2021 IEEE 6th International Conference on Intelligent Computing and Signal Processing (ICSP 2021)

A Novel Cloud Data Integrity Verification Scheme Based on Dynamic Array Multi-branch Tree

Ting Zhao[1,2], Zhi-yong Zhang[1,2*] and Li-li Zhang[1,2]
[1)] Information Engineering College, Henan University of Science and Technology, Luoyang, China
[2)] Henan International Joint Laboratory of Cyberspace Security Applications, Henan University of Science and Technology, Luoyang, China
E-mail: xidianzzy@126.com

2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP) | 978-1-6654-0413-6/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICSP51882.2021.9408776

***Abstract*—With the continuous development of cloud storage technology, more and more users choose to store their data to the cloud server to save local storage costs, but at the same time, they also lose direct control over their data, so ensuring data integrity becomes a challenge. A cloud storage data integrity verification scheme based on Dynamic Array Multi-branch Tree (DA-MBT) is proposed. The leaf node is set as an array structure, which reduces the height of the tree, improves the utilization rate of nodes, simplifies the process of dynamic update, reduces the query time of the data block, and improves the verification efficiency. Finally, it can be concluded that, the proposed DA-MBT scheme supports full dynamic operation, realizes the privacy protection of users, can effectively verify the integrity of a large number of data, and can effectively reduce the communication and computing overhead during the verification process.**

Keywords- cloud storage; Integrity verification; Public verification; Dynamic update; Privacy protection

I. INTRODUCTION

Cloud storage is to expand users' storage space at a lower cost through virtualization technology, enabling users to access cloud data anytime and anywhere [1]. However, users also lose the ability to physically control the cloud data when they store it in the cloud. Cloud service providers may deliberately discard some data that users do not often access in order to save storage costs [2]. However, to maintain their reputation or to avoid the problem of compensation, cloud service providers may conceal these accidents. These data security problems have greatly reduced people's trust in cloud storage services and seriously affected the promotion and application of cloud storage services. Therefore, cloud data integrity verification has become an urgent problem to be studied.

At present, data integrity verification schemes can be divided into two categories: proof of data possession (PDP) and proof of retrievability (POR) [3]. PDP focuses on identifying whether the data is damaged at a high rate, while POR is to detect the data is damaged and recover the damaged data with a small probability [4][5]. The difference between the two lies in the different emphasis when verifying the integrity of user data.

Ateniese et al. [6] first proposed the provable data possession (PDP) scheme, which used sampling technology and RSA homomorphism label to check the integrity of cloud data. The scheme effectively reducing communication and computing overhead in the verification process, but did not take into account the dynamic nature of data in cloud storage. Later, Erway et al. [7] and Wang et al. [8] proposed the PDP scheme of full dynamic update one after another. In the scheme, Erway et al. introduced the data structure of authentication hop table to construct the PDP protocol, but authentication hop table requires too much auxiliary information when calculating the path, with low efficiency and large communication overhead. Wang et al. used another data structure, the Merkle Hash Tree (MHT), to achieve dynamic updating. Again, too much auxiliary information was needed to calculate the path. Yang [9] and others use the index hash table structure to store the index changes of data blocks, which also achieves the dynamic operation of the scheme. Recently, Li Yong et al. [10] adopted a Large Branching Tree (LBT) dynamic data structure to optimize the MHT structure. The LBT structure reduces the height of the construction tree and simplifies the process of data updating. However, the scheme does not take into account the possibility that the returned proof could be stolen by an untrusted third party. Zhu [11] et al. proposed a weighted single linked list large branching tree scheme (WSLBT) to optimize the LBT scheme, reduce the height of the tree and further improve the verification efficiency. Xie [12] et al. adopted the structure of Multi-Branch Tree (MBT), which effectively reduced the height of the Tree and supported multi-user authentication, but the communication and calculation overhead were high. According to the above problem, this paper proposes a Dynamic Array based on Dynamic Array Multi - branch Tree (DA-MBT) cloud storage data integrity verification scheme, which set the leaf node into Array structure, reduce the height of the Tree, in the guarantee data block content and position of the right, at the same time reduces the cloud server data block search speed. The proposed scheme effectively reduce the computational and communication overhead in the process of the validation.

