International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. #, No.# (Mon-20..)

An Efficient Secure Auditing Framework for Big Data Storage in Cloud Computing Environment

Sameen Fatima[1], Dr. Shafiq Hussain[2], and Rana Abu Bakar[3]

[1] Department of Computer Science & Engineering, University of Engineering & Technology, Lahore, Pakistan
[2]Department of Computing Sciences, University of Sahiwal, Sahiwal, Pakistan
[3]Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Thailand

E-mail address: sameenfatima35@gmail.com, drshafiq@uosahiwal.edu.pk, ranaabubakar@ieee.org

Received ## Mon. 20##, Revised ## Mon. 20##, Accepted ## Mon. 20##, Published ## Mon. 20##

Abstract: Cloud storage enables to use and manage remote data efficiently but increase the risk of tampering data. This proposed paper a public auditing-based framework to develop a novel system for secure data auditing in cloud storage. We build a dynamic index table with no requirement to transfer elements in the update operation for inserting or deleting. If data is not incorporate in the cloud, the auditor from a third party can identify the corrupted block. The authorization is implemented between cloud storage and third-party to prevent dos attack. The proposed model is flexible, efficient, and secure as per detailed analysis and simulation results.

Keywords: Cloud Computing, Big Data, Secure, Indexing, Auditing

3. INTRODUCTION

Cloud computing's progress is a rapidly growing platform for computing business and has many advantages, including large volume, less expensive, and high scalability. Most of users are eager to upload their massive amount of data for storage and analysis to cloud servers and delete their data from local servers whenever they want to. Despite cloud storage ease, storing data on cloud servers without original content can cause many security issues. Cloud servers also experience different malicious attacks and failures of hardware or software [1-3]. The CSP did not inform the clients about the existence of faults or attacks. Worse though, to save maintenance costs or cloud storage space, cloud service providers may abandon information that users do not or do not have access to [4][6]. Cloud Service Providers must also provide evidence to verify that data is appropriately stored on the cloud. In the cloud computing environment most challenging and significant problems is ensuring data protection on cloud servers. Data auditing methods may allow users to check the quality of the data they store without accessing it on remote cloud servers. The data audit method has two parts based on the position of the verifier:

- Private Auditing
- Public Auditing

Users checked the data integrity in private auditing methods [7][9] and this increases the overhead to consumers they can't afford to pay. Since public audit methods enable the audit process to be performed by any public verifier having a public key for the consumer. It usually involves an expert third-party auditor (TPA) to carry out the verification mission. For examining the data quality in the cloud, several auditing methods are implemented. Nonetheless, when the data is changed, these methods cannot verify which block is corrupt. Moreover, as data needs to be modified regularly, there is no reliable authenticated data system to achieve accurate auditing. However, it is important to introduce an effective cloud public auditing method for dynamic cloud storage. The following research proposes a novel dynamic-public auditing framework by implementing a novel data framework called the Dynamic Index Table (DIT). Without the adjustments of the elements, our method will attain dynamic updating via DIT. Our framework can also assess missing and corrupt block, when data integrity is unable to achieve. Following describe our contributions:

- We implement a dynamic public auditing process to verify the corrupt block.
- The Dynamic Index Table (DIT) is a critical authenticated data structure used to hold the block

engranaabubakar@gmail.com

properties to assist TPA during data auditing and modifying without transfer items.

- The security of the method which is proposed is proved and verified. The findings indicate that the proposed method is much more effective than others.

This whole paper is structured according to this. Section 2 contains the detailed literature. Section 3 elaborates the framework concept, the threat model, and the method's design objectives. Section 4 is preliminaries. The proposed method is presented in depth in Section 5. The Section 6 reveals the security analysis, and section 7 indicates the performance evaluation of the proposed method. Lastly, this paper is concluded in Section 8.

2. Related Work

To date, several standard public auditing methods are introduced to check out the data integrity, which is stored in an un-trusted environment. The first public audit method [10] implemented the Provable Data Possession in 2007. It enables any public verifier to verify the data quality devoid of recalling. After all, only the integrity of static data can be verified by this method. Later, they suggested another method [11] to inspect the dynamic data in the cloud servers depending on the PDP symmetric critical method. This model enables deletion and dynamic alteration operations, but it does not make it useful during insertion operations. An authenticated data structure is often implemented to increase the update performance. In his DPDP method, Tamassiaet al. [12] implemented a skip list authentication. Wang et al. [13] later introduced a complex, Merkle Hash Tree (MHT)-based public audit method. The device could perform complicated data operations, but it generates multiple processing and overhead communications during the process of verification. Method [14] implemented the Index Hash Table (IHT) stored in the method on the TPA side to assist with dynamic verification. It is more efficient compared to other systems in terms of production and communication costs. After all, an IHT is a sequential data structure in the update process and concluded in a decrease in the device's performance. In 2013, an index table (ITable) was introduced by Yang and Jia [15] to store the abstract block's details in each block. It is successful in refraining from a replay attack. All block tags after deletion or insertion need to be recalculated in the insert and delete operations as these block indexes shift. With useful verifiable, fine-grained updates, Liu et al. [16] put forward an approved Big Data audit method. In 2017, the author [17] subsequently implemented a cloud storage audit method based on Dynamic-Hash-Table. Gan et al. [18] developed a powerful and robust algebraic signature audit method for outsourced Big Data in 2018. The authors [19] have introduced storage for the cloud to look upon the big shared data. Pan et al. [20] introduced a method for integrity testing on mobile devices' IoT in 2020. In [41], Homomorphic message authentication code (MAC) and homomorphic Signature are combined to form a new auditing scheme, but assigning a key to the user is a problem. In [42], Merkle Hash Tree and B* tree are combined to form a new auditing scheme that includes complicated data operations. However, during the integrity testing process, all methods' data structures do not guarantee a replay attack. It is, therefore, necessary to establish a more efficient audit method to achieve complicated services for integrity verification. In Table 1, different methods are compared with each other:

Table 1: Contrast Between Different Schemes

Method	Dynamic Auditing	Batch Auditing	Authorized Auditing	Data Structure
[11]	✓	×	×	×
[12]	✓	×	✓	×
[13]	✓	✓	✓	×
[14]	✓	✓	✓	×
[15]	✓	✓	×	×
[16]	✓	×	✓	✓
[17]	✓	✓	✓	×
Proposed	✓	✓	✓	✓

Several different integrity verification systems have been placed everywhere now and have caused cloud computing to boost security. Since the data which is stored on the cloud for sharing will encounter many different privacy challenges.

