EAI Endorsed Transactions
on Cloud Systems
Research Article EAI.EU

Outsourced Auditing With Data Integrity Verification Scheme (OA-DIV) and Dynamic Operations for Cloud Data with Multi-Copies

Arjun U[1], Vinay S[2]

[1]Department of Information Science and Engineering, PESITM, Shimoga Karnataka, India
arjuninformation@gmail.com
[2]Department of Information Science and Engineering, P E S College of Engineering, Mandya, Karnataka, India
vinaymanyan@gmail.com

Abstract

Data storage in cloud is widely utilized by different commercial, educational, scientific, healthcare applications and much more. The major concern apart from data security is data integrity. Our work focuses on data integrity. In spite of the presence of Service Level Agreement (SLA) between the data owner and CSP, the data integrity may be affected and also todays' cloud computing platform processes more of real-time data, which involves dynamic data operations. Most of the existing works intend to verify the cloud data integrity however the issues related to data replicas are not usually considered and most of the existing cloud techniques fail to focus on dynamic data operations. This article presents an outsourced auditing with data integrity verification scheme (OA-DIV) scheme that can handle multiple copies of cloud data and supports dynamic data operations such as data insertion, deletion and updation. The performance of the proposed work is tested with respect to communication cost and processing time, while comparing the results with the existing approaches.

Keywords: Cloud Computing, Data Auditing, Integrity, Dynamic Operation.

Received on 08 March 2021, accepted on 19 April 2021, published on 27 April 2021

doi: 10.4108/eai.27-4-2021.169423

1. Introduction

Cloud storage service brings numerous attractive benefits to the organizations and other business communities, such that it is employed widely. Though the hassles involved in data storage are addressed by cloud service provider, the major challenging concerns are data integrity and privacy. The data owners are concerned about the consistency of the outsourced data, as any modification or tamper may lead to serious problems, in addition to data recovery. There are several other related problems such as maintaining the data integrity of multiple copies of cloud data and performing dynamic operations over cloud data.

Hence, the data owners perform data integrity checks then and there, in order to confirm the correctness of the data. There are various techniques in the existing literature for checking the integrity of data. One of the famous verification techniques for cloud data integrity is the Provable Data Possession (PDP) [1, 2]. The PDP based techniques employ the sampling process for data integrity verification and the data inconsistency can immediately be detected without the need of downloading the complete data. However, this method cannot perform well when the data volume is more and incurs high computational cost.

Proof of Retrievability (PoR) is another technique, which helps to verify the data integrity and based on this technique, the authors of [3,4] proposed a data proof with retrievability. However, this approach requires 'sentinels' and can provide a static count of audits alone. Additionally, most of the existing works focus on audit operation alone and the data organization or the data duplicate management are rarely discussed.

In order to increase data availability, multiple copies of the outsourced data are stored in the remote cloud servers that exist in different geographical locations. The main challenge associated with multiple data copies is that when the parent data is modified, all the replicas are needed to be

modified. Failing to achieve this leads to data inconsistency, which seriously affects the performance of the cloud system.

The real-time scenario demands the cloud environment to be more realistic by providing support to dynamic operations that includes data insertion, updation and deletion [5-11]. However, most of the existing literature are based on static data and do not support dynamic operations. The proposed work intends to address this issue by presenting a scheme for cloud computing environment, which can handle dynamic data operations such as data insertion, deletion and updation [40-45].

Understanding the necessity of data organization and duplication management while performing data audit, this paper presents a OA-DIV scheme that focuses on both the above mentioned issues. The work highlights of this paper are listed as follows.

- This paper considers multiple replicas of the cloud stored data.
- The data organization is taken care of by means of tree based structure.
- Perform the dynamic operations such as data insertion, deletion and updation.
- The audit effectiveness of the work is proven to be better than the compared schemes.

The remainder of this paper is arranged in the following manner. Section 2 discusses the related review of literature, section 3 presents the proposed auditing scheme and section 4 presents the dynamic operations. The performance of the proposed work is tested in section 5 and the conclusions are summarized in section 6.

2. Literature Review

This section aims to review the existing literature with respect to data integrity preservation and dynamic operation in cloud computing environment.

In [11], a user revocation scheme is presented for identity based cloud storage auditing meant for big data. This work is based on effective key generation and the updation of private keys. The user revocation is carried out by upgrading the private keys of non-revoked users and not considering the authenticators. This work does not require any complex certificate management schemes, as in the case of standard Public Key Infrastructure (PKI) systems.

A data integrity scheme for cloud computing environment is presented on the basis of distributed machine learning in [16]. Initially, a Provable Data Possession (PDP) sampling based auditing algorithm is presented for verifying the data integrity. A random number is then generated and the Discrete Logarithm Problem (DLP) is utilized for proof formation. Finally, an identity based cryptography is employed for generating the public/private keys.

In [13], a trustworthy data integrity is presented for cloud storage system by employing collaborative auditing block chain mechanism. The authors of this work intend to handle the trust issue between the data owners and Cloud Service Providers (CSP). Here, all the nodes perform auditing assignments and store them. An attribute based cloud data integrity auditing scheme is proposed in [14] for cloud storage. This work defines specific attribute set for enabling the users to upload data. This work claims to provide privacy preservable attributes and avoids collusion attacks.

A data integrity auditing scheme based on algebraic signature is presented in [15]. This work focuses on confidentiality through batch auditing process. This work supports dynamic operations on data as well and the security analysis of the work is performed. In [16], a privacy-preserving audit scheme is proposed for cloud computing, which also can support dynamic data operations of Unmanned Aerial Vehicles (UAV). This technique works on the basis of distributed string equality check protocol in addition to Merkle hash tree. Initially, a Third Party Auditor (TPA) is included that takes care of digital sign, integrity checks and dynamic data operations. The security is further improved by means of equality check protocol.

