International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8 Issue-6, March 2020

VASD^{2}OM: Virtual Auditing and Secure Deduplication with Dynamic Ownership Management in Cloud

Geeta C M, Nikhil R C, Raghavendra S, Rajkumar Buyya, Venugopal K R

Abstract: ***In cloud repository amenities, deduplication technology is often utilized to minimize the volume and bandwidth by removing repetitious information and caching only a solitary duplicate of them. Deduplication is extremely productive when many customers deploy the identical information to the cloud repository, but it aggrevates problems pertaining to safety and proprietorship. Proof-of-ownership mechanism authorize any possessor of an identical information to approve to the cloud repository server that he possess the information in a dynamic way. In repository utilities with enormous information, the repository servers may want to minimize the capacity of cached information, and the customers desires to examine the integrity of their information with a reasonable cost. Aiming at realizing integrity auditing and deduplication with effective proprietorship administration of information in the cloud, we propose Virtual Auditing and Secure Deduplication with Dynamic Ownership Management in Cloud (VASD^{2}OM) mechanism. It empowers the distributed server to limit access to the deployed information though the proprietorship transforms actively. Further, the mechanism supports safe and effective virtual auditing of the documents during the download process. In addition, the proposed mechanism lowers the burden of dataowner to audit documents by himself and there is no need to authorize auditing to a Third Party Auditor (TPA). Experimental results demonstrate that the virtual auditing has low auditing time cost relative to the existing public auditing schemes.***

Keywords: Cloud storage, Deduplication, Encryption, Proof-of-Ownership, Revocation, Virtual auditing.

I. INTRODUCTION

Cloud computing delivers extensible, inexpensive and location-independent online facilities extending from simple backup facilities to distributed repository frameworks. Now a days optical networks [1], [2] have been deployed all over the globe for efficient information communication. The rapid expansion of information capacity stockpiled in the distributed repository has compelled to a rising need for procedures for conserving disk capacity and network bandwidth.

To minimize resource utilization, various distributed repository utilities, such as Dropbox [3], Google Drive [4], make use of deduplication procedure, where the distributed repository saves one solitary duplicate of repetitious information and furnishes links to the copy rather than of saving other genuine copies of that information, irrespective of the number of customers request to save the information.

In order to secure their personal information from unapproved external attackers and from the Cloud Service Provider (CSP) [5], customers encode their documents before deploying to the distant server. While, traditional encryption cannot be used to carry out deduplication due to the following reasons; Deduplication method takes benefit of information equivalence to recognize the identical information and decrease the repository capacity. On the contrary, the cryptographic algorithms shuffle the encoded documents in order to make ciphertext equivalent from theoretically random information. Encryption of the identical documents by distinct customers with distinct encryption keys results in distinct ciphertexts makes it critical for the distributed server to determine if the plaintext are identical and deduplicate them.

Concurrent encryption [6] solves this issue successfully. The concurrent encryption algorithm encodes an input document with the hash value of the input document as an encode key. The ciphertext is transmitted to the server and the customer keeps the encode key. Since concurrent encryption is imperative, alike documents are encoded into equivalent ciphertext, despite of who encodes them. Thus, the distributed repository server can carry out deduplication over the encoded document. and all proprietors of the document can retrieve the encoded document (after executing the proof-of-proprietorship convention) and decode it subsequently since they possess the identical encode key for the document. In the case of ownership repudiation, assume numerous customers possess proprietorship of a ciphertext deployed in distributed repository.

Manuscript received on February 10, 2020.
Revised Manuscript received on February 20, 2020.
Manuscript published on March 30, 2020.
* Correspondence Author
Geeta C Mara*, CSE, University Visvesvaraya College of Engg, Bangalore University, Bangalore, India. Email: geetacmara@gmail.com
Nikhil R C, ASE, Jain University, Bengaluru, India. Email:nikhilrc1504@gmail.com
Raghavendra S, CSE,University Visvesvaraya College of Engg, Bangalore University, Bangalore, India. Email: raghush86@gmail.com
Rajkumar Buyya, CSE, The University of Melbourne, Australia.Email:rbuyya@unimelb.edu.au
Venugopal K R, Bangalore University, Bangalore, India. Email: venugopalkr@gmail.com

With passage of time, a few of the customers might demand the distributed server to remove or alter their documents, and the server removes the proprietorship information of the customers from the proprietorship list for the equivalent documents. Then, the repudiated customers are prohibited from retrieving the documents saved in the distributed repository after the deletion or alteration inquiry (forward privacy).

Contrarily, when a customer deploys document that formerly prevails in the distributed repository, the customer is prevented from retrieving the document that were cached before he acquired the proprietorship by uploading it (backward privacy). Therefore, as long as repudiated customers possess the encode key, they can retrieve the equivalent document in the distributed repository at any time, despite of the legitimacy of their proprietorship.

When customers utilize cloud repository facilities, cus-mers intend to be assured about the genuineness of their information in the server. Hence, it is an essential prerequisite of customers to regularly audit the present state of their information. So far, numerous mechanisms have been suggested including proof of retrievability and verifiable information possession mechanisms. Reliable deduplication and integrity verification are fundamental functions needed in distributed repository utilities. Accordingly, individual studies have been sincerely carried out on these two topics. Still, comparatively few research [7], [8] have been carried out for implementing a combined mechanism that can support these two functions simultaneously.

In this paper, we introduce Virtual Auditing and Secure Deduplication with Dynamic Ownership Management in Cloud (VASD²OM) that supports virtual auditing and secure deduplication of encoded information that has effective ownership management ability.

A. Motivation

Currently, there is a noticeable growth in the size of information cached in repository utilities, accompanying with dramatic development of networking methods. In repository amenities with very large information, the repository servers may prefer to decrease the capacity of stockpiled information, and the customers may need to check the integrity of their information with a reasonable cost. Effective proprietorship administration is an essential and demanding challenge in secure deduplication over encoded information in distributed repository. Motivated by this factor, we present a new Virtual Auditing and Secure Deduplication with Dynamic Ownership Management in Cloud mechanism that bolsters safe and effective auditing of the data owner's documents by the virtual auditing entity and secure deduplication of encrypted information that has dynamic ownership management capability. In our proposed scheme, the information proprietor has been relieved from the responsibility of verifying the documents by himself and there is no need to assign the verification job to a Third Party Auditor (TPA). Further, the scheme provides efficient data deduplication over encoded information with active ownership management.

B. Contributions

In this paper, we introduce a new Virtual Auditing and Secure Deduplication with Dynamic Ownership Management in Cloud (VASD²OM) mechanism that supports secure and effective virtual auditing and deduplication of encrypted information that has dynamic ownership management capability. Specifically, our contributions can be summarized as follows:

(i) Secure deduplication of encoded information with dynamic proprietorship administration capability.

(ii) Efficient auditing of data owners file virtually by the virtual auditing entity during download process.

