High Secure and Verification Mechanism for Cloud Storage

Lukka Ramesh Babu
Department of CSE
GVP College of Engineering
Email:lukkarami@gmail.com

Vemu Tulasi
Associate Professor
Department of CSE
GVP College of Engineering
Email: tulasi@gvpce.ac.in

Abstract—**Cloud Computing is one of the best architecture of the existing architecture. It stores the applications and databases in one location and those applications and databases can be accessed by any authorised users .That makes users need not to have infrastructure required for them at user side. User feels free and need not to have more knowledge on infrastructure maintenance. However, the fact that users no longer have physical possession of the possibly large size of outsourced data makes the data integrity protection in Cloud Computing a very challenging and very difficult task, mostly for users who have limited computing resources and capabilities. By enabling public auditability for cloud data storage security is very importance so that users can resort to an external audit party to check the integrity of outsourced data when they want to get that data. third party auditor (TPA) should be more secure, to do that it should met the following two fundamental requirements have to be met: 1) TPA should be able to efficiently audit the cloud data storage without storing the copy of the original data, and it should not have additional burden to the cloud user; 2) he third party auditing process should bring in no new vulnerabilities towards user data privacy. In this paper, to maintain integrity, we introduced the new concept by finding the resultant value of data block by using the proposed algorithm and that will be stored at TPA(Third party auditor) and client. performance analysis shows proposed schema is more secure and highly efficient.**

keywords—**Integrity,verification,homomorphic authentication.**

I. INTRODUCTION

Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network. cloud-shaped symbol is used as its name as an abstraction for the complex infrastructure. In general Cloud computing is responsible for remote services with a user's data, software and computation. At beginning cloud computing is not much secure, but it appears to derive from the practice of using drawings of stylized clouds to denote networks in diagrams of computing. The word cloud is used as a symbol for the Internet, based on the use of a cloud-like shape to denote a network on telephony schematics and later to depict the Internet in computer network diagrams as an abstraction of the underlying infrastructure it represents. Consider the large size of the outsourced electronic data and the client's limited resources,the problem can be generalized as how can the client find an efficient way to perform periodical integrity verifications without having the copy of original data files.To solve the problem of data integrity checking, many schemes are proposed security models [1],[2], [3], [4], [5], [6], [7].all these schemes, great efforts are made to design solutions that meet various requirements such as efficiency, Stateless verification and retrievability of data, etc. all the schemes presented before fall into two categories: private auditability and public auditability. private auditability can achieve through higher efficient scheme, public auditability allows anyone, not just the client , to challenge the cloud server for correctness of data storage without keeping no private information. Then, clients should choose for the evaluation of the service performance to third party auditor (TPA), without devotion of their resources.

clients are unreliable or may not be able to afford the overhead of performing frequent integrity checks in the cloud. For efficiency purpose, the outsourced data themselves need not be required by the verifier for the verification purpose.

II. EXISTING SYSTEM

Data encryption is an existing system for data privacy. data encryption is one way to mitigate this privacy concern, but it is only complementary to the public auditing scheme to be proposed in this paper.

Through general encryption, it is not enough to prevent data from flowing away towards external parties during the auditing. so, it can not fully solve the problem of protecting data secrecy it can only reduce up to certain level. data leakage still remains a problem due to the exposure of encryption keys. Outsourced data can not be handle by the traditional hash function and signature approaches. So it is not a real time solution for data verification by downloading them, because it will be more expensive communications, in the case of large size files. it is important to know that public audit ability for CSS, so that data owners may go for third party auditor. TPA contains expertise and capabilities that a common user cannot handle.Audit service is very important in clouds.

In existing third party auditing schemes mostly uses probalistic proof technique for a storage provider to prove that clients' data remain as orginal data.

Disadvantages with this scheme are: i)Lack of rigorous performance analysis for constructed audit system greatly affects the practical application ii)It is very important to develop a more efficient mechanism for dynamic audit services, in which possible advantage through dynamic data operation can be done. iii)Single TPA for all auditing will take more time.

