Dynamic Auditing Protocol for Efficient and Secured Data Storage in Cloud Computing

R.Parameshwaran
Masters of Computer Science and Engineering
M.A.M. College of Engineering
Trichirapalli, India
cseparamu021@gmail.com

A.N.Gnana Jeevan
Associate Professor,
Department of Computer Science and Engineering,
M.A.M. College of Engineering
Trichirapalli, India
jeevan@mamce.org

Abstract

Cloud computing, where the data has been stored on cloud servers and retrieved by users (data consumers) the data from cloud servers. However, there are some security challenges which are in need of independent auditing services to verify the data integrity and safety in the cloud. Until now a numerous methods has been developed for remote integrity checking whichever only serve for static archive data and cannot be implemented to the auditing service if the data in the cloud is being dynamically updated. Therefore, it is expected to design an efficient and secure dynamic auditing protocol to convince the data owners for the security and integrity of their data. In this paper, we intent to construct an auditing framework for cloud storage systems for efficient privacy-preserving auditing service. Then, our auditing protocol is extended to support the data dynamic operations for secure auditing in the random oracle model. In addition, our auditing protocol is improved to support batch auditing for both multiple owners and multiple clouds without any trusted organizer. Our proposed auditing protocols will be proved for their secure and efficient computation with reduced cost for the auditing.

I INTRODUCTION

Cloud storage is one of the data storage systems which known as cloud computing [1]. It permits data owners to store data in cloud from their local computing systems (data hosting service). Presently cloud computing is used by more and more owners for storage of data [2] in remote locations to reduce the storage cost in their own system and comfortable in carrying. However, data stored in cloud also introduces some challenges [3] like security and integrity of data. In addition, the data could be lost in the cloud infrastructure, no matter what high degree of reliable measures cloud service providers would take [4]. Sometimes, cloud service providers remove some stored data whichever have not been used for long time to save their storage space and dishonestly convince the owners that the data are correctly stored in the cloud.

Usually, the data integrity is checked by two-party storage auditing protocols [5], [6], [7]. However, this cloud storage system could mostly not be guaranteed to provide unbiased auditing result which is inappropriate for any storage. Now-a-days, third-party auditing is widely chosen for the storage auditing in cloud computing.

A Third-Party Auditor (auditor) can convince both cloud service providers and owners by its capabilities to do a more efficient work. There are some important requirements for the third-party auditing in cloud storage systems [8], [9] which are as follows:

1) *Confidentiality*. The auditing protocol should keep owner's data confidential against the auditor.

2) *Dynamic auditing*. The auditing protocol should support the dynamic updates of the data in the cloud.

3) *Batch auditing*. The auditing protocol should also be able to support the batch auditing for multiple owners and multiple clouds.

At present, there are variety of protocols has been developed and implemented for checking remote integrity of the data by the auditor on the remote server [10],[11]. They cannot be applied to cloud storage systems because of that they do not have the ability to preserve privacy of the data and also cannot support the data dynamic operations.

We can find a numerous records on the dynamic auditing protocols whichever have their own pros and cons as per cloud storage servers. Wang et. al., [12] proposed a dynamic auditing protocol that can support the dynamic operations of the data on the cloud servers, but this method may leak the data content to the auditor because it requires the server to send the linear combinations of data blocks to the auditor. The same group [10] already studied their dynamic auditing scheme to be privacy preserving and support the batch auditing for multiple owners. However, the large number of data tag used in their scheme gain a heavy storage overhead on the server. Zhu et al., [13] introduced a cooperative provable data possession scheme and found that it can support the batch auditing for multiple clouds and the dynamic auditing [11]. However, their scheme was not valid in the batch auditing for multiple owners due to different in the parameters for generating the data tags used by owners and also have drawback in combining the data tags from multiple owners for batch auditing. Another drawback of their protocols is the need of an additional trusted organizer to forward a job due to the applied mask technique to ensure the data privacy for the multiple cloud batch auditing. Additionally, both Wang's schemes and Zhu's schemes are in need of heavy computation cost of the auditor that restricts the good performance of the auditor

In this paper, an efficient and secure dynamic auditing protocol has been proposed to overcome the drawbacks and fulfil the requirements to store data with dynamic operations. The encryption and decryption of the data can be possible which is generated by using the Bilinearity property of the bilinear pairing that can be very useful to check the correctness of the proof. Here is no need of the mask technique, so our method avoids the requirement of trusted organizer during the batch auditing for multiple clouds. Further, our method provides the possibility to verify the correctness of the proof with an intermediate value. Therefore, the computing problems of the auditors in the cloud storage server can be reduced.

