UTM UNIVERSITI TEKNOLOGI MALAYSIA
INTERNATIONAL JOURNAL OF INNOVATIVE COMPUTING
ISSN 2180-4370
Journal Homepage : https://ijic.utm.my/

Data Integrity Schemes for Outsourced Data on Untrusted Cloud Storage: A Review

Yasir Ali Panhwer, Mazleena Salleh
Faculty of Computing,
Universiti Teknologi Malaysia,
81310 UTM Johor Bahru, Johor, Malaysia
yasirpanhwer@gmail.com, mazleena@utm.my

Submitted: 12/01/2018. Revised edition: 16/05/2018. Accepted: 21/05/2018. Published online: 31/05/2018

***Abstract*—Cloud Computing have emerged into a very vast growing area which have a great impact towards the development other new technologies and application such as internet of things, sensors, artificial intelligence, social networks and business applications. With the new development of technology and applications, the production of data is increased and data have the property of being updated dynamically. That dynamic data is stored on cloud storages provided by third party service providers. The third party cloud storages cannot be solely trusted and the user does not have the control over the possession or integrity of data. The integrity is the major concern for that data as it is being intact not deleted, modified or destroyed intentionally or unintentionally. The researchers have presented several protocols such as Provable Data Possession (PDP) techniques which provide probabilistic approach for data integrity verification in which the verification is done on block level. To support dynamic nature of data the researchers have introduced different data structure along with PDP. Due to enormous amount of data and its dynamic nature, the integrity verification schemes causes high computational and communicational cost for metadata generation and node rebalancing of the data structures**.

***Keywords*—Cloud Computing, Data Dynamics, Data Integrity, Provable data possession, Cloud Storage**

I. INTRODUCTION

In the 21st century age of information technology, data is growing exponentially day by day. That exponentially growing data are termed as big data which specifically refer to data sets that are large, complex, structured and non-structured that is produced by day-to-day business [1]. This increasingly huge amount of big data is part of day-to-day business which is produced by technological advanced application such as internet, social networking sites, healthcare application, sensor networks and many other companies and it is increasing dynamically. All of these dynamically changing big data need enormous powers for processing and storage so are stored and processed in emerging technology "Cloud Computing".

Cloud computing is the latest technology in the world of computing, which is emerging with broad ranging effects across IT, business, health care, software engineering and data storage [2]. As the enterprise, organizations and individual users produce enormous amounts of data day by day, the task of storing, processing and maintaining security of that data on local storage is a great challenge for the companies which incur a lot of expense for the infrastructure to store and process the data [3]. Cloud computing, mainly provides three service delivery model to its customers, which are Infrastructure as a service (IaaS), Software as a Service (SaaS), and Platform as a service (PaaS) with four deployment models that is private, public, hybrid and community cloud defined by the National Institute of standard and technology (NIST) [2]. Cloud computing implements the virtualization technique to efficiently provide resources to end users. The main characteristics of cloud computing are that it provides on demand service, resource pooling, highly scalable, flexible, low cost computational powers for application, storage and platforms. The major cloud services are used for storing data, sharing data and application services. Most of the enterprise companies are motivated towards the cloud computing, hence moving their application development, data storage (financial, personnel, healthcare) towards cloud storage.

II.CLOUD COMPUTING

Industry of computing and networking has been revolutionized by World Wide Web (internet) which provides sharing of resources to global users. Due to the invention of cloud computing the classical computing paradigm is greatly changed because it made the resources available to everyone with enormous repository of data storage, computational powers, application and services through internet. Cloud computing is basically a large pool of computational power and storage which is made publically available through source of internet and services are paid based on usage. It gives distinctive benefits such as increased network access, scalability, flexibility, usage effectiveness and resource sharing. Due to this, cloud computing paradigm adoption is widely increasing and implemented in various domains like social networking, government, health care applications and scientific adoptions.

The usability of cloud computing services is simple that the client only needs a computing device with stable internet. Apart from the easy usability the cloud computing provides many benefits to users such as global access, standard platform, massive scalability, dynamic infrastructure, administrative function and majorly its minimal usage costs. Such seamless and feasible connectivity with cloud have advanced many organizations to grow at a faster rate. The advancement in cloud computing services have great impact on the organizations and individuals to fulfill organizational goals and objectives. The cloud computing technology has greatly increased the advancements to the organizations in terms of cost, globalization, flexibility, scalability and revenue.