In the current scenario, we can't assure any security to our data which is stored in different web servers and their databases while two or more servers trying to communicate with each other to exchange the data between them. Because when some other users trying to hack the information from the cloud environment, we are allowing them to retrieve some information which may need to be secured from the application point of view.

A. *Problem Definition*

We consider a cloud data storage service involving three different entities : the cloud user (U), who has large amount of data files to be stored in the cloud; the cloud server (CS), which is managed by cloud service provider (CSP) to provide data storage service and has significant storage space and computation resources (we will not 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 assess the cloud storage service security on behalf of the user upon request.

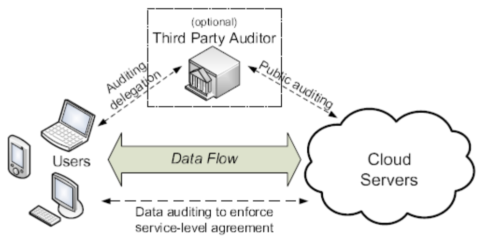

Fig. 1: Cloud storage service architecture

Users trusts on the CS for cloud data storage. They may also interact with the CS to access their stored data in the cloud for various application purpose whenever they want. We cannot trust the TPA auditing because it may leak the data ,so it should be able to efficiently audit the cloud data storage without storing the copy of the original data and it should not be extra on-line burden to cloud users. Any possible leakage of user's outsourced data in TPA through the auditing protocol should be avoided.

Public-key based homomorphic authenticator which enables TPA to perform the auditing without demanding the copy of the original data so that it will reduces the communication and computation overhead as compared to the direct data auditing approaches. By using the random mask authenticator with the homomorphic authenticator technique TPA can not know any thing about the data stored in the cloud server. For batch auditing our aggregation properties will help.

To securely introduce an effective third party auditor (TPA), the following two fundamental requirements have to be met:

1) TPA should be able to efficiently audit the cloud data storage without copy of the original data, and

introduce no extra on-line burden to the cloud user;

2) The third party auditing process should contain extra vulnerabilities towards user data privacy.

III. THE PROPOSED SCHEMES

In Quang's paper, they concentrated on only verification . But here we are providing algorithms for both verification and security also.

Following we provided both algorithms for storing the data and retrieving the data.

In our formal adversarial model, we let a system HAIL consist of the following functions:

keygen(1^λ)→κ: Generates a keyκ= (sk,pk) of size security parameterλ. (For symmetric-key systems,pk may be null.)

encode$(\kappa, F, \ell, n, b) \rightarrow \{F_0^{(i)}\}_{i=1}^n$: Encodes F as a set of file segments, where $F_0^{(i)}$ is the segment designated for server i. The encoding is designed to provide`-out-of-n redundancy across servers and to provide resilience against an adversary that can corrupt at most b servers in any time step.

Decode (κ,t,$\{F\text{'}_t{}^{(i)}\}^n{}_{i=1}$)→ F: Recovers the original file Fat time t from a set of file segments stored at different servers.

Challenge $(\kappa) \rightarrow \{C_i\}_{i=1}^n$: Generates a challenge value C_i for each server i.

verify (κ,j,$\{C_i,R_i\}^n{}_{i=1}$) {→0,1}. Checks the response of server j, using the responses of all servers $R_1,...,R_n$ to challenge set$C_1,...,C_n$. It outputs a '1' bit if verification succeeds, and '0' otherwise. We assume for simplicity that verify is sound, i.e., returns 1 for any correct response.

Redistribute $(\kappa, t, \{\hat{F}_t^{(i)}\}_{i=1}^n) \rightarrow \{F_{t+1}^{(i)}\}_{i=1}^n \cup \perp$: Is an interactive protocol that replaces the fragment $\hat{F}_t^{(i)}$ stored at server I with $F_{t+1}^{(i)}$. It implements a recreation and distribution of corrupted file segments, and outputs ⊥ if the file cannot be reconstructed.