The outline of our proposed method has the following essential qualities:

1) Our proposed system can efficiently store the data with a great privacy. The cryptography method and the Bilinearity property of the bilinear pairing are used to develop our auditing protocol which ensures the data privacy. Our method reduces the computation and communication cost.
2) Data dynamic operations are possible by our auditing protocol.
3) Our protocol also provides good efficiency in batch auditing for both multiple clouds and owners without any additional trusted organizers with large-scale cloud storage systems.

The following sections demonstrate the basic concepts and proposed protocol model.

II BACKGROUND

A. System Model

In general any auditing systems for cloud storage have three essential members as shown in Fig (1).

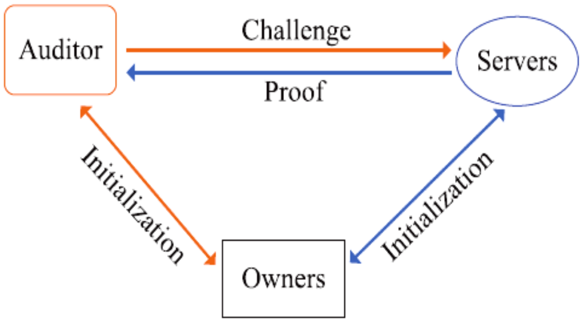


Fig. I - System model of the data storage auditing

They are data owners (owner), the cloud server (server), and the Third-Party Auditor (auditor). The owners are the data creators who store their data in the cloud. The cloud server stores the owners' data and provides the data access to users (data consumers). The auditor is a trusted third-party who expertise in data storage and has capabilities secure auditing service for both the owners and servers. The auditor can be a trusted organization managed by the government, which can provide unbiased auditing result for both data owners and cloud servers.

Symbol	Physical Meaning
sk_t	secret tag key
pk_t	public tag key
sk_h	secret hash key
M	data component
T	set of data tags
n	number of blocks in each component
s	number of sectors in each data block
M_{info}	abstract information of M
$\mathcal{C}$	challenge generated by the auditor
$\mathcal{P}$	proof generated by the server

TABLE I - Notations

The cloud services have some of the protocol definitions which are listed in Table I.

Protocol Definitions in Storage auditing protocol

A storage auditing protocol consists of the following five algorithms:

1. KeyGen $(\lambda) \rightarrow (sk_h, sk_t, pk_t)$

   This key generation algorithm takes no input other than the implicit security parameter λ. It outputs a secret hash key sk_h and a pair of secret-public tag key (sk_t, pk_t).

2. TagGen $(M, sk_t, sk_h) \rightarrow T$

   The tag generation algorithm takes as inputs an encrypted file M, the secret tag key sk_t, and the secret hash key sk_h. For each data block mi, it computes a data tag t_i based on sk_h and sk_t. It outputs a set of data tags $T=\{t_i\}_{i€[1,n]}$.

3. Chall $(M_{info}) \rightarrow \mathcal{C}$

   The challenge algorithm takes as input the abstract information of the data M_{info} (e.g., file identity, total number of blocks, version number, time stamp, etc.). It outputs a challenge $\mathcal{C}$.

4. Prove $(M, T, \mathcal{C}) \rightarrow \mathcal{P}$

   The prove algorithm takes as inputs the file M, the tags T, and the challenge from the auditor $\mathcal{C}$. It outputs a proof $\mathcal{P}$.

