IJCSE International Journal of Computer Sciences and Engineering Open Access

Research Paper Volume-5, Issue-5 E-ISSN: 2347-2693

Multiple Auditing Schemes with Integrity and Reliability in Cloud Computing

MS. Sulthana[1*], T. Samatha[2], V. Sravani[3], A. Mahendra[4]

[1]Dept. of Computer Science and Engineering, IIIT- RGUKT, Kadapa, India
[2]Dept. of Computer Science and Engineering, IIIT- RGUKT, Kadapa, India
[3]Dept. of Computer Science and Engineering, IIIT- RGUKT, Kadapa, India
[4]Dept. of Computer Science and Engineering, IIIT- RGUKT, Kadapa, India

[*]Corresponding Author: m.s.sulthana2012@gmail.com, Tel.: 9490409972, 8897925585

Online Available at: www.ijcseonline.org

Received: 20/Apr/2017, Revised: 28/Apr/2017, Accepted: 21/May/2017, Published: 30/May/2017

Abstract— Many users store their data in the cloud storage and benefit from high quality applications and services from a common group of configurable computing resources like networks, servers, storage, applications, and services, by these users can avoid the load of local data storage and protection. However, the fact that users no longer have physical control of the large size of data makes data reliability protection in Cloud computing a challenging task, especially for users with constrained computing resources. Cloud computing is used by many software industries nowadays, since security is not provided in cloud, many companies adopt their unique security structure. To avoid this problem, users can route data to a third party auditor (TPA) he can check the integrity of rooted data.TPA can be securely introduced such that the auditing process should not create any problems towards user data privacy, and should not bring in no added load to user. In this paper, we are securing the user data and providing privacy. We further expand the TPA to carry out multiple auditing tasks concurrently and powerfully. Wide-range of security and performance investigation shows the proposed schemes are provably secure and highly efficient.

Keywords— Data Storage, Privacy-Preserving, Public Review Ability, Cryptographic Protocols, Cloud Computing

I. Introduction

Cloud computing is a model for providing convenient, on-demand network access to a shared pool of computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. As disruptive technology with profound implications, Cloud Computing is specifies the nature of how businesses use information technology. One fundamental characteristic of this model is that data is being centralized or rooted to the Cloud. From users' point of view, storing data to the cloud provides several benefits, free from the burden of data storage, everywhere data access, maintaining cost is low.

While cloud computing provides several advantages, it also brings new security threats towards the user data as in [1]. Cloud service providers are separate entities, data rooting is actually giving up user's ultimate control over the data. So that the reliability of the data in the cloud, being put at risk due to several reasons. First one is infrastructures under the cloud are facing both internal and external threads, Second one is, CSP retrieve data for financial reasons by removal of data that has not been or is not often accessed, or even hide data loss incidents so as to maintain a status. In short, although data rooting is attractive it does not provide guarantee on user data reliability and availability. Thus, in this paper we are enabling a privacy-preserving third-party auditing protocol, which is independent on data encryption.

Here our main work is: support privacy-preserving public auditing in Cloud Computing, with a focus on data storage. as well, with the prevalence of Cloud Computing, a probable increase of auditing tasks from different users may be delegated to TPA .In order to perform individual auditing tasks is inefficient for that purpose a natural demand is then how to enable the TPA to efficiently perform multiple auditing tasks in a batch manner, i.e., simultaneously.

To solve these problems, our work utilizes the technique of public key based homomorphic linear authenticator (or HLA for short), which Enables TPA to perform the auditing without asking the local copy of data and thus significantly minimizes the communication and computation overhead.
1. The proposed system provides privacy preserving public auditing which enables the third party auditor for checking the integrity of the data without learning the data content.
2 Privacy preserving public auditing system provides scalable and well-organized public auditing.
3. We prove the security and rationalize the performance of our proposed schemes through existing experiments and comparisons with the up to date.