III. SERVICE MODELS OF CLOUD COMPUTING

Service models is defined as the usability of the cloud computing platform that is delivered to the client on demand. The major cloud services models are (i) Infrastructure as a service (ii) Software as a service (iii) Platform as a service. The details of the service models are given below:

IaaS is the core service for a cloud provider firms. IaaS provides the computer hardware (network, storage, virtual server, data center, processor and memory) resources to its customer. Cloud service provider (CSP) manages above resources and the customer can access the resources with the internet [4]. This all based on virtualization technique and the clients have to pay for the resources they consumes.

PaaS is middleware model which provide services such as program development tools, platform and frameworks, virtual container to run application to the customer to build their own application. It considerably lowers the cost of managing and marinating additional hardware and software required to build the application. The PaaS provides pre-packaged disk images and software stack to the customers that share underlying resources of cloud such as runtime components, libraries and database engines. Google App Engine, AWS elastic beanstalk, apprenda are real time example of PaaS providers which provide ready features such as software developments kits (SDKs) for python, java and .NET respectively [5].

SaaS is the application layer for cloud computing. SaaS provides its customer demand to use the hosted application over the internet and pay according to the usage of the resources. SaaS is the top of the line model for the customers to take benefit of it such as improved operational efficiency and reduced cost of managing the application by themselves [6]. Due to reduced cost and maintenance overhead for the application the SaaS layer is becoming very prevalent cloud business for IT enterprises.

IV. DEPLOYMENT MODEL OF CLOUD COMPUTING

Deployment models are visualized as the physical existence of cloud service infrastructure is either on premises or off premises. The deployment model are majorly of four types, that is private, public, hybrid and community cloud as defined by National Institute of standard and technology (NIST) [2].

Public cloud is cloud infrastructure which is available for organization or users to use by paying what they only use. Public cloud can provide all the services such as PaaS, SaaS, IaaS and more. Google cloud, window azure HP hellion is some third party cloud service providers in the market and can be used on demand.

Private clouds are on premises clouds, managed and maintained by the organization themselves and used within the organization or enterprises. They have dedicated team to manage and maintain their datacenters for providing cloud services to their enterprise applications. They are preferably more secure in terms of trustworthiness of the secure service.

Hybrid cloud which uses both the cloud services private and public clouds are said to have hybrid cloud structure in their organization. So in this type of organizations data is classified between different measures of security. The most sensitive data which can cause a major damage to the organization are kept within premises (private cloud) while the data which is less important is kept in off premises (public cloud) so the organization are free from managing less important data.

Community cloud setup is shared and controlled by multiple organizations which usually support common goal or interest. The cloud can be placed within the premises or off-premises of the organization. The cloud is either maintained by the controlling organization or by any third party organization. It reduces the security risk of public cloud and the cost of private cloud and the participating organization can freely access the data available on the cloud [5].

V. DATA INTEGRITY

The confidentially, integrity and availability (CIA) are one of the core security features for any organizational information on cloud computing. The primary focus for this research project is on the data integrity. Data integrity can be defined as “the sense of assurance that data is accurate and consistent throughout its entire life cycle. Further data integrity insures that the data is intact that is, not lost or damaged intentionally or accidentally. Integrity of data is intensively been researched for many years and integrity verification can be classified into two major approaches:

i) Deterministic approach in which whole file is checked for the integrity verification. It guarantees 100% possession of data and is good for checking the small sized data.

ii) Probabilistic approach where only few required blocks are checked to ensure the integrity of file. It does not guarantee 100% assurance of the integrity but good for data files.

First traditional way is to compute message authentication code (MAC) of the file is the simple way to assure the integrity of file. Firstly before outsourcing the file to remote cloud storage data owner computes the MAC of the whole file. The MAC of the file is kept at data owner's local storage while deletes the file from the local storage. When data integrity is required to be verified, the verifier sends a request to retrieve the file from cloud storage. Then the verifier re-computes the MAC of the outsourced file. It compare the local stored MAC with the outsourced re-computed MAC to verify the integrity of data [7]).