5. Verify $(\mathcal{C}, \mathcal{P}, sk_h, pk_t, M_{info}) \rightarrow 0/1$

   The verification algorithm takes as inputs $\mathcal{P}$ from the server, the secret hash key sk_h, the public tag key pk_t, and the abstract information of the data M_{info}. It outputs the auditing result as 0 or 1. **[14]**

B. Security Model

The auditor and server are play the most essential role in the security of the cloud computing. In general, the auditor can stand as side of the owner who audits the whole procedure with sufficient honest, but he is curious about the received data. Whereas, the sever could be dishonest and may launch the following problems to the clouded data:

1. *Replace attack*. The server may choose another valid and uncorrupted pair of data block and data tag m_k,t_k to

replace the challenged pair of data block and data tag (m_i,t_i), when it already discarded m_i or t_i.

2. *Forge attack*. The server may forge the data tag of data block and deceive the auditor; if the owner's secret tag keys are reused for the different versions of data.

3. *Replay attack*. The server may generate the proof from the previous proof or other information, without retrieving the actual owner's data. **[14]**

III EFFICIENT AND PRIVACY-PRESERVING AUDITING PROTOCOLS

The basic techniques used to construct our dynamic auditing service will be presented along with the necessary algorithms applied to the design of the auditing protocol for cloud storage systems.

A. Importance of Our Solution

It is important to design a data storage auditing protocol which can protect the privacy of the data against the auditor. The reasons are as follows:

1) The auditor may collect the data information from the data proof through the data blocks in the case of public data.

2) In the case of encrypted data, the auditor may somehow get the content keys by anyway and also can decrypt the data.

The solution for these issues can be developed by generating an encrypted proof with the challenge stamp by using the bilinearity property of the bilinear pairing. So the auditor can only check the correctness of the data without decrypting it. And further, our method instructs the server to compute the proof as an intermediate value of the verification. Therefore, the auditor can only verify the intermediate value which improves the security of owner's data and also reduce the computing loads of the auditor.

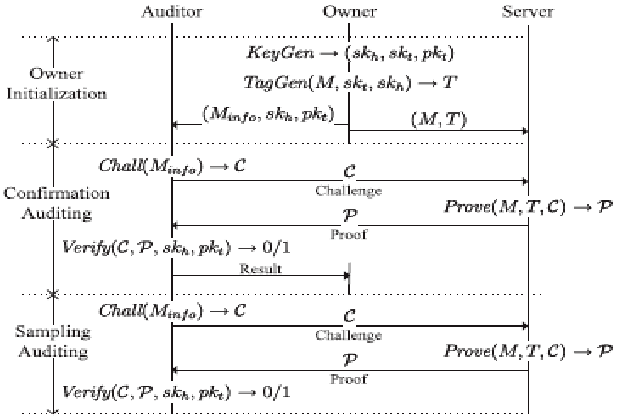

Auditor
Owner
Server
Owner Initialization
$KeyGen \rightarrow (sk_h, sk_t, pk_t)$
$TagGen(M, sk_t, sk_h) \rightarrow T$
(M_{info}, sk_h, pk_t)
(M, T)
Confirmation Auditing
$Chall(M_{info}) \rightarrow \mathcal{C}$
$\mathcal{C}$
Challenge
$Prove(M, T, \mathcal{C}) \rightarrow \mathcal{P}$
$\mathcal{P}$
Proof
$Verify(\mathcal{C}, \mathcal{P}, sk_h, pk_t) \rightarrow 0/1$
Result
Sampling Auditing
$Chall(M_{info}) \rightarrow \mathcal{C}$
$\mathcal{C}$
Challenge
$Prove(M, T, \mathcal{C}) \rightarrow \mathcal{P}$
$\mathcal{P}$
Proof
$Verify(\mathcal{C}, \mathcal{P}, sk_h, pk_t) \rightarrow 0/1$

Fig. II Framework of our privacy-preserving auditing protocol.

And further, our method instructs the server to compute the proof as an intermediate value of the verification. Therefore, the auditor can only verify the intermediate value which improves the security of owner's data and also reduce the computing loads of the auditor. Our method reduces the storage loads and communication cost by applying the data fragment technique and homomorphic verifiable tags. Therefore, the performance of the auditing system can be improved due to the reduced number of tags and data blocks.