Most studies reported that retain auditing protocols to avoid leakage of privacy [2],[21]-[28]. Simultaneously, lightweight methods [29]-[35] are being implemented to advance the IoT and Smart devices to meet the audit process's productivity needs. Simultaneously, lightweight methods [29]-[35] are being implemented with the advancement of the IoT and smart devices to meet the productivity needs of the audit process. Over the last few years, many methods have been proposed on attribute-based and identity-based encryption to explain data sharing with other registered clients in the cloud [36]-[42].

3. SYSTEM MODEL, SECURITY REQUIRENMENT & ARCHITECTURE

As shown in fig.1, we define the system model. In our model, there are four objects: client (USER), third-party auditor (TPA) and key generation center (KGC), and cloud service provider (CSP). The user creates and outsources vast quantities of information to cloud servers (CS), which are extremely capable of maintaining a customer's Data. CSP operates servers in the cloud and provides users with access everywhere with a connection on the Internet. TPA is a body approved by the user and has a lot of knowledge and check data integrity efficiently and accurately.

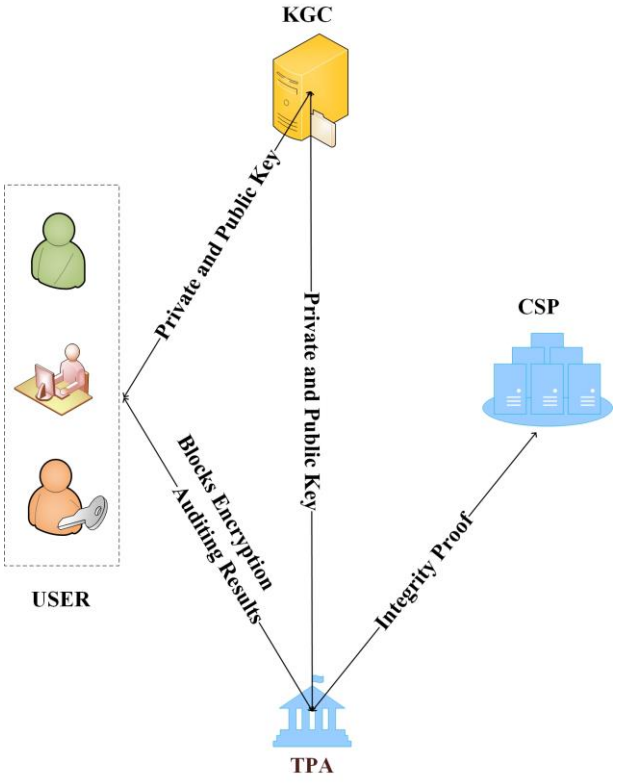

Figure 1: Illustration of Model

We assume that in our proposed method CSP and TPA are both in semi-trusted zone. Since he might be curious about user details, TPA is semi-trusted. The framework must maintain TPA's outsourced data protection. CSP is semi-trusted as it can initiate forge attack or replace TPA attack for economic reasons if any data is corrupt or lost on cloud servers.

Following are the security requirements:

Auditing in public: Publicly, TPA will check the credibility of the user's outsourced data.

Auditing authorization: To avoid a replay attack, just the approved TPA can initiate an auditing challenge.

Privacy of Data: In public auditing procedures, the content of data stored on cloud servers is not learned by TPA.

Unforgeability: The block tags for auditing can be generated only by the user.

Integrity of storage: Only if data blocks are stored correctly by CSS and the related block tags will integrity verification is accomplished.

Following properties should be achieved by our method:

Requirement of Security: Data Privacy, Authorization, and unforgeability could achieve by our method.

Lightweight Operations: The cost of communication and computations are reduced in our auditing method.

Effectiveness: User's authorization should effectively achieve the data auditing process.

4. PRELIMINARIES

The perliminaries of the proposed method is stated below

A. Notations:

Description of notations used in paper is represented in Table 2:

Table 2: Notations of Proposed Method

Notation	Meaning	Notation	Meaning
$\mathbb{G}1, \mathbb{G}2$	Multiplicative Group	F	Plaintext of file
e	Bilinear Map	M	Encrypted file
H, h, π	Secure Hash functions	mac_i	Encrypted blocks
p	Prime order of group	δ_i	Block tag
Gp	Generator of $\mathbb{G}1$	P	Integrity proof
sk	User Secret Key	U_{id}	User identity
pk	User Public Key	$sigP$	Authorization
a	TPA Secret Key	$ChallT$	Challenge
w	TPA Public Key	θ	Group system

B. Bilinear Map:

Let G be a (multiplicative) group and there are G1 → G2 be a group homomorphism, and having same large prime order q and $\mathbb{G}$ is a generator of $\mathbb{G}1$. A bilinear map from G1 × G2 to G1 is a function e which is denoted by $e: \mathbb{G}1 \times \mathbb{G}2 \rightarrow \mathbb{G}1$. Following are some properties:

Computability:

$\forall u, v \in \mathbb{G}1$, an evaluation Algorithm $eA(u, v)$.

Binarity:

$$\forall a, b \in Z_q, \exists e(u^a, v^b) = eA(u, v)^{ab}$$

Non-degeneracy value:

$$eA[Gp, Gp] \neq 1$$

Security:

The most difficult task is to evaluate Discrete Logarithm in $\mathbb{G}1$.

C. Complex Assumptions:

Discrete Logarithms:

Let's assume that g generates $\mathbb{G}$, a multiplicative cyclic group having a prime factor of q. Probabilistic polynomial time doesn't exist on inputting$y \in \mathbb{G}$, that resulted a value $x \in Z_q^*$ where$g^x = y$.

Computational Diffie-Hellman:

Let's assume that g generates $\mathbb{G}$, a multiplicative group having a prime order of$q$. Probabilistic polynomial time doesn't exist on inputting$g^x g^y \in \mathbb{G}$, that resulted a value $g^{xy} \in \mathbb{G}$.