II. INTRODUCTION OF THE DYNAMIC ARRAY MULTI - BRANCH TREE

In the DA-MBT, the leaf node corresponds to a dynamic array, in which the hash value of data blocks is stored. The introduction of the DA-MBT structure not only ensures the correctness of the location of data blocks, but also makes up for the deficiency of the linear growth of the depth of the tree with the increase of data blocks. In the DA-MBT, for a node x, if it is a leaf node, the stored value is $H(x)$, which represents the

978-1-6654-0413-6/21/$31.00 ©2021 IEEE

value obtained after the hashing operation of all data blocks in the corresponding dynamic array of the x node. If it is a non-leaf node, it represents the value obtained by hashing again after linking all child nodes of the x node. L Represents the length of the dynamic array. The DA-MBT structure is shown in Fig. 1.

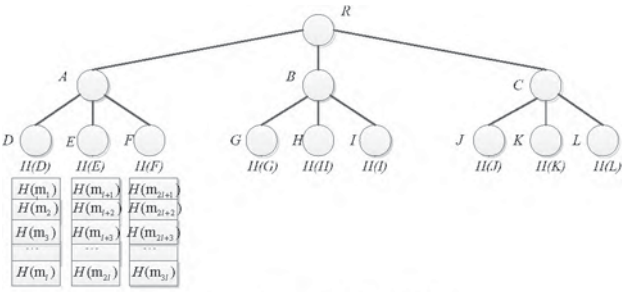

Figure 1. Structure of DA-MBT

III. DATA INTEGRITY VERIFICATION SCHEME

A. Basic operation algorithm

In this paper, the scheme is divided into four phase: initialization phase, challenge phase, response phase, audit phase.

Initialization phase: Includes two polynomial time algorithms KeyGen and SigGen.

In the KeyGen algorithm, private keys x are randomly selected from Z_p, $e: G \times G \to G_T$ is a bilinear mapping, g is a generator of G, calculate the public key $v = g^x$. $H: \{0,1\}^* \to G$ and $h: G_T \to Z_p$ are hash functions.

In the SigGen algorithm, first the user divides the file F into n blocks $\{m_1, m_2, ..., m_n\}$, and then randomly selects parameters $u \leftarrow G$ to generate a label $T_i = (H(m_i) \cdot u^{m_i})^x$ for each data block. All the labels T_i form the label set $\phi = \{T_i\}(i = 1, 2, \ldots, n)$. The user constructs DA-MBT according to the divided data blocks and iterates upward to calculate the value of the root node $H(R)$. The root node is signed with the private key $Sig_{sk}(H(R)) = H(R)^x$. Finally the user sends the file data block $\{m_1, m_2, ..., m_n\}$, label set ϕ and root node signature $Sig_{sk}(H(R))$ to the cloud server. Parameters g 、 v and u are all public parameters. The cloud server builds and saves DA-MBT structure in the same steps according to the files sent by users.

Challenge phase: This phase is mainly completed by the third party auditor TPA, which is authorized by the user to periodically challenge the cloud server to verify the integrity of the data. In order to ensure the randomness of the challenge, the TPA randomly selects c values from the index set $S = \{S_1, S_2, \cdots, S_c\}$ of the data block to form the index set. The TPA randomly selects a random non-negative number v_i from Z_p for each index data block to form a challenge pair $chal = \{(i, v_i)\}_{i \in S}$ and send it to the server.