II. PROBLEM STATEMENT

2.1 The System and Threat Model

As in [2] we mention a cloud storage service which contains three different entities illustrated in Fig 1:*User* (U),contains large amount of data files to be stored in the cloud; the *cloud server* (CS), which is managed by the *cloud service provider* (CSP) to provide data storage service and has significant differentiate CS and CSP hereafter); the *third party auditor* (TPA), who has expertise and capabilities that cloud users do not have and is trusted to review the cloud storage service reliability on behalf of the user upon request. Users rely on the CS for cloud data storage and maintenance. They may also dynamically interact with the CS to access and update their stored data for various application purposes. To save the computation resource as well as the online burden, cloud users may resort to TPA for ensuring the storage integrity of their outsourced data, while hoping to keep their data private from TPA.

2.2 Design Goals

To enable privacy-preserving public auditing for cloud data storage under the aforementioned model, our protocol design should achieve the following security and performance guarantees.
1) Public audit ability: to allow TPA to verify the correctness of the cloud data on demand without retrieving copy of the data.

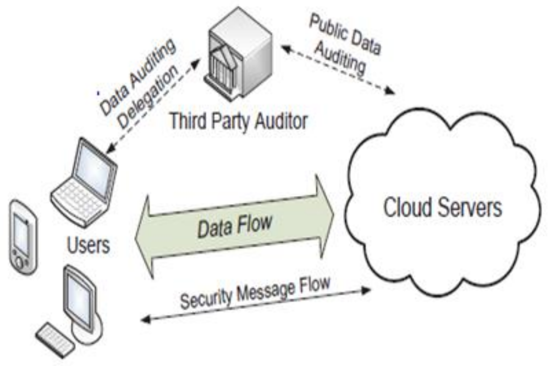


Fig 1: The architecture of cloud data storage service.

2) Storage correctness: to ensure that the data stored in the cloud should be correct.
3) Privacy-preserving: to ensure that the TPA cannot retrieve the user data content.
4) Batch auditing: to make third party auditor in order to perform possibly large number auditing tasks of different users concurrently.
5) Lightweight: to allow TPA to perform auditing with minimum communication and computation overhead.

III. PROPOSED SCHEMES

This system provides complete solution to the rooting of data along with integrity .in this we are discussing two schemes and their drawbacks. The extension our main scheme to support batch auditing for the TPA upon delegations from multiple users. Finally, we discuss how to generalize our privacy-preserving public auditing scheme and its support of data dynamics.

3.1 Definitions and framework

The proposed system contains four algorithms (KeyGen, SigGen, and Gen Proof Verify Proof). KeyGen is a key generation algorithm that is run by the user to setup the scheme. SigGen is used by the user to generate verification metadata, which may consist of MAC, signatures, or other related information. GenProof is run by the cloud server to generate a proof of data storage correctness Verify Proof is run by the TPA to audit the proof from the cloud server. Public auditing system consists of two phases, Setup and Audit.

Setup: By using the KeyGen user initializes the public and secret parameters, SigGen used to pre-process the data and generates verification metadata .The user then stores the data file F and verify the metadata at the cloud server, and deletes its local copy.

Audit: After the setup phase the cloud user sends a challenge to the cloud server for checking that the server retained the data file correctly. Then the server will derive a response message from a function of the stored data file F and its verified metadata by executing GenProof. The TPA then checks the response via Verify Proof. Our outline assumes the TPA is stateless, which is a attractive property achieved by our proposed solution.

3.2 Notation and Preliminaries

F – The data file to be outsourced, denoted as a sequence of m blocks a1. . . an ∈ Zq for some large prime q.

MAC (·) (·) – message authentication code (MAC) function, defined as: s× {0, 1} ∗ → {0, 1}

Where, s denotes the key space. H (·), h (·) – cryptographic hash functions.

3.3 The Basic Schemes

Before going to our main result, we study two classes of schemes in [3] as a warm-up. The first one is a MAC-based solution which suffers from unwanted organized drawbacks enclosed usage and stateful verification, this may create extra online burden to users, in a public auditing setting. The second one is a system based on homomorphic linear authenticators (HLA), which covers many recent proofs of storage systems. We will identify the reason why all existing HLA-based systems are not privacy-preserving. The analysis of these basic schemes leads to our main result, which overcomes all these drawbacks. Our Main scheme to be presented is based on a specific HLA scheme.