B. Our Privacy-Preserving Auditing Protocol - Process

The construction of our auditing protocol consists of three important steps as shown in Fig. II. They are owner initialization, confirmation auditing, and sampling auditing which are described below in detail.

Step 1: *Owner initialization*-(owner generates the keys and the tags for the data)

The owner runs the key generation algorithm *KeyGen* to generate the secret hash key sk_h, the pair of secret-public tag key $(sk_t,\ pk_t)$. Then, it runs the tag generation algorithm *TagGen* to compute the data tags. After all the data tags are generated, the owner sends each data component $M = \{m_i\}_{i\in[1,n]}$ and its corresponding data tags $T=\{t_i\}_{i\in[1,n]}$ to the server together with the set of parameters $\{u_j\}_{j\in[1,s]}$. The owner then sends the public tag key pk_t, the secret hash key sk_h, and the abstract information of the data M_{info} to the auditor, which includes the data identifier FID, the total number of data blocks *n*.

Step 2: *Confirmation auditing*-(owner confirms the data stored in the server with auditor) In our auditing construction, the auditing protocol only involves two-way communication: Challenge and Proof. During the confirmation auditing phase, the owner requires the auditor to check whether the owner's data are correctly stored on the server. The auditor conducts the confirmation auditing phase as

a) The auditor runs the challenge algorithm **Chall** to generate the challenge $\mathcal{C}$ for all the data blocks in the data component and sends the $\mathcal{C} = (\{i,v_i\}_{i\in Q}\, R)$ to the server.

b) Upon receiving the challenge $\mathcal{C}$ from the auditor, the server runs the **prove** algorithm Prove to generate the proof $\mathcal{P} = (TP,\ DP)$ and sends it back to the auditor.

c) When the auditor receives the proof $\mathcal{P}$ from the server, it runs the verification algorithm **Verify** to check the correctness of $\mathcal{P}$ and extract the auditing result.

The auditor then sends the auditing result to the owner. If the result is true, the owner is convinced that its data are correctly stored on the server, and it may choose to delete the local version of the data.

Step 3: *Sampling auditing*-(owner deletes the local copy & auditor conducts sampling auditing)

The auditor will carry out the sampling auditing periodically by challenging a sample set of data blocks. The frequency of taking auditing operation depends on the service agreement between the data owner and the auditor (and also depends on how much trust the data owner has over the server). Similar to the confirmation auditing in Step 2, the sampling auditing procedure also contains two-way communication as illustrated in Fig. II.

IV SECURE DYNAMIC AUDITING

In the case of dynamic data storage, the data has been frequently updated by the owners. So the auditing service should have the capability to retain the static archive data along with the updated one. Here is more insecurity to the dynamic data by the server. Especially, the server may attack the dynamic data by the following ways:

1) *Replay attack*. The server fails to update the data correctly and sometimes it may use the previous version of the data during auditing.

2) *Forge attack*. The server have possibility to copy the data at the time of dynamic data operations. Then the server can use any data irrespectively to the auditing.

A. Keys for Security in dynamic auditing

The replay attack can be prevented by introducing an *index table* (ITable) to record the abstract information of the data. The ITable consists of four components: Index, *Bi*, *Vi*, and *Ti*. The Index denotes the current block number of data block mi in the data component *M*. *Bi* denotes the original block number of data block mi, and *Vi* denotes the current version number of data block mi. *Ti* is the time stamp used for generating the data tag.

Owner creates the ITable during the owner initialization. The auditor further uses it to update the message with owner. The storage of data can be up-to-date to both owner and the auditor in the data dynamic operation.

The forge attack can be managed by modifying the tag generation algorithm *TagGen*. Specifically, when generating the data tag t_i for the data block m_i, we insert all the abstract information into the data tag by setting W_i=$FID\|_i\|B_i\|_{Vi}\|T_i$, such that the server cannot get enough information to forge the data tag from dynamic operations.

B. Dynamic Auditing: Algorithms & Process

The construction of the dynamic auditing protocol has four steps such as owner initialization, confirmation auditing, sampling auditing, and dynamic auditing.

