International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-5, June 2019

# Kernel filtering for Multi-level integrity verification approach for cloud data storage

**K.Shirisha Reddy, M. Balaraju**

***Abstract:** User data privacy is essential in the cloud storage for protecting the user data in the cloud server. Register the user's data and accessing their respective data from the cloud server is an active area in the research community. Due to the data validation conditions, hardware, and operational complexity constraints, it is a challenging issue for the user to verify the integrity of data. Hence, in this paper, a new approach named Multi-level integrity verification scheme is proposed to ensure the data integrity of the user. The main focus of this work is to protect the user data privacy, and to forward the user signature and the files from the user to the Cloud Server Provider (CSP) and Third Party Auditor (TPA). The user files are verified based on the dual control mechanism and the file signature is verified by the Cloud Server Provider. The privacy of the signature information and the cloud storage space utilization is effectively enhanced. The performance of the proposed Multi-level integrity verification scheme is evaluated using different parameters, like computational time and memory usage of 889ns, and 4083bytes for 400kb dataset, respectively.*

***Keywords:** Cloud Service Provider (CSP), Data Integrity,Kernel filtering, Multi-level integrity verification, Third Party Auditor (TPA).*

## I. INTRODUCTION

Cloud computing is a paradigm in the internet, where the network resources are accessed and shared through the communication link as a service [9]. In the worldwide technology and in the medical area, the cloud computing is utilized in most of the situations, as it offers high computational performance and supports enormous data storage. The main benefit in using the cloud computing is the pay as use, resource allocation, and active provision. Accordingly, the cloud services are based on different dependencies, such as incompatibilities, and vulnerability in creating the holes. Hence, security in the cloud computing is a major concern in the research community, as it is important for the decision makers [12] [19] [20]. In the cloud computing framework, one of the most significant services offered is the cloud data storage, where the data is managed, maintained, remotely backed-up, and made the services easily accessible to the user through the internet. Moving the information in the cloud service significantly provides more expediency to the users, as the user does not care regarding the hardware management difficulties [9] [15] [16]. The cloud storage mainly depends on the growth of the cloud computing framework.

**Manuscript published on 30 June 2019.**

***** Correspondence Author (s)
.**Shirisha Reddy,** Research Scholar, JNTUH, Hyderabad, Telangana state, India.
**Dr. M. Balaraju,** Professor & Principal, SVIT, JNTUH, Hyderabad, Telangana state, India

The development and the boon of the cloud service offer the cloud users with more storage space, but still the cloud storage faces a lot of drawbacks, namely data deletion or tampered, and data missing. Therefore, data security in the cloud data storage becomes a major requirement in enhancing the cloud data services. In the huge data processing scheme, cloud storage offers a new solution in the cloud era. Security in the cloud services involvesAuthenticity, integrity, and confidentiality [1]. The service in the cloud storage offers the user to outsource their information to the cloud server, and this outsourced information will be remotely accessed from various places, e.g. GoogleDrive, Dropbox, and OneDrive. These services offer the user in a flexible and efficient way in managing their information without maintaining and deploying the storage service [4] [25].

Security mechanism should be offered to protect the confidential and the sensitive cloud data. The audit services and the cryptography concept ensure the integrity and the confidentiality of the outsourced data [9] [17] [18].Cloud storage offers more significant attention in both the industrial field and the educational field. In addition to the advantages of the cloud services, reliable data storage and security enhancement is a challenging task in the cloud storage. In the cloud storage, data integrity is a challenging task associated in the cloud services [14]. Therefore, data integrity is a major security concern in the cloud computing framework, as the users are worrying about the confidential of the data because the users are not physically rendered access to the data before it is outsourced to the cloud server. Checking the integrity of the user data is the basic need of the cloud server, as the cloud server generates a better integrity report with high reputation even when the data may be missed or damaged. The cloud user is responsible in preventing the cloud server from falsification [4] [21] [22]. The user loss will be huge when the cloud data gets damaged and at the same time, the attacks in the network cause irregularity between multiple information copies. Hence, it is required to check the data integrity regularly to enhance the consistency and correctness of multiple information copies [11]. Moreover, the external adversary affects the user data in the cloud server due to the political or the financial reasons. Hence, an effective secure verification approach is introduced to enhance the data integrity of the user. Some of the data verification methods depend on the users for executing the verification. Therefore, the user in the cloud storage has to spend more computation and communication costs for verifying the data integrity [4]. The data integrity control approach is used by the data owners to protect the outsourced storage based on the cryptographic hash mechanism.

Digital signature scheme is widely used in the data storage, while the data is shared among various users, and the Message authentication codes (MAC) is used to store the private data, while the data is shared by the single user. The outsourced data integrity is verified using the method named Third party auditor (TPA), which is referred as public verification [2]. To achieve the integrity control mechanism, it is necessary for the user to have the keys, like public key pair, and private key for the digital signature and the symmetric key for the MAC. Hence, the user data is stored in the form of MAC or as signature, which are computed using the symmetric key or the private key of the user. When the user knows that the data is preserved, then the user downloads the information and their significant MAC or signature from the cloud and verifies for the matching results. When the MAC or signature of the user matches, then it ensures the data integrity so that the user assured that the data is not corrupted in the cloud storage [10]. Accordingly, it is required to bear the huge verification and communication burden by the user for accessing the data. Moreover, public verification scheme is introduced to minimize the integrity verification burden. Hence, an independent and external auditor is adopted to check the integrity of data periodically with respect to the users [4].

The primary goal of this research is to design and develop a Kernel filtering for Multi-level integrity verification approach of cloud data storageto protect the user data privacy. The proposed data verification scheme involves six different phases, namely Setup (), Encode (), Dual control (), Challenge (), Prove () and Verify ().The data integrity approach uses three entity participants, namely CSP, TPA, and user [26]. In the Setup phase, the username and the password of the user is verified with the CSP. The kernel filtering is used in the Encode phase to divide the user file into different data blocks. However, the user forwards the audit request to the TPA for verifying the user data file and the verified challenge is send back by the TPA to the CSP. Finally, the TPA verifies the data file based on the dual control mechanism.

The major contribution of this paper is elaborated as follows:

- The proposed Multi-level integrity verification approach ensures the data integrity of the user by protecting the user data privacy in the cloud storage.
- The user credentials are verified and stored in the CSP and the kernel filtering is used for partitioning the user data files. The parameter, like session passwords, Chebyshev polynomial, hashing operation, and ECC is used in the mathematical model along with the Kernel filtering theory to ensure the data integrity.

