia, Cadi Ayyad University, Morocco. He writes and presents widely on issues of IT Security, Cryptographic Systems and Cloud Computing. He is the author of numerous publications related to his research interests.

Latifa Er-Rajy

She is a Doctor at the Computer Science Department of Cadi Ayyad University in Marrakesh, Morocco. Her research interests are Android Applications, Mobile security and Security in Cloud Computing. She is the author of several publications related to his research interests.

Yassine El Khanboubi

Hereceived his Master's degree in Information Systems Engineering from Cadi Ayyad University Marrakesh, Morocco. He is currently a PhD candidate at the Hassan II university. His research interests include Computer Network Security, Mobile and Wireless Communication Security.