A. *Notations and preliminaries:*

F – the data file to be outsourced, denoted as a sequence of n blocks $m_1, \ldots, m_n \varepsilon Z_p$ for some large prime p.

$f_{key}(\cdot)$ – pseudorandom function (PRF), defined as: $\{0, 1\}^{*} \times key \rightarrow Z_p$.

$\Pi_{key}(\cdot)$ – pseudorandom permutation (PRP), defined as:

$\{0, 1\}^{\log_2(n)} \times key \rightarrow \{0, 1\}^{\log_2(n)}$

$MAC_{key}(\cdot)$ – message authentication code (MAC) function defined as: $\{0, 1\}^* \times key \rightarrow \{0,1\}^1$.

Cval– user calculate the hash value and store with him until acknowledge comes from TPA.

Rval- It is the resultant value of all the files stored in CSS.

F_{md} – TPA calculated message digest value.

We now introduce some necessary cryptographic background for our proposed scheme.

Bilinear Map Let G_1, G_2 and G_T be multiplicative cyclic groups of prime order p. Let g_1 and g_2 be generators of G_1 and G_2, respectively. A bilinear map is a map

$e : G_1 \times G_2 \rightarrow GT$ with the following properties(D.Boneh,C.Gentry,jan2003) :

1) Computable: there exists an efficiently computable algorithm for computing e;

2) Bilinear: for all $u \varepsilon G_1$, $v \varepsilon G_2$ and $a, b \varepsilon Z_p$, $e(u^a, v^b) = e(u, v)^{ab}$;

3) Non-degenerate: $e(g_1, g) \neq 1$;

4) for any $u_1, u_2 \varepsilon G_1$, $v \varepsilon G_2$, $e(u_1u_2, v) = e(u_1, v) \cdot e(u_2, v)$.

Algorithm: Storing the Data

From client side generate a key for each file using key Generator G_k by taking file size as input value and then eccrypt the file

Now randomise the encrypted data after that using sign generator S_G create the signature.

.Now client Cval after that encrypt the data with the shared secret key S_{ut} between TPA and user.

Now TPA will calculate msgdigest and store that original data in cs and keeps that msgdigest itself and send Rval and F_{md} to the client.

Now client verifies the msgdigest with calculated Cval value.

If it matching with calculated value then it update the result Rval.

Else

It will resend the data to TPA.

Algorithm 2:Retrieving The Data

When ever user request the data again TPA will send to the user only requested data by using the shared secret key S_k.

if it is delete or update operation in that case TPA will update its Rtval and then send it to the user by encrypting the shared secret key S_k.

After receiving user decrypt the cipher text using shared secret key and then verify it and update R_{val}.

B. *Support for public auditability*

public auditing scheme consists of four algorithms (KeyGen, SigGen, GenProof, VerifyProof). 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, it contains MAC, signatures, or other relavent information that is used for auditing. GenProof algoithm will uses the cloud server to generate a proof of data storage correctness, while VerifyProof algorithm will run by the TPA to audit the proof from the cloud server.

Our public auditing system can be constructed from the proposed auditing scheme in two phases they are Setup and Audit:

Setup: The user initializes the public and secret parameters of the system by executing G_k, and encrypt the original data file F by using S_G is used to generate the verification metadata. The user then stores the data file F at the cloud server, deletes its local copy, and publishes the verification metadata to TPA for later audit. As part of pre-processing, the user may alter the data file F by expanding it or including additional Meta data to be stored at server.

*Audit:*The TPA issues an audit message or challenge to the cloud server to make sure that the cloud server has retained the data file F properly at the time of the audit. The cloud server will derive a response message from a function of the stored data file F by executing GenProof. Using the verification metadata, the TPA verifies the response via VerifyProof.