The rest of the paper is organized as: the literature survey of the existing methods along with their merits and demerits are elaborated in section 2. The proposed Multi-level integrity verification approach is described in section 3, and the results along with the performance analysis are elaborated in section 4. Finally, the conclusion is made in section 5.

## II. MOTIVATION

The motivation behind the integrity verification approachesalong with the merits and demerits of the existing methods are discussed in this section.

### 2.1 Literature survey

Various existing methods are surveyed as: Yan Y *et al.* [1] developed data integrity verification approach to protect the user data.This approach effectively eliminated the quantum computer attacks. It enhancedthe usage of storage space in the cloud server ultimately. However, securitywas a major issue in the cloud storage data. Fan Y *et al.* [2] introduced a secure identity-based aggregate signature (SIBAS) approach to verify the data integrity in the cloud data. It was more efficient and feasible in solving the vulnerability problem. It verified the aggregate signature and minimized the key leakage, but the computational time of the server was increased. Ferretti L *et al.* [3] modelled a symmetric cryptographic approach to reduce the network and the storage overhead based on the structure of the database. In the realistic scenario, this approach effectively minimized the resource utilization. However, it failed to provide the data freshness and the data completeness using the bloom filter. Zhang Y *et al.* [4] introduced a public verification mechanism for cloud storage. It verified the message using the authentication code and performed the multi verification framework based on the auditor verification. The dynamic operations, like updating, deletion, and insertion were performed in the cloud data. This scheme achieved efficient computation and communication efficiency. However, the performance was not comparatively balanced. Yannan Li *et al.* [5] developed a fuzzy-based auditing approach for verifying the data integrity in the key management scheme. This approach ensured the data integrity in the outsourced data, but was not applicable in the real-world scenario. Jun Li [6] introduced a availability and integrity verification framework to ensure the data security in the cloud storage. This approach guaranteed the integrity of data based on the private verifiability. However, the performance of the data accessibility was better, but selecting the random data was not robust. Jinxia Wei *et al.* [7] modelled a provable data possession scheme for generating various data replicas. It supported the data integrity by verifying the file copies without leakage of any information. It provided seamless access to the users and supported the data update operation effectively. Even though it achieved better efficiency, controlling the data expansion was not effective. Willy Susilo *et al.* [8] modelled an extendable access control approach to verify the integrity data. It enhanced the security framework and achieved better performance in the access control and integrity check. Generating the new prefix was complex, as it took either the linear size of the server or the user.

### 2.2 Challenges

The major challenges of the research are:

- Selecting the parameter between the cloud and the user must be coordinated, as verifying the integrity in the cloud storage is a challenging task in the cloud computing [10].
- Third party auditor makes the auditing process of the data owner or the client using the encrypted data files. Public auditing in the cloud server is a major challenge associated in the verification scheme [9].
- In the multiple replicas, verifying the data integrity efficiently is a challenging problem in the cloud storage, as the third party auditor increases the user's auditing cost and the data in the cloud is stored integrally as a computing service [11].

- Measuring the secure storage and delivering the integrity measures is a major challenge in the cloud storage. The efficiency of the block chain in the cloud data storage results in the poor performance with respect to the access control in the cloud [12].
- Data storage in the cloud server results in the computational and time overhead. The sensitive and the sanitized information were not recorded in the cloud storage, which is a major drawback of the cloud auditing framework [13].

## III. PROPOSED MULTI-LEVEL INTEGRITY VERIFICATION APPROACH FOR USER DATA PRIVACY

Multi-level integrity verification approach is proposed to verify and ensure the data integrity of the user. The data integrity verification scheme is mainly used in the cloud storage to protect the user data privacy. In the cloud storage, the user needs to register their account and their own data to the cloud server. The overall procedure of the proposed data integrity approach is performed using six important phases, such as, Setup (), Encode (), Dual control (), Challenge (), Prove () and Verify ().The username and the password of the user is verified and stored into the CSP of the cloud storage in the Setup phase. The kernel filtering-based verification approach is used in the Encode phase to partition the user data file into the data blocks. The user forwards the audit request to the TPA for verifying the user file and thereby, the TPA sends the challenge to the CSP in the Challenge phase. In the Verify phase, the TPA verifies the user files based on the dual control mechanism, and the Dual control phase is carried out inside the Prove phase. Based on these phases, the data integrity verification scheme uses three entity participants, namely CSP, User, and TPA. By accommodating these phases, a new protocol is developed using a new mathematical model, where different factors and parameters, such as session passwords, Chebyshev polynomial, hashing operation, and ECC are used along with the Kernel filtering theory. Figure 1 shows the architecture of the cloud storage model.

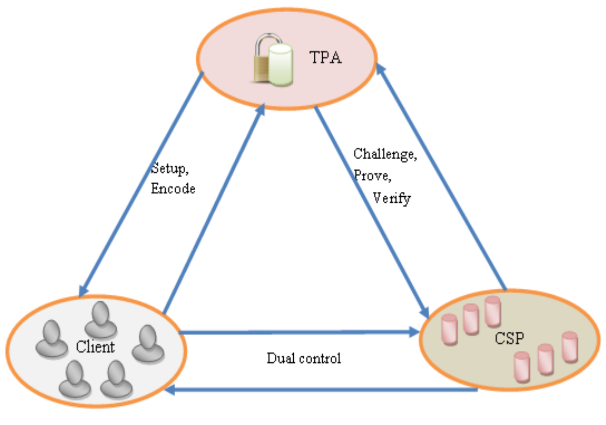


**Figure 1.** Architecture of the cloud storage model

### 3.1 Setup phase

In the Setup phase, the username and password of the user is created and is exchanged among the CSP and the user entity participants. Accordingly, the session password and the private key for user is also created and stored in the entity participants. Moreover, the name, private key and the session password for TPA is created and verified by the TPA entity. At first,the user credentials, like username and the password are created by the user and forwarded it to the CSP entity. The CSP stores the username and password of the user and create a session password for the user and forward the user session password to the respective user. Table 1 demonstrates the symbol description of the proposed integrity verification approach.