In [17], a publicly verifiable with data deletion scheme is proposed for cloud computing. This work focuses on public and private verifiability of the data storage and deletion with the help of invertible bloom filter. Suppose, when the cloud server does not work normally, then the users can detect the abnormality with the probability. The auditing services for cloud storage systems are presented on the basis of Merkle hash tree in [18]. This work detects the suspicious entities, while supporting the dynamic updation.

An integrity check method is performed on cloud storage with multi-copy data in the work of [19]. The key generation process is performed by bilinear mapping and the multi-copy data is handled by multi-branch authentication tree. The correlation of tasks is represented by Directed Acyclic Graph (DAG) by considering the Quality of Service (QoS) demand.

In [20], a lattice based verifier auditor scheme is presented for electronic medical data in cloud storage. This lattice based scheme allows a patient to employ a verifier for performing data audit. A Provable Data Possession (PDP) protocol without certificate is presented for handling multiple copies in cloud is proposed in [21]. This work presents an identity based PDP protocol that can maintain multiple data copies on multiple cloud servers. The security model is built initially and the protocol is formed.

In [22], a lightweight verifiable data aggregation that ensures privacy preservation is proposed. Pailier homomorphic encryption scheme with signature technique is proposed for privacy preservation. The data integrity is provided by q-Strong Diffie Hellman (q-SDH) assumptions. A PDP is presented for dynamic multiple copies of data in cloud storage [23]. The vector dot products are utilized in PDP and the dynamic data structure is presented on the basis of Divided Address Version Mapping Table (DAVMT).

A certificate based auditing scheme is presented on the basis of asymmetric bilinear pairing in [24] for cloud storage. This work is implemented on type D curve of pairing library and the authors claim that the tag generation cost is minimized. In [25], a data integrity scheme is proposed on the basis of short signature for cloud based Internet of Things (IoT). The utilization of short signature algorithm verifies the

data integrity and preserves privacy, while the public auditing is performed by TPA.

In [26], a data integrity verification scheme is proposed on the basis of cryptographic accumulator. This work permits the data owner to perform several rounds of integrity checks. Dynamic data operations are supported and data leakage, replay attacks are tackled by this work. In [27], a dual access control and data integrity verifiable scheme is proposed for cloud computing applications. A time tree is formed for attribute based encryption by employing hierarchical identity based encryption for fixing the access and decryptable time for the keys and data. The data decryption is possible upon the meet of access conditions of data owner. The data verification tree is built by Merkle hash tree.

An identity based data integrity audit with sensitive data hiding scheme is proposed in [28]. Here, all the sensitive data blocks are sanitized and the signatures of these blocks are transformed. The signatures are employed for verifying the integrity of data. In [29], a fuzzy identity based data auditing method is proposed. This work is based on Merkle hash tree and Index Logic Table (ILT), which can support dynamic operations.

The authors of [30] improved the basic scheme on the basis of the PDP approach and suggested an accurate and stable PDP scheme. This work employs symmetrical encryption and protection under the Random Oracle model could be verified, including dynamic data operation, i.e. data changes, such as deletion and attachment. The user sets the total count along with the challenge content and saves the answer as metadata, when the scheme is initialised. However, public auditing is not maintained in this approach, because of the use of symmetric cryptography. Despite the improvisation, this work has rendered little focus on storage and computing performance, as well as lots auditing and privacy.

The authors of [31] suggested two complex PDP schemes by introducing a PDP scheme that could support all dynamic data operations. Initially, this work employs a level authentication jump table and the second is on the basis of RSA tree structure. The objective is to maintain dynamics, especially insertion and so this scheme focuses to compute over high-cloud servers. Yet, this work focuses on the efficiency of contact and dynamic operation alone.

A data protection and public audit system that would be more useful in practise in the area of computation and overhead communication is discussed in [32]. Asymmetric pairing was used to improve performance and also to support complex operations in their scheme. Their scheme demonstrated increased productivity compared to previous works. This paper does not address problems such as storage performance, number of verifications, batch auditing and proven protection.

In [33], a technology known as a balanced update tree is utilized. In terms of the parameters, the authors concentrated on the process of the data update. The average measurement and coordination of this scheme is minimal than the existing schemes.

In [34], the authors suggested the implementation of a stable signature system and the broad branching tree for a new dynamic PDP scheme (LBT). Their framework provides support to complete dynamic modifications with adjustment, insertion and removal. By modifying the Merkle Hash Tree (MHT) in association with LBT, the effectiveness of the communication costs is increased. They use a safe signature algorithm that significantly reduces both CSP and client computing costs.

The authors of [35] suggested the first dynamic POR system in order to handle dynamic data safely. This work presents a new POR method, which inherits the dynamic data setup, known as fairness. Due to confusion, fraudulent consumers are genuinely able to exploit their data with truthful cloud storage servers. They also founded two new tools, one is a 2-3 tree authentication data structure (rb23Tree), and the other the incremental signature method, which is called Hash-Compress-And-Sign (HCS). Help for dynamic activity is the subject of this paper.

The issue of performance is focussed in [36]. They also expanded the static POR to a dynamic framework, where a consumer can make changes that include insert, delete and change. The authors built a fresh version of a B+ with a Merkle hash tree authenticated data structure called Cloud Merkle B + tree (CMBT), while presenting a dynamic POR scheme in combination of the CMBT with BLS signature.

In [37], a dynamic storage POR scheme is suggested that allows clients to read and write arbitrarily at any server location by using a protocol efficient and performing audit protocols to ensure a server maintains the latest data edition. The complexity of computation and communication in their protocol was only multi-log data size. The concept is to divide the data into several tiny blocks and to encrypt every block separately, so that the internalisation of the data block would only affect those code character numbers. Computing and coordination complexity were significantly reduced in their protocol. Other requirements were not considered except for computational efficiency and complex operational support.