We start from the overview of our public auditing system and discuss two straightforward schemes and their demerits. Then we present our main result for privacy-preserving public auditing to achieve the aforementioned design goals. We also show how to extent our main scheme to support batch auditing for TPA upon delegations from multi-users. Finally, we discuss how to adapt our main result to support data dynamics.

Scheme1

This scheme doesn't provide privacy preserving but it is light weight. The cloud user pre-computes MACs $\sigma_i = MAC_{sk}(i||m_i)$ of each block mi (i $\varepsilon\{1, . . . , n\}$), sends both the data file F and the MACs $\{\sigma_i\}_{1\leq i\leq n}$ onto the cloud server, and releases the secret key sk to TPA. During the Audit phase, the TPA requests from the cloud server a number of randomly selected blocks and their corresponding MACs to verify the correctness of the data file. The insight behind this approach is that auditing most of the file is much easier than the whole of it. However, this simple solution suffers from the following severe drawbacks: 1) The audit from TPA demands retrieval of users' data, which should be prohibitive because it violates the privacy-preserving guarantee; 2) Its communication and computation complexity are both linear with respect to the sampled data size, which may result in large communication overhead and time delay, especially when the bandwidth available between the TPA and the cloud server is limited.

Scheme2

This scheme provides privacy preserving but its usage is limited (bounded).To avoid retrieving data from the cloud server, one may improve the above solution as follows: Before storing the data in cloud storage, the cloud user should chooses s random message authentication code keys $\{sk_r\}_{1 \leq r \leq s}$, pre-computes s MACs, $\{MAC_{skt}\ (F)\}_{1 \leq r \leq s}$ for the entire original data file F,and publishes these verification metadata to TPA. The TPA can each time sends a secret key skt to the cloud server and request for a fresh keyed MAC for comparison, thus achieving efficent auditing. However, in this method: 1) the number of times a particular data file can be audited is limited by the number of secret keys that must be a fixed priori. Once all possible secret keys are not effective, cloud user then has to retrieve data from the server in order to re-compute and re-publish new MACs value to the TPA. 2) The TPA should contain and update state of audits, i.e., keep a track on the calculated MAC keys. Considering the very large number of audits from multiple users, maintaining such states for TPA can be more difficult.

Note that another common drawback of the above basic schemes is that they can only support the case of static data, they cannot deal with data dynamic operations. For the reason of brevity and clarity, our main result will focus on the static and dynamic data, too.

To effectively support public auditability without having to retrieve the data blocks themselves, we adopt to the homomorphic authenticator technique (G.Ateniesemarch2007,Q.Wangsep.2009,H.Schacham Dec 2008) .Homomorphic authenticators are unforgeable verification metadata generated from each individual original data blocks, which can be securely aggregated in such a way to assure an auditor that a linear combination of data blocks is correctly computed by verifying only the aggregated authenticator. However, the direct adoption of these techniques is not suitable for our purposes, since the linear combination of blocks may potentially reveal user data information, thus violating the security. If sufficient 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 without calculating for each block separately.

To achieve efficient public auditing, we propose unique integrate the homomorphic authenticator with random mask technique. In our proposed schema, the linear combination of sampled blocks in the server's response is masked with randomness generated by a pseudo random function. With random mask, the TPA no longer has all the required information to build up a correct group of linear equations and therefore cannot derive the user's data,it is not a matter how many linear combinations of the same set of file blocks can be collected. Meanwhile, due to the algebraic property of the homomorphic authenticator, the correctness validation of the block-authenticator pairs will not be affected by the randomness generated from a PRF, which will be shown shortly. Note that in our proposed scheme, we used the public key based homomorphic authenticator; specifically.Its exibility in signature aggregation will further benefit us for the multi-task auditing.