**Table 1.** Symbol description of the proposed Multi-level integrity verification approach

| Symbol | Description |
|---|---|
| $U_{name}$ | Username |
| $U_{pwd}$ | User password |
| $*$ | Stored |
| $\sim$ | Received |
| $U_{pvtkey}$ | private key for user |
| $cby_{pynl}^{user}$ | chebyshev polynomial for user |
| $cby_{pynl}^{TPA}$ | chebyshev polynomial for TPA |
| $TPA_n$ | TPA name |
| $TPA_{pd}$ | TPA password |
| $pb_{key}$ | public key |
| $h(.)$ | Hashing |
| $C[\ ]$ | Elliptic Curve Cryptography encryption |
| $S_{pvtkey}$ | private key for server |
| $TPA_{pwdses}$ | session password for TPA |
| $TPA_{pvtkey}$ | private key for TPA |
| $U_{pwdses}$ | session password for user |
| $cby_{pynl}^{TPA}$ | chebyshev polynomial for TPA |
| Aud req | Audit request |
| Chall req | Challenge request |
| // | Concatenation |
| $\oplus$ | Ex-or operation |
| $\otimes$ | Interpolation |
| $D_1, D_2, D_3, D_4$ | Signatures |
| $p_1$ | random prime number |

The user session password is created by the CSP entity by performing the Ex-OR operation with the ECC encryption and the hashing function, which is represented as,

$$U_{pwdses} = C\left[pb_{key} // S_{pvtkey}\right] \oplus h\left(cby_{pynl}^{user}\right) \tag{1}$$

The CSP creates the user private key by performing the Ex-or operation with the hashing of user password and the ECC encryption of the server private key, which is expressed as,

$$U_{pvtkey} = h\left(U_{pwd}^{*}\right) \bmod p_1 \oplus C\left(S_{pvtkey}\right) \tag{2}$$

where, $p_1$ denotes the random prime number. CSP creates the Chebyshev polynomial for user by using the below equation as,

$$cby_{pynl}^{user} = 8m^4 - 8m^2 + 1 \tag{3}$$

where,

$$m = U_{pwdses} \oplus h\left(U_{pvtkey} // pb_{key}\right) \tag{4}$$

The server private key and the public key are concatenated and the result is fed to the ECC encryption process. On the other hand, the hashing function is performed in the Chebyshev polynomial for TPA. The resulted ECC encrypted function and the hashed function are further processed by the Ex-or operation to create the session password for TPA, which is represented as,

$$TPA_{pwdses} = C\left[pb_{key} // S_{pvtkey}\right] \oplus h\left(cby_{pynl}^{TPA}\right) \tag{5}$$

The hashing process is applied to the TPA password and the result is modulated with the random number $p_2$, and the server private key is encrypted using the ECC encryption. Both the encrypted function and the hashed function are applied to the Ex-or operation to compute the private key for TPA. Hence, the computed TPA private keyis represented as,

$$TPA_{pvtkey} = h\left(TPA_{pd}\right) \bmod p_2 \oplus C\left(S_{pvtkey}\right) \tag{6}$$

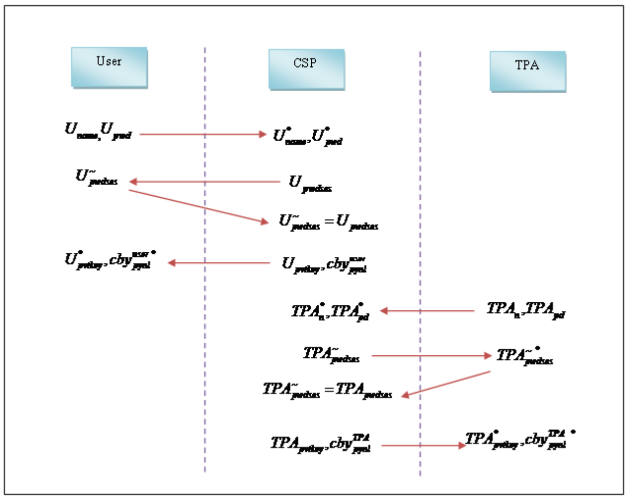

User
CSP
TPA
$U_{name}, U_{pwd}$
$U^{*}_{name}, U^{*}_{pwd}$
$U^{\sim}_{pwdses}$
$U_{pwdses}$
$U^{\sim}_{pwdses} = U_{pwdses}$
$U^{*}_{pvtkey}, cby^{user\,*}_{pynl}$
$U_{pvtkey}, cby^{user}_{pynl}$
$TPA^{*}_{n}, TPA^{*}_{pd}$
$TPA_{n}, TPA_{pd}$
$TPA^{\sim}_{pwdses}$
$TPA^{\sim\,*}_{pwdses}$
$TPA^{\sim}_{pwdses} = TPA_{pwdses}$
$TPA_{pvtkey}, cby^{TPA}_{pynl}$
$TPA^{*}_{pvtkey}, cby^{TPA\,*}_{pynl}$

**Figure 2.** Setup phase of the proposed Multi-level integrity verification scheme

Figure 2 shows the setup phase of the proposed integrity verification approach. The CSP received the user session password from the user and compared with the user session password, which is already stored in the CSP. The CSP creates the private key and chebyshev polynomial for the user and send it to the respective user, where the user stores the private key and the chebyshev polynomial for user, which is send by the CSP. The TPA creates the TPA name and TPA password and forward it to the CSP, while the CSP receives and stores the TPA name and password, respectively. The CSP creates a session password and send to the TPA entity, where the session password for TPA is stored secretly. The CSP entity receives the session password for TPA and verifies with the TPA session password, which is already stored in the CSP entity. The private key and the chebyshev polynomial for TPA are created by the CSP entity participant and forwarded to the TPA entity, where the private key and chebyshev polynomial are stored. Thus, the CSP creates the chebyshev polynomial for TPA using the below equation as,

$$cby_{pynl}^{TPA} = 64n^7 - 112n^5 + 56n^3 - 7n \tag{8}$$

where, $n$ is computed using the below expression as,

$$n = TPA_{pwdses} \oplus h\left(TPA_{pvtkey} // pb_{key}\right) \tag{9}$$

Finally, the TPA received the private key and chebyshev polynomial and stored in the TPA entity.