Response phase: this phase is mainly completed by the cloud server provider CSP. After receiving the challenge information sent by the TPA, the CSP runs the GenPro algorithm. The CSP according to the index value of the challenge information, find the corresponding data block $H(m_i)$ and the leaf node $H(k_i)$ where the data block is located, k_i represent the leaf node corresponding to the data block i. Then backtrack the DA-MBT to find all the sibling nodes on the path from the leaf node to the root node to form the auxiliary information $\{\Omega_i\}_{i \in S}$. The CSP calculate equation $\mu = r + \gamma \sum_{i \in S} v_i m_i$ and $T = \prod_{i \in S} T_i^{v_i}$, there r from Z_p, $\gamma = h(\gamma')$ and $\gamma' = e(u, v)^r$. Here a random mask technique is used to hide the linear combination $\sum_{i \in S} v_i m_i$, prevent untrusted TPA to use Gauss elimination method to steal user data. Finally, the cloud server sends the generated evidence $Pro = \{\mu, T, \gamma', H(m_i)\{H(k_i), \Omega_i\}_{i \in S}, Sig_{sk}(H(R))\}$ to TPA as response information.

Audit phase: This phase is mainly completed by TPA. The TPA after receiving cloud server response information, perform VerPro algorithm. First of all, the TPA recalculates the root node $H(R)'$ according to $H(k_i)$ and $\{\Omega_i\}_{i \in S}$. Then the TPA judging equation $e(Sig_{sk}(H(R)), g) = e(H(R)', g^x)$ holds or not. If not, FALSE is returned. Otherwise, the TPA calculates $\gamma = h(\gamma')$ and continues to verify whether equation $\gamma' e(T^\gamma, g) = e(\prod_{i \in S} H(m_i)^{v_i \gamma} \cdot u^\mu, v)$ is true. If not, the TPA returns FALSE, otherwise it return TRUE.

B. Dynamic update of data

In practical applications, users may conduct dynamic operations on data, such as adding, deleting and modifying data. Therefore, dynamic update becomes an important attribute of data integrity verification. The user needs to provide relevant information to the CSP when the user updates the data, such as the location of the data block and the label of the data block. After receiving the information from the client, the CSP updates the DA-MBT tree according to the requirements and recalculates the values of all nodes on the path from the leaf node to the root node. Next, we mainly introduce the data modification operation, other operations and so on. It seems that we won't do a detailed introduction here.

In the DA-MBT, data modification means that CSP can replace the original data block with the new data block according to the user's request, without downloading the entire file. Suppose the user wants to modify block i, the user needs to calculate $T_i' = (\mathrm{H}(m_i') \cdot u^{m_i'})^x$ from m_i' and constructs the update request $modify = \{M, i, m_i', T_i'\}$. The request $modify$ consists of a quad, M represents the data modification operation, i represents the location of the data block to be

modified, $m_i^{'}$ represents the value of the new data block, and $T_i^{'}$ represents the new data block label. The user sends the request *modify* to the CSP. After receiving the update request, the CSP determines that this is a data modification operation according to M .The CSP then replaces the old data m_i with the new block $m_i^{'}$ and the old label values T_i with the new label value $T_i^{'}$ .Finally, the CSP calculate and update the hash values of leaf node and all nodes on the path from leaf node to root node. The process of modifying a data block (block 20) is shown in Fig. 2 and Fig. 3.

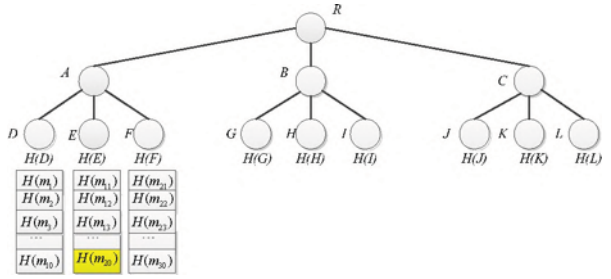

Figure 2. Initial DA-MBT structure

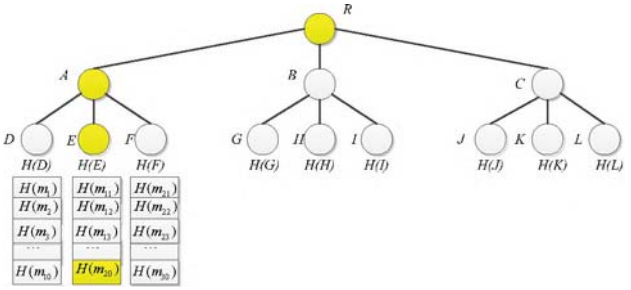

Figure 3. The modified DA-MBT structure

IV. CORRECTNESS AND SAFETY ANALYSIS

A. *Correctness Analysis*

In this scheme, if both TPA and CSP enforce the agreement as required, and the cloud data is not damaged, then the proof from CSP can be proved by the TPA.