The dynamic auditing protocol holds the similar three steps as in the privacy preserving auditing protocol that described earlier.

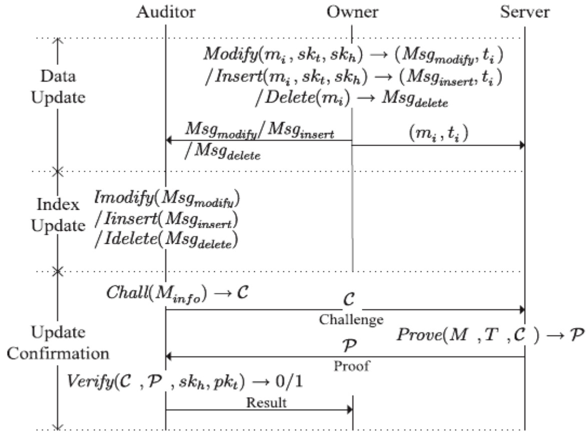
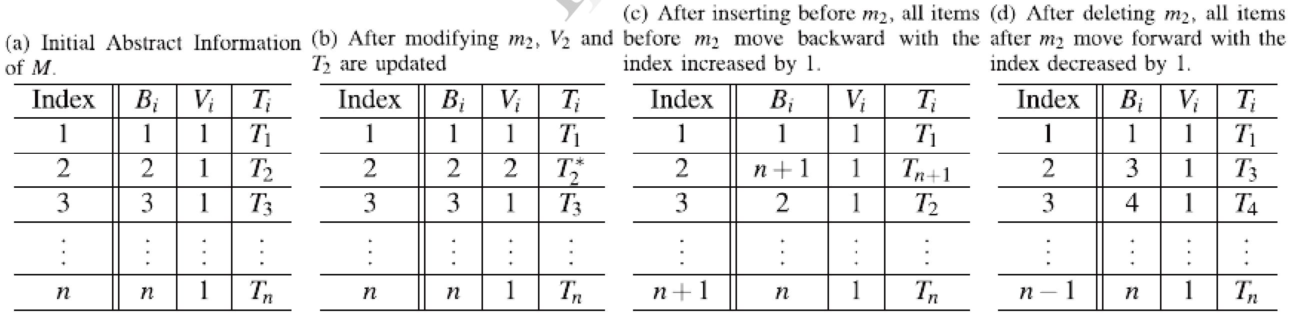

Auditor
Owner
Server
Data Update
$Modify(m_i, sk_t, sk_h) \rightarrow (Msg_{modify}, t_i)$
$/Insert(m_i, sk_t, sk_h) \rightarrow (Msg_{insert}, t_i)$
$/Delete(m_i) \rightarrow Msg_{delete}$
$Msg_{modify}/Msg_{insert}$
$/Msg_{delete}$
(m_i, t_i)
Index Update
$Imodify(Msg_{modify})$
$/Iinsert(Msg_{insert})$
$/Idelete(Msg_{delete})$
$Chall(M_{info}) \rightarrow \mathcal{C}$
$\mathcal{C}$
Challenge
Update Confirmation
$Prove(M, T, \mathcal{C}) \rightarrow \mathcal{P}$
$\mathcal{P}$
Proof
$Verify(\mathcal{C}, \mathcal{P}, sk_h, pk_t) \rightarrow 0/1$
Result

Fig. III Framework of auditing for dynamic operations.

The main change in them is the generation of *TagGen and ITable* keys during owner initialization step (Fig.3). Further, the fourth step namely dynamic auditing contains three steps: data update, index update, and update confirmation which can be explained as follows:

Step 1: *Data update*. Data update operations can be carried out by three types such as modification, insertion, and deletion by the owner. We can use the corresponding algorithm for each update operation to process and facilitate the future auditing.



(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

TABLE II
ITable of the Abstract Information of Data M

Step 2: *Index update*. Index update operations also have three types which is used by the auditor: Index modification, Index insertion, and Index deletion. The auditor may modify three types of algorithms in the ITable corresponding to the update messages. The changes can be made in the ITable.