The efficiency is addressed in dynamic POR scheme using homomorphic controls with constant customer storage in [38]. They also demonstrated how their scheme can be tested publicly. In [39], a new cloud storage approach called OPoR is proposed to improve the POR model to help complex data operations and withstand cloud storage server reset attacks during the upload process. OPoR supplies heavy tag generation calculations to the cloud audit server and removes user participation in the audit and pre-processing phases. However, this article considers the data dynamics and safety. Motivated by these works, this article presents a data integrity verification scheme that can handle multiple copies of the cloud data along with the dynamic data operation.

3. Proposed OA-DIV Cloud Data Auditing Scheme

The proposed OA-DIV cloud auditing scheme is elaborated in this section, which includes all the involved modules and sub-modules. This work splits the work into three modules

such as Cloud Data Owner (CDO), Third Party Auditor (TPA) and CSP. For each module, certain sub-modules are assigned and are overviewed in this section. The CDO module is comprised of phases such as key establishment, data copy creation, signature creation and so on. The auditing strategy formation and integrity challenge placement come under the control of TPA. CSP responds to the challenge with the proof. The overall flow of the work is shown in figure 1. All these modules are outlined in the following sub-sections.

3.1 CDO

The CDO is the owner of the outsourced data and is the most concerning entity regarding the data integrity and security. The integrity of the data is maintained as per the Service Level Agreement (SLA), however in certain cases the SLAs are violated by the CSP. Thus, there is a need to check the data integrity at timely intervals. The CDOs are responsible for carrying out the following.

- Key Establishment – The CDO runs the key establishment algorithm for providing the public (pb) and private (pr) keys.
- Data Copy Creation – The copy of data is created for ensuring better data availability and this algorithm generates the copy. Whenever a data file (D) is passed as input, a group of copied data blocks (D') are formed.
- Signature Creation – The signature creation algorithm is meant for creating signature blocks in order to provide authentication. The inputs of this algorithm are pr and D, while it returns the signature block β and the signature value of the root SV. This processed data block D along with β and SV are passed to the storage server.

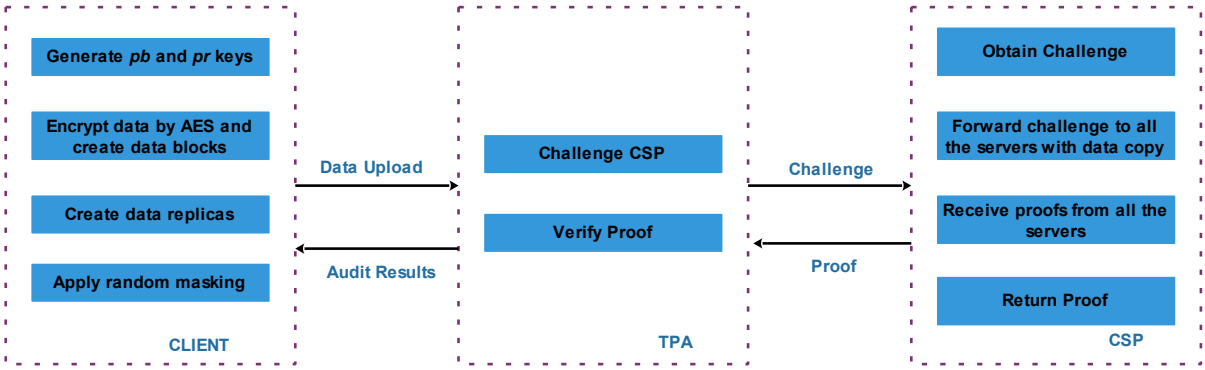

Figure 1. Overall flow of the OA-DIV Scheme

3.2 TPA

TPA is responsible for checking the integrity of data by passing a challenge to the CSP, in order to verify the data correctness. The activities performed by the TPA are forming the audit strategy formation and integrity challenge placement.

- Audit strategy formation – This algorithm forms the audit strategy, which is being executed by the TPA. This algorithm accepts a set of data attributes DA and returns audit strategy QA, which includes the probability of file tamper (pb), data access rate (ar), timestamp (TS).
- Integrity challenge placement – The challenge details are formed by this phase, which accepts the data block group, which is needed to be tested and a random number RN=(i,q_i). The challenge message is forwarded to the cloud storage server.

3.3 CSP

The CSP is the entity that provides the user with the cloud storage space that follows SLA. Whenever the CSP is challenged by the CDO for the proof of data integrity, the CSP has to respond with the proof of integrity.

- Proof Formation – This phase intends to present the data integrity proof for the data being challenged. The input of this phase are the data D', β and CHAL, for which the algorithm returns the proof P.

As soon as the proof is received, the TPA verifies the P and returns with either TRUE or FALSE. The value TRUE indicates the proof is verified and vice-versa. Hence, the CDO generates β and the root node's signature, where every parent node authenticates its child node and the child node authenticates the data. This idea makes it simple to maintain the data replicas and to check the integrity of the data. All the phases are explained as follows.

3.4 Proposed Work

Let CG_1 and CG_2 be the cyclic groups with PMprime order and cg_1, cg_2 form these cyclic groups. The bilinear map is shown as $e: CG_1 \times CG_1 \rightarrow CG_2$, anti-collision hash function $HF: MS \rightarrow \{0,1\}$, where MS is the message space. Choose $a_0, a_1, a_2, \ldots, a_n \leftarrow \mathbb{Z}_q$ and $HS \in CG_2$.

Compute $HS_0 = H^{\frac{1}{a_0}}, HS_i \in CG_2, i = \{1,2,\dots,n\}$, in order to public keys and private keys denoted by $pb = (HS, HS_0, HS_1, \dots, HS_n)$, $pr = (a_0, a_1, a_2, \dots, a_n)$.