### 3.2 Encode Phase

In the Encode phase, the user partitions the data file $B$ into $I$ number of data blocks. Let $I_1, I_2, \ldots I_k, \ldots, I_n$ denotes the data blocks and the signatures, such as $D_1, D_2, D_3$, and $D_4$ are computed based on the data blocks in the Encode phase. The signature $D_1$ is computed by interpolating the data blocks and hashing the chebyshev polynomial of user with the username, which is represented as,

$$D_1 = \left(\sum_{k=1}^{l} I_k \oplus I_{k+1}\right) \oplus h\left(U_{name} // cby_{pynl}^{user\,*}\right) \tag{10}$$

The signature $D_2$ is calculated by interpolating the kernel filtering with the hashing of username, which is expressed using the below equation as,

$$D_2 = h\left(U_{name}\right) \otimes n \tag{11}$$

where, $n$ indicates the kernel filtering, and $\otimes$ represents the interpolation. The kernel filtering is calculated by applying the exponential to the chebyshev polynomial for user.

$$n = \exp\left(cby_{pynl}^{user\,*}\right) \tag{12}$$

The user password is concatenated with the kernel filtering and the result is fed to the hashing process, which is modulated with the public key to obtain the signature $D_3$. Hence, the signature $D_3$ is expressed as,

$$D_3 = (pb_{key}) \bmod h(U_{pwd} // n) \tag{13}$$

The user session password received by the user is concatenated with the kernel filtering and the resulted operation is passed into the hashing process, which is used to perform the EX-OR operation with the ECC encrypted username to obtain the signature $D_4$. Hence, the signature $D_4$ is represented as,

$$D_4 = h(U^{\sim}_{pwdses} // n) \oplus C(U_{name}) \tag{14}$$

Finally, the user forwards all the computed signatures, data blocks, an d TPA to the CSP entity in the Encode () phase.

**3.3 Challenge phase**

Once the signatures are send to the CSP entity then, the Challenge () phase begins to process its functions. In the Challenge () phase, the user broadcast the Aud req to the TPA to perform the verification process. The TPA receives the Aud req and verifies it to authenticate the identity of user. Once the user identity is authenticated, the TPR broadcast the Challreq to the CSP entity.

**3.4 Prove phase**

The challenge is broadcasted by the TPR to the CSP entity in the Challenge () phase. In the Prove phase, the CSP accepts the Chall req, which is send by the TPR through the authentication. The CSP generates the file signature for the corresponding Chall req and broadcast the file signature to the TPA entity.
The summation of the data blocks and the total received data blocks are allowed to perform the EX-OR operation. The username stored in the CSP entity is concatenated with the chebyshev polynomial for user and the result is applied into the hashing function. Finally, the results from hashing issubjected to the EX-OR operation with $\sum_{k=1}^{l} I^{\sim}_{k} \oplus I^{\sim}_{k+1}$. The summation of the data blocks is passed into the EX-OR operation with $I^{\sim}_{k+1}$.Hence, the file signature $D^{\sim}_1$ received by the TPA is expressed as,

$$D^{\sim}_1 = \left( \sum_{k=1}^{l} I^{\sim}_{k} \oplus I^{\sim}_{k+1} \right) \oplus h(U^{*}_{name} // cby^{user}_{pynl}) \tag{15}$$

The received file signature $D^{\sim}_2$ contains the parameter of kernel filtering and the hashed user password. The received file signature $D^{\sim}_2$ is represented using the below equation as,

$$D^{\sim}_2 = f(U^{\sim}_{pwd}) \otimes n \tag{16}$$

where, the kernel filtering $n$ is defined as the exponential of the chebyshev polynomial for user, which is indicated using the below equation as,

$$n = \exp(cby^{user}_{pynl}) \tag{17}$$

The received file signature $D^{\sim}_3$ has the hashed function of the user password and the kernel filtering with the modulated public key, which is indicated using the below expression as,

$$D^{\sim}_3 = pb_{key} \bmod f(U^{*}_{pwd} // n) \tag{18}$$

The received $D^{\sim}_4$ file signature has the EX-OR operation of the concatenated function with the ECC encrypted username.

$$D^{\sim}_4 = (U_{name} // n) \oplus C(U^{*}_{name}) \tag{19}$$

Hence, the CSP generates the file signatures for each Chall req and broadcast the file signatures to the TPA entity. The file signatures that are to be broadcasted from the CSP to the TPA is indicated as, $D^{\sim}_1$ $D^{\sim}_2$ $D^{\sim}_3$ ,and $D^{\sim}_4$ , respectively.

**3.5 Verify phase**

The file signatures are generated by the CSA in the Prove phase and are broadcasted to the TPA. In the verify phase, the TPA receives the file signatures, which is send from the CSA and verifies the data files based on the dual control mechanism.

**3.6 Dual control phase**

In the Dual control phase, the dual control mechanism is carried out for verifying the file signatures, which are generated by the CSP entity. The intermediate message $X_1$ is computed using the below expression as,

$$X_1 = D_1 \oplus D_2 \tag{20}$$

The intermediate message $X_2$ is computed as,

$$X_2 = D^{\sim}_1 \oplus D^{\sim}_2 \tag{21}$$

The intermediate message $X_1$ and $X_2$ are compared, if $X_1 = X_2$ then, it is considered as verification-1. Moreover, the intermediate messages $X_3$ and $X_4$ are computed as,

$$X_3 = f(D_3 // D_4) \tag{22}$$

$$X_4 = f(D^{\sim}_3 // D^{\sim}_4) \tag{23}$$

The intermediate messages $X_3$ and $X_4$ are compared, if $X_3 = X_4$ then,verification-2 terminates. Hence, in the dual control mechanism, the verification is performed at two levels. Figure 3 shows the Dual control phase of the proposed integrity verification approach.

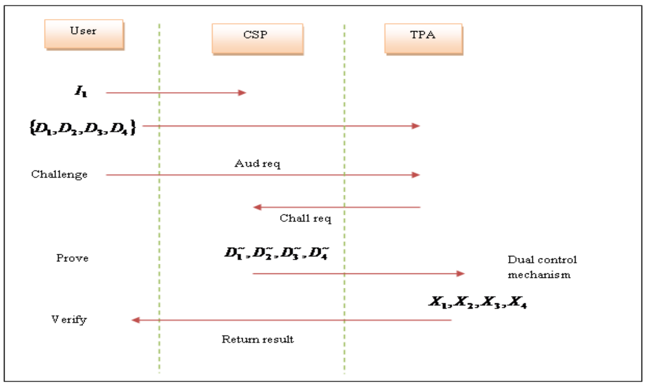


**Figure 3. Dual control phase of the proposed integrity verification approach**

## IV. RESULTS AND DISCUSSION

This section describes the results and discussion of the proposed Multi-level integrity verification approach to ensure the privacy of the user data integrity. The performance of the proposed verification scheme is analyzed and the comparative discussion with respect to the existing methods is elaborated.