Step 3: *Update confirmation*. After the necessary modifications have done on the ITable by the auditor, the owner can find the updated data which will be considered for the confirmation of auditing. Once the owner comes to know the status of data update through auditing, he will delete the local version of data.

V Batch Auditing in Multiowner and Multicloud

Presently, the auditing on the stored data serves the owners to check the integrity of data on the cloud servers in cloud computing. But when large number of owners request for the auditing, the auditor have to combine these auditing requests together and conduct the auditing for multiple owners simultaneously which is known as batch auditing. This will improve the system performance. The work reported in ref [13] cannot support the batch auditing for multiple owners due to variation in the parameters for generating the data tags. Therefore, the auditor cannot combine the data tags from multiple owners to perform the batch auditing. In some cases, the data owners use more data which may store on more than one cloud servers which challenges the auditor to construct data tags and verify all the proofs. This will be very tedious process to the auditor and have more computation cost. So, it is essential to combine and verify all the data together.

Mostly mask technique has been used to ensure the data privacy which requires an additional trusted organizer in multicloud batch auditing [13]. Here, we have employed the encryption method with the bilinearity property of the bilinear pairing to protect the data privacy in multicloud batch auditing without any additional trusted organizer.

A. Construction of Batch Auditing

The batch auditing can be constructed by the following steps.

First of all, the multiple owners and multiple clouds have to be batched assets. For example, the set of owners can be taken as “O” and the set of cloud servers can be taken as “S”.

Step 1: *Owner initialization*. At first the owners $O_k(k\text{€}O)$ have to create their own key generation algorithm KeyGen to generate the pair of secret-public tag key $(sk_{t,\ k}, pk_{t,k})$ and a set of secret hash key $\{sk_{h,kl}\}_{l\text{€}S}$. If the owner using different cloud servers, then it is essential to create a different secret hash keys.

Step 2: *Batch auditing for multiowner and multicloud*. After the owner initialization, batch auditing can be started by defining the each owners and cloud servers as O_{chal} and S_{chal} respectively. The batch auditing can be carried out by three steps such as batch challenge, batch proof, and batch verification.

Step (B.i): *Batch challenge*. *BChall* algorithm is used to run the batch challenge C for a set of challenged owners O_{chal} and a set of clouds S_{chal}.

Step (B.ii): *Batch proof*. After the batch challenges has been received, each server $S_l(l\text{€}S_{chal})$ will create a proof $P_l=(TP_l,DP_l)$ which can be run by *BProve* (Batch prove algorithm). The proof P_l will be received by the auditor.

Step (B.iii): *Batch verification*. At the final step, the received proofs P_l from the challenged servers will be checked for their correctness by the batch verification algorithm *BVerify*.

VI RELATED WORKS

Now-a-days cloud computing is the essential process to process global storage through servers which reduce the computational loads for owners in storing data. Therefore, there are a numerous investigations has been available on cloud computing. Though the dynamic auditing on cloud servers is being new and studied by few groups. Recently, Ateniese et al. [18] proposed a dynamic provable data possession protocol based on cryptographic hash function and symmetric key encryption. They develop a protocol to precompute a certain number of metadata within a set up period. So, this method limits the number of updates and needs update operations to recreate all the remaining metadata which cannot be applied for large files. Another group (Erway et al. [19]) followed the PDP model to maintain the dynamic data updates on the stored data. Though they also developed two dynamic provable data possession scheme by using a new version of authenticated dictionaries based on rank information, this scheme causes heavy computation burden to the server due to the PDP scheme.

Kan Yang et al [14], and Cong Wang Et al [20] also proposed different ideas to improve the security and integrity of the cloud computing both in static and dynamic operations. In some cases, which have the leakage of data content to the auditor [12], have heavy storage overhead on servers [10], some can perform batch auditing for multiple clouds only not to multiple owners [11], [13] and also need of additional trusted organizers.

Our proposed method which has been described in the previous sections has significant improvements and importance for dynamic data auditing over the other existing reports.