Second simple approach is that first data owner divides the file into *n* blocks, then computes the MAC of each block with a secret key. The owner sends the file and the MAC to cloud server and deletes the computed MAC and file itself from local storage. The data owner only stores the secret key. For verification the verifier requests the file block and the corresponding MAC from the remote server. The verifier computes the MAC with the secret key and compares it with the corresponding received MAC from the server.

The above traditional approaches can work great when the data size is small and the data is not changed frequently once stored on the server. But as for the data both approaches have serious flaws and are impractical. In the first approach it have high communication cost such as if the 10GB or 100GB file is outsourced so every time to verify the integrity the data owner need to download the file from the cloud storage which is impractical because of bandwidth and data usage. While with the second approach cannot cover the nature of dynamic updates of data. So in nutshell, due to data the above approaches have severe disadvantage such as they have high communication and computational cost and does not support the dynamic nature of data [8].

VI. DATA INTEGRITY SCHEMES

Initially the research on the data integrity verification schemes started with static data and continued to dynamic data (create, update, and delete operations). Further private verifiability and public verifiability [7] with or without inference of third party auditing was supported. Because with introducing a TPA created a loophole for privacy concerns, so privacy preserving data integrity protocol were introduced.

Researchers have used homomorphic tags, bilinear pairing [7], algebraic signatures [9], foundation codes, erasure codes [10], RS codes based on cauchy metrics and other techniques as metadata generation in data integrity verification [11]. To support dynamic data updates the researchers have used the ITable, skip list [12], divide and conquer table [13], Merkle hash tree. Integrity schemes generally have following phases:

A. Preprocessing Phase

The original data is preprocessed with some predefined algorithm to generate the metadata. The metadata (for verification) and the original file are uploaded to the cloud service provider.

B. Verification Phase

This phase involves auditor (TPA or owner) who sends the challenge request to the CSP that generate the proof of possession using the original data and metadata. The challenge proof is provided to the auditor who ensures the data integrity within cloud storage being intact.

VII. PROVABLE DATA POSSESSION CHARACTERISTICS

The characteristics of a data integrity schemes can be categorized with the functionality it provides to verify the possession of data. The functionally can further be categorized into the security services, features, performance metrics, data verification coverage and the state of verifiability. The security services it must provide is data integrity and also can address confidentiality and availability. The features should be robust integrity, soundness which provide correct information regarding data and block less verification without downloading the actual data. The performance of the protocol must be efficient in terms of computational cost of metadata data and data verification, communication expenses of exchange of data, storage expenses and detection probability. Data verification coverage covers the static data or dynamic data verification. State of verifiability depends on either it provide public or private verifiability shown in Fig. 1.

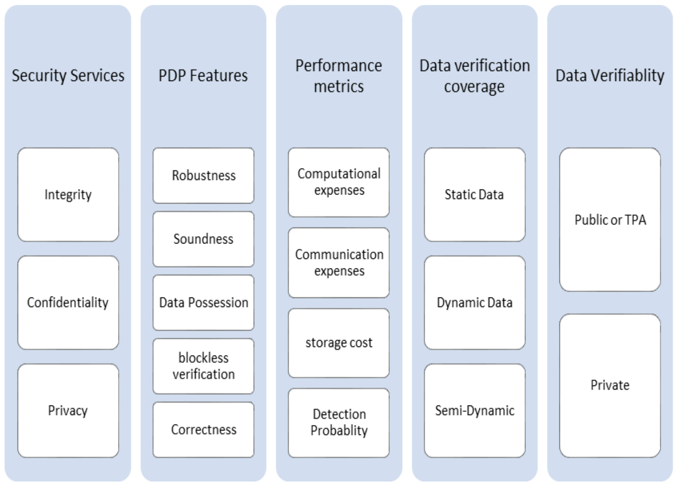


Fig. 1. Provable Data Possession Characteristics (Sookhak *et al*., 2014)

VIII. PROVABLE DATA POSSESSION TECHNIQUES

The provable data possession (PDP) is the techniques to ensure the data possession in cloud storage. This section reviews the state of the art existing PDP techniques of different researchers that verify the integrity of outsourced data without retrieving the original data from cloud storage. The section will also reviews how the dynamically changing nature of data is resolved in the data integrity verification protocols by using different data structures.