### 4.1 Comparative methods

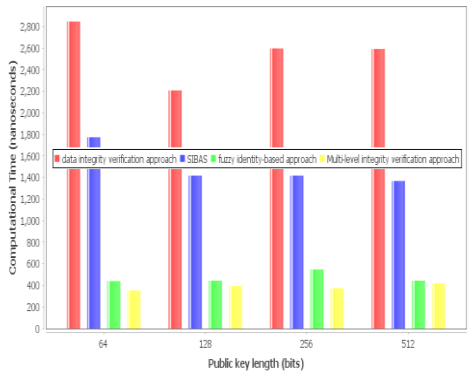
The performance of the proposed Multi-level integrity verification approach is evaluated and analyzed, and the proposed approach is compared with the existing methods, like data integrity verification approach [1], Secure Identity Based Aggregate Signature approach (SIBAS) [2], fuzzy identity-based approach [5], respectively.The performance is analyzed using the metrics, namely computational time and memory. The computational time is the time required to execute the computational operations. Memory is the storage area utilized in the computational functions.

### 4.2 Comparative analysis

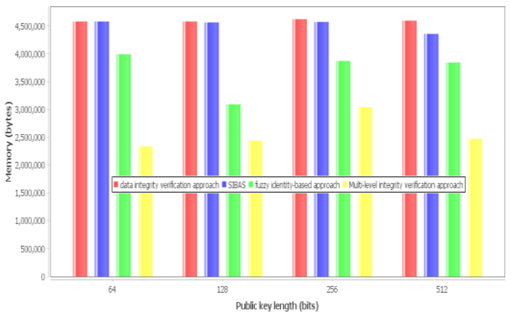
This section elaborates the comparative analysis of the proposed Multi-level integrity verification approach based on the computational time and memory.

***a) Analysis using 100kb dataset:*** Figure 4 shows the comparative analysis of the proposed verification approach using 100kb dataset based on the computational time and memory. Figure 4 a) depicts the analysis of the computational time by varying the public key length. When the length of the public key is 64 bits, the computational time of the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach is 2846ns, 1774ns, and 440ns, whereas the computational time of the proposed Multi-level integrity verification approach is less as 352ns. When the length of the public key is 128 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach executes the data integrity scheme with the maximum computational time of 2208ns, 1417ns, and 444ns, whereas the proposed Multi-level integrity verification approach executes the data integrity operations with the less computational time of 395ns. When the length of the public key is 256 bits, the computational time of the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach is 2597ns, 1417ns, and 547ns, while the computational time of the proposed Multi-level integrity verification approach is less as 374ns. When the public key length=512 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach executes the data integrity scheme with the maximum computational time of 2592ns, 1368ns, and 444ns, whereas the proposed Multi-level integrity verification approach executes the data integrity functions with the less computational time of 418ns, respectively.

Figure 4 b) depicts the analysis of storage by varying the public key length. When the length of the public key is 64 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the storage of 4579bytes, 45787bytes, and 3992bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 23367bytes. When the length of the public key is 128 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the maximum storage of 4579bytes, 4560bytes, and 3091bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 2439bytes. When the length of the public key is 256 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the maximum storage of 44620bytes, 4571bytes, and 3870bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 3043bytes, respectively. When the length of the public key is 512 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the maximum storage of 4594bytes, 4356bytes, and 3844bytes,whereas the proposed Multi-level integrity verification approach use the less storage of 2472bytes.



**(a)**

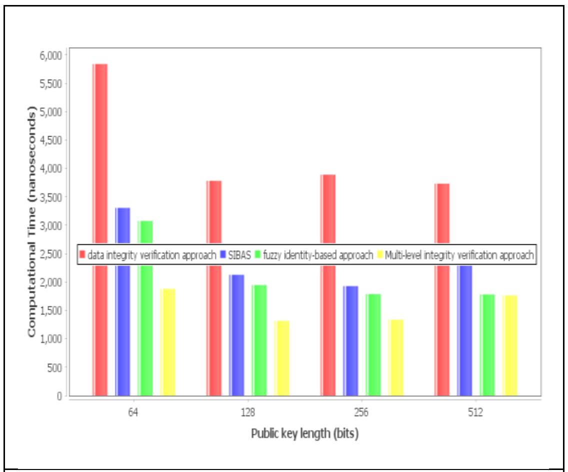


**(b)**

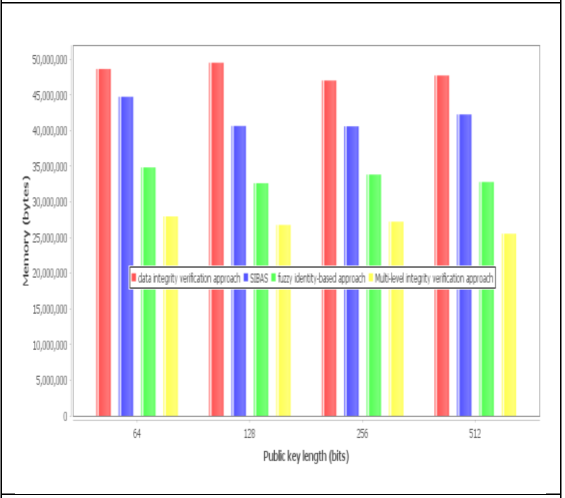
**.Figure 4. Comparative analysis using 100kb dataset, a) computational time, b) memory**

***b) Analysis using 200kb dataset:*** Figure 5 shows the comparative analysis of the proposed verification approach using 200kb dataset based on the computational time and memory. Figure 5 a) depicts the analysis of the computational time by varying the public key length. When the length of the public key is 64 bits, the computational time of the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach is 584ns, 330ns, and 307ns, whereas the computational time of the proposed Multi-level integrity verification approach is less as 1881ns. When the length of the public key is 128 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach executes the data integrity scheme with the maximum computational time of 3782ns, 2127ns, and 1948ns, whereas the proposed Multi-level integrity verification approach executes the data integrity operations with the less computational time of 1317ns. When the length of the public key is 256 bits, the computational time of the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach is 3891ns, 1929ns, and 1787ns, while the computational time of the proposed Multi-level integrity verification approach is less as 1337ns. When the public key length=512 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach executes the data integrity scheme with the maximum computational time of 3734ns, 2672ns, and 1782ns, whereas the proposed Multi-level integrity verification approach executes the data integrity functions with the less computational time of 1766ns, respectively.