The data (D) is encrypted by applying the elegant and mathematically efficient cryptographic Advanced Encryption Standard (AES) algorithm and the data blocks (db) are formed upon the encrypted data $(E(D))$, such that the equation for encrypted data $E(D) = (db_1, db_2, \dots, db_n)$. The CDO creates R number of replicas for the data, which is represented by the following.

$$E(D) = \{db_{r,1}, db_{r,2}, \dots, db_{r,n}\}; 1 \le r \le R \quad (1)$$

Random masking technique is then applied for altering the data and the condition is given as follows.

$$D' = (D_r)_{1 \le r \le R} \quad (2)$$

$$= \{bl_{r,i}\}_{1 \le i \le n, 1 \le r \le R} \quad (3)$$

$$= \{bl_{r,1}, bl_{r,2,\dots}, bl_{r,n}\} \quad (4)$$

Here,

$$bl_{r,i} = db_{r,i}^{l} + y_l \quad (5)$$

$$db_{r,i} = \{db_{r,i}^{1}, db_{r,i}^{2}, \dots, db_{r,i}^{n}\} \quad (6)$$

$$y_l = \{f(HS_i||r||l); 1 \le i, l \le n; 1 \le r \le R \quad (7)$$

This value is formed by the pseudo random number generator. All the data copies are maintained by a tree structure and when the challenge is forwarded to the CSP, the following actions take place.

Whenever a CHAL message is received by the CSP, the children nodes of a specific data blocks are checked and certain information are obtained.

$$CHAL = \{(i, q_i)\}_{l_1 \le i \le l_t} \quad (8)$$

This is followed by the detection of the copies of the specific data file RP_1' and the value of root node $\widehat{RP}_0$. The CSP then passes the CHAL message to the servers $(R-1)$ that hold copy of the data, which is as follows.

$$CHAL_r = \{r, i, q_i\}_{l_1 \le i \le l_t, 2 \le r \le R} \quad (9)$$

The r^{th} storage server is responsible computes the value RP_r' and the proofs $\mathcal{M}_r'$ and ρ_r' are formed for $CHAL_r$.

$$RP_r' = HS(HS(RP_{r,l1}) \left|\left|HS(RP_{r,l_2})\right|\right| \dots | \left|HS(RP_{r,l_t})\right|) \quad (10)$$

$$\mathcal{M}_r' = \sum_{i=l_i}^{l_t} q_i . bl_{r,i} \quad (11)$$

$$\rho_r' = \prod_{i=l_i}^{l_t} \beta_{r,i}^{q_i} \quad (12)$$

The proofs of all the data copies $\{RP_r', \mathcal{M}_r', \rho_r'\}$ are forwarded to the central server that has passed the challenge request. As soon as the proof is obtained, the correctness of the data is checked by $\{RP_r'\}$.

$$\widehat{R_0} = HS(HS(RP_1')||HS(RP_2')|| \dots ||HS(RP_R')) \quad (13)$$

The central server verifies the condition of the equation and computes the following.

$$\mathcal{M}_r = \mathcal{M}_1 + \mathcal{M}_r' = \sum_{i=l}^{l_t} q_i . bl_{1,i} + \mathcal{M}_r' \quad (14)$$

$$\rho_r = \rho_1' . \rho_r' = \left(\prod_{i=l_i}^{l_t} \beta_{1,i}^{q_i}\right) . \rho_r' \quad (15)$$

Suppose when these equations do not satisfy, then it is declared that the data integrity is affected. The verification evidence is computed by the following.

$$\mathcal{M} = \sum_{r=1}^{R} \mathcal{M}_r \quad (16)$$

$$\rho = \prod_{r=1}^{R} \rho_r \quad (17)$$

The corresponding server sends the computed proof $P = \{RP_0, \mathcal{M}, \rho\}$ to the TPA. By this way, the integrity of the data is checked and the organization of the replicas helps in carrying out dynamic operations.

4. Dynamic Operations over Outsourced Cloud Data

This section elaborates the proposed approach that supports dynamic operations such as data insertion, deletion and modification over outsourced cloud data. The dynamic operations are described one after the other with the proposed algorithms in the coming sub-sections.

4.1 Data Insertion (DI)

This process intends to insert a fresh data block to the existing data at a particular location of data file 'DF' and the process of data insertion does not affect the logical structure of the original data. Whenever the data owner requires to insert a new data block db_{new} after the specific block of data db_i, the following steps are followed. The overall algorithm of data insertion operation is shown as follows.

Initially, the data owner creates the new version of data along with the time stamp (TS_{new}), which is represented as follows.

$$DO \rightarrow TPA: DI(db_{new}, TS_{new}) \quad (18)$$

Where, DO is the data owner that sends the 'Data Insert' request to the Third Party Auditor (TPA), which includes the

new version of the data (db_{new}) along with the timestamp of the new data block (TS_{new}). Figures 2 and 3 show the before and after structure of data insertion operation.

Algorithm 1 Data Insertion

//Input: Data block to be inserted
//Output: Inserted block with updated file

1. *Begin*
2. *//Data Owner Side*
3. *Begin*
4. *Create new data information* (db_{new}, TS_{new}) *for the new data block;*
5. *Send data update request to TPA;*
6. *End*
7. *//TPA side*
8. *Begin*
9. *Detect the last data block;*
10. *Insert the new data block;*
11. *Update with* (db_{new}, TS_{new})*;*
12. *Increment the pointer;*
13. *End;*
14. *//CSP Side*
15. *Begin*
16. *Generates new version of file* DF_{new} *and tag set;*
17. *End;*
18. *End;*

Sl.No	Block ID	Db_i	TS_i
1	1	Db_1	TS_1
2	2	Db_2	TS_2
. . .	. . .	. . .	. . .
i	i	Db_i	TS_i
. . .	. . .	. . .	. . .
n-1	n-1	Db_{n-1}	TS_{n-1}
n	n	Db_n	TS_n
i_{new}	i_{new}	Db_{new}	TS_{new}