A) Homomorphic verifiable tag based PDP

[14] presented first model of PDP which performs the data verification without downloading the original data on untrusted cloud server. The author presented PDP to overcome the integrity checking problems such as deterministic approach followed by other protocols which incur expensive computational server computation on entire file. Expensive communication complexity as well owner local storage overhead by keeping some metadata for later auditing task.

By definition, the author categorized PDP with four polynomial time algorithms (*KeyGen, TagBlock, GenProof, CheckProof*) such that:

KeyGen (1^k) → (*pk, sk*) *is an algorithm run by client to initialize the scheme. The purpose of this algorithm is to generate the public and private key pair (pk, sk) which will be used through the scheme for metadata generation, generating and verifying proofs.*

TagBlock(pk, sk, f) → T_m*: Tagblock is executed at the client side to generate the metadata tags based on input file, public and private key pair.*

GenProof(pk, F, cal, Σ) →V: GenProof is executed at the cloud server after the challenge message to compute the proof of the given challenge. The input it takes is the public key, f blocks, a challenge and the metadata tags corresponding to the f challenged blocks.

CheckProof(pk, sk, chal,V) →{"success", "failure"}: is run by the client after the proof of challenge is received. The input it takes are public, private key pair, the challenge and the proof of possession sent received from the cloud storage.

[12] proposed dynamic provable data possession (DPDP) which is extended version of the protocol which was presented by [14] to support the limitation of data dynamics. The protocol is based on homomorphic verifiable tags and also proposed second construction based on authenticated dictionary using RSA tree [15]. The protocol works on a file F with n blocks. It supports the insertion of new block at any i^{th} position, modification of existing block and deletion of any block of the file.

[16] also proposed the efficient and secure provable data possession which is based on Homomorphic verifiable tags. The protocol supports the data dynamic operation (insert, delete, update, insert) with public verifiability which means and authorized third party can verify the data is intact or not. Privacy is major concern when involving the TPA because the TPA can recover the data from data proof. The author also ensures the privacy in the design of protocol.

B) Identity based PDP

[17] proposed the scheme which is based on identity of the data owner. Identity based PDP is proposed to eliminate the complexity of certificate management. The certificate management in the protocol makes it inefficient. The proposed is based on RSA assumption with large public exponents in random oracle model and it supports the variable-sized data blocks and public auditing.

The main entities in the protocol are: (i) user, (ii) cloud server, (iii) third party auditor (TPA), and (iv) Private key generator (PKG). The identity based integrity checking have procedural algorithm as follows:

- Setup it which outputs master secret key and master public key.
- Extract is the one which makes the secret key based on the identity of the user.
- TagGen takes the input of file and identity ID to produce the metadata tags of each file block.
- Challenge is responsible to construct a challenge based on user ID.
- ProofGen outputs the proof of the challenge
- ProofCheck is to verify the proof computed by the server [17].

C. Symmetric Key Cryptography based PDP

[18] proposed another Scalable and efficient PDP which is solely based on cryptographic hash function and symmetric key cryptography but does not require bulk encryption and also supported the dynamic data operation which are block modification, deletion and append. Earlier protocol used asymmetric key cryptography which was using high computation for large files and dynamic data updates were not addressed before.

The fundamentals presented for scalable and efficient PDP by [19] are that the data, data owner, server, hash function such as SHA-1, SHA-2, authenticated encryption / decryption scheme which provides both privacy and authenticity, pseudo random function is efficiently computable random value, pseudo-random permutation is indexed under key. A good PRP having random sequence from specified range, AES is considered as good PRP.

D. BLS Signature

[7] proposed PDP which enable the public auditing and data dynamics. The main entities involved in the protocol are the client who stores the file, Cloud Storage server (CSS) where the data is stored and third party auditor (TPA) who can challenge the CSS to check that the data is intact. The idea of third party is that the client might not have enough time, feasibility or resources to monitor their data on cloud. So any trusted TPA can be verifier who has the possession public key of the client.