Figure 5 b) depicts the analysis of storage by varying the public key length. When the length of the public key is 64 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the storage of 4863bytes, 4473bytes, and 3482bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 2797bytes. When the length of the public key is 128 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the maximum storage of 4951bytes, 4065bytes, and 3262bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 2676bytes. When the length of the public key is 256 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the maximum storage of 4701bytes, 4059bytes, and 3383bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 2722bytes, respectively. When the length of the public key is 512 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the maximum storage of 4772bytes, 4226bytes, and 3279bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 2554bytes.



**(a)**



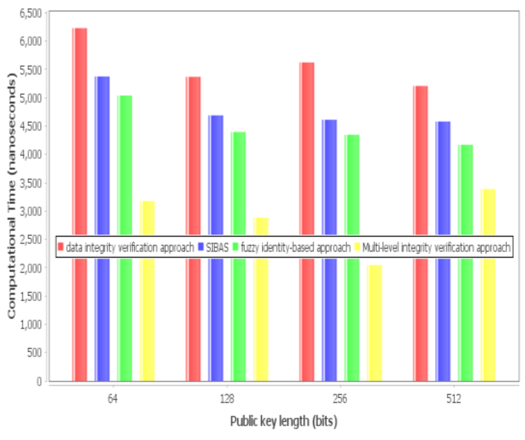
**(b)**

**Figure 5. Comparative analysis using 200kb dataset, a) computational time, b) memory**

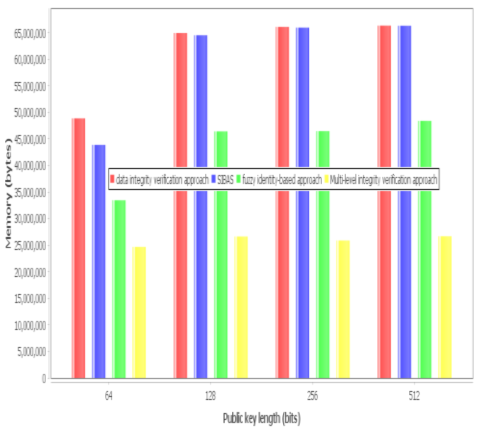
***c) Analysis using 300kb dataset:*** Figure 6 shows the comparative analysis of the proposed verification approach using 300kb dataset based on the computational time and memory. Figure 6 a) depicts the analysis of the computational time by varying the public key length. When the length of the public key is 64 bits, the computational time of the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach is 6229ns, 5376ns, and 5041ns, whereas the computational time of the proposed Multi-level integrity verification approach is less as 3175ns. When the length of the public key is 128 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach performs the data integrity operations with the highest computational time of 5370ns, 4689ns, and 4396ns, whereas the proposed Multi-level integrity verification approach executes the data integrity operations with the less computational time of 2884ns. When the length of the public key is 256 bits, the computational time of the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach is 5623ns, 4612ns, and 4347ns, while the computational time of the proposed Multi-level integrity verification approach is less as 2045ns.

When the public key length=512 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach executes the data integrity scheme with the maximum computational time of 5208ns, 4580s, and 4170ns, whereas the proposed Multi-level integrity verification approach executes the data integrity functions with the less computational time of 3387ns, respectively.

Figure 6 b) depicts the analysis of storage by varying the public key length. When the length of the public key is 64 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the storage of 4886bytes, 4389bytes, and 3347bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 2469bytes. When the length of the public key is 128 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach uses the maximum storage of 6495bytes, 6453bytes, and 4642bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 2665bytes. When the length of the public key is 256 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the maximum storage of 6611bytes, 6595bytes, and 4648bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 2589bytes, respectively. When the length of the public key is 512 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the maximum storage of 6634bytes, 6630bytes, and 4841bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 2668bytes.



(a)



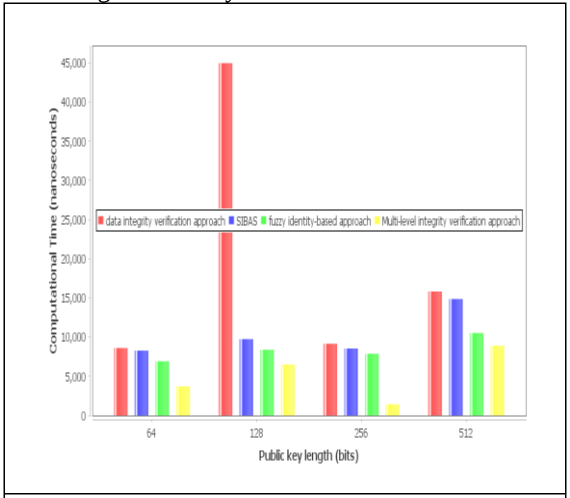
(b)

**Figure 6. Comparative analysis using 300kb dataset, a) computational time, b) memory**

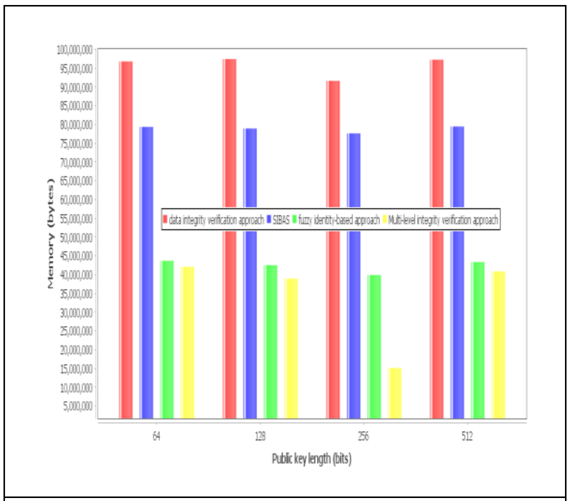
***d) Analysis using 400kb dataset:*** Figure 7 shows the comparative analysis of the proposed verification approach using 400kb dataset based on the computational time and memory. Figure 7 a) depicts the analysis of the computational time by varying the public key length. When the length of the public key is 64 bits, the computational time of the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach is 8622ns, 8275ns, and 6923ns, whereas the computational time of the proposed Multi-level integrity verification approach is less as 3718ns. When the length of the public key is 128 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach performs the data integrity operations with the highest computational time of 4497ns, 9750ns, and 8399ns, whereas the proposed Multi-level integrity verification approach executes the data integrity operations with the less computational time of 6517ns. When the length of the public key is 256 bits, the computational time of the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach is 9164ns, 8544ns, and 7879ns, while the computational time of the proposed Multi-level integrity verification approach is less as 1428ns. When the public key length=512 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach executes the data integrity scheme with the maximum computational time of 1528ns, 1486s, and 1053ns, whereas the proposed Multi-level integrity verification approach executes the data integrity functions with the less computational time of 8898ns, respectively.