Figure 2. Data structure before performing insertion operation

The TPA then takes action to insert the new data db_{new} at the completion of data file DF by creating a new data block and increments the pointer by one. The user creates a new signature σ_{new} for the new data block db_{new} and forwards the data update request to the CSP, which is represented as follows.

$$DO \rightarrow CSP: DI(DF, i, db_{new}, TS_{new}), \sigma_{new} \qquad (19)$$

Sl.No	Block ID	Db_i	TS_i
1	1	Db_1	TS_1
2	2	Db_2	TS_2
. . .	. . .	. . .	. . .
i	i	Db_i	TS_i
i_{new}	i_{new}	Db_{new}	TS_{new}
i+1	i+1	Db_{i+1}	TS_{i+1}
. . .	. . .	. . .	. . .
n-1	n-1	Db_{n-1}	TS_{n-1}
n	n	Db_n	TS_n
n+1	n+1	Db_{n+1}	TS_{n+1}

Figure 3. Data structure after performing insertion operation

As soon as the receipt of this request from the CSP, the CSP creates the fresh version of this file, which is represented as DF_{new} along with the tag set $TG_{new} = DF_{new}||n_{db}||Sig_{sk}(DF_{new}||n_{db})$. The signature is composed of a newly formed file, total count of data blocks and the parameters are signed by the secret key. Hence, the process of data insertion is discussed in this section and the next section presents details of data deletion.

4.2 Data Deletion

When a specific data block has to be deleted, then it succeeding blocks after the processing blocks are needed to be moved. The data deletion procedure of this work is presented as follows. When the data owner wishes to delete a specific data block db_i from the data file DF, the following request is sent to the TPA.

$$DO \rightarrow TPA: DD(DF, db_i) \qquad (20)$$

Here, the data deletion (DD) request is sent to the TPA, which consists of the data file (DF) and the specific data block to be deleted (db_i). In this case, after performing deletion, the pointer is decremented by one. The updated file is then reflected on the cloud server, when the DO forwards the update request to CSP, which is as follows. The data deletion algorithm is presented as follows.

Algorithm 2 Data Deletion

//Input : Data block to be deleted
//Output : Updated file with deleted block

1. *Begin*
2. *//Data Owner Side*
3. *Begin*
4. *Send data delete request to TPA;*
5. $DO \rightarrow TPA: DD(DF, db_i)$;
6. *Decrement pointer;*
7. *End*
8. *Update data in cloud storage;*
9. *//CSP Side*
10. *Begin*
11. *Generates new version of file* DF_{new} *and tag set;*
12. *End;*
13. *End;*

$$DO \rightarrow CSP: DD(DF, i) \quad (21)$$

The cloud server updates the new version of the file, while generating the following tag set.

$$TG_{new} = DF_{new} || n_{db} || Sig_{sk}(DF_{new} || n_{db}) \quad (22)$$

The signature contains the newly updated file that reflects the carried out deletion with the total count of data blocks and the parameters signed with the secret key. Hence, the process of data deletion is explained and the next section considers the process of data updation.

4.3 Data Updation

Data updation is the most important dynamic operation among all the operations, which replaces a particular block with another block. When the data owner intends to update a specific data block with another block, then the update request is forwarded to the TPA. The algorithm for data updation is presented as follows.

Algorithm 3 Data Updation

//Input : Data file to be modified;
//Output : Updated file;

1. *Begin*
2. *// Data owner side*
3. *Begin*
4. *Send update request* $DU(DF, i, db_{new'}, TS_{new'})$;
5. *Update file;*
6. *End;*
7. *// TPA side*
8. *Begin*
9. *Modify old file with newly updated file;*
10. *End;*
11. *//CSP side*
12. *Begin*
13. *Generates new version of file* $DF_{new'}$ *and tag set;*
14. *End;*
15. *End;*

$$DO \rightarrow TPA: DU(DF, i, db_{new'}, TS_{new'}) \quad (23)$$

This data updation (DU) request consists of the file to be updated (DF), block number (i), block update ($db_{new'}$) and timestamp of the data block ($TS_{new'}$). As soon as the update request is obtained from the data owner, the corresponding data file is retrieved and the specific data block is located. The data update request is sent to the CSP, which is represented as follows.

$$DO \rightarrow CSP: DU\left(DF, i, db_i, bl_{new}(db_{new}, TS_{new}), bl_{new}(db_{new'}, TS_{new'})\right), \sigma_{new'} \quad (24)$$

The data owner generates a new signature $\sigma_{new'}$ and the update request is forwarded to the CSP, which is as follows. The CSP performs the replacement operation of the old file to the new file ($db_{new'}$). The tag generation is represented below.

$$TG_{new} = DF_{new'} || n_{db} || Sig_{sk}(DF_{new'} || n_{db}) \quad (25)$$

Hence, the dynamic operations of data insert, delete and update are presented in this section and the next section discusses the experimental results attained by the proposed work and presents the results achieved by the proposed approach.

5. Results and Discussion

The performance of the proposed OA-DIV work is simulated with the help of Eucalyptus tool, which is an open source. The Eucalyptus fast version 3.4.1 is employed for a stand-alone computer with 500 GB hard disk and 8 GB RAM. The experimental results attained by this work are compared with the existing approaches and the results are discussed in this section. The comparison is performed in terms of communication cost, processing time and storage cost.