VII CONCLUSION

In this paper, we proposed and described a protocol to store and retrieve the data in the cloud servers rather than local storage devices. This efficient and inherently secure dynamic auditing protocol can protect the data privacy against the auditor. The combination of the cryptography method with the bilinearity property of bilinear paring can be a very effective tool to secure the large number of owner’s data compared to the mask technique. Therefore, there is no need of additional organizer for the auditing in multicloud and multiowners storages. This also reduces the communication and computation cost by transforming the computing loads of auditing from the auditor to the server. Hence the performance of the cloud computing can be improved and can be useful in large-scale cloud storage systems. Moreover, our method is simple and guarantees the security of the owner’s data from the dishonest auditors by creating intermediate values for updated data.

REFERENCES

[1] H. Shacham and B. Waters, “Compact Proofs of Retrievability,” Proc. Int’l Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology (Asiacrypt), vol. 5350, pp. 90-107, Dec. 2008.

[2] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, “Provable Data Possession at Untrusted Stores,” Proc. 14th ACM Conf. Computer and Comm. Security (CCS ’07), pp. 598-609, 2007.

[3] Juels and J. Burton, S. Kaliski, “PORs: Proofs of Retrievability for Large Files,” Proc. ACM Conf. Computer and Comm. Security (CCS ’07), pp. 584-597, Oct. 2007.

[4] M.A. Shah, R. Swaminathan, and M. Baker, “Privacy-Preserving Audit and Extraction of Digital Contents,” Cryptology ePrint Archive, Report 2008/186, 2008.

[5] M.A. Shah, M. Baker, J.C. Mogul, and R. Swaminathan, “Auditing to Keep Online Storage Services Honest,” Proc. 11th USENIX Workshop Hot Topics in Operating Systems (HotOS ’07), pp. 1-6, 2007.

[6] C. Wang, Q. Wang, K. Ren, and W. Lou, “Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing,” Proc. IEEE INFOCOM, pp. 525-533, 2010.

[7] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S.S. Yau, “Dynamic Audit Services for Integrity Verification of Outsourced Storages in Clouds,” Proc. ACM Symp. Applied Computing, W.C. Chu, W.E. Wong, M.J. Palakal, and C.-C. Hung, eds., pp. 1550-1557, 2011.

[8] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, “Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing,” IEEE Trans. Parallel Distributed Systems, vol. 22, no. 5, pp. 847-859, May 2011.

[9] Y. Zhu, H. Hu, G. Ahn, and M. Yu, “Cooperative Provable Data Possession for Integrity Verification in Multi-Cloud Storage,” IEEE Trans. Parallel and Distributed Systems, vol. 23, no. 12, pp. 2231-2244, Dec. 2012.

[10] Kan Yang and Xiaohua Jia, “An Efficient and Secure Dynamic Auditing Protocol for Data Storage in Cloud Computing”, IEEE Transactions on Parallel and Distributed Systems, VOL. 24, NO. 9, Sep2013

[11] M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M. Isard, “A Cooperative Internet Backup Scheme,” Proc. USENIX Ann. Technical Conf., pp. 29-41, 2003.

[12] Y. Deswarte, J. Quisquater, and A. Saidane, “Remote Integrity Checking,” Proc. Sixth Working Conf. Integrity and Internal Control in Information Systems (IICIS), Nov. 2004.

[13] G. Ateniese, R.D. Pietro, L.V. Mancini, and G. Tsudik, “Scalable and Efficient Provable Data Possession,” IACR Cryptology ePrint Archive, vol. 2008, p. 114, 2008.

[14] C.C. Erway, A. Ku¨ pc¸u¨ , C. Papamanthou, and R. Tamassia, “Dynamic Provable Data Possession,” Proc. ACM Conf. Computer and Comm. Security, E. Al-Shaer, S. Jha, and A.D. Keromytis, eds., pp. 213-222, 2009.

[15] Cong Wang, Sherman S.M. Chow, Qian Wang, Kui Ren and Wenjing Lou,“Privacy-Preserving Public Auditing for Secure Cloud Storage”, IEEE Transactions on Computers, VOL. 62, NO. 2, Feb 2013