C. Organization

The list of the paper is compiled as follows: Related works are discussed in Section 2 which gives the pros and cons on existing integrity auditing and deduplication schemes. In Section 3, earlier models and their drawbacks are presented and several preliminaries are discussed in Section 4. Problem statement and System framework illustrates the operation of the framework and provides the technicalities regarding the design goals, in Section 5. In Section 6, scheme details of Virtual Auditing and Secure Deduplication with Dynamic Ownership Management in Cloud (VASD²OM) has been discussed. Section 7, discusses on Key updation, Data deletion and Data modification. Section 8 presents Security analysis while performance evaluation results are analyzed in Section 9. Conclusions are presented in Section 10.

II. RELATED WORKS

In this section we review the recent works on both integrity auditing and secure deduplication.

D. Integrity Auditing

Zhu *et al.,* [9] created the collaborative *PDP* in multi-cloud repository and Proof of Retrievability (*PoR*) [10] mechanisms that supports integrity verification. In contrast with *PDP, PoR* not only satisfies the distributed servers by retaining the intended documents and also affirms their complete reconstruction. In order to comprehend adequate information dynamics, Wang *et al.,* [11] elaborated the *PoR* scheme by making use of the Merkle Hash Tree generation for chunk label confirmation. Most powerful label producing mechanism [12], enhances protected and adequate label construction time. The advantage of the mechanism is that it minimizes the cost of the information proprietor. Li *et al.,* [13] designed a state-of-the-art distributed repository system with two separate distributed servers for sincerity verification to minimize the estimation cost at the customer side. Raghavendra *et al.,* [14] have suggested a guaranteed multi-proprietor information distribution for dynamic group in the cloud. The advantage of the scheme is that it has reduced depository and computation cost while it does not support multi-media documents. Jiang *et al.,* [15] introduced a dynamic public honesty validation mechanism with defended group customer repudiation. The proposed scheme supports the cluster information encoding and decoding at the time of data repair process and accomplish adept and safe customer renouncement.

TABLE I. COMPARISON OF INTEGRITY AUDITING AND DEDUPLICATION SCHEMES IN CLOUD

Authors	Concept	Performance	Advantages	Disadvantages
Jin and Zhou *et al.*, 2018 [19]	Effective and public verification with appealing adjugation for cloud information.	The scheme provides fairness assurance and controversy adjugation.	The mechanism supports public auditing, efficient data dynamics	Scheme introduces overhead for dynamic update and dispute adjugation.
Razaque *et al.*, 2017 [17]	Data confidentiality conserving Framework provides the verifying ability to all the key participants (cloud users, CSP and TPA).	The functional proficiency and authenticity of the CSPs is accomplished. .	Handles the negative role of the TPA	The efficacy of CSP is decreased
Shen *et al.*, 2017 [18]	Global and sampling verification is used to realize collaborative trust between DO′s and CSP's.	The scheme is less computationally expensive and accomplishes efficient batch auditing.	Supports global and sampling blockless verification	More communication cost.
Li and Jin, 2016 [8]	Secure verification and deduplication of information in cloud.	More auditing time cost.	Secure deduplication on encrypted data.	Increased time cost response.
Wu *et al.*, 2018 [20]	Differentiated caching mechanism to enable primary storage deduplication in cloud.	Deduplication workload is reduced in the post-processing deduplication phase.	Achieves high inline cache efficiency.	The scheme has more computational overhead.
Zheng *et al.*, 2017 [21]	Encoded cloud media center with guaranteed deduplication	Scheme incurs little storage overhead.	Reliable deduplication while firmly preserves the video privacy.	Does not support media files with scalable structures.
He *et al.*, 2016 [22]	Deduplicable robust proof of repository for multi-client environments.	Supports infinite number of verification and update operations.	Supports integrity verification, cross-client Deduplication with excellent consistency.	More communication cost.
Hur *et al.*, 2016 [23]	Cloud server controls access to deployed information though the proprietorship alters dynamically.	Low communication cost	Avoids information leakage to the repudiated clients.	More computational overhead.
Proposed Scheme	Virtual auditing and secure deduplication with dynamic ownership management in cloud.	Virtual auditing is more efficient compared to the public auditing.	Information proprietor has been relieved from the burden of auditing the document and also there is no need for the dataowner to hire the TPA.	Download time is little more compared to the existing scheme. .

The limitation is that the scheme requires costly reckoning effort in the setup time. Mastering C++ is used to carry out simulations in C++ [16]. Razaque *et al.*, [17] proposed an isolated trilateral information secrecy-conserving framework to furnish the verifying proficiency to all the key stakeholders (cloud customers, CSP and TPA). The scheme affords a procedure to examine a TPA for reducing any possible insider assaults. The advantage of the scheme is that it successfully handles the contradictory role of the TPA while the performance of the CSP is poor.

Shen *et al.*, [18] proposed a public verifying mechanism with an innovative compelling framework. The scheme supports blockless validation and cluster verification. Disadvantage of the mechanism is that the transmission cost is more in verify phase. Jin and Zhou [19] proposed a public reviewing convention with public provability, adequate information dynamics and candid controversy negotiation. The scheme yields honesty assurance and controversy negotiation. The limitation is that the scheme introduces overhead for dynamic update and controversy negotiation.

E. Secure Deduplication

Data deduplication is a specific information confining approach for discarding identical documents of repetitious information in repository. The method is utilized to increase repository usage and decrease bandwidth utilization. Li et *al.*, [8] designed two reliable frameworks, *SecCloud and Sec*Cloud+ that achieves information forthrightness and deduplication in cloud. *SecCloud* uses *MapReduce* cloud that creates labels for the information and also validates the truthfulness of the information cached in the cloud. In *SecCloud*, the reckoning cost of the customer is significantly reduced. *SecCloud+* supports sincerity validation and protects deduplication on the encrypted data.

Chen *et al.*, [24] developed chunk-level message-locked encode mechanism. The protocol achieves record level and chunk level deduplication and chunk key administration. The scheme is not computationally productive. Stanek and Kencl [25] discuss about inherent pressure between repository regularizing methods and end-to-end encryption. The proposed mechanism counteracts deduplication of unpopular information. It is re-silient to customer conspiracy attacks and never deduplicates unpopular documents. The drawback of the mechanism is that it has low capacity-saving efficiency. Wu *et al.*, [20] proposed a discerned stockpiling scheme to allow primary repository deduplication in clouds. The scheme achieves inflated inline backup adeptness and decreases the deduplication workload. The scheme has more computational overhead. Zheng *et al.*, [21] outlined and achieved an encoded cloud media center accommodating encoded Scalable Video Coding *(SVC)* videos.

The advantage of the scheme is that it enables secure deduplication and preserves the video privacy. The disadvantage is that it incurs little storage overhead.

He *et al.,* [22] presents deduplicable dynamic confirmation of storage mechanism. An efficient mechanism, *DeyPoS* is constructed to achieve effective proof of repository and safe cross-user deduplication, simultaneously. The scheme supports integrity auditing and cross-customer deduplication. The limitation is that the transmission cost is high.

A comparison of Integrity Auditing and Deduplication schemes of recent works are shown in Table I.