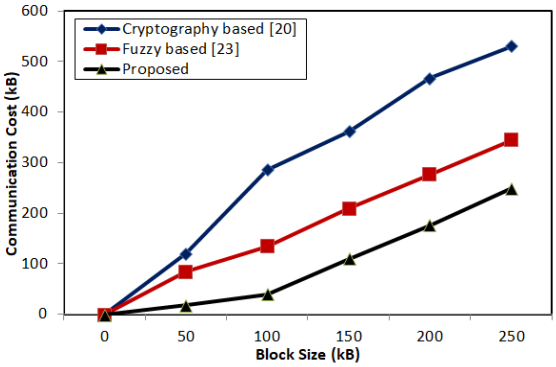

Figure 4. Communication cost w.r.t block size (kB)

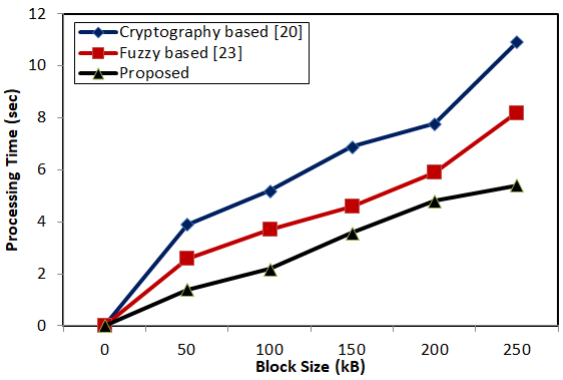

Figure 5. Processing time w.r.t block size(kb)

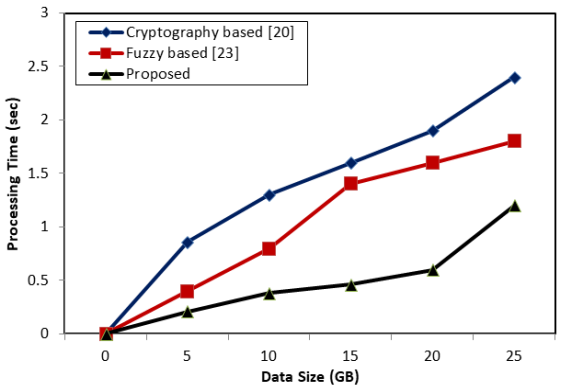

Figure 6. Processing time during integrity verification phase

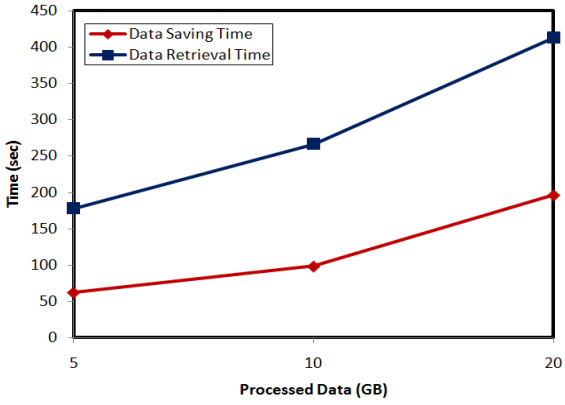

Figure 7. Data storage and retrieval time

From figures 4 to 6, the communication cost, processing time with respect to block size and verification phase of the proposed work are presented. The results show that the proposed data integrity verification scheme consumes minimal communication cost and processing time than the compared approaches. The main reason for attaining better results is the better organization of the data blocks, data copy creation and management. The communication cost of the proposed scheme meant for verification phase is shown in Table 1.

Table 1. Communication cost comparison list

Techniques	Communication Cost	
	Setup	Verification
Cryptography based	vbO(logn)	Cryptography based
Fuzzy based	O(vb)	Fuzzy based
Proposed OA-DIV Scheme	O(vb)	Proposed OA-DIV Scheme

Here, n is the total count of blocks in data and vb is the count of data blocks.

The effectiveness of the dynamic operation is examined with respect to the data storage and retrieval time. The results indicate the data saving time is lesser, when compared to the data retrieval time. The reason for this is that the data are segregated and stored in different cloud servers. Figure 7 shows the time consumption of data storage and retrieval time.

The data retrieval time is measured by retrieving the complete data and thus, the data retrieval time is more. Besides this, while retrieving the data the sequence has to be maintained. On the other hand, when a particular data block alone is retrieved, the retrieval is done in a streak. Both data storage and retrieval time increases with respect to the count of data blocks. The results of the time consumption for data block insertion and deletion are shown in figures 8 and 9 respectively.

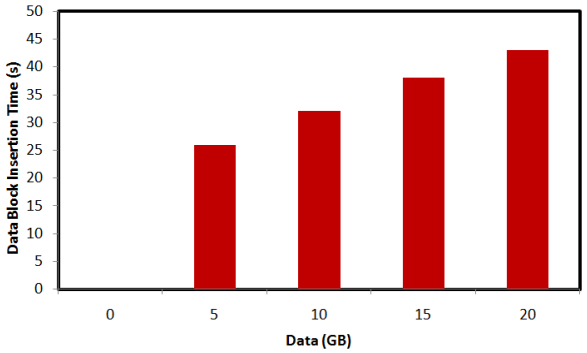

Figure 8. Block insertion operation analysis

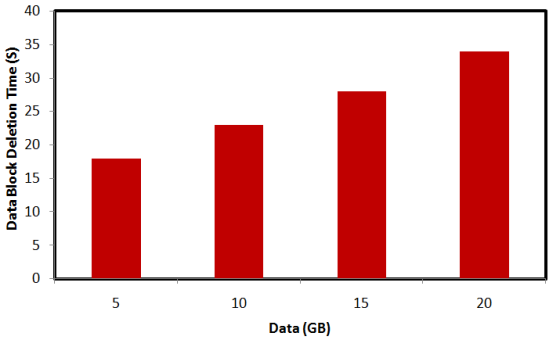

Figure 9. Block deletion operation analysis

The analysis varies the volume of data from 5 to 20 GB and the time consumption is analysed. The experimental analysis confirm that the block insertion and deletion operations consume more time with respect to the size of data, yet is reasonable.

The security analysis of the proposed work is discussed as follows.