1) Equation $e(Sig_{sk}(H(R)),g)=e(H(R)^{'},g^{x})$ is proved as follows:

Because equation $Sig_{sk}(H(R))=H(R)^{x}$ holds, so equation $e(Sig_{sk}(H(R)),g)=e(H(R)^{x},g)$ holds, according to the property of bilinear mapping $e(H(R)^{x},g)=e(H(R),g^{x})$, so equation $e(Sig_{sk}(H(R)),g)=e(H(R),g^{x})$ holds. TPA recalculates the value of the root node $H(R)^{'}$ according to the information returned by the CSP. If and only if $H(R)^{'}=H(R)$, the above equation is true.

2) Equation $\gamma^{'}e(T^{\gamma},g)=e(\prod_{i\in S}H(m_i)^{v_i\gamma}\cdot u^{\mu},v)$ is proved as follows:

$$\gamma^{'}e(T^{\gamma},g)=e(u,v)^{r}\cdot e((\prod_{i\in S}T_i^{v_i})^{\gamma},g)$$

$$=e(u,v)^{r}\cdot e((\prod_{i\in S}(H(m_i)\cdot u^{m_i})^{xv_i})^{\gamma},g)$$

$$=e(u,v)^{r}\cdot e((\prod_{i\in S}(H(m_i)\cdot u^{m_i})^{v_i})^{\gamma},g)^{x}$$

$$=e(u,v)^{r}\cdot e((\prod_{i\in S}H(m_i)^{v_i}\cdot u^{m_iv_i})^{\gamma},v)$$

$$=e((\prod_{i\in S}H(m_i)^{v_i})^{\gamma}\cdot u^{\gamma\sum_{i\in S}m_iv_i+r},v)$$

$$=e((\prod_{i\in S}H(m_i)^{v_i})^{\gamma}\cdot u^{\mu},v)$$

B. *Safety analysis*

The security analysis of the scheme is as follows:

The proposed scheme can resist replay attack: replay attack refers to that the CSP damages, loses or tampers the data stored on it due to its own reasons. After the data integrity verification cannot be completed, the proof provided before is sent back to the user to cover up the fact that its data is not available to deceive the user. If some data blocks are not available due to the CSP, when data integrity validation is performed, proof that has previously passed data integrity validation is returned for validation. However, random number v_i is involved in the generation of proof so that T and μ are different each time, so the CSP cannot pass the verification through replay attack.

The proposed scheme can resist delete attack: delete attack refers to the CSP may delete the data that users seldom access for its own benefit. If the cloud server tries to forge the label to pass the integrity verification, it needs to complete the forgery of T . However, the private key participates in the calculation of T_i , and CSP cannot obtain the private key. So CSP can't use delete attack to cheat TPA.

The proposed scheme can resist tamper attack: tamper attack refers to that the CSP maliciously modifies the data stored by the user, fails to pass the integrity verification, and then provides forged evidence to deceive the user. Assuming that m_i is modified to m_i^{*} , due to the collision resistance of the hash function, the equation $H(m)=H(m^{*})$ does not hold. So it cannot be verified by the TPA.

V. PERFORMANCE ANALYSIS

By comparing with the existing scheme, the results shown in Tab. 1 are obtained, where n represents the number of data blocks, k represents the emergence degree of the tree, and l represents the initial length of the corresponding structure of the leaf node.