III. BACKGROUND WORK

Hur *et al.,* [23] designed an original server-side deduplication mechanism for encoded information. It permits the distributed server to supervise admission to deployed information while the proprietorship alters dynamically by using arbitrary concurrent encryption and guaranteed proprietorship cluster key allocation. This avoids information exposure to renunciated customers though they already possessed the information, and also to cloud repository server. The disadvantage of the mechanism is that it has added computational overhead.

IV. PRELIMINARIES AND DEFINITION

The notations used in this paper are described below:
$l \leftarrow S$ is an operation that selects a random element l from a finite set S and assigns it to *l*. $k \leftarrow A\ (l_1\ ,\ .....)$ denotes executing *A* on inputs $(l_1\ ,\ .\ .\ .)$ and allocating the output to *k*. The security parameter 1^{λ} represents a string of λ ones if $\lambda \in N$. The concatenation of two bit-strings is c and d is denoted by *c* || *d.*

Let U = $(c_1,.... c_n)$ be the universe set of clients. Let Id_t be the identity of the client c_t. Let $G_i \subset U$ be a set of clients that possess the document D_i, that is called as a proprietorship cluster. Let $L_i = (T_i\ ,G_i)$ be a proprietorship list for D_i, retained by the distributed server, that contains label T_i and G_i for D_i. Let K_{Gi} be the proprietorship cluster key that is distributed among the genuine proprietors in G_i [23].

We define the scheme Virtual Auditing and Secure Deduplication with Dynamic Ownership Management (*VASD²OM*) in Cloud. The mechanism comprises of the following functions:

1) *UK←UKGen(U):* The *UK* construction function accepts a set of clients *U* as input, and outputs *UK's* for every client in *U* for safe proprietorship cluster key dissemination.

2) *C←FileUpload(D, 1^{λ}):* This is a file uploading convention that receives as input document *D* and *λ,* and outputs a ciphertext *C* of the document. *C* contains encoded document and its label. The information proprietor transmits *C* with its label to the CSP.

3) *C′ ← ReEncode(C, G):* The re-encode function accepts a ciphertext *C* and a proprietorship cluster *G*, and outputs a re-encoded ciphertext *C′.*

4) *VA ← VAudit(C′,G):* The virtual audit function receives as input re-encoded ciphertext and a proprietorship cluster *G*, and outputs Virtual Audit Report *(V AR)* that consists of re-encrypted ciphertext *C′* attached with audit report of the document.

5) *D←Decrypt(VAR, K, PK):* The decryption function that accepts as input *VAR*, document encode key *K*, and a set of *U K's PK* for encoding a proprietorship cluster key *GK*, and outputs a document *D*, *iff K* is derived from *D* and *GK* is not repudiated from the proprietorship cluster *G* (i.e., the decryptor is in *G*) for *D.*

V. PROBLEM DEFINITION AND SYSTEM MODEL

F. Problem Definition

Given that the original document owner and subsequent document owners upload their respective documents to the cloud, the main objectives are:

1) Secure deduplication of encoded information with dynamic proprietorship administration capability.

2) Efficient auditing of owners file virtually by the virtual auditing agent during download process.

G. System Model

We propose Virtual Auditing and Secure Deduplication with Dynamic Ownership Management in Cloud (VASD² OM). The cloud storage framework (as shown in Fig. 1.) contains three objects:

1) *Information Proprietor:* The Information Proprietor is a customer who owns the information, and outsources it to the distributed storage to save costs. The information proprietor encodes the information and deploys it to the distributed repository with its label information. The original uploader is an information proprietor who deploys information that do not formerly prevails in the distributed repository. Other proprietors might have uploaded the same information earlier; they are called as consequent uploader. Eventually, we refer to a set of information proprietors who allocates the identical information in the distributed repository as a proprietorship cluster. When customers who have genuine proprietorships demand the distributed server to remove or alter the information in the distributed repository, they are repudiated from the proprietorship list and prevented from retrieving the information after erasing the information or alteration such that forward secrecy is accomplished.

2) *Cloud Service Provider:* This is an object that offers distributed repository utilities. It comprises of a distributed server and distributed repository. The distributed server deduplicates the deployed information from customers if required and saves in the distributed repository. The distributed server stores proprietorship lists for deployed information that consists of label for the cached information and the identities of its proprietor. The distributed server manages access to the stockpiled information de-pending on the proprietorship checklist and administers cluster keys for each proprietorship cluster. The distributed server is presumed to be honest-but-curious i.e., it sincerely performs the given jobs in the framework; yet, it would like to acquire knowledge about the ciphertext contents. Thus, the CSP is prevented from retrieving the plaintext of the encoded information though it is genuine.

3) *Virtual Auditing Entity:* This is an inbuilt auditing framework constructed by the information proprietor.
Every time when the information proprietor uploads the document, automatically the metadata information of the document is sent to the virtual auditing framework i.e., the virtual auditing entity consists of the metadata information of all the documents uploaded to the distributed server. When the information proprietor sends the download request for his document, the CSP sends the requested document to the *V AE*. The virtual auditing framework performs auditing of this document and sends the document attached with the auditing report to the information proprietor. The information proprietor receives the requested encrypted document along with auditing report. Hence, in the proposed scheme, the information proprietor has been relieved from the responsibility of verifying the document and also there is no need for the data owner to hire the Third Party Auditor (TPA).

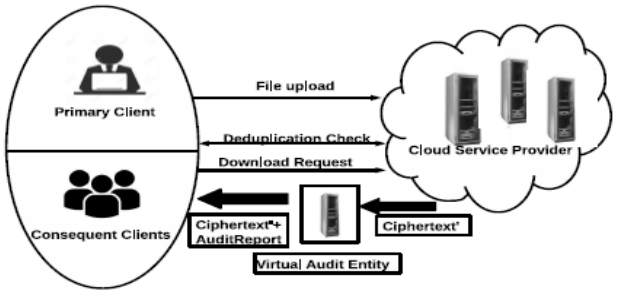


Fig. 1. Cloud Storage Virtual Auditing Model.

VI. THE ALGORITHM

Let $E_K(D)$ be a symmetric encryption of a document D utilizing a key K. A cryptographic hash function H: {0, 1}* → {0,1}* is used to create an encode key and a label from the document.

H. *UK* tree for proprietorship cluster key distribution

The CSP executes *UKGen(U)* [23] and creates *UKs* for customers in *U*. Initially, the CSP sets a binary *UK* tree for the universe of customers *U*, as in Fig. 2, that is utilized to share the proprietorship cluster keys to customers in U $\subset U$. In the tree, every node q_j possess a *UK*, represented by UK_j. customer is indicated by a leaf, and every customer preserves the *UKs* on the path nodes from its leaf to the root and is called as path keys.

For e.g., in Fig. 2, q_2 caches UK_9 ,UK_4 , UK_2 , and UK_1 as its path keys PK_2. For $q_t \in U$, PK_t represents a set of the path keys of q_t.

