iJRASET
International Journal For Research in Applied Science and Engineering Technology

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

**Volume:** 6 **Issue:** IV **Month of publication:** April 2018

**DOI:** http://doi.org/10.22214/ijraset.2018.4376

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887*
*Volume 6 Issue IV, April 2018- Available at www.ijraset.com*

# Secure Cloud Storage using Homomorphic Encryption

Sravan Kumar Talluru[1], Dr. P. Venkateswara Rao[2], Venkateswarlu[3], Sendhil[4], K Karthik[5]
[1, 2, 3, 4, 5]*CSE, Narayana Engineering College*

***Abstract:*** ***Visiting cloud has been the dream of every net surfer. Cloud Computing offers a number of benefits and administrations to its clients who pay the utilize of equipment and program assets (servers facilitated in data centres, applications, computer program...) on request which they can get to through web without the need of costly computers or a huge capacity framework capacity and without paying any equipment maintenance expenses. But these cloud suppliers must give ensures on the security of protection and sensitive information put away in their information centres divided between numerous clients utilizing the concept of virtualization.***
***Keywords: Privacy, Homomorphic Encryption, Security, Cloud Computing, Virtualization.***

## I. INTRODUCTION

**C**LOUD Computing has been the future of information and computation for establishments, due to its great list of unparalleled boons inside the IT history: on-call for self-service, great network get admission to, unbiased aid pooling, speed resource elasticity, usage primarily based pricing and change of access [1]. As a disaster generation with known implications, Cloud Computing is changing the very nature of how organizations use and store data. One key element of this paradigm changing is that data is being centralized or outsourced to the Cloud. From user's view, together with both people and IT businesses, storing data remotely of the cloud to an adaptable on-demand way acquires some points of interest: is the data safe?, is the cloud admin honest etc.Majority of the data outsourcing is really giving up user's amazing control again the fate for their data. To begin with of all, in spite of the fact that the infrastructures under the cloud are much more effective and reliable than individual computing gadgets, they are still facing the wide extend of both inside and external threats for information keenness. Cases of blackouts and security breaches of essential cloud administrations appear from time to time. Furthermore, there do exists various inspirations for (Cloud Service Providers) CSP to act deceiving towards the cloud users with respect to the status of their outsourced information. For illustrations, CSP may recover capacity to cash related reasons eventuallyuser's perusing disposing of data that need not been or may be rarely gotten to, or In fact stow away information reduction stages thus concerning illustration to preserve a reputation. In specific, basically downloading every last one of data to its astuteness affirmation is not an useful result because of those up and down I/O transmission costs. Other than, it is regularly insufflate will identify those data degradation in a manner of speaking the point when gaining entrance to the information, Similarly as it doesn't indicate client's accuracy certification to the individuals un-accessed majority of the data Furthermore may be excessively awful late to recuperate the information passing alternately harm. thinking about the huge length of the outsourced records and the person's restrained useful resource functionality, the responsibilities of auditing the records correctness in a cloud environment can be ambitious and high-priced for the cloud customers [2], [12].

Public auditability allows an external party, in addition to the user himself, to verify the correctness of remotely stored data. However, most of these schemes [8], [10], [13] do not consider the privacy protection of users' data against external auditors. Indeed, they may potentially reveal user data information to the auditors, as will be discussed in Section 3.4. This severe drawback greatly affects the security of these protocols in Cloud Computing. From the perspective of protecting data privacy, the users, who own the data and rely on TPA just for the storage security of their data, do not want this auditing process introducing new vulnerabilities of unauthorized information leakage towards their data security [14]. Moreover, there are legal regulations, such as the US Health Insurance Portability and Accountability Act (HIPAA) [15], want the uploaded data to be hacked by external parties [9]. We want to develop a auditing system that regularly verifies the integrity and existence of the files. Becausea person doing this auditing duties can be tedious and bulky, a good call for thisis a way to enable the auditor to effectively carry out a couple of auditing duties in a batch manner, i.e., parallel. To deal with those troubles, our scheme deals with a public key primarily based on homomorphic linear authenticator (HLA) [8], [10], [13], which enables TPA to perform the auditing without annoying the neighbourhood replica today's information and hence significantly reduces the communiqué and computation overhead compared to the trustworthy records auditing strategies. With the aid of integrating the HLA with random overlaying, our protocol ensures that

the TPA could no longer examine any know-how approximately the records content material stored in the cloud server all through the efficient auditing technique. In particular, our contribution may be summarized as the following 3 components:

*1)* Our scheme accomplishes parallel processing where multiple delegated inspecting errands from diverse clients can be performed at the same time by the TPA.

*2)* Our scheme accomplishes parallel processing where multiple delegated inspecting errands from diverse clients can be performed at the same time by the TPA.

*3)* We prove the security and justify the performance of ourproposed schemes through concrete experiments and comparisons with state-of-the-art.

Our rest of the paper is organized as follows. Section II introduces the system and threat model, and our design goals. Then we provide the detailed description of our scheme in Section III. Section IV provides an algorithm. Section V gives the properties of our methodology. Section VI gives the analysis and evaluation, followed by Section VII which provides the related work. Finally, Section VIII concludes the whole paper.