MAC - based Solution. Especially there are two promising ways to make use of MAC to authenticate the data. A trivial way is just uploading the data blocks with their MACs to the server, and sends the corresponding secret key sk to the TPA. Later, the TPA can randomly retrieve blocks with their MACs and check the correctness via sk. Apart from the high (linear in the sampled data size) communication, computation complexities,

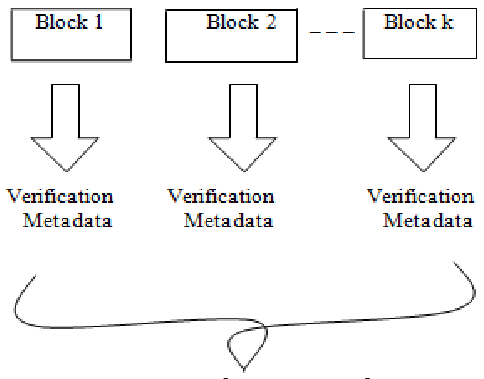
The TPA requires the knowledge of the data blocks for verification. To circumvent the requirement of the data in TPA verification, the idea is as follows. Before data outsourcing, the cloud use chooses r random message authentication code keys {rkτ} 1≤τ ≤r, pre-computes s (deterministic) MACs,{M ACrkτ (F)}1≤τ ≤s for the whole data file F , and publishes these verification metadata (the keys and the MACs) to TPA. The TPA can reveal a secret key rkτ to the cloud server and ask for a fresh keyed MAC for comparison in each audit. This is privacy-preserving as long as it is impossible to recover F in full given MACrkτ (F) and rkτ

Message authentication code algorithm:

```
Function hmac (key, message)
  If (length (key) > block size) then
  Key = hash (key)
  End if
  If (length (key) < block size) then
  Key = key || [0x00 * (block size -     length (key))]
  End if
  o_key_pad = [0x5c * block size] ⨁ key
  i_key_pad = [0x36 * block size] ⨁ key
  Return hash (o_key_pad || hash (i_key_pad ||    message))
  End function
```

On the other hand, it suffers from the following [4] rigorous drawbacks: 1) Once all possible secret keys are used up completely, the user then has to retrieve data in full to recompute and republish new Message Authentication Codes to Third Party Auditor; 2) The Third Party Auditor also has to keep and update state between audits par; 3) This Message Authentication Code based solution supports only for the static data, and cannot support with dynamic data at all.

HLA-based Solution: In order to achieving the Privacy Preserving Public auditing effectively without having to retrieve the contents of data blocks themselves, the HLA technique can be used. HLA, like MACs, are also some Unforgivable verification metadata that authenticate the integrity of a data block. The difference is that HLAs can be aggregated. We can perform aggregated HLA which authenticates a linear combination of the Individual data blocks at a time .the following system represents how HLA-based proof of storage system works First of all The user authenticates each element of a file considering P= (a1, · · · , an) by a set of HLAs Φ. The cloud server stores data file P and set of Authenticators Φ}. The TPA checks the cloud storage by sending a random set of challenge{Ci}, P,Φ and {Ci} are all vectors, so {Ci} is an ordered set or {i, Ci} should be sent).The cloud server then returns μ =∑i Ci ·mi and an aggregated authenticator σ (both are computed fromP , Φ and {vi }) that is supposed to authenticate μ.

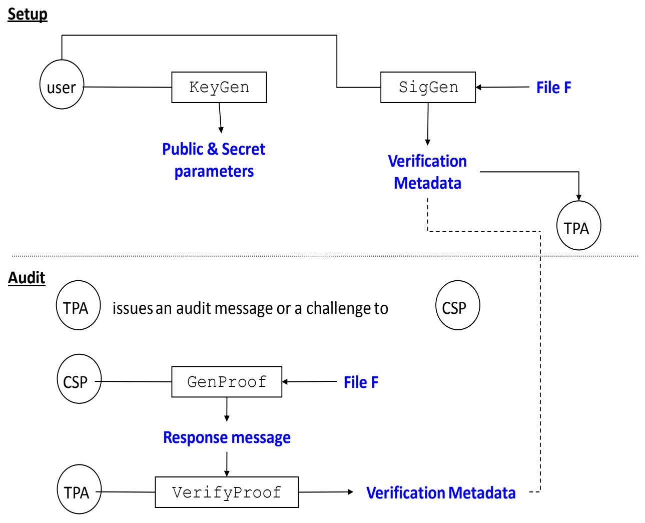


Aggregate verification metadata

Fig 2: Homomorphic linear authenticator (HLA)

Though allowing efficient data auditing and consuming only constant bandwidth, the direct implementation of these HLA-based techniques is still not suitable for our purposes. This is because the linear combination of blocks, μ = ∑ i Ci · mi, may potentially reveal user data information to TPA, and violates the privacy preserving guarantee. Specifically, enough number of the linear combinations of the same blocks are collected, the TPA can simply derive the user's data content by solving a system of linear equations.