5. CONSTRUCTION OF SECURE METHOD

Following are the construction steps of the proposed method:

A. Dynamic Index Table (DIT):

A data structure DIT, which is abbreviated as Dynamic Index Table, is implemented to attain efficient public integrity. DIT is constructed using static linked list to prevent the movement of element whenever blocks are inserted/deleted. Dynamic Index Table is an array which is single dimensional and includes following:

- "Block identity" represented by "Bid_i"
- "Hash Value of Block Number" represented by "$Hash_i$"
- "Time Stamp" represented by "T_i"
- "Version of Block" represented by "V_i"
- "Static pointer which points to next block" represented by "$Next_i$"

Before data integration, $Hash_i$ is used to check which block is corrupted. To avoid attacks Time Stamp T_i and version of block V_i are used, $Next_i$ points to the next block for linking different files. For instance,$Next_i$ is 3 means the succeeding data block of m_2 is m_3 and when $Next_n = 0$ that means that m_n is the last and final block. The initial information of Dynamic Index Table is elaborated in Table 3:

Table 3: Initial Information of Dynamic Index Table

Block	Bid_i	$Hash_i$	T_i	V_i	$Next_i$
1	1	$H(mac_1)$	T_1	1	2
2	2	$H(mac_2)$	T_2	1	3
3	3	$H(mac_3)$	…	…	…
4	4	$H(mac_4)$	T_4	1	5
…	…	…	…	…	…
$i-1$	$i-1$	$H(mac_{i-1})$	T_{i-1}	1	i
i	i	$H(mac_i)$	T_i	1	$i+1$
$i+1$	$i+1$	$H(mac_{i+1})$	T_{i+1}	1	$i+2$
…	…	…	…	…	…
$n+1$	$n+1$	$H(mac_{n+1})$	T_{n+1}	1	n
n	n	$H(m_n)$	T_n	1	0

After deletion of block m_i, the value $Next_{i-1}$ converted to $i+1$ from$i$ which indicates that the upcoming block data is am_{i+1}. Furthermore, the value of $Next_i$ is set to -1 which indicates that m_i is deleted and new one stored.

Table 4:When m_i is deleted from DIT

Block	Bid_i	$Hash_i$	T_i	V_i	$Next_i$
1	1	$H(mac_1)$	T_1	1	2
2	2	$H(mac_2)$	T_2	1	3
…	…	…	…	1	4
4	4	$H(mac_4)$	…	…	…
…	…	…	…	…	…
i	i	$H(mac_i)$	T_i	1	-1
$i+1$	i	$H(mac_{i+1})$	T_{i+1}	1	$i+2$
…	…	…	…	…	…
$n+1$	$n+1$	$H(mac_{n+1})$	T_{n+1}	1	n
n	n	$H(mac_n)$	T_n	1	0

After mac_{i+1} a new block mac_i' is inserted into storage, the value changes from$next\ value_{i+1}$ to i and old $Next_i$ is changed to $i+1$. With the updation of corresponding static pointer, the information about m_i is added to the last position.

Table 5: When a new block m_i' is inserted in DIT

Block	Bid_i	$Hash_i$	T_i	V_i	$Next_i$
1	1	$H(mac_1)$	T_1	1	2
2	2	$H(mac_2)$	T_2	1	3
3	3	$H(mac_3)$	⋯	⋯	⋯
⋯	⋯	⋯	T_4	1	5
⋯	⋯	⋯	⋯	⋯	⋯
$i-1$	$i-1$	$H(mac_{i-1})$	T_{i-1}	1	$i+1$
i	i	$H(mac_i')$	T_i	1	-1
$i+1$	i	$H(mac_{i+1})$	T_{i+1}	1	$i+2$
⋯	⋯	⋯	⋯	⋯	⋯
$n-1$	$n-1$	$H(mac_{n-1})$	T_{n-1}	1	n
n	n	$H(mac_n)$	T_n	1	0

B. Integrity Verification Method in Detail:

There are three phases in auditing method which are as follow:

- Setup
- Integrity verification
- Dynamic update

a. Setup Phase:

System parameters, keys and TPA in *Initial* algorithm are generated by KGC in this phase. Big data is distributed in blocks and then blinds each block using *BlockBlind* algorithm is done by User. In algorithm n , *TPA* is accountable for tags generation and helps in deriving DIT in *DITGen*algorithm. Challenge authority is computed by user for TPA in *AuthorityGen*algorithm. The data flow is described in fig.2:

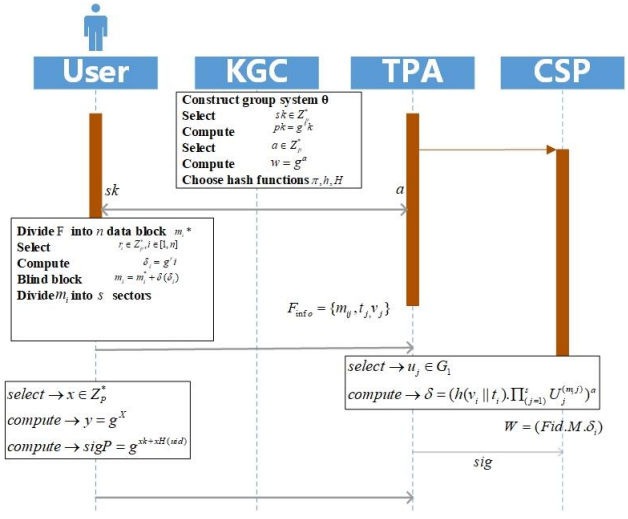

Figure 2: Setup Phase Data Flow

$$\boldsymbol{Initial(\lambda) \rightarrow \{\Theta, \alpha, sk\}}$$:

The parameter is given by λ of system security, a group equipped with bilinear mapping $\Theta = (\mathbb{G}1, \mathbb{G}2, p, g, e)$ is constructed by KGC. User's private key $sk \in Z_p^*$ is selected by KGC and formed public key as $pk = g^{sk}$. Then TPA's private key is selected by KGC as $\alpha \in Z_p^*$and computes public key as $w = g^{\alpha}$. Then secure hash function is chosen by KGC as $\pi: \mathbb{G}1 \rightarrow \mathbb{G}1, h: \{0,1\}^* \rightarrow \mathbb{G}1, H: \mathbb{G}1 \rightarrow Z_p^*$. Lastly, KGC sends user secret key sk with user identity U_{id} , $\alpha$to TPA in a secure manner and make $\{\mathbb{G}1, \mathbb{G}2, g, H, p, \pi, e, H, pk, w\}$ all these to be accessible by anyone.