## II. PROBLEM STATEMENT

### A. *Model of the System*

Our cloud data storage service involving three different entities, as in Fig. 1: the cloud user (U), who has data to be stored in the cloud; the cloud server (CS), which is tangled by the cloud service provider (CSP) to provide data storage services and has significant storage space and computation resources (we won't tell apart CS and CSP hereafter); the third party auditor (TPA), who has expertise and capabilities to verify contents on the cloud 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. We consider the existence of a semi-trusted CS as [16] does. Namely, in most of time it behaves properly and does not deviate from the prescribed protocol execution. However, for their own benefits the CS might neglect to keep or deliberately delete rarely accessed data files which belong to ordinary cloud users. Moreover, the CS may decide to hide the data corruptions caused by server hacks or Byzantine failures to maintain reputation. However, it harms the user if the TPA could learn the outsourced data after the audit. To authorize the CS to respond to the audit delegated to TPA's, the user can sign a certificate granting audit rights to the TPA's public key, and all audits from the TPA are authenticated against such a certificate. These authentication handshakes are omitted in the following presentation.

### B. *Design Goals*

To allow public auditing with privacy guarantee for cloud data,our protocol design have to obtain the subsequent safety and overall performance ensures.

1) *Privacy Keeping:* to make sure that the TPA can't derive users' statistics content material from the statistics collected for the duration of the auditing process.
2) *Integrity:* to ensure that there exists no dishonest cloud server this can bypass the TPA's audit without indeed storing customers' facts intact.
3) *Public audit:* to permit TPA to affirm the correctness of the cloud records on call for without retrieving a duplicate of the complete facts or introducing additional on-line burden to the cloud customers.
4) *Light weight:* to permit TPA to perform auditing with minimal communiqué and computation overhead.
5) *Multi User:* to enable TPA with relaxed and efficient auditing capability to deal with a couple of auditing delegations from probably large variety of different customers concurrently.

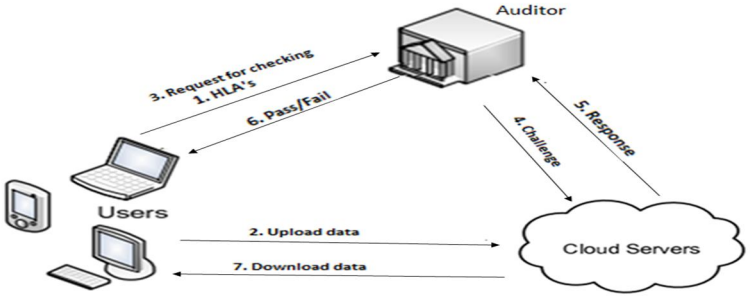



Fig. 1: The architecture of auditor

## III.THE PROPOSED METHOD

This segment presents our open examining scheme which gives a total outsourcing arrangement of data – not as it were the information itself, but too its keenness checking. We begin from a diagram of our open auditing system and talk about two clear plans and their demerits. At that point we display our fundamental scheme and appear how to degree our primary plots to support batch reviewing for the TPA upon assignments from multiple clients. At last, we talk about how to generalize our privacy-preserving open inspecting plot and its support of information elements.

### *A. Definitions and Framework*

The following are the different modules that make up the entire framework.

*1)* *KeyGen:* is a key generation algorithm run by the user to setup the scheme.
*2)* *Sig Gen:* is used by the user to generate verification metadata, which may consist of MAC, signatures, or other related information that will be used for auditing.
*3)* *Gen Proof*: is run by the CS (Cloud server) to generate a proof of data storage correctness
*4)* *Verify Proof:* is run by the TPA to audit the proof from the cloud server.

### *B. Notation and Preliminaries*

The notations and operands used in describing the modules and architectures are clearly described in the below sections

*1)* F – the data to be uploaded, denoted as a sequence of n blocks $m_1, \ldots, m_n \in Z_p$ for some large prime p.
*2)* MAC(·)(·) – message authentication code (MAC) function, defined as: $K \times \{0, 1\} * \rightarrow \{0, 1\} l$ where K denotes the key space.
*3)* H(·), h(·) – cryptographic hash functions. We now put some necessary cryptographic background for our proposed method

We take after a comparative meaning of beforehand proposed conspires with regards to remote information our proposed scheme.

C. *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: $G_1 \times G_2 \rightarrow G_T$ such that for all $u \in G_1$, $v \in G_2$ and a, $b \in Z_p$, $e(u^a, v^b) = e(u, v)^{ab}$. This bilinearity implies that for any $u_1, u_2 \in G_1$, $v \in G_2$, $e(u_1 \cdot u_2, v) = e(u_1, v) \cdot e(u_2, v)$. Of course, there exists an efficiently computable algorithm for computing e and the map should be non-trivial, i.e., e is non-determinant: $e(g_1, g_2) \neq 1$.

### *D. The Basic Methods*

Before giving our fundamental outcome, we consider two classes of conspires as a warm-up. The first is a MAC based solution which experiences unfortunate efficient negative marks – limited utilization and stateful check, which may represent extra online weight to clients, in an open examining setting. This by one means or another likewise appears that the evaluating issue is as yet difficult to tackle indeed; even we have presented a TPA. The second one is a framework in light of homomorphic direct authenticators (HLA), which covers numerous current evidence of capacity frameworks. We will pinpoint the motivation behind why all current HLA-based frameworks are not security safeguarding. The investigation of these essential plans prompts our principle result, which defeats every one of these disadvantages. Our primary plan to be exhibited depends on a particular HLA conspire. MAC based Solution. There are two conceivable approaches to make utilization of MAC to confirm the information. A minor way is simply transferring the information obstructs with their MACs to the server, and sends the comparing mystery key sk to the TPA. Afterward, the TPA can haphazardly recover hinders with their MACs and check the accuracy by means of sk. Aside from the high (direct in the examined information measure) correspondence and calculation complexities, the TPA requires the learning of the information squares for check. To go around the necessity of the information in TPA confirmation, one may limit the check to simply comprise of balance checking. The thought is as takes after. Before information outsourcing, the cloud client picks irregular message validation code keys $\{sk_\tau\}$ $1 \leq \tau \leq s$, pre-processes (deterministic) MACs, $\{MAC_{sk\tau}(F)\} 1 \leq \tau \leq s$ for the entire information document F, and distributes these confirmation metadata (the keys and the MACs) to TPA. The TPA can uncover a mystery key $sk_\tau$ to the cloud server and request a new keyed MAC for examination in each review. This is privacy preserving for whatever length of time that it is difficult to recuperate F in full given $MAC_{sk\tau}(F)$ and $sk_\tau$ . Not withstanding, it endures from the accompanying serious downsides:

*1)* The number of times a specific information record can be evaluated is constrained by the quantity of mystery keys that must be settled a priori. When all conceivable mystery keys are depleted, the client at that point needs to recover information in full to re-register furthermore, re-distribute new MACs to TPA;

*2)* The TPA moreover needs to keep up and refresh state between reviews, i.e., keep track on the uncovered MAC keys. Considering the conceivably huge number of review assignments from different clients, keeping up such states for TPA can be troublesome and blunder inclined;

*3)* It can just help static information, and can't productively manage dynamic information by any stretch of the imagination.

HLAs, like MACs, are additionally some unpenetrable verification metadata that authenticate the integrity of a facts block. The difference is that HLAs can be aggregated. It is viable to compute an aggregated HLA which authenticates a linear combination of static individual blocks. At a high level, an HLA-based proof of storage system works as follow. The user still authenticates each thing of $F = (m_1, m_2, \ldots, m_n)$ *by a set of HLA's* $\Phi$

The cloud server saves {F, Φ}. The auditor checks the cloud storage by giving a random set of challenge $\{\nu_i\}$. (Precisely, F, Φ and $\{\nu_i\}$ are all vectors, so $\{\nu_i\}$ is an ordered set or $\{i, \nu_i\}$ should be sent). The cloud server then responds $\mu = P_{i*}\nu_{i*}$mi and an aggregated authenticator σ (both are computed from F, Φ and $\{\nu_i\}$) that is supposed to authenticate μ. Despite the fact that permitting productive information reviewing and devouring just steady data transfer capacity, the immediate reception of these HLA-based methods is as yet not appropriate for our motivations. This is on account of the direct mix of pieces, $\mu = P_{i*}\nu_{i*}\ m_i$, may conceivably uncover client information data to TPA, and disregards the privacy preserving ensure. In particular, if an enough number of the straight mixes of similar squares are gathered, the TPA can essentially infer the client's information content by comprehending an arrangement of direct conditions.

*E. Privacy-Preserving Public Auditing MethodOverview*

To acquire privatness-retaining public auditing, we advocate to uniquely combining the homomorphic linear authenticator with random overlaying method. In our protocol, the linear mixture of sampled blocks within the server's response is masked with random bits generated by the server. With random overlaying, the TPA now doesn't has all the important records to accumulate a correct group of linear equations and consequently cannot derive the consumer's facts content, no matter how many linear mixtures of the identical set of document blocks can be gathered. at the different hand, the correctness validation of the block authenticator pairs can nonetheless be done in a new way which will be demonstrated without further ado, even with the nearness of the arbitrariness. Our plan makes utilize of an open key based HLA, to prepare the examining convention with open auditability. In particular, we utilize the HLA proposed in [13], which depends on the short signature conspire proposed by Boneh, Lynn and Shacham (hereinafter alluded as BLS signature) [17].

*F. Method Details.*

Let $G_1$, $G_2$ and $G_T$ 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 a generator of $G_2$. H(·) is a secure map-to point hash function: {0, 1} ∗ → $G_1$, which maps strings uniformly to $G_1$. Another hash function h(·): $G_T \rightarrow Z_p$ maps group element of $G_T$ uniformly to $Z_p$. The proposed scheme is as follows:

*1) Setup Phase:* The cloud person runs KeyGen to generate the public and secret parameters. Specifically, the user chooses a random signing key pair ($s_{pk}$, $s_{sk}$), a random $x \leftarrow Z_p$, a random thing $u \leftarrow G_1$, and computes $v \leftarrow g^x$. The secret parameter is sk = (x, $s_{sk}$) and the public parameters are pk = ($s_{pk}$, v, g, u, e(u, v)). Given a statistics file F = ($m_1$,…,$m_n$), the person runs SigGen to compute authenticator $\sigma_i$ for each block $m_i$ : $\sigma_i \leftarrow (H(W_i) \cdot u^{mi})\ x \in G_1$. Here $W_i$ = name||I and title is chosen by way of the user uniformly at random from $Z_p$ as the identifier of file F. Denote the set of authenticators by way of Φ = $\{\sigma_i\}1 \leq i \leq n$. The final phase of SigGen is for ensuring the integrity of the special file identifier name. One easy way to do this is to compute t = name||$SSig_{ssk}$ (name) as the file tag for F, where $SSig_{ssk}$ (name) is the signature on identify below the private key $s_{sk}$. For simplicity, we expect the TPA is aware of the range of blocks n. The consumer then sends F alongside with the verification metadata (Φ, t) to the server and deletes them from local storage.

*2) Audit Phase:* The TPA first retrieves the file tag t. With recognize to the mechanism we describe in the Setup phase, the TPA verifies the signature $SSig_{ssk}$ (name) by means of $s_{pk}$, and stops by discharging FALSE on the off chance that the confirmation fizzles. Something else, the TPA recuperates name. Presently it goes to the "center" some portion of the examining process. To produce the test message for the review "challenge", the TPA picks an irregular c-component subset I = {$s_1$,…., $s_c$} of set [1, n]. For every component I ∈ I, the TPA likewise picks an arbitrary esteem $\nu_i$ (of bit length that can be shorter than |p|, as clarified in [13]). The message "challenge" determines the places of the obstructs that are required to be checked. The TPA sends challenge = $\{(i, \nu_i)\}i \in I$ to the server. After accepting test challenge = $\{(i, \nu_i)\}i \in I$ , the server runs GenProof to create a reaction evidence of information stockpiling rightness. In particular, the server picks an arbitrary component $r \leftarrow Z_p$, and figures R = e(u, v) r ∈ GT .