3.4 Privacy-Preserving Public Auditing Scheme



*Overview***:** Homomorphic linear authenticator with random masking technique [5] by combining these techniques we can achieve privacy preserving public auditing for secure data storage system. In this technique, the linear combination of sampled blocks in the server's response is masked with randomness generated by the server. With random masking, the Third Party Auditor does not need to maintain all the information in order to build a correct group of linear equations and consequently cannot retrieve the user's data content, no matter how many linear combinations of the same set of file blocks can be collected. On the correctness validation of the block-authenticator pairs can still be carried out in a new way which will be shown shortly, even with the presence of the randomness. Our design makes use of a public key based HLA, to equip the auditing protocol with public audit ability. Specifically, we use the HLA proposed in, which is based on the Short signature scheme proposed by Boneh, Lynn and Shacha.

TPA	Cloud Server
1.Retrieve a file tag f, verify its signature quit if failed;	TPA Sends chal
2.Generate a challenge chalk={(icy)ibis	3.Compute α′=∑ CiMi and μ=πi€I μi
	4.Randomly pick r data perform computations
	5.compute α=r+ γ α′
6.compute γ=h(r) Verify α,μ,R	

Fig 3: The privacy preserving public auditing protocol.

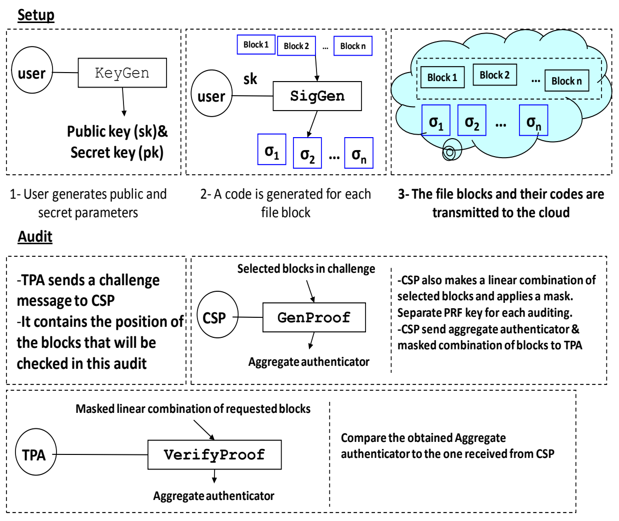

Setup
user
KeyGen
Public key (sk)&
Secret key (pk)
Block 1
Block 2
Block n
user
sk
SigGen
σ1 σ2 … σn
Block 1
Block 2
Block n
σ1 σ2 … σn
1- User generates public and secret parameters
2- A code is generated for each file block
3- The file blocks and their codes are transmitted to the cloud
Audit
-TPA sends a challenge message to CSP
-It contains the position of the blocks that will be checked in this audit
Selected blocks in challenge
CSP
GenProof
Aggregate authenticator
-CSP also makes a linear combination of selected blocks and applies a mask.
Separate PRF key for each auditing.
-CSP send aggregate authenticator & masked combination of blocks to TPA
Masked linear combination of requested blocks
TPA
VerifyProof
Aggregate authenticator
Compare the obtained Aggregate authenticator to the one received from CSP

Fig 4: Privacy preserving public auditing scenario.

3.5 Extension for Batch Auditing

Privacy preserving public auditing is used for performing individual auditing delegations at a time it is not sufficient for the third party auditor for performing individual auditing. We are further extending our scheme to Perform multiple delegations from multiple users on more number of data files performed by the third party auditor by this we can improve the performance of our privacy preserving public auditing scheme. The third party auditor can generate a random set of challenges M for the multiple data files and send that challenge to the cloud server.