Function:KeyGeneration

1) Each client in *U* is allocated to a leaf node of the *UK* tree. Arbitrary keys are created and allocated to every leaf node and internal node.
2) Every client $c_t \in U$ accepts safely the path keys PK_t from its leaf node to the root node of the tree.
3) CSP generates the path keys and utilizes *UKs* to encode the proprietorship cluster keys.

I. File uploading:

Assume a data owner c_t deploys a document D_i to the distributed server. The data owner c_t encodes the document D_i as explained in Algorithm I, Phase I. Data owner c_t estimates a key $K_i \leftarrow H(D_i)$ from D_i and then encodes D_i and K_i and constructs the ciphertext C_i . Next, c_t deploys (T_i , C_i , ID_t) to the CSP. Then, c_t deletes document D_i and keeps only key K_i for repository saving. The CSP receives the ciphertext of the document, and checks for the deduplication. If the document D_i is an original document, the server inserts ID_t into the proprietorship group G_i , generates $L_i = (T_i, G_i)$, and saves C_i in the CSP. This uploader is called as a primary client. If the document is a duplicate, then the CSP executes *PoW* convention with

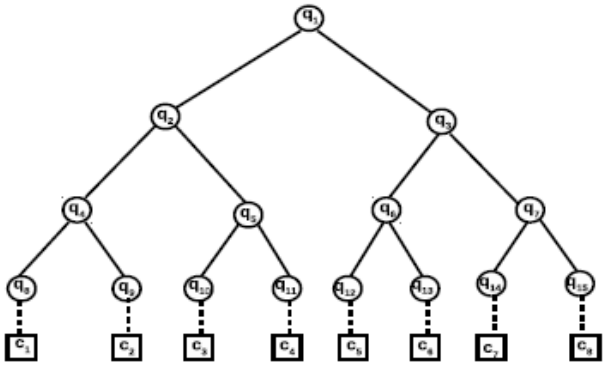


Fig. 2. UK tree for ownership group key distribution.

the data owner. When the data owner proves that he is an authorized person, then the CSP provides a link for the file existing in its server. Next, the uploader is added to the consequent clients group.

J. Data Re-encryption:

Before issuing *Ci*, the distributed server re-encodes it by executing ReEncrypt(C_i,G_i) utilizing the proprietorship cluster information [See Algorithm I, Phase II]. The re-encode function imposes admission control of actively altering proprietor to the deployed information. CSP selects a random proprietorship cluster key GK_i for the proprietorship cluster G_i . Further, CSP re-encrypts $C^1{}_i$ and creates $(C^1{}_i)' = E_{GKi} (C^1{}_i)$. The CSP chooses root nodes of the minimum cover sets in the *UK* tree that covers all of the leaf nodes accompanying with customers in G_i. For example, if $G_i = c_1 , c_2 , c_3 , c_4 , c_7 , c_8$ in Fig. 2, then $UK(G_i) = UK_2$, $U K_7$, because q_2 and q_7 are the root nodes of the minimum cover sets that can cover all of the customers in G_i . Any customer $\boldsymbol{c \notin G_i}$ can by no means learn any *UK* in $UK(G_i)$. CSP constructs $C^3{}_i = \{E_K (GK_i)\}$ where $K \in UK(G_i)$ and transmits the proprietorship cluster keys to genuine customers. On accepting any document request query (T_i , ID_j) from a customer c_j, the distributed server looks up L_i and acknowledges with $(T_i , C'{}_i)$ to the customer, where $C_i' = ((C^1{}_i)', C^2{}_i , C^3{}_i)$ if $ID_j \in G_i$ otherwise, it does nothing.

K. Virtual Auditing

The data owner c_t sends a document request *query* (T_i ,ID_j) to the CSP. Then, the CSP sends a response C_i' to the inbuilt framework, the Virtual Auditing Entity (*VAE*). The *VAE* possesses the metadata information and public keys of the clients, and performs auditing, if ($T'_i \neq T_i$), then *VAE* sends auditing report that the document has been modified else the document is correct [See Algorithm I, Phase III]. *VAE* can audit efficiently any number of documents at the same time. *VAE* sends the ciphertext C'_i appended with the auditing report {C'_i, *VAR*} to the data owner c_t.

L. Data Decryption:

When a customer c_t accepts a ciphertext C_i' from the distributed server, he can decode the document by executing Decrypt(C_i, K_i, PK_t), if $c_t \in G_i$. This phase consists of the following two steps:

Step I: Ownership Group Key Decryption

When the data owner c_t forwards a document request inquiry and accepts (T_i, C_i') from the CSP, c_t first analyzes it as (T_i, C^1_i, C^2_i, C^3_i). The data owner, c_t acquires the proprietorship cluster key from C^3_i. If the data owner c_t has genuine proprietorship, then c_t can decrypt the proprietorship cluster key GK_i as $GK_i = D_{UK}(C^3_i)$ where $UK \in UK(G_i) \cap PK_t$). Any customer $c \notin G_i$ can by no means decode GK_i, even if he conspires with other customers $c' \notin G_i$, that makes the proposed mechanism secure against such a conspiracy assault.

Step II: Message Decrypt

The data owner c_t accepts the ciphertext appended with the *VAR*. Then data owner c_t parses (C_i,*VAR*). The data owner c_t decodes the ciphertext and retrieves the message as follows: $C^1_i \leftarrow D_{GKi}((C^1_i)')$, $L \leftarrow C^2_i \oplus K_i$, $D_i \leftarrow D_L(C^1_i)$, $T'_i \leftarrow H(D_i)$.

The customer c_t obtains the plaintext D_i along with the *VAR* [*See Algorithm 1, Phase IV*].

Summary of the notations used in the Algorithm 1 is illustrated in Table II.

VII. KEY UPDATION, DATA DELETION AND DATA MODIFICATION

M. Key Updation:

Suppose consequent customers are trying to outsource document that is same as the formerly outsourced document by the original uploader, the equivalent ciphertext existing in the CSP need to be re-encrypted to

Algorithm 1: VASD²OM: Virtual Auditing and Secure Deduplication with Dynamic Ownership Management in Cloud.

Input: D_i, L, c_t, T_i, ID_i, G_i

Output: K_i, C^1_i, C^2_i, C^3_i, C'_i, $(C^1_i)'$, C_i, *VAR*

Phase I: File uploading:

- Suppose a data owner c_t outsource document D_i to the cloud server. c_t encrypts the document by executing the Encrypt(D_i, 1^λ) algorithm.
- c_t estimates a key $K_i \leftarrow H(D_i)$ from D_i. Then, c_t encrypts D_i and K_i as $C^1_i \leftarrow E_L(D_i)$ and $C^2_i \leftarrow L \oplus K_i$ and constructs the ciphertext ($C_i = C^1_i, C^2_i$).
- Next, c_t uploads (T_i, C_i, ID_t) to the CSP. Upon receiving the C_i, the CSP checks for the deduplication.
- If D_i is the original file, then the CSP saves the file in the server. Then, the uploader is called as a primary client.
- Otherwise, the CSP executes PoW convention with the data owner. When data owner proves that he is an authorized person, then the CSP provides a link for the document existing in its server. The uploader is added to the consequent clients group.