The author used the public key encryption BLS signature based homomorphic authenticator to support the public verifiability of the data. For the dynamic data operation, the author used Merkle hash tree (MHT) [20]. MHT was built for the purpose to ensure efficiently and securely that the set of elements are not modified or damaged. MHT stores the hashes of the elements at the leaves and constructed as a binary tree. The root node is constructed by the children nodes so any change at children nodes will be detected at root node.

The main procedures that protocol follows are that a file *F* is divided into *n* blocks. The public and private key pair is generated by KeyGen(). The SigGen() takes the private key and file blocks and outputs signature of each block. Then the client generates the root of MHT which is hash of corresponding children nodes. The client signs the root with its private and sends file blocks, signature and MHT to cloud storage. The verifier challenges the data blocks to server, the server construct the proof based on the data blocks, signature and MHT.

E. Algebraic Signature based PDP

[3] proposed a dynamic PDP based on Algebraic signature. Algebraic signature is basically a type of hash function having algebraic properties. The main property which is used for constructing a data verification schemes is that the signature of the sum of some random blocks produces the same result as when taking the sum of signatures of corresponding block [21].

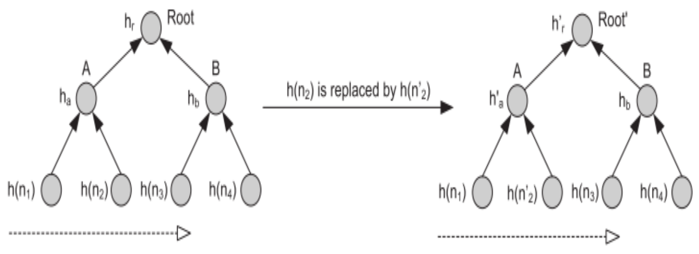
[22] also proposed a scheme based on Algebraic signature which was based on five phases which are Setup, TagBlock, Challenge, ProofGen, and ProofVerify. This scheme provides efficient data verification for static data but does not support dynamic nature of big data.

IX. DATA DYNAMICS

To support data dynamic nature, several researchers have used different data structures to solve the problem. Below is given explanation of state of the art solution.

A. Merkle Hash Tree

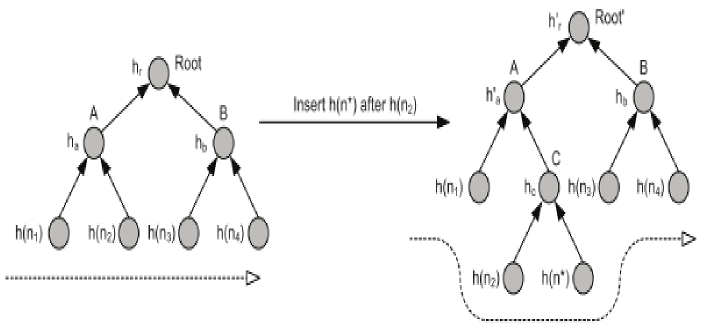
Data modification in the cloud follows the procedure as the client requests the server to modify *i*th block of file. First client compute the signature of the modified block and sends updated data block along with its computed signature. The server upon request modifies the updated data block with previous one, updates the corresponding signature, update the hash of block and construct a new root Fig. 2.



h_r Root
A
B
h_a
h_b
$h(n_1)$ $h(n_2)$ $h(n_3)$ $h(n_4)$
$h(n_2)$ is replaced by $h(n'_2)$
h'_r Root'
h'_a
$h(n_1)$ $h(n'_2)$ $h(n_3)$ $h(n_4)$

Fig. 2. MHT Modification Operation

Data insertion is similar to the data modification in which the client computes the signature and sends the data block along with the signature. The insertion of data block and signature is at the *n*th positions as shown in Fig. 3.



h_r Root
A
B
h_a
h_b
$h(n_1)$ $h(n_2)$ $h(n_3)$ $h(n_4)$
Insert $h(n^*)$ after $h(n_2)$
h'_r Root'
h'_a
C
h_c
$h(n_2)$ $h(n^*)$

Fig. 3. MHT Block Insertion Operation (Wang *et al.,* 2012)

The deletion is done when the owner request to delete any data block. After the delete request the server deletes the data block and corresponding signature. The server also deletes the hash of corresponding block and rebalances the tree to keep the property of binary tree Fig 4.