TPA	Cloud Server
1.Retrieve a file tag f, verify its signature quit if failed;	TPA Sends chal
2.Generate a challenge Chal={(i,Ci)i€I	
	3.Compute α′,μk Kk in single user

	4.compute R=R1…RK
	L = vk1 ‖vk2 ‖ · · ‖vkK
	and γk = h(R‖vk ‖L);
	5. Compute μk = rk + γkμ′k

6. Compute γk = h(R‖vk ‖L) for each user k and perform Batch auditing via Equation 2.	

Fig 4: The batch auditing protocol.

The third party auditor retrieves a file and tags verifies its signature and quits if fails.

IV. EVALUTION

4.1 Security Analysis

From [6] estimate the security of the proposed scheme by analyzing its requirement of the security guarantee described in Section 2.2, namely, the storage correctness and privacy-preserving property. We start from the Single user case, where our main result is maintained. Then we show the security guarantee of batch auditing for the TPA in multi-user setting

4.2 Performance Analysis

We now mention the performance of the Privacy-preserving public auditing schemes to show that they are certainly lightweight. We will focus on the cost of the efficiency of the privacy-preserving protocol and our proposed batch auditing technique. The experiment is conducted using C on a Linux system with an Intel Core 2 processor running at 1.86 GHz, 2048 MB of RAM, and a 7200 RPM Western Digital 250 GB Serial ATA drive with an 8 MB buffer.

4.2.1 Cost of Privacy-Preserving Protocol

We begin by calculating the cost in terms of basic cryptographic operations. Suppose there are c random blocks specified in the Message chal during the Audit phase. Under this setting, we measure the cost introduced of the privacy preserving auditing in terms of server computation, auditor computation as well as communication over head. On the server side, the generated response includes Aggregated authenticator μ=πi€I μi € G1 and a random factor R.

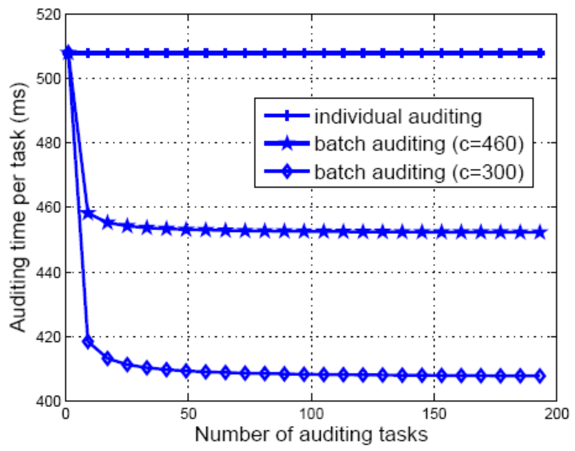


Fig 5: Comparison on auditing time between batch and individual auditing. Per task auditing time denotes the total auditing time divided by the number of tasks.

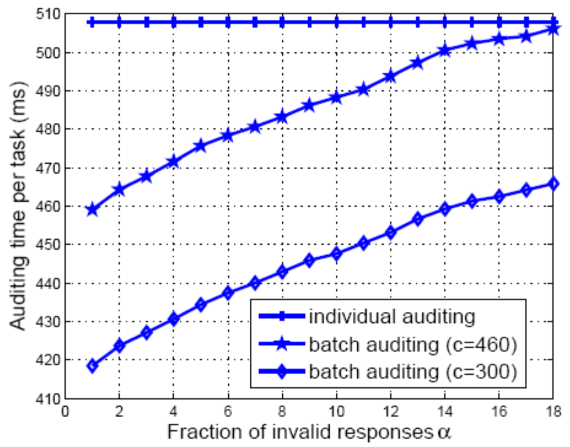


Fig 6: Fraction of invalid responses α

4.2.2 Sorting out Invalid Responses

Now we use testing to justify the effectiveness of our recursive binary search approach for the TPA to sort out the invalid responses when batch auditing fails. This ex+periment is strongly pertained to the work in, which evaluates the batch verification efficiency of various short signatures. To evaluate the feasibility of the recursive approach, we first generate a collection of 256 valid responses, which implies the TPA may concurrently handle 256 different auditing delegations. We then conduct the tests continuously while randomly corrupting an α-fraction, ranging from 0 to 18%, by replacing them with random values. The average auditing time per task against the individual auditing approach is presented in Fig. 5. The result shows that even the number of invalid responses exceeds 15% of the total batch size, the performance of batch auditing can still be safely concluded as more preferable than the straightforward individual auditing. Note that the random distribution of invalid responses within the collection is nearly the worst-case for batch auditing. If invalid responses are grouped together, it is possible to achieve even better results.