Phase II: Data Re-encryption:

- For G_i, CSP chooses an arbitrary proprietorship cluster key GK_i . Then, re-encrypts C^1_i and creates $(C^1_i)' = E_{GKi}(C^1_i)$
- Then CSP chooses root nodes of the minimum cover sets in the *UK* tree that can cover all of the leaf nodes corresponding with the customers in G_i.
- Any customer $c \notin G_i$ can by no means know any *UK* in $UK(G_i)$.
- CSP constructs $C^3_i = E_K(GK_i)$ where $K \in UK(G_i)$ and transmits the proprietorship cluster keys to genuine customers.

Phase III: Virtual Auditing:

1) c_t transmits a document request query (T_i, ID_j) to the CSP.

2) CSP sends C_i' to the VAE. The VAE possesses public keys of the customers, and performs auditing, if ($T'_i \neq T_i$), then VAE transmits auditing report that the document has been modified else the document is correct.

3) VAE can audit efficiently any number of documents at the same time.

4) VAE sends the ciphertext C_i' appended with the auditing report (C_i, V AR) to c_t.

Phase IV:Data Decryption:

This phase consists of the following two steps:

Step I: Proprietorship Cluster Key Decrypt:

- When c_t transmits a document request query and accepts (T_i, Ci') from the CSP, ct first parses it as (T_i, C^1_i, C^2_i, C^3_i).
- c_t acquires the GK_i from C^3_i. If c_t has genuine proprietorship, then c_t can decrypt the GK_i as $GK_i = D_{UK}(C^3_i)$ where $UK \in (UK(G_i) \cap PK_t)$.

StepII: Message Decrypt:

- The customer c_t receives the ciphertext appended with the *VAR*. Then c_t parses Ci' ,*VAR*.
- *ct* decodes the ciphertext and acquire the message as follows:
- $C^1i \leftarrow DGKi(C^1_i)', L \leftarrow C^2_i \oplus Ki$, $Di \leftarrow DL(C^1_i)$, $T'_i \leftarrow H(Di)$
- Then, c_t obtains the plaintext *Di* with *the VAR*.

Table Ii. Summary Of The Notations Used In The Algorithm 1

Notation	Description
L	Random document encryption key.
D_i	i^{th} document
c_t	t^{th} client or data owner.
UK	Keys generated for all the clients in U.
K_i	Key used as UK for encrypting L

T_i	Tag of the document D_i
C_i	Ciphertext of i^{th} document
id_t	Identity of t^{th} client.
L_i	Proprietorship list
G_i	i^{th} Proprietorship group
UL	Customer list
GK_i	Proprietorship group key of i^{th} group.
id_k	k^{th} block identifier
PK_t	Path key of t^{th} path in UK tree.
VAE	Virtual Auditing Entity.
VAR	Virtual Auditing Report
C^1i	Ciphertext of the i^{th} document encrypted with key L.
C^2i	Ciphertext of key $_{Ki}$ encrypted with L.
C^3i	Ciphertext of the Ownership group Key GK_i encrypted with Key UK.
$C'i$	Ciphertext of the i^{th} document requested by the valid client in the proprietorship group.
$(C^1i)'$	Ciphertext of the document obtained after re-encryption of C^1i

prohibit consequent customers from retrieving the earlier encoded information so that backward secrecy is achieved. However, when customers who have legitimate proprietorship demands the distributed server to erase or update the information in the distributed repository, they ought to be repudiated from the proprietorship checklist and prevented from retrieving the information after information removal or transformation so that forward secrecy is accomplished.

Let us assume that a customer c_t encodes the data by executing the *Encrypt*(D_i, 1^{λ}) algorithm and produces ciphertext, $(C^1_i)''$ and deploys $(T_i, (C^1_i)'', ID_t)$ to the CSP. Its equivalent proprietorship list and ciphertext already prevails in the server. The ciphertext $(C'i)''$ created by c_t is not equal to the ciphertext that is already existing in the server because the encode key L is arbitrarily chosen from $\{0,1\}^{k\lambda}$ by distinct customers. Next the key updation and re-encryption processes in the CSP progress as follows:

1. If the label of the document T_i outsourced by the consequent customer is equal to the label of the document T_i outsourced by the original uploader, then CSP adds identity (ID_t) to the proprietorship cluster G_i.
2. CSP encodes the ciphertext component $C^1_i = E_{GKi}(C^1_i)$ in C_i with the prevailing proprietorship cluster key GK_i. Then it selects an arbitrary proprietorship cluster key $GK'i(\neq GK_i)$ and executes the *ReEncrypt*(C_i,G_i) algorithm with the updated proprietorship cluster information G_i and GK_i to assure backward secrecy.

If the consequent customer deploys a new document, then he is considered as the original uploader and the CSP constructs a original proprietorship checklist for the information, inserts ID_t into the recently constructed proprietorship cluster and cache the uploaded information in the distributed server.

N.Data deletion:

When a customer c_t wants to delete document D_i from the distributed repository, the customer transmits the document deletion request messages with *(delete, T_i, ID_s)* to the CSP. Then, the CSP carry out the following strategy:

If $ID_t \in G_i$, it deletes ID_t from G_i. It chooses a arbitrary proprietorship cluster key and executes the *ReEncrypt*(C_i, G_i) algorithm with the updated proprietorship cluster information G_i to assure forward secrecy otherwise it does nothing.

O. Data Modification:

When a customer c_t desires to update the document D_i to D_j , the customer encodes the altered document and produces the ciphertext C_j and its equivalent label T_j. Then, the customer transmits a document modification request message with *(modify, T_i, T_j , C_j ,ID_t)* to the CSP. Then, the CSP runs the information deletion function, followed by data upload function.

In the information deletion procedure, the CSP verifies whether the customer's $ID_t \in G_i$, it removes ID_t from G_i. Further, it chooses an arbitrary proprietorship cluster key and executes the *ReEncrypt*(C_i,G_i) with the updated proprietorship cluster information G_i for assuring forward secrecy. For the label T_j of the document D_j; if there exists $L_j = (T_j,G_i)$, then CSP considers it as a consequent upload otherwise if L_j does not exist in the server then it considers as an original upload.

VIII. SECURITY ANALYSIS

In this section, we prove the security of the proposed mechanism in terms of information secrecy, information sincerity, backward and forward secrecy, and conspiracy resistance.

P. Information Secrecy:

In our system architecture, even though the CSP is truthful, he is curious about the information deployed by the customers. Hence, the information need to be kept confidential from the CSP as well as from illegitimate customers who cannot authenticate proprietorship. The information secrecy for the stockpiled data against illegitimate customers who have never possessed the information can be trivially confirmed. Let us assume that an illegitimate customer demands a document namely D_i with a label T_i and conspires with one of the prevailing customer and transmits the identity $ID_t \in G_i$ (if $ID_t \notin G_i$, the CSP would easily deny the request), and accepts the equivalent ciphertext $(T_i, ((C^1_i)', C^2_i , C^3_i)$ here $C^1_i = E_{GKi}(E_L(D_i))$ and $C_i = L \oplus K_i$. If the customer had never possessed the document D_i, it is computationally impossible to predict K_i and acquire the information encode key L.