Let μ′ signify the direct mix of examined squares determined in challenge: $\mu' = P\ i\in I\ \nu_i m_i$. To dazzle μ′ with r, the server processes: $\mu = r+\gamma\mu' \bmod p$, where $\gamma = h(R) \in Z_p$. In the mean time, the server moreover computes a collected authenticator $\sigma = Q\ i\in I\ \sigma\ \nu_i\ I \in G_1$. It at that point sends {μ, σ, R} as the reaction verification of capacity accuracy to the TPA. With the reaction from the server, the TPA runs VerifyProof to approve the reaction by first figuring γ = h(R) and afterward checking the confirmation condition

$$R.e(\sigma^{\gamma}, g) \stackrel{?}{=} e\left(\left(\prod_{i=s_1}^{s_c} H(W_i)^{v_i}\right)^{\gamma} \cdot u^{\mu}, v\right) \quad (1)$$

The protocol is illustrated in Fig. 2. The correctness of the above verification equation can be elaborated as follows:

$$\begin{aligned} R.e(\sigma^{\gamma}, g) &= e(u, v)^{r} \cdot e\left(\left(\prod_{i=s_1}^{s_c} (H(W_i) \cdot u^{m_i})^{x v_i}\right)^{\gamma}, g\right) \\ &= e(u^{r}, v) \cdot e\left(\left(\prod_{i=s_1}^{s_c} (H(W_i)^{v_i} \cdot u^{v_i m_i})^{\gamma}, g\right)^{x}\right. \\ &= e(u^{r}, v) \cdot e\left(\left(\prod_{i=s_1}^{s_c} H(W_i)^{v_i}\right)^{\gamma} \cdot u^{\mu'\gamma}, v\right) \\ &= e\left(\left(\prod_{i=s_1}^{s_c} H(W_i)^{v_i}\right)^{\gamma} \cdot u^{\mu'\gamma + r}, v\right) \\ &= e\left(\left(\prod_{i=s_1}^{s_c} H(W_i)^{v_i}\right)^{\gamma} \cdot u^{\mu}, v\right) \end{aligned}$$

## IV. ALGORITHMS

So far there are no standard implementations of this Homomorphic Encryption but there exists two partial implementations with a mathematical approach, those are:Elgamal Additive Encryption and Multiplicative RSA Encryption the first one is discussed in this paper.

### *A. Elgamal Additive Encryption*

This algrithms consists of 3 functions: function KeyGen, function Encrypt(m) and function Decrypt(c). The definitions of each function is described below:

*1) Function KeyGen*

*a) Output: public key $k_{pub}$ and private key $k_{pr}$*

*b) function KeyGen*

- *Choose a large prime p*
- *Choose a primitive element $\alpha \in Z_p^*$*
- *Choose an integer $a \in \{0, \ldots, p-2\}$*
- *$\beta = \alpha^a \bmod p$*
- *return $k_{pub} = (p, \alpha, \beta)$, $k_{pr} = a$*

*c) end function*

*2) function Encrypt(m)*

*a) Input: public key $k_{pub} = (p, \alpha, \beta)$ and message m*

*b) Output: ciphertext c*

*c) function Encrypt(m)*

- *Choose $k \in \{2, \ldots, p-2\}$*
- *$x = \alpha^k \bmod p$*
- *$y = \alpha^m \cdot \beta^k \bmod p$*
- *return $c = (x, y)$*

*d) end function*

3) *function Decrypt(c)*
a) *Input: private key $k_{pr}$ = a and ciphertext c = (x, y)*
b) *Output: message m*
c) *function Decrypt(c)*
- $m^* = x^{-a} \cdot y \bmod p$
- *Recover m from* $m^* = \alpha^m$
- *return m*

d) *end function*

Example

An example describing the above algorithms is as follows:

4) *function KeyGen*

| | |
|---|---|
| Choose a large prime p | $p = 13$ |
| Choose a primitive element $\alpha \in Z_p^*$ | $\alpha = 2$ |
| Choose an integer $a \in \{0, \ldots, p-2\}$ | $a = 4$ |
| $\beta = \alpha^a \bmod p$ | $\beta = 2^4 \% 13 = 3$ |
| return $k_{pub} = (p, \alpha, \beta)$, $k_{pr} = a$ | return (13,2,3), 4 |

5) *function Encrypt(m)*

| | let $m_1 = 5$ | $m_2 = 7$ |
|---|---|---|
| Choose $k \in \{2, \ldots, p-2\}$ | $k_1 = 6$ | $k_2 = 8$ |
| $x = \alpha^k \bmod p$ | $x_1 = 2^6 \% 13 = 12$ | $x_2 = 2^8 \% 13 = 9$ |
| $y = \alpha^m \cdot \beta^k \bmod p$ | $y_1 = 2^5.3^6 \% 13 = 6$ | $y_2 = 2^7.3^8 \% 13 = 8$ |
| return c = (x, y) | return (12,6) | return (9,8) |

*Applying some simple operation **
*we took $m_1 = 5$ and $m_2 = 7$*
*After encryption we got*
$E(m_1) = (12,6)$ $E(m_2) = (9,8)$
*Multiplying $m_1$ and $m_2$*
$m_1 * m_2 = 5 * 7 = (12,6)*(9,8) = (12*9\%13, 6*8\%13) =$ ***(4,9)***

6) *function Decrypt(c)*

| | |
|---|---|
| | $C = (4,9)$ |
| $m^* = x^{-a} \cdot y \bmod p$ | $m^* = 4^{-4} \cdot 9 \% 13$ |
| Recover m from $m^* = \alpha^m$ | $m^* = (4^4)^{-1} \cdot 9 \bmod 13 \equiv 1$ |
| return m | $m^* = \{22, 35, 48\ldots\}$ |
| | *recovering m from α powers,* |
| | $35 \% 1 \equiv 36 \% 2 = 0$ **= 35** |

Hence the required operation was performed on encrypted data and decrypted to get the original value as if the operation was done on the plain text itself.