Figure 7 b) depicts the analysis of storage by varying the public key length. When the length of the public key is 64 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the storage of 9683bytes, 7936bytes, and 4367bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 4209bytes.

When the length of the public key is 128 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach uses the maximum storage of 9745bytes, 7894bytes, and 4251bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 3895bytes.



**(a)**



**(b)**

**Figure 7. Comparative analysis using 400kb dataset, a) computational time, b) memory**

When the length of the public key is 256 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the maximum storage of 9165bytes, 7765bytes, and 3986bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 1511bytes, respectively. When the length of the public key is 512 bits, the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach use the maximum storage of 9729bytes, 7946bytes, and 4336bytes, whereas the proposed Multi-level integrity verification approach use the less storage of 4083bytes. Table 2 shows the comparative analysis of the proposed Multi-level integrity verification approach.

**Table 2. Comparative analysis of the proposed Multi-level integrity verification approach**

| **Security Issues** | Huansheng Ning *et al.*'s scheme (2015) | Odelu*et al.*,(2015) | Vivek V. Jog and Senthil Murugan T (2017) | Proposed Multi-level integrity verification approach |
|---|---|---|---|---|
| Provides mutual authentication | Yes | Yes | Yes | Yes |
| Provides multi-level authentication | Yes | Yes | Yes | Yes |
| Requires identity-verification table | Yes | Yes | Yes | Yes |
| Server spoofing attack resistance | Yes | Yes | Yes | Yes |
| Stolen verifier attack resistance | No | Yes | Yes | Yes |
| Privileged insider attack resistance | Yes | Yes | Yes | Yes |
| Password guessing attack resistance | Yes | Yes | Yes | Yes |
| Provides strong user anonymity | Yes | Yes | Yes | Yes |
| Known session-specific temporary information attack resistance | No | Yes | Yes | Yes |
| Impersonation attack resistance | Yes | Yes | Yes | Yes |
| Reply attack resistance | Yes | Yes | Yes | Yes |
| Man-in-the-middle attack resistance | Yes | Yes | Yes | Yes |
| Provision for revocation and re-registration | Yes | Yes | Yes | Yes |
| Free from denial of service attack | No | Yes | Yes | Yes |
| Profile table-stolen resistance | No | No | Yes | Yes |
| Key resilience | No | No | Yes | Yes |
| Reconnaissance attack resistance | No | No | No | Yes |
| Free from theft attack | No | No | No | Yes |
| Dual control | No | No | No | Yes |
| Data loss attack | No | No | No | Yes |
| Integrity verification | No | No | No | Yes |

### 4.3 Comparative discussion

The comparative discussion made using the proposed Multi-level integrity verification approach is elaborated in this section. Table 3 shows the comparative discussion of the proposed integrity verification approach based on the computational time and storage. The computational time of the existing methods, like data integrity verification approach, SIBAS, and fuzzy identity-based approach is high for 100kb dataset with the values of 2592ns, 1368ns, and 444ns, while the proposed integrity verification approach executes the operations with the lower computational time of 418ns, respectively. In the 200kb dataset, the computational time of the existing methods, such as data integrity verification approach, SIBAS, and fuzzy identity-based approach is high as 3734ns, 2672ns, and 1782ns, whereas the computational time of the proposed Multi-level integrity verification approach is 1766ns. The storage utilized by the existing methods, such as data integrity verification approach, SIBAS, and fuzzy identity-based approach in the 200 kb dataset is high of 4772 bytes, 4226 bytes, and 3279 bytes, while the memory usage of the proposed integrity verification approach is low of 2554 bytes, respectively. For 400kb dataset, the computational time of the existing methods, namely data integrity verification approach, SIBAS, and fuzzy identity-based approach is high with the time of 1582ns, 1486ns, and 1053ns, while the computational time of the proposed integrity verification approach is low with the time of 889ns, respectively. In the 400kb dataset, the existing methods, such as data integrity verification approach, SIBAS, and fuzzy identity-based approach use the highest storage of 9729 bytes, 7946 bytes, and 4336 bytes, while the proposed use the less storage 4083 bytes, respectively.

**Table 3. Comparative discussion**

| Metrics/Methods | | Data integrity verification approach | SIBAS | Fuzzy identity-based approach | Proposed Multi-level integrity verification approach |
|---|---|---|---|---|---|
| **100kb** | *Computational time(ns)* | 2592 | 1368 | 444 | 418 |
| | *Memory(bytes)* | 4594 | 4356 | 3844 | 2472 |
| **200kb** | *Computational time(ns)* | 3734 | 2672 | 1782 | 1766 |
| | *Memory(bytes)* | 4772 | 4226 | 3279 | 2554 |
| **300kb** | *Computational time(ns)* | 5208 | 4580 | 4170 | 3387 |
| | *Memory(bytes)* | 6634 | 6630 | 4841 | 2668 |
| **400kb** | *Computational time(ns)* | 1582 | 1486 | 1053 | 889 |
| | *Memory(bytes)* | 9729 | 7946 | 4336 | 4083 |

## V. CONCLUSION

The data integrity verification scheme named Multi-level integrity verification approach is proposed in this paper to verify and ensure the data integrity of the user. In the cloud storage, the user needs to register their account and their own data to the cloud server. The proposed data integrity verification approach involves six different phases, such as, Setup (), Encode (), Dual control (), Challenge (), Prove () and Verify (). The proposed data integrity approach uses three entity participants, namely Cloud Server Provider, User, and Third Party Auditor. The username and the password of the user is verified and stored into the CSP of the cloud storage in the Setup phase. The kernel filtering-based verification approach is used in the Encode phase to partition the user data file into the data blocks. In the Verify phase, the TPA verifies the user files based on the dual control mechanism, and the Dual control phase is carried out inside the Prove phase. By accommodating these phases, a new mathematical model is developed, where different parameter, like session passwords, chebyshev polynomial, hashing operation, and Elliptic Curve Cryptography are along with the kernel filtering theory. The performance of the proposed Multi-level integrity verification approach is better using the security parameters, like computational time and memory usage of 889ns, and 4083 bytes for 400kb dataset, respectively.