On the contrary, if the customer had once possessed the document but does not have genuine proprietorship during request time, the customer may have information of K_i but can by no means acquire the equivalent GK_i from C^3i. Thus, the illegitimate customer cannot acquire D_i from C^1_i without GK_i. Further, the mischievous CSP might instigate an assault strategy.

The CSP can learn the set of ciphertext $C^1_i = (C^1_i, C^2_i)$ with the equivalent L_i. Although the CSP decides the proprietorship cluster key GK_i, it is computationally impossible for the CSP to predict K_i and hence without K_i, it cannot decode C^1_i and acquire D_i. As a result, information secrecy against the honest-but-curious CSP and illegitimate customers is assured.

Q. Information Integrity:

In the proposed scheme, the integrity of the outsourced data is securely and efficiently verified by the *VAE*. The information proprietor constructs and deploys the *VAE*, such that the *VAE* securely and efficiently audits the integrity of the documents during the download process. When the information proprietor sends a document request query *(T i, IDj)* to the CSP, then the CSP sends a response C_i' to the inbuilt framework, the Virtual Auditing Entity (*VAE*). The *VAE* possesses the metadata information and public keys of the clients, and performs auditing, if ($T_i' \neq T_i$), then *VAE* sends auditing report that the document has been modified else the document is correct. *VAE* sends the ciphertext C_i' appended with the auditing report {C_i', *VAR*} to the data owner c_t. Thus the information integrity has been achieved where the information proprietor's worrying factor about the correctness of their outsourced documents has been eliminated and there is no need for the information proprietor to hire a Third Party Auditor (TPA).

R.Backward and Forward Secrecy:

When a customer possess a document that has been earlier outsourced at some time instance and attempts to outsource them to the CSP, the equivalent proprietorship cluster key is revised separately and arbitrarily, say from *GK* to *GK′*, and transferred safely to the genuine proprietors of the information (including the customer) instantaneously. Further, the encoded document $(C^1)' = E_{GK}(C^1)$, that was encoded with *GK*, is re-encoded by the CSP with the revised proprietorship cluster key *GK′* as $C^1 = E_{GK'}(C^1)$ though the customer has cached the former ciphertext before he possess the document, and cannot decode the former ciphertext. Thus, the backward safety of the deployed information is assured.

However, when a customer erase or update the document, the equivalent proprietorship cluster key is also modified separately and arbitrarily, say from *GK* to GK′, and transferred safely to the genuine proprietorship cluster users (excluding the customer) instantaneously. Then, the ciphertext document $(C^1)' = E_{GK}(C^1)$ that was encoded with *GK* is re-encoded by the CSP with a original proprietorship cluster key *GK′* as $(C^1)'' = E_{GK'}(C^1)$. Then, the customer cannot decode the existing ciphertext after his repudiation, because he cannot at all acquire *GK′*. Though the customer has regained and saved the information encode key *L* before he was repudiated from the proprietorship cluster, it would be of any use for acquiring the desired document from C^1 in the consequent ciphertext, as it is re-encoded with a latest arbitrary *GK′*. Thus, the forward secrecy of the deployed information is also assured in the proposed mechanism.

S. Conspiracy Resistance:

To accomplish conspiracy resistance, illegitimate customers who have not genuine proprietorship of cloud information are incapable to decode them though they conspire. In the proposed mechanism, with the purpose of decoding the ciphertext and acquire the plaintext, the customers need to know both the information encode key *L* and the proprietorship cluster key *GK*. Though, few illegitimate customers may acquire the information encode key, it is not possible to acquire the proprietorship cluster key *GK*. Whenever any proprietorship change happens in a proprietorship cluster, the proprietorship cluster key is re-keyed instantaneously and the documents are re-encoded utilizing the modified cluster key. Though the illegitimate customers conspire with one another, they are unable to acquire the latest proprietorship cluster key, because neither of the *UKs* in their path keys in the *UK* tree is utilized to encode and assign the proprietorship cluster key. Thus, the proposed mechanism is safe against a conspiracy assault of the illegitimate customers.

IX. PERFORMANCE ANALYSIS

In this section, we discuss an experimental analysis of our mechanism. We have used Intel(*R*) Core(*TM*) *i5*-5200U, CPU @2.20GHz, 8GB RAM. Each cryptographic operation was implemented utilizing the *Crypto++* library *ver*. 5.6.2 [26] on a 3.4 GHz processor PC. The key parameters were selected to provide a 28-bit security level. The implementation utilizes an *MD5* as a cryptographic hash function to construct a 128-bit key and label, and an *AES* with Electronic Code Book (*ECB*) mode as an encode/decode function. In this simulation, we set the size of the document as $S_D = S_C = 10$MB. To realize a 128-bit security level, we set $S_K = 128$ bits, $S_T = 128$ bits Let *n* be the number of customers in the system and m be the number of proprietors in a proprietorship checklist for a document.

Fig. 3 shows the impact of *t* on average auditing time (ms) per task where *n*=10. For every individual customer auditing request the Third Party Auditor (TPA) [27], [28] performs auditing and the time taken by the TPA [i.e., public auditing] is more as compared to virtual auditing performed by the Virtual Auditing Entity (*VAE*) in the proposed scheme. When the customer or data owner transmits the download request of the document to the CSP, the CSP transmits the ciphertext of the file to the *VAE*. Upon receiving the ciphertext of the requested document, the *VAE*, performs auditing with the metadata information and sends the ciphertext appended with the *VAR* to the dataowner. Hence, in the proposed scheme the time taken by the Virtual Auditing Entity *VAE* is less as compared to the TPA (public auditing). In public auditing, the TPA uses pairing operations which takes more time to audit while the *VAE* audits the file by comparing the signatures of the file which takes less time. , it is observed in Fig.3, the time

TABLE III. Comparison of virtual auditing with individual auditing performed by TPA.

Total number auditing tasks where n=10	Average auditing time (ms)		
	Panda	Virtual Audit	DHT-PA
5	280	206	801
10	285	210	803
15	290	218	806
20	295	226	808
25	300	232	810
30	305	237	813
35	310	243	816
40	315	250	819

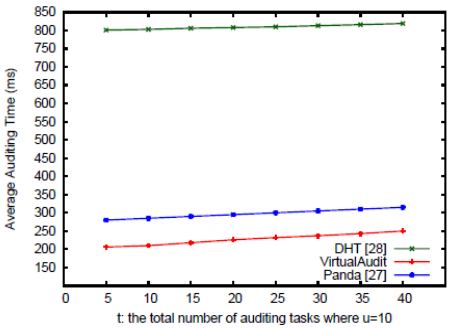


Fig. 3. Comparison of virtual auditing with individual auditing performed by TPA.