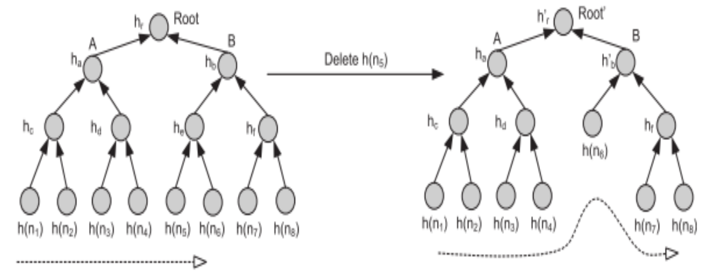
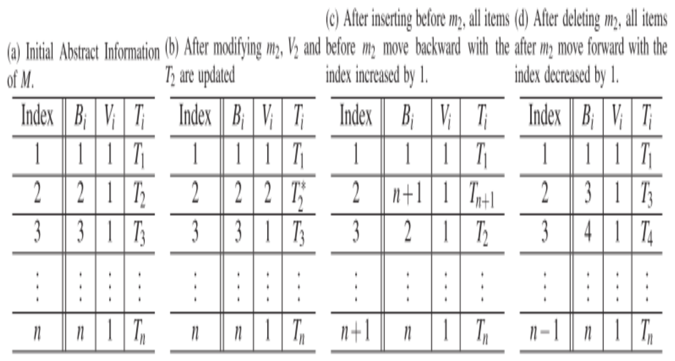


Fig. 4. MHT Block Deletion Operation (Wang *et al*., 2012)

B. ITable

The dynamic data changes in [16] are supported by usage of ITable. ITable consist on entries such as the original number of file block B_i, the version number of data block and time stamp at which time the block is inserted or updated. The Fig 5 shows the updates of the table in ITable.



(a) Initial Abstract Information of *M*.

Index	B_i	V_i	T_i
1	1	1	T_1
2	2	1	T_2
3	3	1	T_3
⋮	⋮	⋮	⋮
n	n	1	T_n

(b) After modifying m_2, V_2 and T_2 are updated

Index	B_i	V_i	T_i
1	1	1	T_1
2	2	2	T_2^*
3	3	1	T_3
⋮	⋮	⋮	⋮
n	n	1	T_n

(c) After inserting before m_2, all items before m_2 move backward with the index increased by 1.

Index	B_i	V_i	T_i
1	1	1	T_1
2	$n+1$	1	T_{n+1}
3	2	1	T_2
⋮	⋮	⋮	⋮
$n+1$	n	1	T_n

(d) After deleting m_2, all items after m_2 move forward with the index decreased by 1.

Index	B_i	V_i	T_i
1	1	1	T_1
2	3	1	T_3
3	4	1	T_4
⋮	⋮	⋮	⋮
$n-1$	n	1	T_n

Fig. 5. ITable Data Dynamic Operations

C. Divide and Conquer Table

Data block insert of new block is done after the *i*th block find the ith block then update the metadata in DCT and also the file Fig 6.

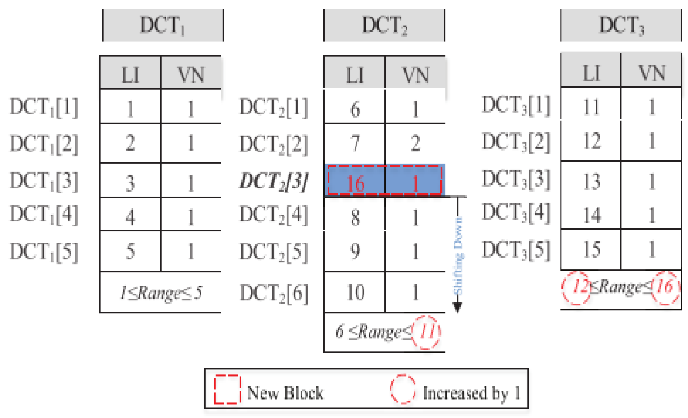


Fig. 6. Block Insertion in DCT

The data append operation means that the data is inserted at the end of the file. At DCT the entries is also updated in the last DCT table with index of appended block and version number of new block as shown in Fig 7. If the DCT is full then the new DCT table is constructed.