## ACKNOWLEDGEMENT

The author would thank to higher authorities of JNTUH to use the research laboratories. The author would thank for the facilities provided by FIST for research in VBIT.

## REFERENCES

1. Yan Y, Wu L, Gao G, Wang H and Xu W, “A dynamic integrity verification scheme of cloud storage data based on lattice and Bloom filter”, Journal of information security and applications, vol. 39, pp.10-18, 2018.
2. Fan Y, Lin X, Tan G, Zhang Y, Dong W and Lei J, “One secure data integrity verification scheme for cloud storage”, Future Generation Computer Systems, 2019.
3. Ferretti L, Marchetti M, Andreolini M. and Colajanni M, “A symmetric cryptographic scheme for data integrity verification in cloud databases”, Information Sciences, vol. 422, pp.497-515, 2018.
4. Zhang Y, Xu C, Liang X, Li H, Mu Y and Zhang X, “Efficient public verification of data integrity for cloud storage systems from indistinguishability obfuscation”, IEEE Transactions on Information Forensics and Security, vol. 12, no. 3, pp.676-688, 2017.
5. Yannan Li, Yong Yu, Geyong Min, Willy Susilo, Jianbing Ni and Kim-Kwang Raymond Choo, "Fuzzy Identity-Based Data Integrity Auditing for Reliable Cloud Storage Systems", Journal of Latex Class Files, vol. 14, no. 8, August 2015.
6. [6] Jun Li, "Novel availability and integrity verification protocol for ISMAC system under cloud environment", Computers & Electrical Engineering, vol. 57, pp.209-219, 2017.
7. Jinxia Wei, Mingxu Yi, Lingwei Song, “Efficient integrity verification of replicated data in cloud computing system”, computers & security, vol. 65, pp.202-212, 2017.
8. Willy Susilo, Peng Jiang, Fuchun Guo, Guomin Yang, Yong Yu and Yi Mu, “EACSIP: Extendable Access Control System with Integrity Protection for Enhancing Collaboration in the Cloud”, IEEE Transactions on Information Forensics and Security, vol. 12, no. 12, pp.3110-3122, 2017.
9. Hiremath S. and Kunte S, “A novel data auditing approach to achieve data privacy and data integrity in cloud computing”, IEEE International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT), pp. 306-310, December 2017.
10. Apolinario F Pardal M. and Correia M, “S-Audit: Efficient Data Integrity Verification for Cloud Storage”, IEEE International Conference on Trust, Security And Privacy In Computing And Communications, pp. 465-474, August 2018.

11. Li H, Li Q and Zhang L, "Multiple-replica cloud data integrity verification scheme based on B+ tree", IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), pp. 2279-2282, March 2017.
12. Zhao B, Fan P and Ni M, "Mchain: A Blockchain-Based VM Measurements Secure Storage Approach in IaaS Cloud With Enhanced Integrity and Controllability", IEEE Access, vol. 6, pp.43758-43769, 2018.
13. Wenting Shen, Jing Qin, Jia Yu, Rong Hao, and Jiankun Hu, "Enabling Identity-Based Integrity Auditing and Data Sharing with Sensitive Information Hiding for Secure Cloud Storage", IEEE transactions on information forensics and security, vol. 12 , no. 2, pp.331-346, 2018.
14. Luo, W and Bai, G "Ensuring the data integrity in cloud data storage", IEEE International Conference on Cloud Computing and Intelligence Systems, pp. 240-243, September 2011.
15. Swapnali Morea and Sangita Chaudhari , "Third Party Public Auditing Scheme for Cloud Storage", International Journal of Procedia Computer Science, Vol. 79, pp. 69-76, 2016.
16. Cong Wang , Sherman S M Chow, Qian Wang, Kui Ren, and Wen jing Lou. "Privacy Preserving Public Auditing for Secure Cloud Storage," IEEE Transactions on Computers, Vol. 62, no. 2, February 2013.
17. Wang C., Wang Q., Ren K., Lou W., "Privacy-preserving public auditing for data storage security in cloud computing", IEEE proceedings in infocom, pp. 1–9, March 2010.
18. Worku S.G., Xu C., Zhao J. and He X., "Secure and efficient privacy-preserving public auditing scheme for cloud storage", Computers & Electrical Engineering, vol. 40, no. 5, pp.1703-1713, 2014.
19. Duan Qiang, Yuhong Yan, and Athanasios V. Vasilakos, "A Survey on Service-Oriented Network Virtualization Toward Convergence of Networking and Cloud Computing," IEEE Transactions on Network and Service Management, vol. 9, no. 4, pp. 373-392, 2012.
20. Rajak S and Verma A, "Secure Data Storage in the Cloud using Digital Signature Mechanism," International Journal of Advanced Research in Computer Engineering & Technology, vol. 1, no. 4, 2012.
21. Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling public auditability and data dynamics for storage security in cloud computing," IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 5, pp. 847–859, 2011.
22. M. Sookhak, A. Gani, H. Talebian, A. Akhunzada, S. U. Khan, R. Buyya, and A. Y. Zomaya, "Remote data auditing in cloud computing environments: A survey, taxonomy, and open issues," ACM Computing Surveys, vol. 47, no. 4, 2015.
23. Twenty newsgroup dataset, "https://www.kaggle.com/crawford/20-newsgroups", accessed on May 2019.
24. Reuter database, https://www.kaggle.com/nltkdata/reuters accessed on May 2019.
25. C. Wang, S. S. Chow, Q. Wang, K. Ren, and W. Lou, "Privacypreserving public auditing for secure cloud storage," IEEE Transactions on Computers, vol. 62, no. 2, pp. 362–375, 2013.
26. K.Shirisha Reddy,Dr.M.BalaRaju,"An integrated approach of data storage and security in cloud computing", International Journal of Application or Innovation in Engineering and Management, vol. 1,Issue 4,pp.72-78,Dec 2012.

## AUTHORS PROFILE

**Mrs. K. SHIRISHA REDDY**, B.Tech(CSIT), M.Tech(SE) ,Ph.D(CSE) pursuing from JNTUniversity Hyderabad and having 13 years of Teaching Experience. Presently working as an Associate Professor, CSE,VBIT.

**Dr.M.BALARAJU,** B.E(E.C.E), M.Tech(CSE), PhD(CSE) and having 25 years of Teaching Experience. He is doctorate from JNTU Hyderabad.He is working as a Principal , Professor,SVIT,Hyderabad.