5.1 Security analysis

It is technically difficult to forge the proof generated by the TPA.

Proof: When the challenge is placed by the TPA, the proposed work considers all the replicas of the data and the challenge is passed to all the servers with $CHAL_r = \{r, i, q_i\}_{l_1 \le i \le l_t, 2 \le r \le R}$. The CSP responds to the challenge with the proof $P = \{RP_0, \mathcal{M}, \rho\}$. When the equation holds, the data integrity is proven to be maintained, else it is not. Hence, the proposed work focuses to maintain the data integrity of the replicas as well and it is difficult to forge the TPA's proof. On the whole, the proposed work is proven with the better verification scheme for maintaining the integrity of data. The conclusions of this work are presented as follows.

Conclusion

This article presents an OA-DIV Outsourced auditing with data integrity verification scheme for cloud computing environment, which even considers the data copies for integrity check. This work relies on three modules such as client, Third Party Auditor (TPA) and CSP. The client uploads the encrypted data blocks to the CSP. The copies of the data are created and maintained in different cloud storage server.

Whenever a challenge message is submitted to the CSP, all the servers that possess the copy of data are also challenged and the integrity of all the copies is also verified. The copies are organized in a tree-like structure, such that it supports to carry out dynamic operations over outsourced cloud data. The dynamic operations include data insertion, deletion and modification. This work handles these dynamic operations by segregating the data into several data blocks, which makes the process easier and efficient. The effectiveness of the proposed work is assessed in terms of time consumption, data storage and retrieval time. The results indicate the effectiveness of the proposed work and this approach can be extended by considering the real-time cloud environment.

References

[1] H. Wang, Z. Chen, J. Zhao, X. Di, and D. Liu, "A vulnerability assessment method in industrial Internet of Things based on attack graph and maximum flow," IEEE Access, vol. 6, pp. 8599-8609, 2018,

[2] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in Proc. Eur. Symp. Res. Comput. Secur., Saint-Malo, France: Springer, 2009,pp. 355-370.

[3] Q. Zhiguang, W. Shiyu, and Z. Yang, "An auditing protocol for data storage in cloud computing with data dynamics," J. Comput. Res. Develop., vol. 52, no. 10, pp. 2192-2199, 2015.

[4] H. Wang, J. Gu, X. Di, D. Liu, J. Zhao, and X. Sui, "Research on classification and recognition of attacking factors based on radial basis function neural network," Cluster Comput., vol. 22, no. S3, pp. 5573-5585, May 2019.

[5] Yuan, Y., Zhang, J., & Xu, W. (2020). Dynamic Multiple-Replica Provable Data Possession in Cloud Storage System. IEEE Access, 8, 120778-120784.

[6] Zhang, Y., Lin, G., Gu, H., Zhuang, F., & Wei, G. (2020). Multi-copy dynamic cloud data auditing model based on IMB tree. Enterprise Information Systems, 1-22.

[7] Chen, X., Shang, T., Zhang, F., Liu, J., & Guan, Z. (2020). Dynamic data auditing scheme for big data storage. Frontiers of Computer Science, 14(1), 219-229.

[8] Li, A., Chen, Y., Yan, Z., Zhou, X., & Shimizu, S. (2020). A survey on integrity auditing for data storage in the cloud: from single copy to multiple replicas. IEEE Transactions on Big Data.

[9] Daniel, E., & Vasanthi, N. A. (2020). ES-DAS: An Enhanced and Secure Dynamic Auditing Scheme for Data Storage in Cloud Environment. Journal of Internet Technology, 21(1), 173-182.

[10] Thokchom, S., & Saikia, D. K. (2020). Privacy preserving integrity checking of shared dynamic cloud data with user revocation. Journal of Information Security and Applications, 50, 102427.

[11] Zhang, Y., Yu, J., Hao, R., Wang, C., & Ren, K. (2018). Enabling efficient user revocation in identity-based cloud storage auditing for shared big data. IEEE Transactions on Dependable and Secure computing.

[12] Zhao, X. P., & Jiang, R. (2020). Distributed Machine Learning Oriented Data Integrity Verification Scheme in Cloud Computing Environment. IEEE Access, 8, 26372-26384.

[13] Huang, P., Fan, K., Yang, H., Zhang, K., Li, H., & Yang, Y. (2020). A Collaborative Auditing Blockchain for Trustworthy Data Integrity in Cloud Storage System. IEEE Access, 8, 94780-94794.

[14] Yu, Y., Li, Y., Yang, B., Susilo, W., Yang, G., & Bai, J. (2017). Attribute-based cloud data integrity auditing for secure outsourced storage. IEEE Transactions on Emerging Topics in Computing.

[15] Shen, J., Liu, D., He, D., Huang, X., & Xiang, Y. (2017). Algebraic signatures-based data integrity auditing for efficient data dynamics in cloud computing. IEEE Transactions on Sustainable Computing.

[16] Liu, J., Wang, X. A., Liu, Z., Wang, H., & Yang, X. (2020). Privacy-Preserving Public Cloud Audit Scheme Supporting Dynamic Data for Unmanned Aerial Vehicles. IEEE Access, 8, 79428-79439.