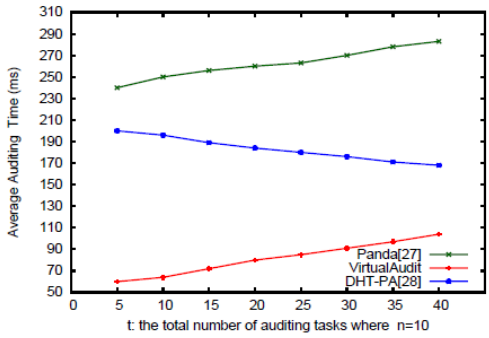


Fig. 4. Comparison of virtual auditing with batch auditing performed by TPA.

TABLE IV. Comparison of virtual auditing with batch auditing performed by TPA.

Total number auditing tasks where n=10	Average auditing time (ms)		
	Panda	Virtual Audit	DHT-PA
5	240	60	200
10	250	64	196
15	256	72	189
20	260	80	184
25	263	85	180
30	270	91	176
35	278	97	171
40	283	104	168

TABLE V. Computation time for download.

File size (MB)	Download time (ms)	
	Hur scheme	VASD^{2}OM
2	6.23	8.46
4	14.1	16.42
6	21.9	25.87
8	25.82	30.87
10	31.55	35.56

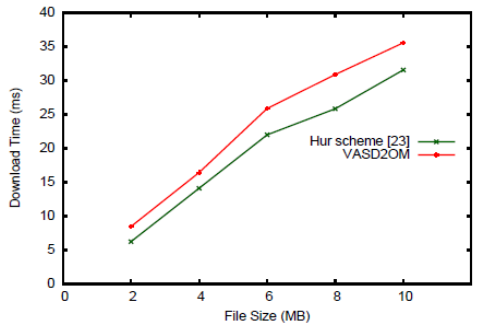


Fig. 5. Computation time for download.

taken by the TPA is more compared to the virtual auditing.

Fig. 4, shows Impact of *t* on average auditing time (*ms*) per task where *n*=10 for batch auditing. When a certain amount of auditing requests arrives in a very short time, our mechanism supports batch auditing where the *VAE* performs auditing of the number of auditing requests simultaneously and gives the auditing report to the data owners. When the data owner's requests for the file download, the CSP sends the respective ciphertext to the *VAE*. Since, the *VAE* is an inbuilt framework by the data owner, *VAE* efficiently and securely performs auditing of all the requested files simultaneously. In Fig. 4, it is observed that the time taken by virtual auditing is lower when compared to the public auditing schemes in [27], [28].

Fig. 5, shows the time taken by the dataowner to download the document in the *VASD^{2}OM* scheme is little more compared to the prevailing scheme [23]. In the *VASD^{2}OM* scheme, an inbuilt *VAE* is constructed where virtual auditing takes place during the download process. The data owner downloads the ciphertext text appended with the auditing report. Hence, the proposed scheme relieves the burden of the data owner to audit the document by himself and there is no need to delegate the auditing of the files to the public verifier.

X. CONCLUSIONS

In this paper, we introduce Virtual Auditing and Secure Deduplication with Dynamic Ownership Management in Cloud (*VASD^{2}OM*) mechanism. The proposed mechanism presents a re-encode method that permits dynamic updates upon any proprietorship changes in the distributed repository and supports efficient auditing of data owners file virtually by the Virtual Auditing Entity (*VAE*) during download process. Whenever a proprietorship transition happens in the proprietorship cluster of deployed information, the information are re-encoded with an instantly revised proprietorship cluster key that is safely forwarded only to the genuine proprietors. Thus, the proposed mechanism improves information secrecy in distributed repository against any customers who do not have genuine proprietorship of the information, as well as against the distributed server. The proposed mechanism has an inbuilt Virtual Auditing Entity (*VAE*) that performs secure and efficient auditing of the requested documents efficiently during the download process. In the proposed scheme, information proprietor has been relieved from the burden of auditing the document and also there is no need for the data owner to hire the Third Party Auditor. The efficiency analysis results demonstrate that in the *VASD^{2}OM* mechanism the virtual auditing is more efficient compared to the public auditing schemes [27], [28] while the download time is almost similar compared to the existing scheme[23].

REFERENCES

1. K. R. Venugopal, E. E. Rajan, and P. S. Kumar, "Performance Analysis of Wavelength Converters in WDM Wavelength Routed Optical Networks", in Proceedings. Fifth International Conference on High Performance Computing (Cat. N 98EX238). IEEE, pp. 239-246, 1998.
2. K. R. Venugopal, E. E. Rajan, and P. S. Kumar, "Impact of Wavelen-gth Converters in