## V. PROPERTIES OF OUR SCHEME

It is obvious that our protocol achieves users auditability. There may be no mystery keying material or states for the TPA to keep or keep among audits, and the auditing protocol does no longer pose any capability on line burden on customers. This technique guarantees the privateness of consumer records content during the auditing manner by using employing a random protecting r to hide μ, a linear aggregate of the facts blocks. Caution that R in our method, which gives the privacy retaining guarantee, won't affect anything, due to the relationship between R and γ in γ = h(R) and the verification eq. Storage integrity thus goes from that of the given protocol [13]. The security of this scheme will be officially proven later. The HLA helps to achieve the constant

communication overhead for server's response during the audit: the size of {σ, μ, R} doesn't rely on the number of sampled blocks c.

*A. Support for Multi User Processing*

With the foundation of protection saving open evaluating, the TPA may simultaneously deal with different examining upon various clients' assignment. The individual inspecting of these undertakings for the TPA can be dull and extremely wasteful. Given K reviewing designations on K unmistakable information records from K extraordinary clients, it is more profitable for the TPA to bunch these numerous assignments together and review at one time. Remembering this regular request, we somewhat adjust the convention in a solitary client case, and accomplishes the collection of K check conditions (for K examining undertakings) into a solitary one, as appeared in Equation 2. Therefore, a protected group evaluating convention for concurrent inspecting of numerous assignments is acquired. The points of interest are portrayed as takes after.

1) *Setup Phase:* Basically, the clients simply perform Setup autonomously. Assume there are K users in the architecture, and eachuser k has an information $F_k = (m_k, m_1, \ldots, m_k, m_n)$ to be uploaded to the cloud, where $k \in \{1, \ldots, K\}$. For straightforwardness, we expect each document $F_k$ has a similar number of n pieces. For a specific user k, indicate the mystery key as $(x_k, s_{skk})$, and the comparing open attribute ($i \in \{1, \ldots, n\}$), in which $W_k,i = name_k||i$. eventually, each user k sends file $F_k$, set of authenticators $\Phi_k$, and tag $t_k$ to the server and deletes them from neighborhood garage.
2) *Audit section:* TPA first gets and verifies report tag $t_k$ for each user k for later auditing. If verification fails, auditor quits by way of emitting false; in any other case, TPA gets back $name_k$. The auditor then sends the audit challenge = $\{(i, v_i)\}i\in I$ to the cloud for auditing data files of all kclients. Ongetting challenge, for each user $k\in \{1, \ldots, k\}$, the server randomly choices $r_k\in Z_p$ and computes $R_k = e(u_k, v_k)^{r}{}_k$ .

Denote $R = R_1 \bullet R_2 \bullet \bullet \bullet R_K$, and $L = vk_1||vk_2|| \bullet \bullet \bullet ||vk_K$, our protocol further requires the server to compute $\gamma_k = h(R||v_k||L)$. Then, the randomly masked responses may be generated as follows:

$$\mu_k = \gamma_k \sum_{i=s_1}^{s_c} v_i m_k, i + r_k \, mod \, p \; and \; \sigma_k = \prod_{i=s_1}^{s_c} \sigma_{k,i}^{v_i}$$

The server then responses the TPA with
$\{\{\sigma k, \mu k\}\ 1\leq k\leq k, R\}$. To affirm the reaction, the TPA can first compute $\gamma_k = h(R||v_k||L)$ for $1 \leq k \leq k$. subsequent, TPA checks if the following equation holds:

$$R \cdot e(\prod_{k=1}^{k} \sigma_k^{\gamma_k}, g) \stackrel{?}{=} \prod_{k=1}^{k} e\left( (\prod_{i=s_1}^{s_c} H(W_k, i)^{v_i})^{\gamma_k} \cdot u_k^{\mu_k}, v_k \right) \quad (2)$$

The batch protocol is illustrated in Fig. three. right here the left-hand side (LHS) of Equation 2 expands as:

$$LHS = R_1 \cdot R_2 \ldots R_k \prod_{k=1}^{k} e(\sigma_k^{\gamma_k}, g)$$

$$= \prod_{k=1}^{k} R_k \, . \, e(\sigma_k^{\gamma_k}, g)$$

$$= \prod_{k=1}^{k} e\left( (\prod_{i=s_1}^{s_c} H(W_k, i)^{v_i})^{\gamma_k} \cdot u_k^{\mu_k}, v_k \right)$$

This is the proper hand side, as required. Note that the ultimate equality follows from Equation 1.

*B. Efficiency Improvement*

As proven in Equation 2, batch auditing now not only allows TPA to perform the a couple of auditing duties simultaneously, but also greatly reduces the computation price on the TPA side. This is due to the fact aggregating K verification equations into one helps minimize the wide variety of incredibly expensive pairing operations from 2K, as required in the individual auditing, to K + 1. Thus, a considerable amount of auditing time is anticipated to be saved.

*ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887*
*Volume 6 Issue IV, April 2018- Available at www.ijraset.com*

*C. Identification of Invalid Responses*

The verification equation (Equation 2) only holds when all the responses are valid, and fails with high probability when there is even one single invalid response in the batch auditing, as we will show in Section 4 In many situations, a response series might also comprise invalid responses, specially $\{\mu_k\}1\leq k\leq K$, brought on via accidental data corruption, or perchance malicious recreation by a cloud server. The ratio of invalid responses to the valid ought to be quite small, and yet a popular batch auditor will reject the entire collection. Going on a recursive divide-and-conquer approach (binary search) can be used to sort out these multi auditing tasks, as advised through [18]. We can certainly divide the collection of responses into two halves, and recurse the auditing on halves by Equation 2 if the multi auditing fails. The auditor now may additionally now require the server to send back all the $\{R_k\}1\leq k\leq K$, as individual auditing. In Section 4.2.2, even if up to 20% of responses are invalid,we can show that the usage of this recursive binary search approach batch auditing still performs faster than individual verification through carefully designed scan.

## VI.EVALUATION

*A. Protection Evaluation*

We examine the safety of the proposed scheme by way of reading its fulfillment of the security assure defined in phase 2.2, specifically, the garage correctness and privateness-retaining property. We start from the unmarried user case, where our major end result is originated. Then we display the security assure of batch auditing for the TPA in multi-consumer setting.

1) *Storage Correctness Assurance:* We need to show that the cloud server can't generate valid response for the TPA without faithfully storing the records, as captured by Theorem 1.

a) *Theorem 1:* If the cloud server passes the Audit section, then it must indeed own the required records intact as it's far.

b) *Proof:* The evidence consists of steps. First, we display that there exists an extractor of μ′ inside the random oracle model. as soon as a valid reaction {σ, μ′} are received, the correctness of this announcement follows from Theorem 4.2 in [13]. Now, the cloud server is dealt with as an adversary. The extractor controls the random oracle h(·) and solutions the hash query issued via the cloud server. For a project $\gamma = h(R)$ returned by means of the extractor, the cloud server outputs {σ, μ, R} such that the following equation holds.

$$R \cdot e(\sigma^{\gamma}, g) = e((\prod_{i=s_1}^{s_c} H(W_i)^{v_i})^{\gamma} \cdot u^{\mu}, v) \quad (3)$$

Assume that an extractor can rewind a cloud server in the convention to the point just before the test h(R) is given. Presently the extractor sets h(R) to be $\gamma * \neq \gamma$. The cloud server yields {σ, μ∗, R} to such an extent that the following condition holds as

$$R \cdot e(\sigma^{\gamma^*}, g) = e((\prod_{i=s_1}^{s_c} H(W_i)^{v_i})^{\gamma^*} \cdot u^{\mu^*}, v) \quad (4)$$

Comparable to the single client case, every client k has as of now haphazardly picked an alternate (with overpowering likelihood) name namek ∈ Zp for his/her record Fk, and has effectively produced the relating document tag $t_k = name_k||SSig_{sskk}(name_k)$. The extractor then gets $\{\sigma, \mu' = (\mu - \mu^*)/(\gamma - \gamma^*)\}$ as a correct response of the given proof of storage system [13]. To verify, recall $\sigma_i = (H(W_i) \cdot u\, m_i)\, x$, divide (3) by (4), we have

$$e(\sigma^{\gamma-\gamma^*}, g) = e((\prod_{i=s_1}^{s_c} H(W_i)^{v_i})^{\gamma-\gamma^*} \cdot u^{\mu-\mu^*}, v)$$

$$e(\sigma^{\gamma-\gamma^*}, g) = e((\prod_{i=s_1}^{s_c} H(W_i)^{v_i})^{\gamma-\gamma^*}, g^x) e(u^{\mu-\mu^*}, g^x)$$

$$\sigma^{\gamma-\gamma^*} = (\prod_{i=s_1}^{s_c} H(W_i)^{v_i})^{x(\gamma-\gamma^*)} \cdot u^{x(\mu-\mu^*)}$$

$$(\prod_{i=s_1}^{s_c} \sigma_i^{v_i})^{\gamma-\gamma^*} = (\prod_{i=s_1}^{s_c} H(W_i)^{v_i})^{x(\gamma-\gamma^*)} \cdot u^{x(\mu-\mu^*)}$$

$$u^{x(\mu-\mu^*)} = (\prod_{i=s_1}^{s_c} \sigma_i / H(W_i)^x)^{v_i})^{\gamma-\gamma^*}$$

$$u^{x(\mu-\mu^*)} = (\prod_{i=s_1}^{s_c} u^{xm_i})^{v_i})^{\gamma-\gamma^*}$$

$$\mu - \mu^* = (\sum_{i=s_1}^{s_c} m_i v_i) \cdot (\gamma - \gamma^*)$$

$$(\sum_{i=s_1}^{s_c} m_i v_i) = (\mu - \mu^*)/(\gamma - \gamma^*)$$

subsequently, we observation that this extraction argument and the random oracle paradigm are also used inside the proof of the underlying scheme [13].

2) *Privacy preserving assure:* We want to make sure that no data has been gained by the auditor in the process of audit.

a) *Theorem 2:*Auditor can't recover μ′ from the server's response {σ, μ, R},

b) .*Proof:* We display the existence of a simulator that can produce a legitimate response even without the understanding of μ ′, in the random oracle model. Now, the TPA is dealt with as an adversary. Given a valid σ from the cloud server, first off, randomly pick γ, μ from $Z_p$, set R ← $e((\prod_{i=s_1}^{s_c} H(W_i)^{v_i})^{\gamma} \cdot u^{\mu}, v)/e(\sigma^{\gamma}, g)$. in the end, back patch γ = h(R) for the reason that simulator is controlling the random oracle h(•). We statement that this back patching approach inside the random oracle model is also utilized in the proof of the underlying scheme [13].