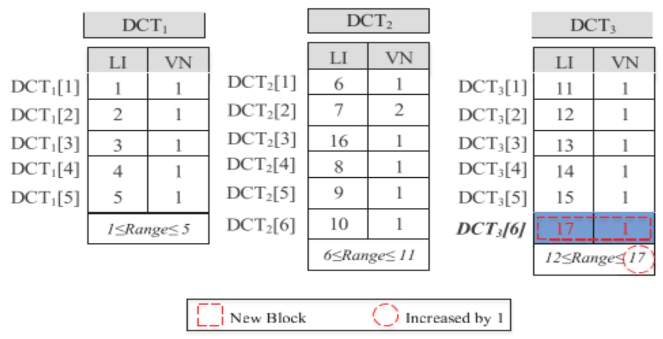


Fig. 7. Block Append in DCT

The data delete operation is to delete some specified block from the original file. The owner request the data block index to delete. At cloud storage first the data block is deleted and its corresponding signature. After that the entry from DCT is deleted by searching the entry. The remaining blocks are rebalanced by shifting upward as shown in Fig 8.

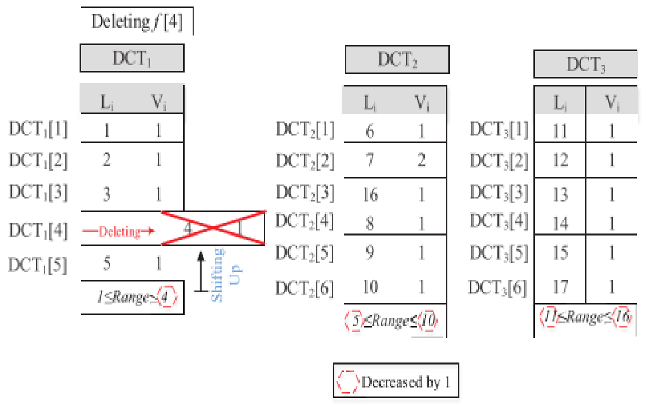


Fig. 8. Block Deletion in DCT

X. Comparison

The Table I shows the comparison of different existing works on cloud data integrity verification. The table also shows the limitation of each work.

TABLE I. Comparison of Existing Work on Data Integrity

Presented	**Approach**	**Dynamic Updates approach**	**Weakness**
Ateniese *et al.*, (2007)	Homomorphic verifiable tags	N/A	Causes high computational expenses because of using RSA numbering.
Erway *et al.*, (2009)	Homomorphic verifiable tags	Using Authenticated rank based skip list	High computational cost and does not provide block less verification.
Ateniese *et al.*, (2008)	Cryptographic hash function and symmetric key cryptography	Token based list manipulation	High computational cost because of node rebalancing after dynamic insertion, deletion.
Wang *et al.*, (2012)	BLS homomorphic authenticator	Merkle hash tree	After every updates function needs to calculate the root which incurs computational cost.
Yang & Jia, (2013)	Homomorphic verifiable tags	ITable	High computational cost because of node rebalancing in ITable after dynamic insertion, deletion.

Presented	**Approach**	**Dynamic Updates approach**	**Weakness**
Yu *et al.*, (2016)	RSA based	N/A	Does not support dynamic data updates.
Chen (2013)	Algebraic signature	N/A	Does not support dynamic data updates.
Sookhak *et al.*, (207)	Algebraic signature	Divide and conquer table	Searching is not efficient and also creates bottleneck because of shifting of DCT entries.

XI. Conclusion

The data on the cloud is increasing day by day and due to less control of the owner of data the data is at risk of different attacks. This paper review the cloud computing organization and data integrity schemes for outsourced data on cloud storage. The limitation in current work is also highlighted.

References

[1] P. Jain, M. Gyanchandani, and N. Khare. (2016). Big data Privacy: A Technological Perspective and Review. *J. Big Data*, 3(1), 25.