Wavelength Routed All-Optical Networks", Computer Communications, vol. 22, no.3, pp. 244–257, 1999.
3. "Dropbox, http://www.dropbox.com/."
4. "Google drive, http://drive.google.com."
5. W. K. Ng, Y. Wen, and H. Zhu, "Private Data Deduplication Protocols in Cloud Storage," in Proceedings of the 27th Annual ACM Symposium on Applied Computing. ACM, pp. 441–446, 2012.
6. J. R. Douceur, A. Adya, W. J. Bolosky, P. Simon, and M. Theimer, "Reclaiming Space from Duplicate Files in a Serverless Distributed File System," in Proceedings 22nd International Conference on Distributed Computing Systems. IEEE, pp. 617–624, 2002.
T.-Y. Youn, K.-Y. Chang, K.-H. Rhee, and S. U. Shin, "Efficient Client-Side Deduplication of Encrypted Data with Public Auditi-ng in Cloud Storage," IEEE Access, vol. 6, pp. 26578–26587, 2018.
7. J. Li, J. Li, D. Xie, and Z. Cai, "Secure Auditing and De-duplica ting Data in Cloud", IEEE Transactions on Computers vol. 65, no. 8, pp. 2386–2396, 2016.
8. Y. Zhu, H. Hu, G.-J. Ahn, and M. Yu, "Cooperative Provable Data Possession for Integrity Verification in Multicloud Stora- ge", IEEE Transactions on Parallel and Distributed Systems,vol. 23, no. 12, pp. 2231–2244, 2012.
9. H. Shacham and B. Waters, "Compact Proofs of Retrieva--bility", of Cryptology, vol. 26, no. 3, pp. 442–483, 2013.
10. Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing," Computer Security–ESORICS, pp. 355–370, 2009.
11. S. Raghavendra, C. M. Geeta, R. Buyya, K. R. Venugopal, S. S Iyengar, and L. M. Patnaik, "MSIGT: Most Significant Index Generation Technique for Cloud Environment," in Proceedin- gs of the Annual IEEE India Conference (INDICON), pp. 1–6, 2015.
12. J. Li, X. Tan, X. Chen, and D. S. Wong, "An Efficient Proof ofRetrievability with Public Auditing in Cloud Computing," in 5th International Conference on Intelligent Networking and Colla-borative Systems (INCoS),. IEEE, pp. 93–98, 2013.
13. S. Raghavendra, P. A. Doddabasappa, C. M. Geeta, R. Buyya, K. R. Venugopal, S. S. Iyengar, and L. M. Patnaik,"Secure Multi-Keywor-d Search and Multi-User Access Control over an Encrypted Cloud Data," International Journal of Information Processing, vol. 10, no. 2, pp. 51–61, 2016.
14. T. Jiang, X. Chen, and J. Ma, "Public Integrity Auditing for Shared Dynamic Cloud Data with Group User Revocation,"IEEE Transact- -ions on Computers, vol. 65, no. 8, pp. 2363– 2373, 2016.
15. K. R. Venugopal and R. Buyya, "Mastering C++," Tata McGr- -raw-Hill Education, 2013.
16. A. Razaque and S. S. Rizvi, "Privacy Preserving Model: A New Sc--heme for Auditing Cloud Stakeholders," Journal of Cloud Computing, vol. 6, no. 1, p. 7, 2017.
17. J. Shen, J. Shen, X. Chen, X. Huang, and W. Susilo, "An Efficient Public Auditing Protocol with Novel Dynamic Structure for Cloud Data," IEEE Transactions on Information Forensics and Security, vol. 12, no. 10, pp. 2402–2415, 2017.
18. H. Jin, H. Jiang, and K. Zhou, "Dynamic and Public Auditing ithFair Arbitration for Cloud Data," IEEE Transactions on Cloud Com--puting, vol. 6, no. 3, pp. 680–693, 2018.
19. H. Wu, C. Wang, Y. Fu, S. Sakr, K. Lu, and L. Zhu, "A Differentia--ted Caching Mechanism to Enable Primary Storage Deduplication in Clouds," IEEE Transactions on Parallel and Distributed Systems, vol. 29, no. 6, pp. 1202–1216, 2018.
20. Y. Zheng, X. Yuan, X. Wang, J. Jiang, C. Wang, and X. Gui,"Tow- -ard Encrypted Cloud Media Center with Secure Deduplication," IEEE Transactions on Multimedia, vol. 19, no.2, pp. 251–265, 2017.
21. K. He, J. Chen, R. Du, Q. Wu, G. Xue, and X. Zhang,DeyPoS:Ded--uplicatable Dynamic Proof of Storage for Multi-User Environments ," IEEE Transactions on Computers, vol. 65, no.12, pp. 3631–3645, 2016.
22. J. Hur, D. Koo, Y. Shin, and K. Kang, "Secure Data Deduplication with Dynamic Ownership Management in Cloud Storage", IEEE Transactions on Knowledge and Data Engineering, vol. 28, no. 11, pp. 3113–3125, 2016.
23. R. Chen, Y. Mu, G. Yang, and F. Guo, "BL-MLE: Block-Level -Message-Locked Encryption for Secure Large File deduplication," IEE-E Transactions on Information Forensics and Security, vol. 10, no. 12, pp. 2643–2652, 2015.
24. J. Stanek and L. Kencl, "Enhanced Secure Thresholded Data Dedu--plication Scheme for Cloud Storage," IEEE Transactions on Depe--ndable and Secure Computing, 2016.
25. "Crypto++ library 5.6.2, http://www.cryptopp.com/."
26. B. Wang, B. Li, and H. Li, "Panda: Public Auditing for Shared Datawith Efficient User Revocation in the Cloud," IEEE Transactions on Services Computing,, vol. 8, no. 1, pp. 92–106, 2015.
27. H. Tian, Y. Chen, C.-C. Chang, H. Jiang, Y. Huang, Y. Chen, and J. Liu, "Dynamic-Hash-Table Based Public Auditing for Secure Cloud Storage," IEEE Transactions on Services Computing, vol. 10, no. 5, pp. 701–714, 2017.

AUTHORS PROFILE

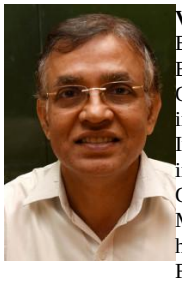
Geeta C M is a research scholar in the department of Computer Science and Engineering, University Visvesvaraya College of Engineering, Bangalore University, Bengaluru. She has received B.E. degree in Electronics and Communication from Basaveshwara College of Engineering, Bagalkot, Karnataka, India and M.E degree in Information Technology, from Bangalore University, Bengaluru, Karnataka, India. Her areas of interest are Cloud Computing and Wireless Sensor networks. She is a student member of the IEEE.

Nikhil R C is a UG student in the department of Aerospace Engineering, International Institute for Aerospace Engineering and Management, Jain University, Bengaluru, India. He is currently pursuing B. Tech degree in Aerospace Engineering. He has carried out projects under Ministry of Defence organization such as DRDO and CSIR-NAL during the course of his bachelor's degree. His areas of interest are Cloud Computing, Aerodynamics, Control Systems and Navigation Systems.

Raghavendra S is an Associate Professor in the department of Computer Science and Engineering, Vivekananda College of Engineering and Technology, Puttur. He received his Bachelor degree in Computer Science and Engineering from BMS Institute of Technology, Visvesvaraya Technological University, Bengaluru and Master degree from R V College of Engineering, Visvesvaraya Technological University, Bengaluru. Dr. Raghavendra S has authored over 20 publications and his research interests include Cloud Computing, Applied Cryptography and Internet of Things. He is serving as editorial board member and Guest editor for a number of prestigious journals, like Elsevier, Springer, KJIP. He is a member of the IEEE.

Rajkumar Buyya is a Redmond Barry Distinguished Professor and Director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne, Australia. He has authored over 625 publications and seven text books including Mastering Cloud Computing published by McGraw Hill, China Machine Press, and Morgan Kaufmann for Indian, Chinese and international markets respectively. He is one of the highly cited authors in Computer Science and Software Engineering worldwide (h-index=121, 78,000+citations). Software technologies for Cloud Computing developed under his leadership have gained rapid acceptance and are in use at several academic institutions and commercial enterprises in 40 countries around the world.

Dr. Buyya is recognized as a Web of Science Highly Cited Researcher in 2016, 2017 and 2018 by Thomson Reuters, a Fellow of IEEE, and Scopus Researcher of the Year 2017 with Excellence in Innovative Research Award by Elsevier for his outstanding contributions to Cloud Computing.



Venugopal K R is currently the Vice Chancellor, Bangalore University, Bengaluru. He obtained his Bachelor of Engineering from University Visvesvaraya College of Engineering. He received his Masters degree in Computer Science and Automation from Indian Institute of Science Bangalore. He was awarded Ph.D in Economics from Bangalore University and Ph.D in Computer Science from Indian Institute of Technology, Madras. He has a distinguished academic career and has degrees in Electronics, Economics, Law, Business Finance, Public Relations, Communications, Industrial Relations, Computer Science and Journalism. He has authored and edited 64 books on Computer Science and Economics, which include Petrodollar and the World Economy, C Aptitude, Mastering C, Microprocessor Programming, Mastering C++ and Digital Circuits and Systems etc., He has filed 101 patents. During his three decades of service at UVCE he has over 640 research papers to his credit. His research interests include Computer Networks, Wireless Sensor Networks, Parallel and Distributed Systems, Digital Signal Processing and Data Mining. He is a Fellow of IEEE and ACM Distinguished Educator.