Scheme Details: Let $G1$, $G2$ and GT be multiplicative cyclic groups of prime order p, and $e : G_1 \times G_2 \rightarrow G_T$ be a bilinear map as introduced in preliminaries. Let g be the generator of G_2. H(.) is a secure map-to-point hash function:$\{0, 1\}^* \rightarrow G_1$, which maps strings uniformly to G_1. Another hash function $h(\cdot) : G_1 \rightarrow Z_p$ maps group element of G_1 uniformly to Z_p. The proposed scheme is as follows:

Setup Phase:

1)The cloud user runs KeyGen to generate the system's public and secret parameters. He chooses a random $x \leftarrow Z_p$, a random element $u \leftarrow G_1$, and computes $v \leftarrow g^x$ and $w \leftarrow u^x$. The secret parameter is sk = (x) and the public parameters are pk = (v,w, g, u). Given data file $F = (m_1, \ldots, m_n)$, the user runs SigGen to compute signature σ_i for each block mi:$\sigma i(H(i) \cdot u^{mi})^x\ \varepsilon_{,}\ G_1\ (i = 1, \ldots, n)$. Denote the set of signatures by $\phi = \{\sigma_i\}_{1 \leq i \leq n}$. The user then sends $\{F, \phi\}$ to the server and deletes them from its local storage.

Audit Phase:

2) During the auditing process, to generate the audit message *"chal"*, the TPA picks a random *c*-element subset I = {s1, . . . , sc} of set [1, *n*],

where sq = Πkprp (q) for 1≤q≤*c* and kprp is the randomly chosen permutation key by TPA for each auditing. We assume that s1 ≤ · · · ≤ sc. For each element i ε I, the TPA also chooses a random value i (of a relative small bit length compared to |p|). The message "chal" specifies the positions of the blocks that are required to be checked in this Audit phase. The TPA sends the chal = to the server.

3) Upon receiving challenge chal $=\{(i,v_i)\}_{i\varepsilon I}$, the server runs GenProof to generate a response proof of data storage correctness. Specifically, the server chooses a random element $r \leftarrow Z_p$ via $r = f_{kprf}$ (chal), where kprf is the randomly chosen PRF key by server for each auditing, and calculates $R= (w)^r=(u^x)^r \varepsilon G_1$. Let denote the linear combination of sampled blocks specified in chal: $= \Sigma_{i\varepsilon I} v_i m_i$. To blind μ′ with *r*, the server computes: $\mu=\mu'+rh(R)\varepsilon Z_p$ .Meanwhile, the server also calculates an aggregated signature $\sigma=\Pi_{i\varepsilon I}\sigma_i^{vi}\varepsilon G_1$.It then sends {μ,σ,R} as the response proof of storage correctness to the TPA runs VerifyProof to validate the response by checking the verification equation:

$$e(\sigma.(R^{h(R)}),g)=e(\Pi_{i=s1}^{sc} H(i)^{vi}. u^{\mu},v)\text{…………..}(1)$$

the correctness of the above verification equation can be elaborated as follows:

$$e(\sigma. R^{h(R)} ,g)=e(\Pi_{i=s1}^{sc}\sigma_i^{vi}.(u^x)^{r.h(r)},g)$$

$$=e(\Pi_{i=s1}^{sc} (H(i).u^{mi})^{x.vi}.(u^{r.h(R)})^x,g)$$

$$=e(\Pi_{i=s1}^{sc} (H(i)^{vi}.u^{mivi}).(u^{r.h(R)}),g)$$

$$=e(\Pi_{i=s1}^{sc} (H(i)^{vi}).u^{\mu'+r.h(R)},g^x)$$

$$=e(\Pi_{i=s1}^{sc}(H(i)^{vi}). u^{\mu},v)$$

It is clear that the random mask R has no effect on the validity of the checking result.