$\boldsymbol{BlockBlind(F, SK) \rightarrow M}$**:** By erasure code algorithm, the user divides plain text file $F$with its own identifier F_{id} into n block of data which is named as m_i'. Before outsourcing F to CSS, user blind each block to keep the data private from other users. Randomly,$r_i \in Z_p^*, i \in [1, n]$ selects by user and computes $\delta_i = g^{ri}$. Then each block is blinded by user as $m_i = m_i^* + \pi(\delta_i)$ and indicates $M = \{m_i\}_{i \in [1,n]}$. Moreover, each block m_i is divided into $s$sectors by the user. This means $M = \{mac_i\}, mac_i = \{mac_{ij}\}, i \in [1, n], j \in [1, s]$. Lastly, a cloud client sends $F_{info} = \{m_{ij}, t_i, v_i\}$ to TPA.

$\boldsymbol{TagGen(M, \alpha) \rightarrow \sigma_i}$**:** Selection of $U_j \in \mathbb{G}1, j \in [1, s]$ by TPA and computes $\delta_i$with block tags every storage block $mac_i, i \in [1, n]$ as follows:

$$\delta_i = (h(v_i \parallel t_i).\prod_{j=1}^{s} U_j^{m_{ij}})^{\alpha} \quad (1)$$

Then $W = (F_{id}, M, \delta_i)$ is send to CSP by TPA.

$\boldsymbol{DITGen(F_{info}) \rightarrow DIT}$:DIT including $Bid_i, Hash_i, T_i, V_i, Next_i$are generated by TPA and stored it for dynamic updates. m_i is deleted from local server by TPA to save space.

$\boldsymbol{AuthorityGen(sk) \rightarrow sigP}$: For prevention from DDoS distributed denial of service) attack on CSP by the malicious attackers, auditing challenge is launch by an authorized TPA. The user having identity U_{id} randomly selects $x \in Z_p^*$ and generates $y = g^x$. Authorization generated by user for TPA is as follow:

$$sigP = g^{sk + xH(U_{id})} \quad (2)$$

b. Integrity Verification Phase

First of all, a challenge is generated by TPA in algorithm *ChallGen* and forward to CSP. After that CSP calculate the integrity proof and then again send it to the TPA in

algorithm $ProofGen$ for verification. The data flow is given in the fig. 3.

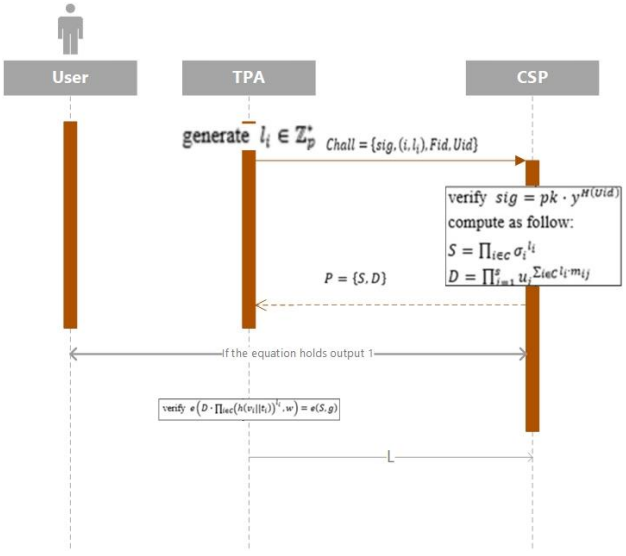

Figure 3: Flow of Data in Integrity Verification Phase

$\boldsymbol{ChallGen\ (F_{info})}$**:** When a user gives verification to TPA, some block is selected by user to form a random number subset element from set of $[1, n]$ and random number is generated $l_i \in Z_p^*, i \in C$.

Then a challenge $ChallT = \{sigP, (i, l_i), Fid, U_{id}\}, i \in C$ is send to CSP by the TPA.

$\boldsymbol{aProofGen\ (aF, T, ChallT)}$**:** When CSP received a challenge, equation $sigP = pk.y^{H(U_{id})}$ is verified. If it fails to verify then it resulted as NO and if it verified then tag and data proof are computed as follow:

$$S = \prod_{i \in C} \delta_i^{li} \quad (3)$$

$$D = \prod_{j=1}^{s} u_j^{\sum^{i \in C} li.m_{ij}} \quad (4)$$

After that $P = \{S, D\}$ is send to TPA by CSP.

$\boldsymbol{ProofVerify(P, w) \rightarrow \{1.0\}}$**:** TPA verify the proof after receiving it from CSP as

$$eA(D.\prod_{i \in C}(h(v_i \parallel t_i)^{li}, w) = eA(S, g). \quad (5)$$

If it verifies, the algorithm resulted as 1. If not, the algorithm then points out which of the block isn't stored correctly. Request to verify it out, it is sent by TPA to CSP. After that CSP generates $L = \{h_i^{'} = H(mac_i), i \in C$, such that m_i is a data block which get stored on CSS as well as L and is then transferred to the TPA. After that TPA contrast L with Hash $H(mac_i)$ present in Dynamic Index Table. If $h_i^{'} \neq H(mac_i)$,then TPA informs that i[th] block is corrupt and recover it to CSP.

c. Dynamic Update Phase

The data which is outsourced to the cloud can be updated by the user when required. The user should perform the block level insertion, block level deletion, and block level modification operation tasks. $BlockInsert$, $BlockDelete$, $BlockModify$, algorithms are used for block insertion, block deletion and block modification respectively.

$\boldsymbol{BlockInsert(m_i^{'}, i, SK)}$**:** Assume that after block m_i, another block $m^{'}$ is then inserted. First of all, $BlockBlind$ algorithm by the user as $m^* = m^{'} + \pi(\delta_i)$. $TagGen$ Algorithm is called by TPA to evaluate a tag $a\delta^{'}$ afor $am^{'}$ and then transfer $a\{m^*, \delta^{'}\}$. Then TPA evaluates $aH(m_i^{'})$. TPA after evaluation add $a(i+1, Hash(mac^{'}), t^{'}, v^{'})$ to the end location of DIT or where $Next_i$ is at -1.

$\boldsymbol{BlockDelete\ (m_i)}$**:** Suppose deletion of m_i block is to be done. CSP deletes m_i and δ_i. TPA, on the basis of block number, finds the location of m_i and changes $Next_i$ to -1.

$\boldsymbol{BlockModify(m_i^{'}, \alpha, sk)}$**:** Suppose that the block m_i is modified to new state that is $m_i^{'}$. To blind the block, first of all $BlockBlind$ algorithm is called by the user like $m_i^{'} = m_i^{'} + \pi(\delta_i).TagGen$ Algorithm is called by TPA to evaluate a new tag $a\delta^{'}$ for $am^{'}$ and then transfer $\{m^*, \delta^{'}\}$ to CSP. Then TPA evaluates $H(mac_i^{'})$ and does an addition of new $(i, H(m^{'}), t^{'}, v^{'})$ and modify the old item to new one in DIT. The user appointed the TPA to check which of the block is updated. The user then deleted the local data when verification is done.

C. Batch Auditing from multiple users:

This auditing method can simultaneously process different verification from multiple users. Assume that U is a group of multiple users k . CSP evaluates two different proofs, one is tag $S_i, i \in [1, k]$, and other is data $D_i, i \in [1, k]$ when k challenges are received by multiple k users. According to the following equations the CPU gets S_U by assembling with S_i and D_U by assembling with D_i.

$$S_U = \prod_{i=1}^{k} S_i \quad (6)$$

$$D_U = \prod_{i=1}^{k} D_i \quad (7)$$

TPA then justify the proof with the equation given below:

$$e(D_U.\prod_{i=1}^{k}\left(\prod_{j \in C}\left(h\left(v_{i,j} \parallel t_{i,j}\right)\right)^{l_{i,j}}\right), w_i) = e(S_U, g) \quad (8)$$

If all the files are accurately stored on cloud servers then the equation outputs YES otherwise NO.

6. SECURITY ANALYSIS

After Correctness, unforgeability and privacy of the proposed method is analyzed.

Theorem 1:

To verify the file integrity stored in cloud, the authorized verifier is used in our proposed method.

Proof:

It is proved through eq (5).

$$\begin{aligned} & e(D.\prod_{i\in C}(h(v_i \parallel t_i)^{li}, w) \\ &= e(\prod_{j=1}^{s} u_j^{\sum_{i\in C} l_i m_{ij}}.\prod_{i\in C}(h(v_i \parallel t_i)^{li}, w) \\ &= e(\prod_{i\in C}\prod_{j=1}^{s} u_j^{l_i m_{ij}}\prod_{i\in C}(h(v_i \parallel t_i)^{li}, w) \\ &= e(\prod_{i\in C}\prod_{j=1}^{s} u_j^{l_i m_{ij}}\prod_{i\in C}(h(v_i \parallel t_i)^{li}, g^{\alpha}) \\ &= e(S, g) \end{aligned}$$

Theorem 2: To attain verification publicly, if Computational Diffie Hellman is endure in a bilinear mapping then it is improbable for CSP to make a proof of integrity.

Proof: The $P = \{S, D\}$ is send to TPA after receiving challenge $Chall = \{sig, (i, l_i), F_{id}\}, i \in C$ from CSP. Assume that a wrong proof is generated to TPA by CSP $P' = \{S, D'\}$ and $D' = \prod_{j=1}^{s} u_j^{\lambda_j'}, \lambda_j' = \sum_{i\in C} l_i \times m_{ij}', j \in [1, s]$. $\lambda_j = \sum_{i\in C} l_i \times m_{ij}', \Delta\lambda_j = \lambda_j - \lambda_j'$. It is clear that at least one $\Delta\lambda_j \neq 0$. If verification is passed with P' by CSP, it will win otherwise fails.

Assume that if CSP the wins the game, it can be inferred from the following equation:

$$= e(\prod_{j=1}^{s} u_j^{\lambda_j'}.\prod_{i\in C}(h(v_i \parallel t_i)^{li}, w) = e(S, g)$$

Moreover, the proof $P = \{S, D\}$ is also an accurate one, it can be satisfied from following equation:

$$= e(\prod_{j=1}^{s} u_j^{\lambda_j}.\prod_{i\in C}(h(v_i \parallel t_i)^{li}, w) = e(S, g)$$

It can be concluded from the equations written above and from the properties of bilinear map that $\prod_{j=1}^{s} u_j^{\lambda_j'} = \prod_{j=1}^{s} u_j^{\lambda_j} = \prod_{j=1}^{s} u_j^{\Delta\lambda_j} \Longrightarrow 1$. It is because $\mathbb{G}_1$ is a cyclic group then $b_1, b_2 \in \mathbb{G}_1, \exists x \in Z_p$ such that $b_2 = b_1^x$. Moreover, b_1, b_2, u_j can be formed as $u_j = b_1^{u_j} b_2^{v_j} \in \mathbb{G}_1$, where $u_j, v_j \in Z_p$. Then we get following $\prod_{j=1}^{s} u_j^{\Delta\lambda_j} = \prod_{j=1}^{s} (b_1^{u_j} b_2^{v_j})^{\Delta\lambda_j} = b_1^{\sum_{j=1}^{s} u_j \Delta\lambda_j}.b_2^{\sum_{j=1}^{s} u_j \Delta\lambda_j} = 1$. An approach to the problem of DL can be sought. Unless $\Delta\lambda_j = 0$, the x value can be obtained as following:

$$b_2 = b_1^x = b_1^{\frac{\sum_{j=1}^{smax} u_j \Delta\lambda_j}{\sum_{j=1}^{smax} v_j \Delta\lambda_j}}$$

$$x = \frac{\sum_{j=1}^{smax} u_j \Delta\lambda_j}{\sum_{j=1}^{smax} v_j \Delta\lambda_j}$$

Theorem 3: During the integrity verification, as much as DL assumptions hold, it is not possible for TPA to attain any data which is private.

Proof: CSS sends data proof $D = \prod_{j=1}^{s} u_j^{\sum i\in C^{li.m_{ij}}}$ to TPA after it gets challenge $Chall$. It is impossible for TPA to attain any data which is private for user because according to DL assumption $\sum i \in C^{li.m_{ij}}$ is at the exponential position of D.

7. PERFORMANCE TESTING

Performance testing is based on the following parameters:

A. Communication Cost:

Major cost of communication is formed between the user to TPA and the TPA to CSS, in our proposed method. Assume that the size of element of Z_p is $|p|$. After user blinds each block with the help of $BlockBlind$ algorithm, it sends $F_{info} = \{m_{ij}, t_i, v_i\}$ to TPA. Hence the communication costs occur as $n|p| + n(|t_i| + |v_i|)$. TPA sends $W = (F_{id}, M, \delta_i)$ to CSP in $TagGen$algorithm and communication costs occur is $2n|p| + 1$. The major communication cost occurs between TPA and CSS during integrity verification phase. In algorithm $ChallGen$, TPA sends $Chall = \{sig, (i, l_i), F_{id}, U_{id}\}$ to CSP. The cost of communication occurs in bits like $c(|i| + |p|$. CSP then dispatch $P = \{S, D\}$ to TPA in $ProofGen$ algorithm. The constant communication cost occurs like $2|p|$which can be neglected. The cost of communication is constant during updating phase between the user and TPA. This also occur in between TPA and the CSP. The comparison is done between different methods as shown in Table 6 and can be inferred that our method has more efficient communication cost than others.

Table 6: Contrast of Communication Costs of Different Methods

Method	Setup	Verification	Updating
[12]	$O(n)$	$O(logn)$	$O(logn)$
[13]	$O(n)$	$O(logn)$	$O(logn)$
[14]	$O(n)$	$O(nc)$	$O(1)$
[17]	$O(n)$	$O(nc)$	$O(1)$
Proposed Method	$O(n)$	$O(nc)$	$O(1)$

B. Storage Cost

There are three phases in our method and storage costs usually occur in setup phase. Assume that the file named as F and having n blocks, size of $|p|$, is outsourced on cloud. In $BlockBlind$ algorithm, the user sends $F_{info} = \{m_{ij}, t_i, v_i\}$ to TPA.

TPA generates DIT including $Bid_i, Hash_i, T_i, V_i, Next_i$ in $DITGen$ algorithm. TPA deletes m_i from the server to save the storage. Hence, the cost of TPA total storage in set phase be nl_3 and $l_3 = |Bid_i| + |t_i| + |v_i| + |Hash_i| + |Next_i|$. TPA sends $W = (F_{id}, M, \delta_i)$ to CSP in $TagGen$ algorithm and CSP saves W. The cost of storage generated by M and δ_i in setup phase is $2n|p|$. Index Hash Table (IHT) is utilized in method [14].

IHT specifies the changes occurring in blocks and during integrity verification process, value of hash block is generated. The cost of TPA storage of is nl_1, where $l_1 = |B_i| + |V_i| + |R_i|$. In method [17], Dynamic hash table is used and the cost of TPA storage is nl_2, where $l_2 = |v_i| + |t_i| + |Next_i|$. The storage costs of different method are computed in Table 7. The size of l_3 is much greater than l_1 and l_2. Because the hash value employed for each block, DIT is much secure and efficient as compared to IHT and DHT.

Table 7: Contrast of Storage Costs of Different Methods

Method	TPA	CSP
[12]	0	$\|p\|(2^{logn+1} + n - 1\|$
[13]	0	$\|p\|(2^{logn+1} + n - 1\|$
[14]	$(n+1)l_1$	$2n\|p\|$
[17]	$(n+1)l_2$	$2n\|p\|$
Proposed Method	nl_3	$2n\|p\|$

C. Computation Cost

The time of computation of the proposed method is evaluated with the experiment. It is then compared with the method [14]. The method's implementation is done on a Linux system having 1GB Ram, 1.6 GHz processor. For our simulation, Pairing Based Cryptography having 0.5.13 model is utilized. During these experiments, results represent the only an average of 20 trails.

i. User computation time in the setup:

In these experiments of multiple blocks, test our proposed method, computation numbers with a maximum block size of 1 KB. It can be assumed from Fig. 4 that the computational time of the user is to the number of blocks, and our protocol's computing cost is lower than method [14].

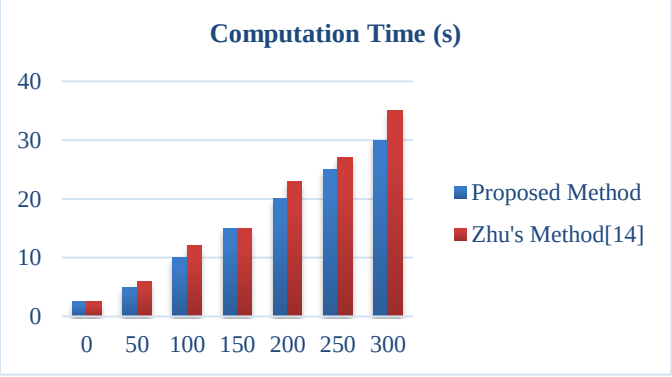

Figure 4: Contrast of Setup Phase Computation time

ii. Computation Costs in Verification phase:

The relation between block size and computation time is checked at the same file size of 1 MB during the verification stage. 20 % of the overall block number corresponds to the challenged block number during the simulation. In Fig.5, we can assume that the cost of verifying our proposed method is declining as the block size increases. The time of verification in the method [14], however, is increasing, as the verification equation in the method [14] relates to each block sector.

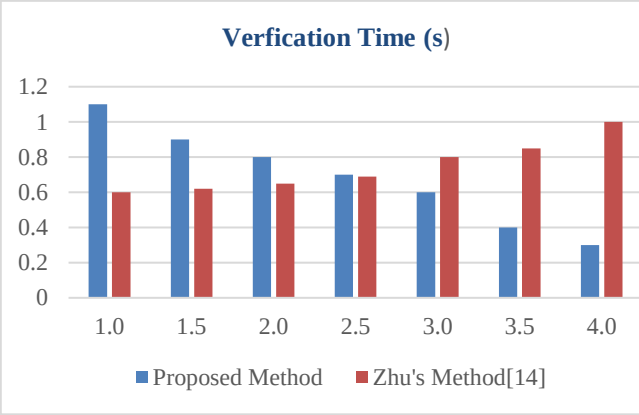

Figure 5: Contrast of Verification Time

iii. Computation Cost in updating:

It assumes that the size of block is of maximum 1 KB in the updating phase experiment. The update time is checked with a file size from 1 MB to 50 MB. Out of Fig. 6 and Fig. 7, we may infer that our method is more effective in insertion and as well as in deletion operation. As IHT is a sequence data structure in Zhu 's method and about half o elements are adjusted and resulting in a decrease in the efficiency of the update process. In our method, without moving objects, just the static pointer have to be modified.

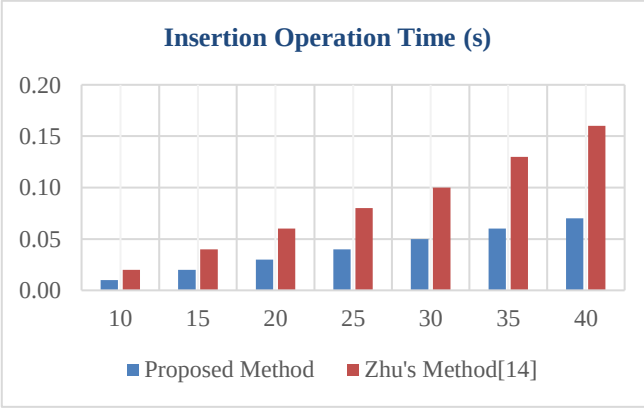

Figure 6: Contrast of Insertion Time

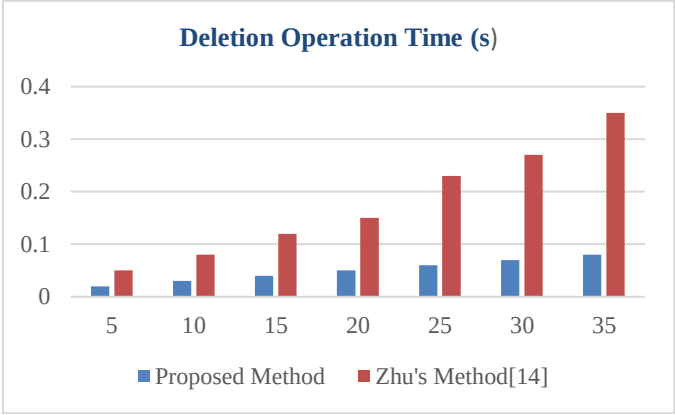

Figure 7: Contrast of Deletion Time

8. CONCLUSION

This research proposes an effective hierarchical auditing method for cloud servers for data that is outsourced. In the proposed methodology, a dynamic index table (DIT) is utilized. There is no need for elements to be shifted during insertion, deletion, and update operations in this method and designed to increase data updates' efficiency. Also, TPA will detect and recuperate the corrupt block if the cloud file is not incorporated. Also, a permit is used to avoid denial of service attacks between users and cloud servers. A secure and efficient integrity verification can be carried out for big data in the proposed method. Results indicated that the secure storage application method costs a minimum, then previous methods for cloud storage and computation cost.

In the future, we must figure out the possibility to further improve the security and efficiency of the integrity verification system since these are the most critical aspects in ample data cloud storage. To improve the speed of integrity testing between cloud and user, we can lessen the communication cost to enhance performance. Besides, cloud server storage costs are also addressed. Since privacy is also another crucial aspect in cloud computing, the privacy of user data is concentrated and main aspects of our future work are efficiency and security.

REFERENCES

[1] Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., ... &Stoica, I. (2009). Above the clouds: A berkeley view of cloud computing. Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS, 28(13), 2009.

[2] Lu, X., & Cheng, X. (2019). A Secure and Lightweight Data Sharing Method for Internet of Medical Things. IEEE Access, 8, 5022-5030.

[3] Zhang, Y., Qiu, M., Tsai, C. W., Hassan, M. M., &Alamri, A. (2015). Health-CPS: Healthcare cyber-physical system assisted by cloud and big data. IEEE Systems Journal, 11(1), 88-95.

[4] Guan, Z., Lv, Z., Du, X., Wu, L., &Guizani, M. (2019). Achieving data utility-privacy tradeoff in Internet of medical things: A machine learning approach. Future Generation Computer Systems, 98, 60-68.

[5] Chang, V. (2017). Towards data analysis for weather cloud computing. Knowledge-Based Systems, 127, 29-45.

[6] Lewko, A., & Waters, B. (2011, May). Decentralizing attribute-based encryption. In Annual international conference on the theory and applications of cryptographic techniques (pp. 568-588). Springer, Berlin, Heidelberg.

[7] Deswarte, Y., Quisquater, J. J., &Saïdane, A. (2003, November). Remote integrity checking. In Working conference on integrity and internal control in information systems (pp. 1-11). Springer, Boston, MA.

[8] Juels, A., &Kaliski Jr, B. S. (2007, October). PORs: Proofs of retrievability for large files. In Proceedings of the 14th ACM conference on Computer and communications security (pp. 584-597).

[9] Yamamoto, G., Oda, S., & Aoki, K. (2007, June). Fast integrity for large data. In Proc. ECRYPT Workshop Software Performance Enhancement for Encryption and Decryption (pp. 21-32).

[10] Ateniese, G., Burns, R., Curtmola, R., Herring, J., Kissner, L., Peterson, Z., & Song, D. (2007, October). Provable data possession at untrusted stores. In Proceedings of the 14th ACM conference on Computer and communications security (pp. 598-609).

[11] Ateniese, G., Di Pietro, R., Mancini, L. V., &Tsudik, G. (2008, September). Scalable and efficient provable data possession. In Proceedings of the 4th international conference on Security and privacy in communication netowrks (pp. 1-10).

[12] Erway, C. C., Küpçü, A., Papamanthou, C., &Tamassia, R. (2015). Dynamic provable data possession. ACM Transactions on Information and System Security (TISSEC), 17(4), 1-29.

[13] Wang, Q., Wang, C., Ren, K., Lou, W., & Li, J. (2010). Enabling public auditability and data dynamics for storage security in cloud computing. IEEE transactions on parallel and distributed systems, 22(5), 847-859.

[14] Zhu, Y., Ahn, G. J., Hu, H., Yau, S. S., An, H. G., & Hu, C. J. (2011). Dynamic audit services for outsourced storages in clouds. IEEE Transactions on Services Computing, 6(2), 227-238.

[15] Yang, K., & Jia, X. (2012). An efficient and secure dynamic auditing protocol for data storage in cloud computing. IEEE transactions on parallel and distributed systems, 24(9), 1717-1726.

[16] Liu, C., Chen, J., Yang, L. T., Zhang, X., Yang, C., Ranjan, R., &Kotagiri, R. (2013). Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates. IEEE Transactions on Parallel and Distributed Systems, 25(9), 2234-2244.

[17] Luo, H. S., Jiang, R., & Pei, B. (2017, December). Cryptanalysis and Countermeasures on Dynamic-Hash-Table Based Public Auditing for Secure Cloud Storage. In 2017 10th International Symposium on Computational Intelligence and Design (ISCID) (Vol. 1, pp. 33-36). IEEE.

[18] Gan, Q., Wang, X., & Fang, X. (2018). Efficient and secure auditing method for outsourced big data with dynamicity in cloud. Science China Information Sciences, 61(12), 122104.

[19] Zhang, Y., Yu, J., Hao, R., Wang, C., & Ren, K. (2018). Enabling efficient user revocation in identity-based cloud storage auditing for shared big data. IEEE Transactions on Dependable and Secure computing.

[20] Lu, X., Pan, Z., & Xian, H. (2020). An integrity verification method of cloud storage for internet-of-things mobile terminal devices. Computers & Security, 92, 101686.

[21] Zhang, Q., Yang, L. T., & Chen, Z. (2015). Privacy preserving deep computation model on cloud for big data feature learning. IEEE Transactions on Computers, 65(5), 1351-1362.

[22] Ming, Y., & Zhang, T. (2018). Efficient privacy-preserving access control method in electronic health records system. Sensors, 18(10), 3520.

[23] Wang, C., Chow, S. S., Wang, Q., Ren, K., & Lou, W. (2011). Privacy-preserving public auditing for secure cloud storage. IEEE transactions on computers, 62(2), 362-375.

[24] Wang, B., Li, B., & Li, H. (2014). Oruta: Privacy-preserving public auditing for shared data in the cloud. IEEE transactions on cloud computing, 2(1), 43-56.

[25] Yu, J., & Hao, R. (2019). Comments on" SEPDP: Secure and Efficient Privacy Preserving Provable Data Possession in Cloud Storage". IEEE Transactions on Services Computing.

[26] Zhou, Q., Tian, C., Zhang, H., Yu, J., & Li, F. (2020). How to securely outsource the extended euclidean algorithm for large-scale polynomials over finite fields. Information Sciences, 512, 641-660.

[27] Halperin, D., Heydt-Benjamin, T. S., Fu, K., Kohno, T., & Maisel, W. H. (2008). Security and privacy for implantable medical devices. IEEE pervasive computing, 7(1), 30-39.

[28] Li, Y., Xia, H., Zhang, R., Hu, B., & Cheng, X. (2020). A Novel Community Detection Algorithm Based on Paring, Splitting and Aggregating in Internet of Things. IEEE Access, 8, 123938-123951.

[29] Yang, G., Xie, L., Mäntysalo, M., Zhou, X., Pang, Z., Da Xu, L., ... & Zheng, L. R. (2014). A health-IoT platform based on the integration of intelligent packaging, unobtrusive bio-sensor, and intelligent medicine box. IEEE transactions on industrial informatics, 10(4), 2180-2191.

[30] Zhang, J., Ren, F., Gao, S., Yang, H., & Lin, C. (2014). Dynamic routing for data integrity and delay differentiated services in wireless sensor networks. IEEE Transactions on Mobile Computing, 14(2), 328-343.

[31] Lai, J., Deng, R. H., Guan, C., & Weng, J. (2013). Attribute-based encryption with verifiable outsourced decryption. IEEE Transactions on information forensics and security, 8(8), 1343-1354.

[32] Shacham, H., & Waters, B. (2008, December). Compact proofs of retrievability. In International conference on the theory and application of cryptology and information security (pp. 90-107). Springer, Berlin, Heidelberg.

[33] Cash, D., Küpçü, A., &Wichs, D. (2017). Dynamic proofs of retrievability via oblivious RAM. Journal of Cryptology, 30(1), 22-57.

[34] Sun, Y., Liu, Q., Chen, X., & Du, X. (2020). An Adaptive Authenticated Data Structure With Privacy-Preserving for Big Data Stream in Cloud. IEEE Transactions on Information Forensics and Security, 15, 3295-3310.

[35] Cai, H., Xu, B., Jiang, L., &Vasilakos, A. V. (2016). IoT-based big data storage systems in cloud computing: perspectives and challenges. IEEE Internet of Things Journal, 4(1), 75-87.

[36] Hu, C., Li, W., Cheng, X., Yu, J., Wang, S., &Bie, R. (2017). A secure and verifiable access control method for big data storage in clouds. IEEE Transactions on Big data, 4(3), 341-355.

[37] Zhao, J., Xu, C., Li, F., & Zhang, W. (2013). Identity-based public verification with privacy-preserving for data storage security in cloud computing. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 96(12), 2709-2716.

[38] Zhang, Y., Zheng, D., & Deng, R. H. (2018). Security and privacy in smart health: Efficient policy-hiding attribute-based access control. IEEE Internet of Things Journal, 5(3), 2130-2145.

[39] Liu, X., Ma, J., Xiong, J., Zhang, T., & Li, Q. (2013, October). Personal health records integrity verification using attribute-based proxy signature in cloud computing. In International Conference on Internet and Distributed Computing Systems (pp. 238-251). Springer, Berlin, Heidelberg.

[40] Jin, H., Zhou, K., Jiang, H., Lei, D., Wei, R., & Li, C. (2018). Full integrity and freshness for cloud data. Future Generation Computer Systems, 80, 640-652.

[41] Bilin Shao, Yanyan Ji .(2021) . Efficient TPA-based auditing scheme for secure cloud storage. Cluster Computing.

[42] Luo, W., Ma, W. & Gao, J. MHB*T based dynamic data integrity auditing in cloud storage. Cluster Computing (2021). https://doi.org/10.1007/s10586-021-03248-w

Sameen Fatima received the Master degree from the renowned institution University of Engineering and Technology, Lahore, Pakistan. Her current research interests are mainly focus on Cloud Computing, Big Data, Cyber Security and IoT security

Dr. Shafiq Hussain received his M.Sc degree in Computer Science from Bahauddin Zakariya University, Multan, Pakistan and the PhD degree in Computer Science from the University of Sunderland, the UK in 2015. He is currently working as Associate Professor of Computer Science in Department of Computer Science, University of Sahiwal, Pakistan. He is also founder and Chairman, Department of Computer Science at the University of Sahiwal. He has completed the HEC funded project as Project Director. He is also working Director IT and Project Director of the Higher Education Commission (HEC) of The University of Sahiwal. He has supervised more than 30 postgraduates research students and published many research papers in academic journals.

Rana Abu Bakar received a master degree in Computer Science (Network Security) from the Virtual University of Pakistan. Currently pursuing a PhD in Computer Engineering from Chulalongkorn University (Bangkok, Thailand). His research interests include networks, vehicular networks, mobile ad-hoc networks, secure UAV drones, cryptography, cryptography threat modeling, IoT security, wireless communications and networking, big data security, next-generation mobile computing and cloud computing security.