3) *Protection Assure for Batch Auditing:* Now we show that our way of extending our result to a multi-person putting will not affect the aforementioned safety insurance, as proven in Theorem 3.

1) *Theorem 3:* Privateness guarantee and equal storage will be guaranteed by our batch audit protocol as in single client case.

2) *Proof:* The privacy-preserving warranty in the multi-user placing is very comparable to that of Theorem 2, and consequently not noted here. For the storage correctness guarantee, we are going to minimize it to the single-user case. We use the forking approach as in the proof of Theorem 1. However, the verification equationfor the batch audits includes K challenges from therandom oracle. This time we want to make certain that allthe other K − 1 challenges are decided earlier than the forking of the worried random oracle response. This can be accomplished the use of the notion in [24]. As quickly as the adversary problems the very first random oracle query for $\gamma_i$ = h(R||$v_i$ ||L) for any i ∈ [1, K], the simulator immediately determines the values $\gamma_j$ = h(R||$v_j$ ||L) for all j ∈ [1, K]. This is viable due to the fact that they are all using the identical R and L. As there is only one of the $\gamma_k$'s in Equation 2 are equal, so a correct response can be taken out similar to the single client case in the justification of Theorem 1. 4.2 Performance Analysis We now determine the overall performance of the proposed privacy-preserving public auditing schemes to show that they are certainly lightweight. We will center of attention on the value of the effectivity of the privacy preserving protocol and our proposed batch auditing technique. The scan is carried out using C on a Linux system with an Intel Core 2 processor going for walks at 1.86 GHz, 2048 MB of RAM, and a 7200 RPM Western Digital 250 GB Serial ATA pressure with an 8 MB buffer. Our code makes use of the Pairing-Based Cryptography (PBC) library model 0.4.18. The elliptic curve utilized in the experiment is a MNT curve, with base discipline measurement of 159 bits and the embedding degree 6. The security level is chosen to be 80 bit, which potential |$v_i$|= 80 and |p| = 160 All experimental outcomes signify the mean of 20 trials.

## VII. RELATED WORK

Ateniese et al. [8] are the primary to take into account public auditability in their described "provable records possession" (PDP) model for making sure ownership of data files on untrusted storages. Ourprotocol uses the RSAbased homomorphic linear authenticators for outsourced statisticsauditing and indicates randomly sampling a few blocks of the file. but, the general public auditability in their scheme needs the linear aggregate of sampled blocks exposed to outside auditor. When used on demand, their scheme isn't always privacy keeping, and might leak user data to the auditor. Juels et al. [11] describe a "proof of retrievability" (PoR) model, in which spot-checking and errors-correcting codes are used to make sure both "possession" and "retrievability" of

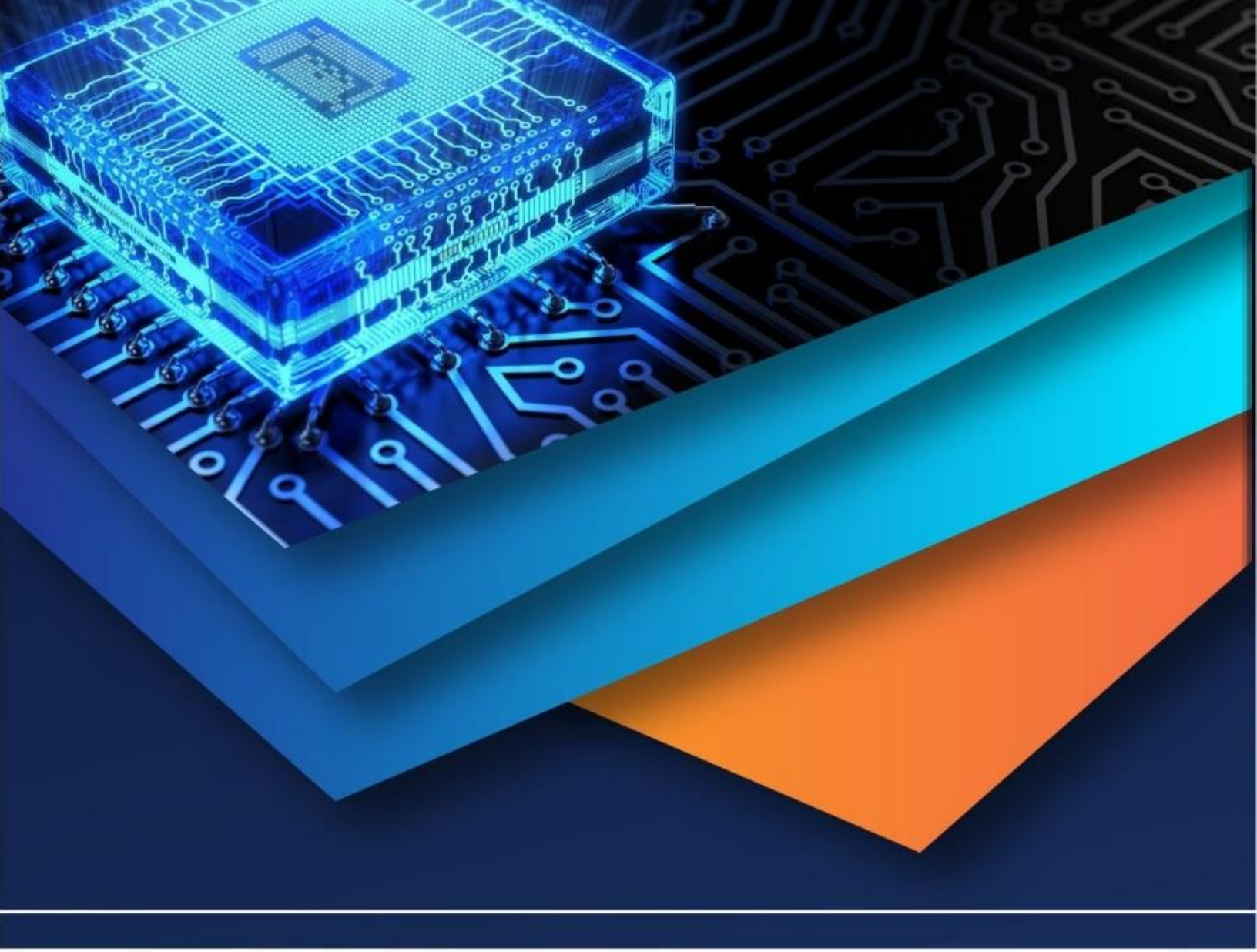
facts files on faraway archive carrier systems. however, the variety of audit demanding situations a user can perform is fixed a priori, and public auditability is not supported in their principal scheme. although they describe a truthful Merkle-tree construction for public PoRs, this method best works with encrypted facts. Dodis et al. [25] supply a take a look at on exceptional versions of PoR with non-public auditability. Shacham et al. [13] layout an progressed PoR scheme built from BLS signatures [17] with complete proofs of security inside the security version described in [11]. just like the construction in [8], they use publicly verifiable homomorphic linear authenticators which can be constructed from provably comfortable BLS signatures. primarily based on the elegant BLS production, a compact and public verifiable scheme is obtained. once more, their method does no longer aid privateness-preserving auditing for the identical motive as [8]. Shah et al. [9], [14] recommend allowing a TPA to preserve on-line storage sincere via first encrypting the information then sending more than a few of pre-computed symmetric-keyed hashes over the encrypted statistics to the auditor. The auditor verifies both the integrity of the facts report and the server's possession of a previously committed decryption key. This scheme handiest works for encrypted documents, and it suffers from the auditor statefulness and bounded utilization, which might also doubtlessly carry in online burden to users when the keyed hashes are used up.