Support for batch auditing

With the establishment of public auditing in Cloud Computing, TPA may concurrently handle multiple auditing delegations upon different users' requests. The individual auditing of these tasks for TPA can be tedious and very inefficient. Given K auditing delegations on K distinct data files from K different users, it is more advantageous for TPA to batch these multiple tasks together and audit at one time. Keeping this natural demand in mind, we propose to explore the technique o f bilinear aggregate signature(D.Boneh and C.Gentry april 2003) ,which supports the aggregation of multiple signatures by distinct signers on distinct messages into a single signature and thus provides efficient verification for the authenticity of all messages. Using this signature aggregation technique and bilinear property, we can now aggregate K verification equations (for K auditing tasks) into a single one, as shown in equation (2), so that the simultaneous auditing of multiple tasks can be achieved.

By integrating the homomorphic authenticator with random mask technique, we can guarantee that TPA could not learn any knowledge about the data content stored in the cloud server during the efficient auditing process. The aggregation and algebraic properties of the authenticator further benefit our design for the batch auditing.

Batch auditing is the capability that multiple delegated auditing tasks from different users can be performed simultaneously by the TPA.

Support for Data Dynamics

In Cloud Computing, outsourced data might not only be accessed but also updated frequently by users for various application purposes. Hence, support+ing data dynamics for privacy-preserving public risk auditing is also of paramount importance. Now we show how our main scheme can be adapted to build upon the existing work to support data dynamics, including block level operations of modification, deletion and insertion. Data dynamics support is achieved by replacing the index information i with the m_i in the computation of block signatures and using the classic data structure-Merkle hash tree (MHT) for the underlying block sequence enforcement.

As a result, the signature for each block is changed to $\sigma_i=(H(m_i).u^{mi})^x$. We can adopt this technique in our design to achieve privacy-preserving public risk auditing with support of data dynamics. Specifically, in the Setup phase, the user has to generate and send the tree root T R_{MHT} to TPA as additional metadata, where the leaf nodes of MHT are values of $H(m_i)$. In the Audit phase, besides {μ, σ, R} , the server's response should also include $\{H(m_i)\}_{i\varepsilon I}$ and their corresponding auxiliary authentication information (AAI) in the MHT. Upon receiving the response, TPA should first use T R_{MHT} and the AAI to authenticate $\{H(m_i)\}_{i\varepsilon I}$ computed by the server. Once $\{H(m_i)\}_{i\varepsilon I}$ are authenticated, TPA can then perform the auditing on $\{\mu,\sigma,R,\{H(m_i)\}_{i\varepsilon I}\}$ via equation (1) where $\Pi_{s1\leq i\leq sc}H(i)^{vi}$ is now replaced by $\Pi_{s1\leq i\leq sc}H(m_i)^{vi}$. Note that the data privacy is still preserved due to the random mask R.

IV. PERFORMANCE ANALYSIS

In this section, we analyze our proposed data storage scheme in terms of security and efficiency. Further discussion on other attacks are also discussed. Basedon the analysis, some enhanced techniques are presented.

A. Security and Dependability of Initial Data Storage

The security proof in [21] and [22] ensures share generation scheme for initial data storage is unconditionally secure and (m − 1)-collusion resistant. That is, up to (m − 1) colluding nodes reveal no information on the encoded data. In practice, requirements may vary depending on different applications, thus the threshold m can be adjusted to accommodate the special needs

B. Security of Dynamic Integrity Assurance

1) *Modifying Data Shares*: After an adversary has compromised set of nodes, it would like to modify the data shares in order to prevent authorized users from recovering the original data blocks correctly. We first analyze the case that the adversary modifies data shares accidentally and study the probability of a false negative result in the process of data integrity checks.

C. Efficiency

We now assess the performance of the proposed distributeddata storage scheme. The notation of cryptographic operationsis summarized in table III.