[17] Yang, C., Liu, Y., Tao, X., & Zhao, F. (2020). Publicly Verifiable and Efficient Fine-Grained Data Deletion Scheme in Cloud Computing. IEEE Access.
[18] Rao, L., Zhang, H., & Tu, T. (2017). Dynamic outsourced auditing services for cloud storage based on batch-leaves-authenticated Merkle hash tree. IEEE Transactions on Services Computing, 13(3), 451-463.
[19] Wang, C., & Di, X. (2020). Research on Integrity Check Method of Cloud Storage Multi-Copy Data Based on Multi-Agent. IEEE Access, 8, 17170-17178.
[20] Zhang, X., Huang, C., Zhang, Y., Zhang, J., & Gong, J. (2020). LDVAS: Lattice-Based Designated Verifier Auditing Scheme for Electronic Medical Data in Cloud-Assisted WBANs. IEEE Access, 8, 54402-54414.
[21] Bian, G., & Chang, J. (2020). Certificateless Provable Data Possession Protocol for the Multiple Copies and Clouds Case. IEEE Access, 8, 102958-102970.
[22] Zhang, J., Zhao, Y., Wu, J., & Chen, B. (2020). LVPDA: A Lightweight and Verifiable Privacy-Preserving Data Aggregation Scheme for Edge-Enabled IoT. IEEE Internet of Things Journal, 7(5), 4016-4027.
[23] Yuan, Y., Zhang, J., & Xu, W. (2020). Dynamic Multiple-Replica Provable Data Possession in Cloud Storage System. IEEE Access, 8, 120778-120784.
[24] Wang, F., Xu, L., Choo, K. K. R., Zhang, Y., Wang, H., & Li, J. (2019). Lightweight Certificate-Based Public/Private Auditing Scheme Based on Bilinear Pairing for Cloud Storage. IEEE Access, 8, 2258-2271.
[25] Zhu, H., Yuan, Y., Chen, Y., Zha, Y., Xi, W., Jia, B., & Xin, Y. (2019). A secure and efficient data integrity verification scheme for cloud-IoT based on short signature. IEEE Access, 7, 90036-90044.
[26] Khedr, W. I., Khater, H. M., & Mohamed, E. R. (2019). Cryptographic accumulator-based scheme for critical data integrity verification in cloud storage. IEEE Access, 7, 65635-65651.
[27] Zhang, Q., Wang, S., Zhang, D., Wang, J., & Zhang, Y. (2019). Time and attribute based dual access control and data integrity verifiable scheme in cloud computing applications. IEEE Access, 7, 137594-137607.
[28] Shen, W., Qin, J., Yu, J., Hao, R., & Hu, J. (2018). Enabling identity-based integrity auditing and data sharing with sensitive information hiding for secure cloud storage. IEEE Transactions on Information Forensics and Security, 14(2), 331-346.
[29] Zhao, C., Xu, L., Li, J., Wang, F., & Fang, H. (2019). Fuzzy Identity-Based Dynamic Auditing of Big Data on Cloud Storage. IEEE Access, 7, 160459-160471.
[30] G. Ateniese, R. D. Pietro, L. V. Mancini and G. Tsudik, "Scalable and Efficient Provable Data Possession," Proceedings of the 4th international conference on Security and privacy in communication networks (SecureComm '08), Sep. 2008.
[31] C. C. Erway. A. Küpçü, C. Papamanthou and R. Tamassia, "Dynamic Provable Data Possession," ACM Transactions on Information and System Security (TISSEC), vol. 17, no. 15, pp. 213-222, Jan. 2009
[32] C. Gritti, W. Susilo and T. Plantard, "Efficient Dynamic Provable Data Possession with Public Verifiability and Data Privacy," Australasian Conference on Information Security and Privacy (ACISP '15), pp. 395-412, Jun. 2015.
[33] Y. Zhang and M. Blanton, "Efficient dynamic provable possession of remote data via balanced update trees," computer and communications security, 2013.
[34] G. Yao, Y. Li Y, L. Lei L, H. Wang and C. Lin. "An Efficient Dynamic Provable Data Possession Scheme in Cloud Storage," grid and pervasive computing, 2016.
[35] Q. Zheng and S. Xu, "Fair and Dynamic Proofs of Retrievability," Proceedings of the first ACM conference on Data and application security and privacy (CODASPY '11), pp. 237-248, Feb. 2011.
[36] Z. Mo, Y. Zhou and S. Chen, "A Dynamic Proof of Retrievability(POR) Scheme with O(kogn) Complexity," International Conference on Communications, 2012.
[37] D. Cash, A. Küpçü and D. Wichs, "Dynamic Proofs of Retrievability via Oblivious RAM," Journal of Cryptology, vol. 30, no. 1, pp.22-57, Sep. 2015.
[38] E. Shi, E. Stefanov and C. Papamanthou, "Practical dynamic proofs of retrievability," Computer and Communications Security, 2013.
[39] J. Li, X. Tan, X. Chen, D. S. Wong and F. Xhafa, "OPoR: Enabling Proof of Retrievability in Cloud Computing with Resource-Constrained Devices," IEEE Transactions on Cloud Computing, vol. 3, no. 2, pp. 195-205, Apr/Jun. 2015
[40] Laghari, A. A., He, H., Khan, A., Kumar, N., & Kharel, R. (2018). Quality of experience framework for cloud computing (QoC). IEEE Access, 6, 64876-64890.
[41]Kumar, V., Laghari, A. A., Karim, S., Shakir, M., & Brohi, A. A. (2019). Comparison of fog computing & cloud computing. Int. J. Math. Sci. Comput, 1, 31-41.
[42]Shafiq, M., & Khan, A. (2018). Assessment of quality of experience (QoE) of image compression in social cloud computing. Multiagent and Grid Systems, 14(2), 125-143.
[43]Memon, K. A.., Halepoto, I. A., & Khan, A. (2019). Quality of experience (QoE) in cloud gaming models: A review. Multiagent and Grid Systems, 15(3), 289-304.
[44]Laghari, Asif Ali, Hui He, Muhammad Shafiq, and Asiya Khan. "Assessing effect of Cloud distance on end user's Quality of Experience (QoE)." In 2016 2nd IEEE international conference on computer and communications (ICCC), pp. 500-505. IEEE, 2016.
[45]Laghari, A., He, H., Laghari, R., Khan, A., & Yadav, R. (2019). Cache Performance Optimization of QoC Framework. EAI Endorsed Transactions on Scalable Information Systems, 6(20).