[2] S. Singh, Y. S. Jeong, and J. H. Park. (2016). A Survey on Cloud Computing Security: Issues, Threats, and Solutions. *J. Netw. Comput. Appl.* 75, 200-222.

[3] M. Sookhak, A. Gani, M. K. Khan, and R. Buyya. (2017). Dynamic Remote Data Auditing for Securing Big Data Storage in Cloud Computing. *Inf. Sci. (Ny)*, 380, 101-116.

[4] M. Ali, S. U. Khan, and A. V. Vasilakos. (2015). Security in Cloud Computing: Opportunities and Challenges. *Inf. Sci. (Ny)*, 305, 357-383.

[5] D. A. B. Fernandes, L. F. B. Soares, J. V Gomes, M. M. Freire, and P. R. M. Inácio. (2014). Security Issues in Cloud Environments: A Survey. *Int. J. Inf. Secu.*, 13(2), 113-170.

[6] S. Subashini and V. Kavitha. (2011). A Survey on Security Issues in Service Delivery Models of Cloud Computing. *J. Netw. Comput. Appl.*, 34(1), 1-11.

[7] Q. Wang, S. Member, C. Wang, S. Member, and K. Ren. (2012). Enabling Public Auditability and Data Dynamic in Cloud Computing. *IEEE Trans. Parallel Distrib. Syst.*, 22(5), 847-859.

[8] K. Zeng. (2008). Publicly Verifiable Remote Data Integrity. 419-434.

[9] Y. Yu, Y. Zhang, J. Ni, M. H. Au, L. Chen, and H. Liu. (2015). Remote Data Possession Checking with Enhanced Security for Cloud. *Futur. Gener. Comput. Syst.*, 52, 77-85.

[10] L. Chen, S. Zhou, X. Huang, and L. Xu. (2013). Data Dynamics for Remote Data Possession Checking in Cloud Storage. *Comput. Electr. Eng.*, 39(7): 2413-2424.

[11] F. Zafar *et al.* (2017). A Survey of Cloud Computing Data Integrity Schemes: Design Challenges, Taxonomy and Future Trends. *Comput. Secur.*, 65, 29-49.

[12] C. C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia. (2009). Dynamic Provable Data Possession. *CCS '09 Proc. 16th ACM Conf. Comput. Commun. Secur.*, 213-222.

[13] M. Sookhak, H. Talebian, E. Ahmed, A. Gani, and M. K. Khan. (2014). A Review on Remote Data Auditing in Single Cloud Server: Taxonomy and Open Issues. *J. Netw. Comput. Appl.*, 43, 121-141.

[14] G. Ateniese *et al.* (2007). Provable Data Possession at Untrusted Stores. *Proc. 14th ACM Conf. Comput. Commun. Secur. - CCS '07*, 1, 598.

[15] C. Papamanthou, R. Tamassia, and N. Triandopoulos. Authenticated Hash Tables Categories and Subject Descriptors. 437-448.

[16] K. Yang and X. Jia. (2013). An Efficient and Secure Dynamic Auditing Protocol for Data Storage in Cloud Computing. *IEEE Trans. Parallel Distrib. Syst.*, 24(9), 1717-1726.

[17] Y. Yu *et al.* (2016). Cloud Data Integrity Checking with an Identity-based Auditing Mechanism from RSA. *Futur. Gener. Comput. Syst.*, 62, 85-91.

[18] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik. (2008). Scalable and Efficient Provable Data Possession. *Proc. 4th Int. Conf. Secur. Priv. Commun. Netowrks - Secur. '08*, 1.

[19] G. Ateniese. (2009). Proofs of Storage from Homomorphic Identification Protocols, 1-14.

[20] R. C. Merkle. (1979). Ralph C. Merkle ELXSi International, pp. 122-134.

[21] S. J. Thomas Schwarz and E. L. Miller. (2006). Store, Forget, and Check: Using Algebraic Signatures to Check Remotely Administered Storage. *Proc. - Int. Conf. Distrib. Comput. Syst.*

[22] L. Chen. (2013). Using Algebraic Signatures to Check Data Possession in Cloud Storage. *Futur. Gener. Comput. Syst.*, 29(7), 1709-1715.