1) *Initial Data Storage*: In terms of computational overhead,before the share generation process the data source nodev has to compute one keyed hash value h(data, kr) and perform two symmetric-key encryptions {data, h(data, kr)}kr and {kr}KUV . The data share generation requires two sets of polynomial evaluations. v first encodes {data, h(data, kr)}kr into n fragments by employing a (m, n) RS code, wherem denotes the number of partitioned data blocks and ndenotes the number of selected neighbours (i.e., share holders).Assume each partitioned data block Di contains c symbols,there are totally n · c polynomial evaluations in this step.Then v employs a (m, n) SS scheme to obtain n shares of{kr}KUV . The construction for the counteracting vector canbe considered as n signature generations with vector size ofn. Finally, a parity based on all shares will be generatedfor each share holder. Let l denote the size of data||kr, the total computation cost at the data source node is hence Hash1l+ SymEncr2 + PolyEvaln·(c+1)m + AlgSigGennn+ParGennk. Correspondingly, the cost at each share holder is only SymDecr1.

NOTATIONS OF CRYPTOGRAPHY

$Hash^t_l$	t has operations with input size of *l*.
$SymEncr^t$	t symmetric-key ecncryption oprations
$PloyEval^t_m$	t polynomial evaluations with polynomial of degree m.
$ParGen^t_k$	t parity generations with vector of size k.
$AlgsigGen^t_k$	t algebraic signature generation with vector of size k.

2*) Dynamic Data Integrity Check*: Consider a share holderw who initiates a data integrity check to verify the integrity ofdata. It will broadcast a challenge message {w, seqno,α, r}to all the share holders. In addition, a 1-symbol algebraicsignature based on its own data share is generated and includedin the challenge. Hence, the communication overhead involvedin this broadcast message is 5 · q bits. Upon receiving the challenge, each share holder needs to compute a 1-symbolalgebraic signature and return it to the check initiator. Thus,for each share holder (including the initiator) the computationalcost is just AlgSigGen1k. The communication overheadinvolved in every response message sigα(Si) is q bits. Afterobtaining all sigα(Si)s (i = 1, . . . , n), each share holder can act as a verifier to check the integrity of data. It isclear that in this step the computational cost at each nodeis ParGen1k + AlgSigGen1k.

V. RELATED WORK

Juels[7] described a formal “proof of retrievability”(POR) model for ensuring the remote data integrity.Their scheme combines cheking and error correcting code to ensure both possession and retrievability of files on archive service systems. Shacham et al. [12] built on this model and constructed a random linear function based homomorphic authenticator which enables unlimited number of challenges and requires less communication overhead due to its usage of relatively small size of BLS signature. Bowers et al. [14] proposed an improved framework for POR protocols that generalizes both Juels and Shacham’s work. Laterin their subsequent work.scheme rests primarily on the pre-processing steps that the user conducts before outsourcing their data file F. Any modification to the contents of original data file F, for few bits, must propagate through the error correcting code and the corresponding random process, thus introducing significant computation and communication complexity. Recently, Dodis et al. [19] gave theoretical studies on generalized framework for different variants of existing POR work.

VI. CONCLUSION

Cloud computing can bring revolution the way we use the Internet, it will become most useful resource for the cloud storage users. But we should more be cautious about. Their outsourced data. There are many new emerging technologies are coming rapid rate, each with new technological advancements to make the human life easier. However, those people who are using cloud services must be very careful to understand the security risks and challenges posed in these technologies. Since the cloud computing is most fascinating technology risks also be more. Main aim of this paper is security problem that will cause due to key mechanism and challenges that are currently faced in the Cloud computing are described. Cloud computing has the potential to become secure and economically viable IT solution in the future. We utilize the homomorphic authenticator and random mask technique to guarantee that TPA can not know any thing about the original data content stored in the cloud storage server during the efficient auditing process, it not only eliminates the burden of cloud user and also less sever the users’ fear of their outsourced data leakage.

TPA can handle multiple audit sessions from different users for their outsourced data and it should extend our secure public auditing protocol into a multi-user setting, where TPA can perform the multiple auditing tasks in a batch manner, i.e., simultaneous. security and performance analysis shows that the proposed schemes are provably secure and highly efficient. We believe all these advantages of the proposed schemes will be small effect on economies of scale for Cloud Computing.