4.2.3 Batch Auditing Efficiency

Batch auditing efficiency can be calculated as follows [7], by considering only the total number of pairing operations. However, on the practical side, there are additional less expensive operations required for batching, such as modular exponentiations and multiplications. In the meantime, the different sampling strategy, i.e., different number of sampled blocks c, is also a variable factor that affects the batching efficiency. Thus, whether the benefits of removing pairings significantly outweighs these additional operations is remained to be verified. To get a complete view of batching efficiency, we conduct a timed batch auditing test, where the number of auditing tasks is increased from 1 to approximately 200 with intervals of 8.

The performance of the corresponding non-batched (individual) auditing is provided as a baseline for the measurement. Following the same experimental settings c = 300 and c = 460,the average per task auditing time, which is computed by dividing total auditing time by the number of tasks, is given in Fig. 4 for both batch and individual measurement. Following the same experimental settings c = 300 and c = 460, the average per task auditing time, which is computed by dividing total auditing time by the number of tasks, is given in Fig. 4 for both batch and individual reducing the TPA's computation cost, as more than 11% and 14% of per-task auditing time is saved, when c is set to be 460 and 300, respectively.

Batch auditing efficiency can be improved when compared to the individual auditing. Through batch auditing third party auditor can perform multiple auditing delegations from more users can be performed at a time.

V. CONCLUSION AND FUTURE SCOPE

The main intention of this paper is for providing the privacy preserving public auditing system for data storage security in Cloud Computing. By using the homomorphic linear authenticator with random masking technique, we are providing the security for the cloud users and also assuring that the TPA would not learn any knowledge about the data content, stored on the cloud server during the well-organized auditing process, which not only avoids the burden of cloud user from the monotonous and possibly cost effective auditing task, but also alleviates the users' fear of their outsourced data leakage. Considering TPA may handles multiple audit tasks at a time from different users for their outsourced data files, we further extending our privacy-preserving public auditing protocol into a multi-user setting, where the TPA can perform multiple auditing tasks in a batch manner for better efficiency. Extensive analysis shows that our Schemes are provably secure and highly efficient.

REFERENCES

[1]. P. Mell, T. Grance, "*The NIST Definion of Cloud Computing*", National Institute of Standards and Technology, Vol.53, Issue.6, pp.1-50, 2009.

[2]. M. Armbrust, A. Fox, R. Griffith, AD. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia, " *A view of cloud computing*", Communications of the ACM, Vol.53, Issue.4, pp.50-58, 2010.

[3]. M. Bellare, P. Rogaway, "*Random oracles are practical: A paradigm for designing efficient protocols*", In First Conference on Computer and Communications Security, USA, pp. 62-73, 1993.

[4]. A. Hirne, S. Namdev, H.S. Tomar, "*Secure Data Distribution in Cloud Environment Using Key Aggregation Cryptic*", International Journal of Scientific Research in Computer Science and Engineering, Vol.4, Issue.2, pp.15-19, 2016.

[5]. G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "*Provable data possession at untrusted stores*", Proceedings of the 14th ACM conference on Computer and communications security, Alexandria, pp. 598–609, 2007.

[6]. C Wang, SSM Chow, Q Wang, K Ren, W. Lou, "*Privacy-preserving public auditing for secure cloud storage*", IEEE transactions on computers, Vol.62, Issue.2, pp.362-75, 2013.

[7]. Q. Wang, C. Wang, J. Li, K. Ren, W. Lou, "*Enabling public verifiability and data dynamics for storage security in cloud computing*," in *Proc. of ESORICS'09 (volume 5789 of LNCS* Springer-Verlag), pp. 355–370, 2009.