Table 1. Performance comparison.

plan	Public validation	Dynamic update	security	Server lookup speed	Root node computing complexity	Communication complexity
Reference [8]	NO	NO	NO	$O(\log_2 n)$	$O(\log_2 n)$	$O(\log_2 n)$
Reference [10]	YES	YES	NO	$O(\log_k n)$	$O(\log_k n)$	$O(\log_k n)$
Reference [12]	YES	YES	YES	$O(\log_k n)$	$O(\log_k n)$	$O(\log_k n)$
This paper	YES	YES	YES	$O(\log_k \frac{n}{l}) + O(1)$	$O(\log_k \frac{n}{l})$	$O(\log_k \frac{n}{l})$

VI. Conclusion

With the popularization and application of cloud storage services, the traditional data integrity verification scheme cannot meet the needs of the existing environment. So in this paper, a data integrity verification scheme was proposed, which is suitable for the cloud environment, the structure of Dynamic Array Multi-branch Tree has be used to realize the dynamic update, the introduction of a third party auditor realizes the public verification, random mask technology be apply to protect the users' privacy. Compared with other schemes, ours has less computational costs and communication overhead.

Acknowledgment

The work was supported by National Natural Science Foundation of China Grant No.61972133, Project of Leading Talents in Science and Technology Innovation for Thousands of People Plan in Henan Province Grant No.204200510021, and Program for Henan Province Key Science and Technology No.212102210383. We thank reviewers and editors for their valuable suggestions, comments and helps.

References

[1] CHEN F, MENG F, XIANG T, et al. Towards Usable Cloud Storage Auditing [J]. IEEE Transactions on Parallel and Distributed Systems, 2020, 31(11): 2605-2617.

[2] MILTON G, MANIKANDANS P. An Efficient Integrity Verification and Authentication Scheme over the Remote Data in the Public Clouds for Mobile Users [J]. Security and Communication Networks, 2020.

[3] CHOON B T, MOHD H A H, LIM Y, GANI A. A survey on Proof of Retrievability for cloud data integrity and availability: Cloud storage state-of-the-art, issues, solutions and future trends[J]. Journal of Network and Computer Applications, 2018, 110: 75-86.

[4] JUELS A, KALISKI B S. Pors: proofs of retrievability for large files[C] // ACM Conference on Computer and Communications Security. ACM, 2007: 584-597.

[5] LIANG W, FAN Y, LI K C, ZANG D and GAUDIOT J. Secure Data Storage and Recovery in Industrial Blockchain Network Environments [J]. IEEE Transactions on Industrial Informatics, 2020, 16(10): 6543-6552.

[6] ATENIESE G, BURAN R, CURTMOLA R, et al. Provable data possession at untrusted stores[C] // ACM Conference on Computer & Communications Security. ACM, 2007: 598-609.

[7] ERWAY C C, KÜPÇÜ A, PAPAMANTHOU C, et al. Dynamic Provable Data Possession [J]. ACM Transactions on Information & System Security, 2015, 17(4): 1-29.

[8] WANG Q, WANG C, REN K, et al. Enabling public auditability and data dynamics for storage security in cloud computing [J]. IEEE Transactions on Parallel & Distributed Systems, 2011, 22(5): 847-859.

[9] Yang, Kan, Jia, et al. An Efficient and Secure Dynamic Auditing Protocol for Data Storage in Cloud Computing [J]. IEEE Transactions on Parallel and Distributed Systems, 2013, 24(9):1717-1726.

[10] LI Y, YAO G, LEI L N, et al. Cloud storage data integrity verification mechanism based on large branching tree [J]. Journal of Tsinghua University (Science and technology), 2016, 56(5): 504-510.

[11] ZHU Y, CHEN Y, YAN X C, et al. A cloud data integrity verification scheme for weighted single linked list multi branch tree [J]. Journal of Chinese Mini-Micro Computer Systems, 2020, 041(003): 575-580.

[12] XIE S J, JIA B, WANG E, et al. Cloud storage big data integrity proof mechanism based on multi-branch tree [J]. Computer Science, 2019, 046(003):188-196.