REFERENCES

[1] C. Wang, Q. Wang, K. Ren, and W. Lou, “Privacy-preserving public auditing for storage security in cloud computing,” in Proc. of IEEE INFOCOM’10, March 2010.

[2] M. A. Shah, R. Swaminathan, and M. Baker, “Privacy-preserving auditand extraction of digital contents,” Cryptology ePrint Archive, Report2008/186, 2008.

[3] A. Juels and J. Burton S. Kaliski, “PORs: Proofs of retrievability for large files,” in Proc. of CCS’07, October 2007, pp. 584–597.

[4] M. A. Shah, R. Swaminathan, and M. Baker, “Privacy-preservingaudit and extraction of digital contents,” Cryptology ePrintArchive, Report 2008/186, 2008, http://eprint.iacr.org/.

[5]R. Curtmola, O. Khan, and R. Burns, “Robust remote data checking,” in Proc. of the 4th ACM international workshop on Storage security and survivability (StorageSS’08), 2008, pp. 63–68.

[6] R. C. Merkle, “Protocols for public key cryptosystems,” in Proc. OfIEEE Symposium on Security and Privacy, Los Alamitos, CA, USA,1980.

[7]G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, “Scalable and efficient provable data possession,” in Proc. of SecureComm’08, 2008, pp. 1–10.

[8]C. Wang, Q. Wang, K. Ren, and W. Lou, “Towards secure and dependablestorage services in cloud computing,” IEEE Transactions on Service Computing, 2011, to appear.

[9]G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson,and D. Song, “Provable data possession at untrusted stores,”in Proc. of CCS’07, Alexandria, VA, October 2007, pp. 598–609.

[10] D. L. G. Filho and P. S. L. M. Barreto, "Demonstrating datapossession and uncheatable data transfer," Cryptology ePrintArchive, Report 2006/150, 2006, http://eprint.iacr.org/.

[11]Amazon.com, "Amazon web services (aws)," Online at http://aws.amazon.com/, 2009.

[12]S. Goldwasser and S. Micali, "Probabilistic encryption," Journal of Computer and System Sciences, vol.28, no. 2, pp. 270–299, 1984.

[13]1,A Platform Computing Whitepaper. "Enterprise Cloud Computing: Transforming IT."*Platform Computing*, pp6, 2010.

[14]Cloud Security Alliance(CSA) Available:http://www.cloudsecurityalliance.org [Mar.19,2010]

[15]C. Weinhardt, A. Anandasivam, B. Blau, and J. Stosser. "Business Models in the ServiceWorld." *IT Professional*, vol. 11, pp. 28-33, 2009.

[16]C. Soghoian. "Caught in the Cloud: Privacy, Encryption, and Government Back Doors inthe Web 2.0 Era" The BerkmanCenter for Internet & Society Research Publication Series.

[17] A. Shamir, "How to share a secret," Communications of the ACM,vol. 22, no. 11, pp. 612–613, Nov. 1979.

[18] S. Reed and G. Solomon, "Polynomial codes over certain finite fields,"SIAM Journal of Applied Mathematics, vol. 8, pp. 300–304, 1960.

[19] P. Devanbu, M. Gertz, C. Martel, and S. Stubblebine, "Authentic third-party data publication," in IFIP DB Sec'03, also in Journal of Computer Security, Vol. 11, No. 3, pages 291-314, 2003, 2003.

[20] E. Mykletun, M. Narasimha, and G. Tsudik, "Authentication and integrity in outsourced databases," in ISOC NDSS'04, 2004.

[21] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in ACM CCS'07, Full paper available on e-print (2007/202), 2007.

[22] M. Bellare, R. Canetti, and H. Krawczyk, "Keying hash functions for message authentication," in CRYPTO'96, 1996.