## VIII. CONCLUSION

In this paper, we propose a protection saving open inspecting framework for information stockpiling security in Cloud Processing. We use the homomorphic direct authenticator what's more, irregular concealing to ensure that the TPA would not take in any information about the information content put away on the cloud server amid the productive inspecting process, which not just disposes of the weight of cloud client from the dreary and potentially costly examining errand, yet additionally mitigates the client's dread of their outsourced information spillage. Considering TPA may parallelly deal with different review sessions from various users for their uploadeddata, we additionally broaden our security protecting open evaluating convention into a multi-client setting, where the TPA can play out different reviewing undertakings in a bunch way for better proficiency. Broad investigation demonstrates that our plans are provably secure and profoundly effective.

## REFERENCES

[1] Privacy-Preserving Public Auditing for Secure Cloud Storage Cong Wang, Student Member, IEEE, Sherman S. M. Chow, Qian Wang, Student Member, IEEE, Kui Ren, Member, IEEE, and Wenjing Lou, Member
[2] P. Mell and T. Grance, "Draft NIST working definition of cloud computing," Referenced on June. 3rd, 2009 Online at http://csrc.nist.gov/groups/SNS/cloudcomputing/index.html, 2009.
[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A Berkeley view of cloud computing," University of California, Berkeley, Tech. Rep. UCB-EECS-2009-28, Feb 2009.
[4] M. Arrington, "Gmail disaster: Reports of mass email deletions," Online at http://www.techcrunch.com/2006/ 12/28/gmail-disasterreports-of-mass-email-deletions/, December 2006.
[5] J. Kincaid, "MediaMax/TheLinkup Closes Its Doors,"Online-at-http://www.techcrunch.com /2008/07/10/mediamaxthelinkup-closes-its-doors/, July 2008.
[6] Amazon.com, "Amazon s3 availability event: July20,2008,Onlinehttp://status.aws.amazon.com/ s3-20080720.html, 2008.
[7] S. Wilson, "Appengine outage," Online athttp://www.cioweblog.com/50226711/appengineoutage.php, June 2008.
[8] S Ezhil Arasu, B Gowri, and S Ananthi. Privacy-Preserving Public Auditing in cloud using HMAC Algorithm. International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277, 3878, 2013.
[9] IK Meenakshi and Sudha George. Cloud Server Storage Security using TPA. International Journal of Advanced Research in Computer Science & Technology (IJARCST) ISSN: 2347-9817, 2014.
[10] Jadhav Santosh and B.R. nandwalkar. Privacy Preserving and Batch auditing in Secure Cloud Data Storage using AES. Proceedings of 13th IRF International Conference, ISBN: 978-93-84209-37-72014.
[11] P. Mell, T. Grance, "Draft NIST Working Definition of Cloud Computing", National Institude of Standtard and Technology, 2009.
[12] M. Armbrust, "Above the Clouds: A Berkeley View of Cloud Computing", Tech. Rep. UCBEECS-2009–28, Feb. 2009.
[13] K Chen, WM Zheng, "CloudComputing: System instances and current research", Journal of Software, vol. 20, no. 5, pp. 1337-1348, 2009.
[14] DG Feng, M Zhang, Y Zhang, Z Xu, "Study on cloud computing security", Jounal of Software, vol. 22, no. 1, pp. 71-83, 2011.
[15] M. Arrington, Gmail Disaster: Reports of Mass Email Deletions, Dec. 2006.
[16] M. Krigsman, "Apple's MobileMe Experiences Post-Launch Pain", July 2008, [online] Available: http://blogs.zdnet.com/projectfailuresl?p~908.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY