DESIGN AND ANALYSIS OF EFFICIENT PROTOCOLS FOR DATA SECURITY IN CLOUD COMPUTING

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

PAROMITA GOSWAMI

MZU REGN NO : 2006580

PH. D. REGN NO : MZU/Ph.D/1655 OF 06.11.2020

DEPARTMENT OF COMPUTER ENGINEERING
SCHOOL OF ENGINEERING AND TECHNOLOGY

MAY, 2024

DESIGN AND ANALYSIS OF EFFICIENT PROTOCOLS FOR DATA SECURITY IN CLOUD COMPUTING

BY

PAROMITA GOSWAMI
DEPARTMENT OF COMPUTER ENGINEERING

Supervisor: Prof. Ajoy Kumar Khan
Joint-Supervisor: Dr. Somen Debnath

Submitted

In partial fulfilment of the requirement of the Degree of Doctor of Philosophy in Computer Engineering of Mizoram University, Aizawl.

CERTIFICATE

This is to certify that the thesis entitled "*Design and Analysis of Efficient Protocols for Data Security in Cloud Computing*" is a record of the research that Paromita Goswami completed under our supervision and guidance, and it is submitted to the Mizoram University in the Department of Computer Engineering under the School of Engineering and Technology, in partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy in Computer Engineering.

All help received by her from various sources has been duly acknowledged. No part of this thesis has been reproduced elsewhere for the award of any other degree.

Prof. Ajoy Kumar Khan
Supervisor & Professor
Dept. of Computer Engineering
Mizoram University,
Aizawl-796 004

Place:

Date:

Dr. Somen Debnath
Joint Supervisor & Associate Professor
Dept. of Computer Science & Engineering
Tripura University,
Agartala-799022

Place:

Date:

DECLARATION

I Paromita Goswami, hereby declare that the subject matter of this thesis is the record of work done by me, that the contents of this thesis did not form basis of the award of any previous degree to me or to do the best of my knowledge to anybody else, and that the thesis has not been submitted by me for any research degree in any other University/Institute.
This is being submitted to the Mizoram University for the degree of Master of Philosophy/Doctor of Philosophy in Computer Engineering

(PAROMITA GOSWAMI)

(Prof.AJOY KUMAR KHAN)

Supervisor

(Dr. V. D. AMBETH KUMAR)

Head

(Dr. SOMEN DEBNATH)

Joint Supervisor

ACKNOWLEDGEMENTS

Foremost, heartfelt appreciation goes to the supervisor, Prof Ajoy Kumar Khan, the Department of CE, Mizoram University. The consistent guidance, profound insights, and continuous encouragement provided by Prof. Khan have been invaluable in shaping the direction of the research.

I would also like to extend my special sincere appreciation to my joint supervisor, Dr. Somen Debnath, Department of Computer Science & Engineering, Tripura University, for his insightful contributions and support. His extensive knowledge in the area of cryptography, cloud security & Information security and his willingness to engage in meaningful discussions have significantly enriched the quality of this research.

I'm grateful to the Department of Computer Engineering, Mizoram University, for fostering an ideal research environment and providing crucial resources.I acknowledge the Department of CEA, GLA University, for encouraging my research journey to continue seamlessly.

My heartfelt thanks to all professors and colleagues for their expertise and feedback, shaping my research at various stages.I also would like to express my heartfelt thanks to my family and friends for their unwavering support and understanding throughout this journey.

Last but not least, my heartfelt gratitude goes to the All Mighty God, whose divine presence guided me through uncertainties and granted me resilience.

(PAROMITA GOSWAMI)

Contents

List of Tables

List of Figures

Chapter 1

INTRODUCTION

Cloud Computing's flexibility through SLA-based services grants users access to extensive resources. It encompasses private, public, hybrid, and community clouds, offering IaaS, PaaS, and SaaS. Providers like Google Compute Engine and Amazon EC2 optimizes performance and reduces costs for clients. This evolution transforms sectors by aiding cost reduction, performance monitoring, and resource management, streamlining resource allocation and workload distribution, and fostering efficiency and scalability in digital landscapes This chapter aims to address this gap by offering an in-depth discussion on the security challenges within physical cloud storage, potential threats, attacks, and their mitigations. It will also categorize data integrity schemes, outline their phases and characteristics, provide a comparative analysis, and project future trends. This comprehensive approach underscores the significance of data integrity schemes in securing cloud storage.

1.1 Overview of Cloud Computing

In the commercial and economic world, many researchers have tried to define that what is cloud computing and its characteristics; according to Buyya et al. [1] "Cloud is a parallel and distributed computing system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resources based on service-level agreements (SLA) established through negotiation between the service provider and consumers". A report from the University of California Berkeley [2] summarized the key characteristics of cloud computing as: "(1) the illusion of infinite computing resources; (2) the elimination of an up-front commitment by cloud

users; and [3] the ability to pay for use as needed.
The National Institute of Standards and Technology (NIST) characterizes cloud computing as a pay-per-use model for enabling available, convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage, applications, services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.[4]"
Although there exist many definitions of cloud computing, most of them describe the common characteristics of the cloud such as pay-per-use service having infinite resources, elastic capacity, virtualized resources, and self-service interface.

1.1.1 Service and Deployment Model of Cloud Computing

Cloud computing services are categorized into three classes based on the abstraction level and service model: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS), as outlined in the reference model by Buyya et al[1].

- **Infrastructure as a Service(IaaS):**The virtualized resources such as computation, storage, and communication are provided on demand by cloud computing is known as infrastructure as a service (IaaS). Cloud infrastructure enables on-demand provisioning of servers running several choices of operating systems and a customized software stack[1].

- **Platform as a Service (PaaS):**Platform as a service (PaaS) offers a higher level of abstraction which makes a cloud easily programmable for infrastructure–oriented service. A cloud platform offers an environment for developers where they create and deploy applications and does not require knowing about how many processors or how much memory is used by those applications. In addition, multiple programming models and specialized services are offered by this service for new applications [1].

- **Software as a Service (SaaS):**Applications are placed on the top of the

cloud stack. Service provided by this layer is accessed by end users through web portals. Therefore, consumers are increasingly shifting from locally installed computer programs to online software services that offer the same functionality. The 7-model delivery of the application is known as software as a service (SaaS), which facilitates the burden of software maintenance for customers and simplifies development and testing for providers [1].

Many issues arise when an application moves in a cloud environment, so the deployment model considers variations in a physical location and distribution and classifies the cloud into three types [1].

- **Public Cloud:** The services that are rendered over a network that is open for everyone is called public cloud[2]. These services may be free or offered on demand. It allows users to interact with local data and use the services across virtual servers and pay for only those services used by them. This cloud is scalable because it can increase or decrease the services according to the number of users. It is less secure because it is available for everyone.

- **Private Cloud:**Private cloud [1] is operated by a single organization or managed by a third party where the data center is used by the organization, not available to the general public. It is not accessed publicly so it is more secure but available at much cost for upfront and maintenance.

- **Hybrid Cloud:**The hybrid model is also known as the community model. This model is shared by several organizations and specific communities that share the concerns (mission, security requirements, and policy and compliance considerations) [3]. When the private cloud is supplemented with computing capacity from the public cloud becomes a hybrid cloud.

1.1.2 Security Issues on Cloud Data Storage

Different service layers have different issues and all security issues and privacy are divided into four parts: data integrity, data confidentiality, data availability, and data privacy [5].

(a) Data Integrity: Data integrity safeguards data from unauthorized changes, deletions, or fabrications. Access management ensures that only authorized entities can access enterprise resources, preventing misuse or theft. In standalone systems, database constraints and ACID transactions maintain data integrity, while authorization controls access to secure resources based on user privileges.

(b) Data Confidentiality: To maintain cloud reliability and trustworthiness, data confidentiality, authentication, and data access control security issues are addressed [5].

(c) Data Availability: Data availability means the following: when accidents such as hard disk damage, IDC fire, and network failures occur, the extent to which user's data can be used or recovered, and how the users verify their data by techniques rather than depending on the credit guarantee by the cloud service provider alone [5].

(d) Data Privacy: Privacy is the ability of an individual or group to seclude themselves or information about themselves and thereby reveal them selectively [5]. Privacy has the following elements:

- **When:** a subject may be more concerned about the current or future information being revealed than information from the past.
- **How:** a user may be comfortable if his/her friends can manually request his/her information, but the user may not like alerts to be sent automatically and frequently.
- **Extent:** a user may rather have his/her information reported as an ambiguous region rather than a precise point.

1.1.3 Security Threats under Cloud Storage

Cloud users benefit from inexpensive computing resources with IaaS services, without infrastructure maintenance responsibilities. However, the lack of trusted actors makes cloud storage vulnerable to attacks from CSPs and insiders. This study outlines cloud storage issues and possible attacks, offering mitigating solutions in Table 1.1.

- In capability of CSP: Managing big cloud storage may create a data loss problem for CSP due to lack of insufficient computational capacity, sometimes not meeting user's requirements, missing a user-friendly data serialization standard with easily readable and editable syntax, due to changes of a life cycle in a cloud environment [27].

- Loses control of cloud data over a distributed cloud environment may give vulnerable chances to unauthorized users to manipulate valuable data of valid one [28].

- Lack of Scalability of physical cloud storage: Scalability means all hardware resources are merged to provide more resources to the distributed cloud system. It might be beneficial for illegitimate access and modify cloud storage and physical data centers [29].

- Unfair resource allocation strategy: Generally, monitoring data is stored in a shared pool in a public cloud environment which might not be preferable to cloud users who are not interested to leave any footprint on their work distribution/data transmission by a public cloud-hosted software component which will be the reason for a future mediocre of original data fetching [30].

- Lack of performance monitoring of cloud storage: Generally, monitoring data is stored in a shared pool in a public cloud environment which might not be preferable to cloud users who are not interested in leaving any footprint on their work distribution/data transmission by a public cloud-hosted

Table 1.1: Potential Types of Vulnerable Attacks and Threats at Storage Level Data Integrity with Mitigating Solutions

Potential Attacks	**Storage Issues**	**Threats**	**Mitigation Solution with references**	**Applied Methods**
DoS	No prediction format to formulate required time/storage to store/process data into cloud storage, data threat	Vulnerable service takes place instead of original service	Proposed authentication & authorization protocol[6, 7] Proposed signature-based scheme[8] Proposed intrusion detection/prevention scheme[9] [10, 11]	Kerberos protocol Attribute-Based Proxy Signature Improved Dynamic Immune Algorithm(IDIA)
Phishing	Lack of storage monitoring, Unaccredited access to physical cloud storage	Data confidentiality disclose	Propose phishing detection technique[12]	a hybrid classifier approach and hyper-parameter classifier tuning
Brute Force Attack /Online Dictionary Attack	Unaccredited access to physical cloud storage	Data confidentiality disclose, Violation of Data Authenticity	Propose data obfuscation scheme[13]	Least Significant Bit(LSB) substitution method
MITC Attack	Improper security against internal and external malicious attacks	Abnormality in service availability	Propose string authentication technique[14]	Chaotic maps and fuzzy extractors
Risk Spoofing	Incapability of CSP's monitoring, Untrusty cloud storage, data lock-in	Lack of internal security, logging violation	Monitoring secure data policies [15]	Symmetric Searchable Encryption (SSE) or Attribute-Based Encryption (ABE)

Port Scanning	Improper security internal and external malicious attack	Abnormality in service availability	firewall policies [16]	Distributed firewalls/Controllers
Identity Theft	Unaccredited access to physical cloud storage, Untrusty cloud storage, data threat	SLA violation, security policies violation	Password based authentication scheme[17, 18, 19], privacy beach prevention [20]	key-based semantic secure Bloom filter (KSSBF), compact password-authenticated key exchange protocol (CompactPAKE), OTP, EMPPC
Data Loss/Leakage	Incapability of CSP, continuous storage monitoring, lack of scalability	Malicious Insider, Malicious Cloud Storage Provider	Propose Data integrity technique [21, 22, 23, 24, 25]	Data encryption method, Public data auditing technique
Shared Technology issue	Unfair resource allocation strategy, no standard data storing format, Shared Technology Issue	VMs are become vulnerable due to loose control of a hypervisor	Virtual Machine Monitoring Scheme [26]	Xen, KVM

software component which will be the reason for a future mediocre of original data fetching [30].

- Data threat: Cloud users store sensitive data in cloud environments about their personal information or business information. Due to the lack of data threat prevention techniques of cloud service providers, data may be lost or damaged[31, 25].

- Malicious cloud storage provider: Lack of transparency and access control policies are basic parameters of a cloud service provider being a malicious storage provider. Due to the missing of these two parameters, it's quite

easy to disclose confidential data of cloud users to others for business profit [32].

- Data Pooling: Resource pooling is an important aspect of cloud computing. Due to this aspect, data recovery policies and data confidentiality schemes are broken[33].

- Data lock-in: Every cloud storage provider does not have a standard format to store data. Therefore, cloud users face a binding problem in switching data from one provider to another due to dynamic changes in resource requirements[34].

- Security against internal and external malicious attack: Data might be lost or data can be modified by insider or outsider attacks[35, 36, 37, 38].

1.1.4 Potential Attacks in Storage Level Service

Cloud computing's storage-level service offers resource computation, virtual network, and shared storage over the internet, providing flexibility and scalability. However, it's susceptible to malicious attacks aiming to steal computing resources or data exfiltration, potentially compromising data integrity and overall security. This study enumerates possible attacks on storage-level services.

- DoS/DDoS: The ultimate purpose of this attack is to do unavailable original services to users and overload the system by flooding spam results in a single cloud server. Due to the high workload, the performance of cloud servers slumps, and users lose the accessibility to their cloud services.

- Phishing: Attackers steal important information in the form of a user's credentials like name, password, etc. after redirecting the user to a fraud webpage as an original page.

- Brute Force attack/ Online dictionary attack: It's one type of cryptographic

hack. Using an exhaustive key search engine, malicious attackers can violate the privacy policy of the data integrity scheme in cloud storage.

- MITC: Man in the cloud attack helps attackers gain the capability to execute any code on a victim machine by installing their synchronization token on a victim's machine instead of the original synchronization token of a victim machine and using this token, attackers get control over the target machine while target machine synchronizes this token to the attacker's machine.

- Port scanning: Attackers perform port scanning methods to identify open ports or exposed server locations, analyze the security level of storage, and break into the target system.

- Identity theft: Using the password recovery method, attackers can get account information of legitimate users which causes loss of credential information of the user's account.

- Risk spoofing: Resource workload balancing is a good managerial part of cloud storage but due to this aspect of cloud computing, attackers can steal the credential data of cloud users, able to spread malware code in host machines, and create internal security issues.

- Data loss/leakage: During data transmission time by external adversaries, incapability of cloud service providers, by unauthorized users of the same cloud environment, and by internal malicious attackers, data can be lost or manipulated.

- Shared technology issue: Compromising hypervisors, cloud service providers can run concurrently multiple OS as guests on a host computer. For the feebleness of hypervisor, attackers create vulnerabilities like data loss, insider malicious attacks, outsider attacks, loss of control on machines, and service disruption by taking control over all virtual machines.

1.1.5 Attacks on Cloud Data Security: A Classification

Based on cloud components, attacks are categorized below [39]:

(a) Network Based Attack

All cloud machines are connected with each other and outsider machines of the cloud environment via a network. Using outsider machines through a network connection, intruders steal or modify sensitive data.

- Port Scanning:It checks the status of executing service on the target machine via a port on a server and causes a DoS service attack on target machines. The intrusion detection systems or firewalls may be used to detect and hinder such attacks [39].

- Botnet Attack: A botnet is used to steal information from a source machine and communicate with a bot master in which commands and control systems are embedded [39]. The main aim of the botnet is to interrupt filtering packets in the network and cause DDoS attacks.

- Spoofing Attack: Malicious attackers modify the source address of IP packets, pretend impersonate to the destination address, and redirect network traffic to an attacker's system. IP spoofing, ARP spoofing, DDoS, and DNS spoofing are examples of spoofing attacks [39].

(b) Storage Based Attacks

Unlike Active attack, a passive attack does not disrupt the normal operation of the

- Data Deduplication Attack: Data deduplication means single-instance storage which minimizes both storage overhead and bandwidth requirements. All redundant data blocks are replaced by a data pointer which is retained in main cloud storage. Using malicious software, attackers can access whole data by stealing the table of data pointers [39].

- Data Scavenging: Data scavenging means during erasing data from a cloud storage device, data are not completely deleted from the storage device and the removed data may be recovered by attackers [39].

(c) Application Based Attack

There are different kinds of intrusion attack and some of the common types of attacks are discussed here:

- Malware Injection and Steganography Attack: A malicious code may be inserted into an application if a cloud platform allows for an insecure interface for application development. With a steganography attack, the attackers embed malicious code within files being transmitted over the network. The transmission of malicious code may then be ignored by security systems for which it seems as if a normal file is being sent. The schemes like StegAD- may be used to identify the files and possible locations of injected code in steganography files [39].

- Web Service and Protocol based Attack: The web services use various protocols such as SOAP whose message header can be manipulated to contain invalid requests. Security policies and validation mechanisms may be implemented to ensure valid requests for the smooth running of services. To cope with possible website-based threats, application firewalls may be used to identify and block the attacks [39].

- Shared Architecture: On a shared architecture, the execution path of a victim's application can be traced. It can be further used to detect the victim's activities and hijack his account. To detect the possibility of being exploited by shared architectures, the application binary code may be analyzed [39].

(d) VM Based Attack

It violates data protection in VMs on a cloud system, exploits vulnerabilities in

VMs, and harms cloud services because multiple VMs are being hosted on a single system.

- VM Creation Attack: Virtual machine means emulation or virtual representation of a physical computer. The hypervisor allows the physical computer to segregate its OS and application from its hardware and form a VM. Sometimes, it can form multiple VMs from one VM by dividing storage, memory, and processors. In these scenarios, vulnerabilities may appear in a cloud environment. If a malicious code is placed in one VM somehow, then it can be replicated to other VMs that are created from the affected VM [39].

- VM Migration and Rollback Attack: When a runtime VM is migrated from one host machine to another machine, the content of the active VM of attack [39].

- VM Scheduler Attack: A few vulnerabilities of the scheduler may result in resource stealing or theft-of-service [39]. For example, a VM may be scheduled to run after a specific time while retaining the credit balance of the VM execution time slice.

- Cross VM Side Channel Attack:Cache and shared memories of a physical server are affected by this type of attack. Attackers can take out information regarding resource usage, cryptographic keys, and other information from a target VM which is residing on the same physical machine as that of the attacker VM [39].

1.1.6 Key Management Techniques in Cloud Storage

To prevent data leakage and increase the difficulty of attack, this paper presents a method combining data distribution and data encryption to improve data storage security. This work has listed here some key techniques used in cloud storage to enhance security and transparency between cloud storage, and cloud users.

- Hierarchical Key Technique: Some research articles[40] provide secret sharing and key hierarchy derivation techniques in combination with user passwords to enhance key security, protecting the key and preventing the attacker from using the key to recover the data.

- Private Key Update Technique: This identity-based encryption technique [41] helps to update the private keys of the non-revoked group users instead of the authenticators of the revoked user when the authenticators are not updated, and it does away with the complex certificate administration found in standard PKI systems.

- Key Separation Technique: This cryptographic method aids in maintaining the privacy of shared sensitive data while offering consumers effective and efficient storage services [42].

- Attribute-based Encryption Key Technique: Instead of disclosing decryption keys, this method achieves the conventional notion of semantic security for data secrecy, whereas existing methods only do so by establishing a lesser security notion [43, 44].

- Multiple Key Technique: This k-NN query-based method improves security by assisting the Data owner(DO) and each query user in maintaining separate keys and not sharing them [45].

Data owners hesitate to trust cloud service providers (CSPs) due to concerns about data integrity and the shared nature of cloud storage. Cloud computing, especially Infrastructure as a Service (IaaS), lacks visibility into data location and processing, posing privacy and security challenges[46, 47]. Malicious users exacerbate data integrity and confidentiality issues, impacting trust in remote cloud storage. Data integrity is crucial for Database Management Systems (DBMS) and transaction properties like ACID. CSPs struggle to guarantee data accuracy and completeness, undermining client trust. This highlights a critical security challenge for cloud storage, essential for the success of cloud computing[48].

Researchers are advancing data integrity in cloud computing through improved verification techniques and privacy-preserving methods. Techniques include Proof of Work (PoW), Proof of Data Possession (PDP), and Proof of Retrievability (PoR). Message Authentication Code (MAC) with a unique key enhances verification, addressing inefficiencies of RSA-based encryption[49]. Provable Data Possession (PDP) validates data possession by a cloud server, while innovations like Transparent PDP [50]and Certificateless PDP refine these algorithms. Proof of Retrievability (PoR) and Proof of Original Ownership (PoW) protocols prevent errors and malicious adversaries. Research continuously introduces improved algorithms[51, 52, 53], such as DHT-PDP[54] and Certificateless dynamic Multiple-Replica PDP[55, 56, 57], aiming to bolster security[58] and integrity in cloud data storage. Fully Homomorphic Encryption (FHE) ensures privacy by encrypting data with operations supported on ciphertext[59]. Somewhat Homomorphic Encryption (SHE) overcomes FHE's complexity[60]. Recent research includes biometrics[61], privacy-preserving auditing for Cloud Storage[62], and data preservation approaches[63].

Recently, Google Cloud has introduced Zebra technologies based on a security command center (SCC) and security operation center (SOC) to point out some harmful threats such as crypto mining activity, data exfiltration, potential malware infections, brute force SSH attacks, etc. to maintain data integrity of business organization's information[64].

Recent years have seen a proliferation of cloud data integrity schemes and survey papers, albeit with limited coverage. Surveys [34] address data auditing, Proof of Retrievability[65], integrity techniques, and protocols[66]. However, they often lack a comprehensive taxonomy and detailed analysis of schemes, security challenges, and design considerations.

1.2 Data Integrity Strategy of Cloud Computing

Data integrity always keeps the promise of data consistency and accuracy of data at cloud storage. Its probabilistic nature and resistance capability of storing data from unauthorized access help cloud users gain trust for outsourcing their data to remote clouds. It consists of mainly three actors in this scheme: Data owner (DO), Cloud Storage/Service Provider (CSP), and Third-Party Auditor(TPA) [34] as depicted in Figure 1.1.

The data owner produces data before uploading it to any local cloud storage to acquire financial profit. CSP is a third-party organization offering Infrastructure as a service (IaaS) to cloud users. TPA exempts the burden of management of data of DO by checking the correctness and intactness of outsourced data. TPA also reduces communication overhead costs and the computational cost of the data owner[67, 68]. Sometimes, DO own-self takes responsibility for data integrity verification without TPA interference. There are three phases in data integrity strategy described below in Table 1.2:

- Data processing phase: In data processing phase , data files are processed in many way like file is divided into blocks[21], applying encryption technique on blocks [74], generation of message digest [71], applying random masking number generation[72], key generation and applying signature on encrypted block [77] etc. and finally encrypted data or obfuscated data is outsourced to cloud storage.

- Acknowledgement Phase: This phase is totally optional but valuable because sometimes there may arise a situation where CSP might conceal the message of data loss or discard data accidentally to maintain their image [72]. But most of the research works skip this step to minimize computational overhead costs during acknowledgment verification time.

- Integrity verification phase: In this phase, DO/ TPA sends a challenge message to CSP and subsequently, CSP sends a response message as metadata

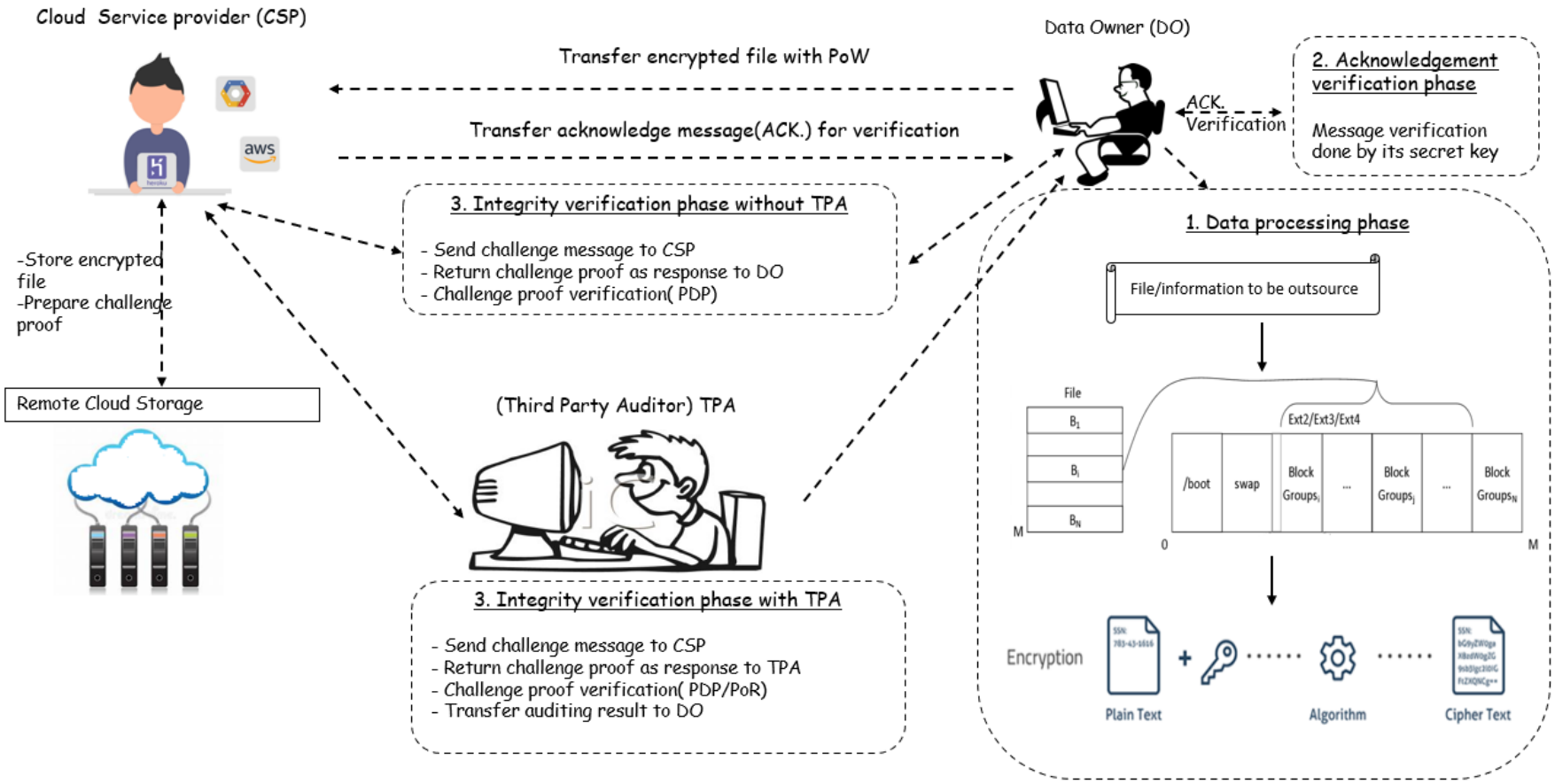

Figure 1.1: Entire Cycle of Data Integrity Technique

Table 1.2: Classified Phases of Data Integrity Schemes

Ref.	Technical Methods	Data Processing Phase			Acknowledgement Phase	Auditing Phase			
		Initial Phase	Key & Signature Generation Phase	Encryption		Using TPA	Using Data Owner /Client	Challenge phase	Proof Verification Phase
[69]	Confidentiality Preserving Auditing	Yes	Yes	Yes	No	Yes	No	Yes	Yes
[21]	Ensuring of confidentiality and integrity data	Yes	No	Yes	No	Yes	No	No	No
[70]	Privacy preserving integrity checking model	Yes	Yes	Yes	No	Yes	No	Yes	Yes
[22]	Verifying Data Integrity	Yes	Yes	Yes	No	No	Yes	Yes	Yes
[71]	Data auditing mitigating with data privacy and data integrity	Yes	No	Yes	No	Yes	No	Yes	Yes
[72]	Public Verification of Data Integrity	Yes	Yes	Yes	No	Yes	No	Yes	Yes
[73]	Ternary Hash Tree Based Integrity Verification	Yes	Yes	Yes	No	Yes	No	Yes	Yes
[68]	Third-party auditing for cloud service providers	Yes	Yes	No	No	Yes	No	Yes	Yes
[74]	Identity-Based Integrity Auditing and Data Sharing	Yes	Yes	Yes	No	Yes	No	Yes	Yes
[75]	A Secure Data Dynamics and Public Auditing	Yes	No	Yes	No	Yes	No	Yes	Yes
[67]	Oruta: privacy-preserving public auditing	Yes	Yes	No	No	Yes	No	Yes	Yes
[76]	Dynamic Auditing Protocol	Yes	Yes	No	No	Yes	No	Yes	Yes
[77]	Dynamic Data Integrity Auditing Method	Yes	Yes	No	No	Yes	No	Yes	Yes
[78]	Algebraic Signatures-Based Data Integrity Auditing	Yes	Yes	No	Yes	Yes	No	Yes	Yes
[79]	Efficient public verification on the integrity	Yes	Yes	Yes	No	Yes	No	Yes	Yes
[80]	Attribute-Based Cloud Data Integrity Auditing	Yes	Yes	No	No	Yes	No	Yes	Yes
[81]	Efficient User Revocation in Identity-Based Cloud Storage Auditing	Yes	Yes	No	No	Yes	No	Yes	Yes
[82]	Secure and Efficient Data Integrity Verification Scheme	Yes	Yes	No	No	Yes	No	Yes	Yes

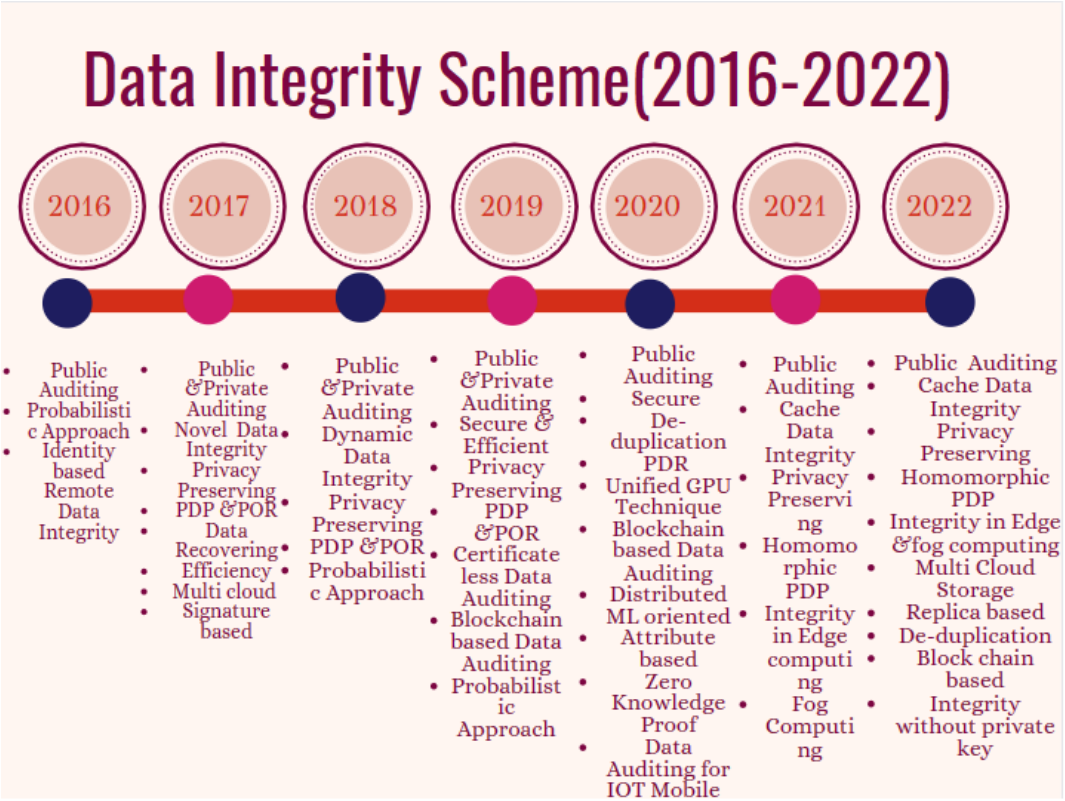

Figure 1.2: Timeline Infographic of Data Integrity

or proof of information to TPA/DO for data integrity verification. The audit result is sent to DO if verification is done by TPA.

In[34], authors have already discussed and shown evolutionary trends of data integrity schemes through a timeline representation from 2007 to 2015 which presented possible scopes of data integrity strategy. Hence, it show a visual representation of all probable trends of the integrity scheme from 2016 to 2022 in the timeline infographic template, Figure 1.2

1.2.1 Characteristics

In this review article, focuses on several quality features of data integrity, which have individually prime importance in cloud storage security. These are:

a) **Public Auditability:** The auditability scheme examines the accuracy of

stored outsourced data from data owner at cloud storage by TPA according to the request of data owners[78, 79].

b) **Audit correctness:** The proof message of CSP can pass the validation test of TPA only if CSP and TPA are being honest and CSP, data owner properly follow the pre-defined process of data storing [73, 81].

c) **Auditing soundness:** The one and only way to pass TPA's verification test is that CSP has to store the data owner's entire outsourced data at cloud storage [74].

d) **Error localization at block level:** It helps to find out error blocks of a file in cloud storage during verification time[73].

e) **Stateless Auditor:** During verification, a stateless auditor is not necessary to maintain, store or update previous results of verification for future usages [72, 79].

f) **Storage Correctness:** CSP prepares a report which shows that all data is entirely stored in cloud storage even if the data are partially tempered or lost. Therefore, the system needs to guarantee data owners that their outsourced data are the same as what was previously stored[83].

g) **Robustness:** In probabilistic data integrity strategy, errors in smaller size data should be identified and rectified [34].

h) **Unforgeability:** Authenticated users can only generate a valid signature/metadata on shared data[83].

i) **Data Dynamic support:** It allows data owners to insert, edit and delete data in the cloud storage by maintaining the constant level of integrity verification support like previous [73].

j) **Dependability:** Data should be available during managing all the file blocks time [73].

k) **Replica Audibility:** It helps to examine the replicas of the data file stored in the cloud storage by TPA on demand with data owners [73].

l) **Light Weight:**It means that due to the occurrence of a large number of blocks and the presence of multiple users in the system, signature process time should be short to reduce the computational overhead of clients[72, 82].

m) **Auditing Correctness:** It ensures that the response message from the CSP side can pass only the verification trial of TPA when CSP properly stores outsourced data perfectly into cloud storage [82].

n) **Privacy Protection:** During verification, the auditing scheme should not expose a user's identity information in front of an adversary[74, 82].

o) **Efficient User Revocation:** The repeal users are not able to upload any data to cloud storage and can not be authorized users anymore [81].

p) **Efficient User Revocation:** Batch Auditing: In the public auditing scheme, batch auditing method is proposed for doing multiple auditing tasks from different cloud users which TPA can instantly perform[79].

q) **Efficient User Revocation:** Data Confidentiality: TPA can not acquire actual data during data integrity verification time [74].

r) **Boundless Verification:** Data owners never give TPA any obligate condition about a fixed no. of verification interaction of data integrity [72].

s) **Efficiency:** The size of test metadata and the test time on multi-owner's outsourced data in cloud computing are both individualistic with the number of data owners[79].

t) **Private Key Correctness:** Private key can pass verification test of cloud user only if the Private key Generator (PKG) sends a right private key to the cloud user[74].

u) **Blockless Verification:** TPA no need to download entire blocks from cloud storage for verification[79].

Table 1.3: Taxonomy of applicable Data Integrity Phases

Type of Proposals				**Category of data**		**Verification Types**				
Deterministic			Probabilistic	Static Data	Dynamic Data	PDP	PoW	PoR	Auditing	
File level Verification	Entire Block Level Verification	Meta data Verification	Random Block level Verification						Public Auditing	Private Auditing
[73, 89]	[71, 90]	[69, 75, 80] [82, 93]	[70, 22, 72] [73, 74, 77] [78, 81, 103] [93, 98]	[91, 92]	[73, 76, 77] [78, 90, 97] [104, 98, 105] [108]	[73, 90, 93] [98, 99, 100] [106]	[72, 94]	[73, 89, 95] [101, 94] [107]	[74, 72, 73] [76, 77, 78] [79, 81, 82] [98, 109] [110]	[71, 90, 96] [102]

1.2.2 Classification

The classification of data integrity relies on various conceptual parameters and sub-parameters, outlined in Table 1.3 with references to elucidate data integrity strategy. Deployment setups vary based on the system environment, including public cloud[84], multi-cloud[85, 86], and hybrid cloud configurations[87]. Each setup poses unique challenges, with multi and hybrid clouds raising significant security concerns regarding data integrity in shared storage structures. Data integrity schemes can be deterministic or probabilistic, with the latter offering better error correction accuracy and lower computational overhead[88], albeit at the expense of overall integrity accuracy compared to deterministic approaches[34].

a) Type of proposal

- File level verification: This is a deterministic verification approach. Here, data integrity verification is generally done by either TPA or the client. The client submitted an encoded file to the storage server and for data integrity verification a verifier verified the encoded file through the challenge key and

secret key which is chosen by the client [89].

- Block Level Verification : This type of verification is a deterministic verification approach. Firstly, a file is divided into blocks, encrypted, generated message digest, and sent encrypted blocks to CSP. Later, CSP sends a response message to TPA for verification and TPA verifies all blocks by comparing the newly generated message digest with the old message digest generated by the client[71].

- Randomly block level verification:This probabilistic verification method involves dividing a file into blocks and generating signatures for each block using hash[70], BLS [72], HLA [111], random masking [72], or ZSS [82]. TPA generates challenge messages for selected blocks, verified by CSP. TPA then verifies proof messages from CSP by comparing signatures [22, 70].

- Metadata verification: In this deterministic approach, firstly cloud users generate a secret key, and using this secret key, cloud users prepare metadata of the entire file through HMAC-MD5 authentication. Later, the encrypted file is sent to CSP, and metadata is sent to TPA. Later this metadata is used for integrity verification via TPA [69].

b) Category of data

- Static data: In static nature, no need to modify data that are stored in cloud storage. In [91], a basic RDPC scheme is proposed for the verification of static data integrity. In remote cloud data storage, all static files are of state-of-the-art nature which gets the main attention but in practical scenarios, TPA gets permission to possess the original data file creates security problems. In [92], the RSASS scheme is introduced for static data verification by applying a secure hash signature (SHA1) on file blocks.

- Dynamic Data: Data owners don't have any restriction policy for applying updation, insertion and deletion operations on outsourced data for unlimited time which is currently stored in remote cloud storage. In [97], a PDP

scheme is introduced by assuming a ranked skipping list to hold up completely dynamic operation on data to overcome the problem of limited no. of insertion and query operation on data which is described in [104]. In [103], dynamic data graph is used to restrict conflict of the dynamic nature of big-sized graph data application.

c) Verification type

- Proof of ownership verification:The Proof of Ownership (PoW) scheme is implemented in the data integrity scheme to verify the data ownership and prevent unauthorized access to outsourced data. It's coupled with a data duplication scheme to mitigate security risks. Different PoW schemes, like s-POW, s-Pow1, and s-Pow2, are defined in [53], each balancing user-side efficiency with the cloud's I/O burden.

- Provable of data possession: The Provable Data Possession (PDP) scheme ensures data integrity verification in cloud storage without downloading data, preventing data leakage attacks. Research describes PDP aspects like computation efficiency, robust verification, and constant communication cost[90]. Certificateless PDP addresses key escrow[99] and key management issues in public cloud storage, enhancing security[100, 106].

- Proof of retrievability verification: Proof of retrievability(PoR) ensures data intactness in remote cloud storage. Both PoR and PDP perform similar functions with the difference that PoR scheme has the ability to recover faulty outsourced data whereas PDP only supports data integrity and availability of data to clients[94]. In [95], IPOR scheme is introduced which ensures 100% retrieval probability of corrupted blocks of original data file. DIPOR scheme also supports data retrieval technique of partial health records along with data update operation [101].

- Auditing verification:Audit verification of outsourced cloud data involves two types: Private auditing (between CSP and data owner) and Public

auditing (CSP and TPA). Various public auditing schemes exist, including privacy-preserving[109, 67], certificate-less[112], optimized[113], bitcoin-based[72], S-audit[94], shared data auditing[67],digital signature(BLS, hash table, RSA etc.) based public auditing scheme [72, 105, 114], and dynamic data public auditing[115]. Additionally, private auditing schemes like SW method[116, 71] have been proposed and reviewed from [117].

1.2.3 Security Challenges in Cloud

Security challenges of data integrity technique in cloud computing always come with some fundamental questions:

- How outsourced data will be safe in a remote server and how data will be protected from any loss, damage, or alteration in cloud storage?
- How security will assure cloud data if a malicious user is present inside the cloud?
- On which location of shared storage, outsourced data will be stored?
- Will legitimate access to the cloud data be by an authorized user only with complete audit verification availability?

All the above questions are associated with the term privacy preservation of data integrity scheme and that's why data integrity in cloud computing is a rapidly growing challenge still now. Refer to Table 1.4, for existing solutions to security challenges and corresponding solutions of data integrity techniques.

a) **Risk to integrity of data**: This security is divided into three parts:

   – during globally acquiring time, cloud services are hampered by many malicious attacks if integrity of database, network etc. are properly maintained.

Table 1.4: Security Challenges of Cloud Storage with its Solutions

Types of Security Issues	**Symptoms**	**Affects**	**Solution with references**
Risk to integrity of data	Unauthorized access, segregation problem of data, lack of maintenance of database	hamperness of cloud storage service, lack of data integrity	Data encryption method, Public data auditing technique[68, 72, 73, 74]
Dishonest TPA	tempering of original file, by the generation of wrong audit message, spoiling of CSP's intention	Lack of data confidentiality, lack of data integrity	hash function with collision resistant property[71], Secure Hash Algorithm (SHA-2)[71], RSA algorithm[75], Threat Model[67] ,panda public auditing(PPA)[118]
Dishonest CSP	Data leakage, data modification, loss of data	loss of reputation of CSP, data unavailability, lack of data integrity	Zero knowledge proof[103],data possession verification scheme[82], string authentication technique[14], firewall policies[16]
Forgery Attack	Forge audit message, forge proof message	Violate data integrity policy, lack of reputation of CSP	soundness criteria[72], one way hash function[67],metadata of storage block order [73], hardness of diffie-Hellman computation in bilinear group[103], pass the challenge proof with non negligible probability[74]
Malicious Insider Attack	Data leakage, data modification, data loss	Violate data integrity policy	string authentication technique[119], digital signature[72, 82, 111, 112]

- Data availability and integrity problems occur if unauthorized changes happened with data by CSP.
- Segregation problem of data among cloud users in cloud storage is another problem of data integrity. Therefore, SLA-based patch management policy, standard validation technique against unauthorized use and adequate security parameters need to be included in data integrity technique [120].

b) **Dishonest TPA**: A dishonest TPA has two prime intentions:

- TPA can spoil the image of CSP by generating wrong integrity verification messages.
- TPA can exploit confidential information with the help of malicious attackers through repeated verification interaction messages with cloud storage.

Hence, an audit message verification method must be included in a data integrity verification scheme to continuously analyze the intentional behavior of TPA

c) **Dishonest CSP**:The adversary CSP has motives to exploit outsourced data for profit or personal use, making detection difficult for data owners. To address this, data integrity schemes should include verification, error detection, and recovery methods to uphold data integrity and confidentiality[73, 121].

d) **Forgery Attack at Cloud Storage**: Outsider attacker may forge a proof message which is generated by CSP for the blocks indicated by challenge message to respond TPA. Malicious auditors may forge an audit-proof that passes the data integrity verification[74, 72].

e) **Data modification by an insider malicious user into cloud storage** : Insider malicious users can manipulate data blocks, deceiving auditors

and owners into trusting cloud storage integrity. To counter this, data integrity techniques should incorporate confidentiality schemes or obfuscation methods[76].

1.3 Motivation

Cloud computing's dynamic and on-demand nature attract users seeking data availability and resource scalability. However, the ease and lower cost also raise concerns of unauthorized access. Despite robust security measures, cloud environments face threats like DDoS attacks, data breaches, and insider thefts, highlighting challenges in data confidentiality, integrity, and authorization.

Reviewing various research papers [2, 69, 72, 122, 123] on cloud services reveals efforts to address specific aspects like data privacy and access control mechanisms. Proposals include improved methods and policy guidelines. However, for service providers' success, ensuring satisfactory services like acknowledgment of stored data to respective owners is essential alongside improvements in other areas.

From our review of research papers[3, 73, 124], it's evident that each study focuses on maintaining data integrity and confidentiality, often through Third Party Auditor (TPA) verification. However, a research gap exists in addressing potential malicious activity by TPAs. There's a need to enhance audit message verification to ensure TPA trustworthiness, preventing deception and data exposure[1]. Proposed solutions should improve cloud security without compromising data privacy[2, 122]. This research aims to introduce new verification techniques to bolster cloud service integrity effectively while safeguarding user data from unauthorized access or manipulation. From the discussions in various research works, it's evident that each researcher endeavors to enhance data integrity verification in the cloud. Papers systematically propose methods with security[72] and computational analysis[74]. Verification success indicates data integrity, but a gap exists in evaluating CSP trust post-failure. Novel trust techniques should assess CSP behavior continuously to ensure trustworthiness amidst potential external

attacks[10, 37]. Integrating these techniques into existing models will bolster cloud service reliability, addressing the need for comprehensive trust evaluation beyond initial service acquisition[2, 1]. This research aims to bridge this gap by incorporating additional trust parameters [125, 126]for continuous CSP trust assessment and system model enhancement.

1.4 Objectives

In light of the aforementioned security concerns and difficulties with the current data integrity mode, the research work's objectives are broken down into the following sections for the proposed work:

- **Modelling a system framework in hybrid cloud environment with all soundness parameters of cloud security:** To establish a strong system frame work with all security parameters likes auditing soundness, protection from external adversary, error localization, dependability, data correctness etc.

- **Develop and Implement an acknowledgement verification technique in same system environment to assure data owner regarding stored data at cloud storage:** To develop an acknowledgement verification technique with security and performance analysis of data verification process in same cloud environment to enlarge aspects of security services.

- **Develop an algorithm of data integrity verification technique with trust parameters in same cloud environment to evaluate trustworthiness of CSP:** To develop a data integrity verification technique to check data correctness at cloud storage by TPA and analysis CSP's trust worthiness for acquiring continuous future cloud services.

- **Develop and add an audit message verification technique in same system environment to evaluate trustworthiness of TPA:** To develop

a data auditing verification technique to check validity of audit message at Data Owner side to evaluate TPA's trustworthiness.

1.5 Methodology

The verification of the data integrity scheme through a third party is comprised of multiple stages. The following points presented briefly in this section serve as a road map for the auditing process of the cloud data.

(i) Review data integrity verfication related papers : Examining relevant papers sheds light on security concerns and attacks on cloud storage, exploring phases, traits, and categorization of data integrity solutions. A comparative study of tactics within cloud storage frameworks is conducted, considering design goals and auditing system overheads. This informs organizations for informed cloud storage decisions addressing performance and security.

(ii) Designing a system model with design goals: There are still security dangers when storing outsourced data in the cloud. Consequently, our research work proposed an effective public data auditing scheme using both stub signatures and ZSS signature that upholds data confidentiality, data privacy, auditing correctness, unambiguity, anonymity, resists collision and forgery attacks, and supports dynamic operations at the sub-block levels in order to address the security issues with cloud storage.

(iii) Pre-processing the Raw Data from The Cloud user's side: During this procedure, the cloud user (CU) determined the file's maximum size and separated it into chunk-size variable pieces. Every block element was then encrypted by CU after it had created pairs of public and private keys. Following that, CU got ready to execute a contract with a cloud service provider to outsource its data.

(iv) Verification of acknowledged data of Cloud Service Provider through Cloud User: The accuracy of the acknowledgment message is achieved

using partial signature on the CU side, guaranteeing that the cloud user's whole file is correctly placed in cloud storage. Furthermore, this useful ACK paradigm shields CU's data privacy from hostile attackers like CSP and other authorised parties during verification in the SCS. It aids in fending off forgeability and replace attacks.

(v) Auditing cloud outsourced data through Third-party Auditor(TPA): We offer a public auditing tool via a TPA to verify the data integrity in shared cloud storage.This auditing system not only maintains data confidentiality, data correctness, auditing correctness, unambiguity, anonymity, etc. during the audit, but it also satisfies the security criteria for protecting the privacy of outsourced data stores in cloud storage.

(vi) Auditing verification of third partie's response through Cloud User: In order to determine whether TPA falsifies the verification result or creates a biassed message without confirming data integrity, we suggested a technique that guarantees an audit message verification test at the cloud user side.

(vii) Recovery of Collude data : At the block index level, we presented the public auditing error localization method. At the DO side, we presented the encrypted data recovery method using a modular technique.

By following this comprehensive methodology and to safeguard data privacy in cloud storage and to reimburse original data through a modular method, we proposed a model for an efficient audit message verification system that utilizes signature-based data integrity and recovery.

1.6 Thesis Organization

The subsequent sections of this thesis are structured into the following chapters:

Chapter 1 establishes the research challenge and framework, introducing concepts like cloud data protection, integrity verification, and data recovery. It emphasizes the roles of symptoms, verification methods, and security challenges in

cloud data integrity. The chapter reviews recent techniques and auditing schemes, offering a comprehensive analysis of the literature which are published in Journal od Cloud Computing,vol.13,pp.-45,2024,doi: 10.1186/s13677-024-00605-z.

Chapter 2 reviews existing literature on cloud data integrity verification schemes, establishing a foundation for understanding different approaches and methodologies.

Chapter 3 proposes a public data auditing approach based on ZSS signatures, ensuring confidentiality and correctness, published in the International Conference on Advanced Computing, Machine Learning, Robotics, and Internet Technologies.

Chapter 4 introduces methods for data security and privacy in shared servers. It includes an outsourced data acknowledgment verification technique using stub signatures, presented in the 6th International Conference on Information Systems and Computer Networks (ISCON).

Chapter 5 proposes a system framework for hybrid cloud environments, adhering to cloud security standards. It introduces a signature-based data auditing technique for efficient verification, detailed in IEEE Access, vol. 12, pp. 58502-58518, 2024, doi: 10.1109/ACCESS.2024.

Chapter 6 presents a signature-based outsourced data auditing scheme ensuring data confidentiality and security while verifying audit messages. It safeguards data uniqueness in cloud storage, resisting external and internal attacks, and is published in IEEE Access, vol. 11, pp. 145485-145502, 2023.

Chapter 7 provides a conclusion to the research, summarizing achievements, reflecting on goals, and suggesting future directions, including Blockchain, fog computing, distributed machine learning, and edge computing. Finally, it is followed by a list of references at the end.

Chapter 2

LITERATURE SURVEY

Numerous studies have explored data integrity and confidentiality, integrating third-party auditors (TPA) to enhance security and fairness. Researchers continually innovate to bolster data privacy and integrity verification in cloud computing, employing methods like Proof of Work (PoW), Proof of Data Possession (PDP), and Proof of Retrievability (PoR). Message Authentication Code (MAC) with unique keys addresses inefficiencies in remote data integrity, while Provable Data Possession (PDP) principles strengthen security by validating data ownership.

This chapter offers a brief overview of past research on third-party data integrity systems, focusing on the methods used in this domain. It provides insights into batch auditing strategies and examines data auditing systems employing cryptography signatures. This review sets the stage for detailed exploration of batch auditing and data integrity verification.

2.1 Background Analysis

Many researchers have given the integrity and confidentiality of user data substantial thought. In order to improve the security and fairness of the data auditing, a TPA is taken into account when designing the auditing protocol. On the other hand, researchers are dedicated to extending their findings to support new functions. In the paper [127], the Researchers suggest a method for verifying the integrity of data based on a short signature algorithm (ZSS signature), that offers privacy and security and public auditing by adding a reliable third party (TPA). Here, by decreasing the hash function complexity in the signature process, the operational complexity is effectively decreased. The study indicates

that the approach is more effective and safer. Using batch auditing, the data integrity auditing scheme this research paper[46] assures the anonymity and integrity of cloud data where the method has the benefit of only requiring one cloud server to support data dynamics and provides favored security characteristics. Based on computational Diffie-Hellman (CDH) assumptions, this study[4] provides a security framework and security verification. According to the research observations, the data integrity audit procedure now includes a blind component and a fog computing layer, which may significantly shorten data transmission lag times while also enhancing data audit security. Researchers develop such a plan and provide a new illustration termed data integrity audits without private key stored in the paper[119]. In this method, Researchers avoid using the hardware token by using biometric information (such as an iris scan or fingerprint) as the user's fuzzy private key, and to verify the user's identity, use a linear sketch with coding and error-correcting procedures. This scheme also provides favored security characteristics. In this research [128], a dynamic data auditing method that keeps up data protection is suggested. Here, in the initialization step, first construct the data authentication structure that is hierarchical and has several branches and secondly, create a data auditing technique, based just on Boneh-Lynn-Shacham digitally signed technique and bilinear coupling mapping technology, and thoroughly outline the data dynamically upgrading procedure. Ultimately, the system assessment process includes security assessment and evaluation metrics. For safe cloud auditing in terms of storage, connectivity, and compute costs, the research work [129], developed a new traditional authentication framework based on Ternary Hash Tree (THT) and Replica-based Ternary Hash Tree (R-THT). TPA will use this framework to carry out data audits. To ensure accessibility and assuring data integrity in the cloud, the proposed method provides auditing at the block, file, and replica levels while using storage block ordering and tree block ordering. Data uploading using proxy-oriented methods is proposed in this papers [130, 48]. Here, the bilinear pairings are used to construct a practical ID-PUIC protocol. On the basis of the computational hardness

of Diffie–Hellman, the ID-PUIC protocol is proven to be secure. The researchers in the paper[49] define formally reversible attribute-based encryption with data intactness protection and present a security architecture (RABE-DI). They provide a specific RABE-DI scheme and demonstrate its integrity and confidentiality inside the specified security architecture. Finally, they demonstrate the effectiveness and applicability of their plan with an execution outcome and effectiveness review. Though numerous research publications suggested efficient techniques to swiftly reply to cloud users in order to maintain a large resource pool [50, 54, 55] but security issues are still present in cloud storage, in this research [131], the researchers suggest a remote data integrity auditing technique that enables data exchange with concealed sensitive data. A sanitizer is utilized in this method to turn the data block's signatures into correct arguments for the cleaned file while also cleaning the data blocks that correspond to the file's sensitive information. In order to fend against outside attackers, researchers offer a traditional authentication approach in that they employ a random masking mechanism rather than secure routes between cloud servers and auditors[132]. In order to fend against outside attackers, researchers offer a traditional authentication approach in that they employ a random masking mechanism rather than secure routes between cloud servers and auditors in [133]. They build a fair challenge message using Bitcoin to deter malevolent auditors from working with cloud servers.

The author of this research paper [52] presented an enhanced kind of a signcryption method based upon the short signature ZSS that may fulfill the aforementioned key data verification criteria. This proposed approach can provide more features for the same computational cost as another ZSS signcryption scheme. The author proposed a blockchain-based system in his research paper [53].BCD-IV is a cloud data integrity verification scheme. TPAs are removed using chain code, which solves collusive attacks. It also created a new homomorphic verification tag (HVT) on the basis of the ZSS signature, which provides blockless verification. The suggested framework enables full dynamic operations with auxiliary data that is kept on a blockchain.

The authors of [59] present a distributed machine learning-focused data integrity verification technique that can solve the key escrow problem while also lowering costs. To assure integrity, this technique is based on identity-based cryptography and two-step key generation technology. The authors use the PDP sampling auditing technique to provide data integrity verification so that the suggested scheme can withstand forgery and tampering assaults.

The author provided a feasible approach for dynamic reviewing in the study paper [60] by using a dynamic list-based index table to verify the integrity of the data, which is more efficient than the state-of-the-art. Furthermore, it has been demonstrated that with a verification structure, communication and storage costs on the client side are effectively reduced. The author introduced a hierarchical trust model to motivate cloud users towards usage of the cloud environment in collaboration of Artificial Bee Colony (ABC) algorithm [123]. Using ABC algorithm, cloud users are capable to find various cloud service provides for providing same services and calculate trusts values of all cloud service providers using Cloud Trust Evaluator (CTE). Here the different cloud entities are divided into three different layers- Cloud User Layer (CUL), Cloud Advisor Layer (CAL) and Cloud Service Provider Layer (CSPL). Author proposed [70] a scheme to prove to be secure against active adversary. This scheme involves three entities: owner of data, TTPA and CSP. In this public audibility data scheme, three phases initialization phase, key generation phase and signature phase are included on the client side. In the research paper [22], author proposed MAC(Message Authentication Code) based integrity checking algorithm which check user's data validation in cloud storage. For that, author introduced three phases: preparation phase, storage phase, verification phase.the research paper [72] proposed a public verification scheme along with a random masking technique instead of secure channels between cloud servers and auditors to resist external adversaries. Using this technique users are able to examine auditors' behaviours to prevent malicious auditors from fabricating verification results and also try to overcome the problem of SWP system.After outsourcing personal data, users don't have

any control over these data. The cloud user is concerned about the integrity of his data stored in the cloud. In research article [71], author established an effective public auditing technique through which third party auditor (TPA) to verify the integrity of data stored in the cloud as it may be attacked by attacker.The research paper [134] established an identity-based signature with light weight and more flexible key management approach to maintain data integrity while physical data is accessed by third party. Using this concept, the key escrow problem for cloud computing can be solved. Author takes up the concept of certificate-based public key crypto systems to attest the fact that a given identity actually belongs to its alleged owner to solve the problem of "unprotected identities". Today's scenario, data privacy, access control and cloud user's privacy are most challenging issues in cloud system. The research paper [7] established a fundamental method to achieve fine grained security with combined approach of Pretty Good Privacy [PGP] and Kerberos in cloud computing. Author used the concept of Kerberos protocol which provides mutual authentication and message integrity as well as data confidentiality. It is based on secret key cryptography and the parameters of this protocols are authentication server(AS), Ticket Granting Server(TGS) and real server to performs secure verification of users and services based on the concept of a trusted third party (KDC) and to prevents eavesdropping or replay attacks.Storing large amount of big data with a high level of fault tolerance is one of mst fabulous feature of cloud computing. Now a days, hybrid and public cloud environment cannot give complete guarantee of data confidentiality while CSPs have full control of cloud user's data. As a result, it looks like lack of data integrity of cloud user's data. In the reasearch paper [21] introduced a method that elaborates the purpose of using separate services outside the cloud environment for authentication, data management and metadata storage to eliminate the possibility of obtaining unauthorized access to data, and the use of metadata to perform integrity control. Using this scheme, data owners have the ability to bounded data access that is stored in an encrypted form and does not allow provider to interact with database.Author established a data protection model

[135], and provides a data access detection algorithm, forms a closed-loop control, provides timely feedback evidence for continuous optimization of data access control strategies, and improves data protection's integrity [135] . Data auditing is a fundamental concept in cloud computing environment (CCE) for maintaining cloud data privacy, integrity data availability and data confidentiality. In public cloud environment, cloud users check their data integrity by either own self or through the help of third party auditor (TPA). In the research paper [69],author introduced a confidentiality preserving auditing scheme by which cloud users can easily verify the risk of the used service from the audit report which is maintained by TPA. This scheme has two benefits.

Vineela.A. et. al [136] established a data integrity auditing scheme for cloud-based medical big data to assist reduce the danger of unwanted data access. Multiple copies of the data are kept to guarantee that it may be restored rapidly in the event of damage. This approach can also be used to allow clinicians to readily track changes in patient's conditions using a data block. The simulation results demonstrated the effectiveness of the proposed method.

Wang.Y. et. al [137] created a Mapping-Trie tree-based data integrity checking scheme. To prevent the leaking of private information, remove the TPA platforms. To audit the data, the method of random sampling is utilized, which considerably reduces the calculation and conveying of resources of cloud users and servers. Both the Hash and Trie schemes have been improved in terms of audit efficiency and storage space savings. Traditional data auditing has issues with data security, processing speed, and communication efficiency. To address these issues, Bian.J. et. al. [47] suggested a data integrity audit scheme based on data blinding. To reduce transmission time, this strategy employs edge devices in the coveying node to construct a fog computing layer between the CSP and the data owner. This paper presented a security model and proof based on computational Diffie-Hellman (CDH) assumptions. The experimental results suggest that including a fog computing layer and a blind element into the data auditing process can effectively minimize data connection delays and increase data audit security.

Zhao.T. et.al. [138] introduced a cloud storage data auditing scheme based on Dynamic Array Multi-branch Tree (DA-MBT) to minimise the height of the tree, enhance node utilisation, simplify the dynamic update process, reduce data block query time, and improve verification efficiency. Finally, the suggested DA-MBT technique allows full dynamic operation, protects user privacy, can successfully verify the integrity of a large number of data, and can significantly reduce communication and processing overhead throughout the verification process. Ali, A.S. et. al [139] suggested a new encryption paradigm as a two-person integrity scheme for cloud-outsourced data is offered here. This strategy employs a trusted third-party manager who serves as a go-between for the data owner and the cloud service provider. The suggested model employs a two-stage encryption process. The key is acquired from the cloud user and data manager in this first stage. The key is significant in the second stage by the data user and the CSP. The suggested system's experimental findings outperform existing cloud security techniques.

Shen.W. et.al. [140] conceived and designed a new trend termed data auditing short of private key storage. To avoid using the hardware token, biometric data is used as the user's fuzzy private key in this technique. It used a linear sketch with coding and error correction methods to certify the user's identity. Furthermore, a novel signature method is proposed here that not only allows blockless verifiability but is also consistent with the linear sketch. This suggested approach delivers desirable security and efficiency, as demonstrated by the security proof and performance analysis.

2.2 Comparative analysis of data integrity strategies

This section presents a comparative study and comparison of data integrity strategies. Table 2.1 shows a comparative analysis of the data integrity strategy of cloud storage for expected design methods with limitations. Zang et.al.[72] introduced

Table 2.1: Comparative Analysis of the data integrity strategy of cloud storage

Year with Ref.	Objectives	Limitations
2023 [141]	Dynamic Proof-of-Storage (PoS) schemes, dynamic on-blockchain auditing, resist all malicious adversary	Due to the missing of data storing acknowledge verification, the reputation of the Cloud server may be destroyed, lack of audit message verification
2023 [142]	Public auditing, scalability, utilization, efficiency,blockchain-based data verification correctness, detection of malicious attack	Insufficient development of energy-efficient consensus mechanisms for data transport in edge computing within the big data environment.
2022 [143]	Correctness of auditing model, unpredictability of tokens, data integrity protection, privacy-preserving, confidentiality, and security of batch verification	Lack of acknowledgment verification, missing data auditing scheme for a multi-cloud environment
2022 [144]	Public auditing, data availability, efficiency, preventing forgery attack	Lack of acknowledgment verification, missing data auditing scheme for a multi-cloud environment, lack of audit message verification
2021 [145]	Public auditing, Public auditing, privacy-preserving, completeness proof, observing CSP's behaviour	Due to the missing of data storing acknowledge verification, the reputation of the Cloud server may be destroyed, lack of audit message verification
2020 [75]	Public auditing, data integrity, dynamic data operation	Acknowledgment message about insert, modification and deletion of data needs to verify otherwise CS may be malicious CS
2020 [146]	Public auditing, dynamic big graph data operation	During verifying time of dynamic graph operations, data privacy is not properly maintained
2019 [82]	Public auditing, reduce computational overhead, resist adaptive chosen-message attack	Validation results need to be verified otherwise TPA may be malicious
2019 [112]	Certificateless public verification	Searching time over encrypted outsourced data in blockchain system takes much time
2019 [109]	Data integrity, resist forge attack	No effective and secure data integrity scheme is present to support the data deduplication process of fog and cloud node

2019 [73]	Public data integrity, error localization, replica level auditing, dynamic update	Due to missing of data storing acknowledge verification, the reputation of CS may be destroyed
2018 [74]	Data integrity auditing, sensitive data hiding	Due to missing of audit message verification scheme, TPA can deceive user about audit message
2018 [69]	Data auditing, privacy-preserving	Audit report needs to verify otherwise TPA may be malicious TPA
2018 [77]	Dynamic update, data integrity auditing, reply forgery and reply attack	An audit message verification scheme needs to be present otherwise TPA may be malicious TPA
2018 [81]	Public auditing, data integrity	Audit message and acknowledge message verification scheme needs to be present otherwise TPA and cloud may be malicious
2017 [22]	Data integrity, resist replay attack and MITC attack	Data privacy issue because after repeatedly passed challenging phase, CSP becomes capable of getting original data block
2017 [115]	Dynamic auditing, dynamic data operation, resist reply attack and replace attack	BLS signature is not suitable for a big data environment
2017 [71]	Public auditing, data integrity	Audit message verification scheme need to be presented otherwise TPA may be malicious
2017 [70]	Data integrity for static data resist from the external adversary	The author assumes that TPA is a trusted one but practically not possible
2016 [147]	Data integrity, privacy-preserving	An audit message verification scheme needs to be present otherwise TPA may be malicious TPA
2016 [72]	Public auditing, resist all external adversaries, protect data from a malicious auditor	Due to the missing of data storing acknowledge verification, the reputation of the Cloud server may be destroyed
2011 [111]	Zero-knowledge public auditing, privacy-preserving	Not applicable for large scale big data and TPA don't have the capability of auditing multiple user's data simultaneously

a random masking technique in public audibility scheme during the computation of proof information generation time. Due to the linear relationship between the data block and proof information, malicious adversaries are capable of inert the effectiveness of the SWP scheme. In the SWP scheme, CSP generates proof information and sends it to TPA for verification. There may be an uncertain situation arise when CSP is intruded on by an external and malicious adversary that can alter every data block's information. To hoax TPA and pass the verification test, a malicious adversary can eavesdrop challenge message and break off the proof message. Therefore, in the SWP scheme, we assume that TPA is the trustworthy element. But practically, it is not possible. To defend against external malicious adversaries without a protective channel, the author proposed here a nonlinear disturbance code as a random masking technique to alter the linear relationship into a nonlinear relationship between data blocks and proof messages. The author applied a BLS hash signature on each block to help the verifier for random block verification. These public audibility verification techniques assure boundless, effective, stateless auditor and soundness criteria with two limitations are that due to the missing data storing acknowledge verification, the reputation of the Cloud services may be destroyed and this scheme is applicable for only static data.

Thangavel et al. [73] proposed a novel auditing framework, which protects cloud storage from malicious attacks. This technique is based on a ternary and replica-based ternary hash tree which ensures dynamically block updating, data correctness with error localization operation, data insertion, and data deletion operations. Shen et al. [74] introduced identity-based data auditing scheme to hide sensitive information at the block level for securing cloud storage during data sharing time. Using this scheme, sanitizer sanitizes data blocks containing sensitive information. Chameleon hash and an unforgeable chameleon hash signature do not provide blockless auditing and require high computational overhead. Hence, this PKG-based signature method assures blockless verification and reduces computational overhead. These public audibility verification techniques assure auditing soundness, private key correctness, and sensitive information hid-

ing one limitation is that due to missing audit messages, TPA can deceive users about data verification. Mohanty et al. [69] introduce a confidentiality-preserving auditing scheme by which cloud users can easily verify the risk of the used service from the audit report which is maintained by TPA. This scheme has two benefits. First, it helps to check the integrity of cloud users' data. Second, it verifies the TPA's authentication and repudiation. In this scheme, the author proposed a system model which supports the basic criteria of cloud security auditing, confidentiality, and availability. HMAC-MD5 technique is used on metadata to maintain data privacy on the TPA side. Chen et. al.[22] proposed MAC oriented data integrity technique based on the metadata verification method which reinforces auditing correctness. These technique helps to protect stored data in cloud storage from MitM and replay attacks. But this scheme needs to improve because, after some repeated pass of challenge-proof messages, CSP will have the ability to get actual block elements of the user's confidential data.

Hiremath. Et. Al. [71] established a public blockless data integrity scheme that secures fixed time to check data of variable size files. For data encryption, the author uses the AES algorithm and SHA-2 algorithm for the data auditing scheme. The author uses the concept of random masking and Homomorphic Linear Authenticator (HLA) techniques to ensure stored data confidentiality during auditing time. But this scheme is only applicable for static data stored in cloud storage. Hence, it needs to expand for dynamic data operations.

Subha. et al.[70] introduced the idea of public auditability to check the correctness of stored data in cloud storage and assume that TPA is a trusty entity. Data privacy mechanisms like Knox and Oruta have been proposed here to grow the security level at cloud storage and resist active adversary attacks. The author uses the Merkle hash tree to encrypt data block elements. Shao. et. al.[77] established a hierarchical Multiple Branches Tree(HMBT) which secures users' data auditing correctness, fulfills the crypto criteria of data privacy, and gives protection against forgery and replay attacks. The scheme uses a special hash function to give BLS signature on block elements and helps in public auditing.DCDV is

a concept based on a hierarchical time tree and Merkle hash tree. Simultaneous execution of access control and data auditing mechanism rarely happens in attribute-based cryptography. Hence, Dual Control and Data Variable(DCDV) data integrity scheme is proposed in[121]. This scheme ensures the solution of the private data leakage problem by the user's secret key and assures the correctness of the auditing scheme.

A PDP technique is proposed for data integrity verification scheme that supports dynamic data update operations, reduces communication overhead for fine-grained dynamic update of Bigdata increases the protection level of stored data at cloud storage, and resists collusion resistance attacks and batch auditing [98]. Another novel public auditing scheme based on an identity-based cryptographical idea ensures low computational overhead from revocated users during the possession of all file blocks. It fulfills the crypto criteria of soundness, correctness, security, and efficiency of revoke users[81].

Some research works introduced BLS cryptographical signature which has the shortest length among all available signatures [72]. This signature is based on a special hash function that is probabilistic, not deterministic. Also, it has more overhead of power exponential and hash calculation. To overcome signature efficiency and computational overhead, a new signature ZSS is proposed [82]. This integrity scheme supports crypto criteria like privacy protection, public auditing correctness, and resisting message and forgery. An attribute-based data auditing scheme is proposed in [147] which proved data correctness and soundness based on discrete logarithm and Diffie-Hellman key exchange algorithm. This scheme maintains the privacy of confidential data of cloud users and resists collusion in blocks during auditing verification time. attacks.

ID-based remote data auditing scheme(ID-PUIC) is introduced here which secures efficiency, security, and flexibility with the help of the Diffie-Hellman problem[84]. It also supports ID-based proxy data upload operation when users are restricted to access public cloud servers. It shows a lower computation cost

of server and TPA than [93]. Both researches works [115] & [91] have worked on public checking of data intactness of outsourced data and reducing communication and computational cost of the verifier. These also support dynamic data auditing, blockless verification, and privacy preservation.

2.2.1 Comparative analysis of desire design challenges of Data Integrity Strategy

Below are the main design issues for data integrity schemes:

a) **Computational overhead**: Data preprocessing, signature generation and audit message verification from data owner side or trusted agent side, challenge message generation, data integrity verification and audit message generation from the TPA side, prof message generation from CSP side all are computational overhead. In [82], the computational overhead of client, CSP and TPA are less than [111] because ZSS signature requires less overhead of power exponential and hash calculation than BLS signature. Table 2.2 compares the computational overhead incurred during public auditing by DO, LCSP, and RTPA. Here, Pair denotes bilinear pairing operatons, Hash denotes hash function, Mul denotes multiplication operation, ADD denotes addition operation, Exp denotes exponential operation, Inv denotes inverse operation, Encrypt denotes encryption operation, decrypt denotes decryption operation, and Sub denotes subtraction operation etc.

b) **Communication overhead**: It means total outsourcing data, which is transferred from client to storage server, transfer of challenge message to CSP, transfer of the proof message to TPA, transfer of audit message to client all are communication overhead. Table 2.3, compares the communication overhead incurred during public auditing by DO, LCSP, and RTPA. Since DO always sends either their original file, an encrypted file, or an encrypted file with a signature to a cloud server, most articles here consider

Table 2.2: Comparison of Computational Overhead between DO, CSP, and TPA During Auditing Phase

Ref.	Data Owner	Cloud Service Provider	Third Party Auditor
[72]	$jHash + (j * k)Exp + jAdd + jExp$	$Hash + kMulExp + kAdd + Exp + (k + 1)Mul$	$Hash + kHash + kMulExp + 3Exp + Mul + 2Pair + Mul$
[73]	$2jHash + 4jExp + 2jAdd + 2jMul + 2jMul$	$Hash + Mul + Exp + Mul + Add + Exp$	$KHash + 2Exp + (k + 1)Mul + Mul + Exp$
[74]	$jHash + jExp + jMul + Add$	$Exp + (k-1)Mul + (k-1)Add + kExp$	$4Pair + 2(k - 1)Add + 2Mul + 2Exp + (k+1)Mul + KHash + (k + 1)Exp$
[148]	$jHash + 2jExp + jAdd + jMul + jMul$	$j + 2Exp + (j + 1)Mul + (k + 1)Exp + kMul$	$4Pair + (k + 2)Exp + (j + 2)Exp$
[81]	Add	$n(2Exp + Mul + Hash$	$KHash + 2Hash + 2(k + 1)Mul + (2k + 3)Exp + 2Pair + (k-1)Add + kMul$
[75]	$4Encrypt + 4Add$	$2Decrypt$	$2Decrypt + Encrypt + Add + Comp$
[82]	$jHash + jMul + jAdd + jInv$	$Hash + 2Add + Mul + Inv + 4Mul$	$Mul + 2Pair + Add$
[149]	$jHash + 2jExp + jAdd + jMul + jMul$	$j * k(Add + Mul + jExp_{G1} + Mul$	$(j + k + 1)Mul + 2Pair + (j + k)Exp$
[78]	$4jMul + jHash + Exp$	$4Mul + Exp$	$kAdd$

Table 2.3: Comparison of Communication Overhead between DO, CSP and TPA During Auditing Phase

Ref.	Data Owner	Cloud Service Provider	Third Party Auditor
[72]	Not Considerable	$log2_c + 160$	$(s+1)p$
[73]	Not Considerable	$2j\|k\| + \|r\|$	$2\|hash\| + 2j\|k\| + 360$
[74]	Not Considerable	$\|p\| + \|q\|$	$c.(\|n\| + \|p\|)$
[148]	Not Considerable	$log2_c$+$(c$+$1)log2_p$	$(s+1)log2_p$
[81]	Not Considerable	$n\|p\| + n\|q\|$	$(c+1).\|q\| + \|p\| + c\|id\|$
[71]	Not Considerable	$j\|k\|$	$j\|Hash\|$
[150]	Not Considerable	$c\|s\| + \|p\|$	$\|p\| + 2\|q\|$
[82]	Not Considerable	$K(\|p\| + \|q\|)$	$2p.(k+q)$
[149]	Not Considerable	$c(\|p\| + \|n\|)$	$(s+1)\|p\|$
[78]	Not Considerable	$\|hash\| + j\|id\| + j\|k\|$	$\|k\|(j+1) + \|c\|$

communication overhead for creating challenge messages and challenge-response messages, which is not included in DO's communication overhead.

c) **Storage overhead**: Entire file or block files, metadata, signature, and replica blocks are required to be stored at cloud storage and at client side depending on the policy of system models. Cloud user storage overhead should be little during auditing verification to save extra storage overhead[151].

d) **Cost overhead**: It denotes the summarized cost of communication overhead, computational overhead, and storage overhead.

e) **Data Dynamic Analysis**: Stored data in cloud storage is not always static. Sometimes, alternation of data, deletion of data or addition of new data with old one are basic functions that come into the practical picture due to the dynamic demanding nature of clients. Therefore, data integrity verification should be done after all dynamic operations on stored data. In [77], the insertion, deletion and updation time of increasing data blocks are less than[110] due to less depth of the authenticated structure of the dynamic data integrity auditing scheme.

Table 2.4: Cloud computing with block chain Technology and its merit with regards to storage level data integrity strategies

Ref.	Issues of Cloud Storage	Merit of Blockchain Technology	Achivements
[152]	In multi-cloud storage environment, majority of the comparable schemes depend on reliable org. like the CSP and the centralised TPA, related and it might be challenging to pinpoint malevolent service providers in the wake of service disputes	used to detect service disputes and accurately identify dishonest service providers, blockchain technology is utilised to record the interactions between users, service providers, and organizers utilized, during the data auditing process	batch verification at a cheap cost without a TPA
[153]	Several TPAs generate challenges for multi-cloud storage, sent to CS to verify data custody. TPAs may dishonestly exploit auditing protocols or collude with CS.	With the usage of blockchain technology, CS might be able to deduce the challenge messages, and there's a chance that user data might be disclosed to the TPA while the audit is being conducted	This ensures decentralized, private audits, allowing public result verification for users
[154]	App development requires data sharing and storage. Functional encryption(FE) solves public-key encryption drawbacks, but requires expensive bilinear pairings.	Cryptocurrency built on the blockchain that allows users to pay third parties when their outsourced decryption is successfully completed	The payment in an FE with outsourced decryption scheme is achieved
[148]	In order to measure cloud data of virtual machines (VMs), two critical concerns in safe IaaS cloud storage are integrity evaluation and decision making.	A two-layer blockchain network can be used to create a revisable user-defined policy-based encryption mechanism and to construct a one-to-one relationship between a user, a node, and a virtual machine.	Enhance data integrity level and aids in controlling the scope of approved verifiers in a flexible manner
[141]	Vast storage proofs and/or vast auditor states are prerequisites for the application of Dynamic Proof-of-Storage techniques designed for traditional cloud storage to Distributed Systems	Static proof-of-success systems promise compact proofs; dynamic on-blockchain auditing protocols can provide concretely tiny auditor states	Index information management is accomplished by optimisation strategies.
[142]	In a multi-cloud storage environment, controlling scalability, data governance,non-tampering, trustworthiness, and transparency are two challenges.	A novel strategy for the security of huge data storage that makes use of highway protocol and blockchain technology to create new blocks that address problems with baseline models	dynamic control over sharing data manipulation is achieved.

2.2.2 Comparative analysis of blockchain-based data integrity

Blockchain technology is a peer-to-peer, decentralised system. It is compatible with distributed and scalable systems where all data are handled as transparent blocks with timestamps to prevent any changes to one data block from affecting all the linked blocks that come after it. Each transparent block also contains the cryptographic hash of the previous block. By enhancing data privacy via the Merkle tree concept, this technology feature preserves the confidence of data owners while optimising cloud storage speed. In [155], a distributed virtual agent model is presented using mobile agent technology to provide cloud data trust verification via multi-tenant and to preserve cloud data reliability. A generic architecture based on blockchain is presented in [156] to improve the security of provenance data in cloud storage, which is crucial for safely accessing cloud data log information. Every research project in [157, 158, 159] aims to use blockchain technology to improve data privacy and uphold data integrity in cloud storage. This article uses blockchain technology to address several cloud storage difficulties, as seen in Table 2.4.

2.3 Summary

This chapter provides a comprehensive assessment of the literature, comparing various methods in the domain of third-party auditing schemes for cloud data. It includes an analysis of relevant research and current state-of-the-art methodologies, along with a comparison of the computational and communication overhead between Data Owner (DO), Cloud Service Provider (CSP), and Third-Party Auditor (TPA) during the auditing phase. Additionally, it explores the integration of cloud computing with blockchain technology and its benefits in terms of storage-level data integrity strategies.

Chapter 3

Signature-based Batch Auditing Verification in Cloud Resource Pool

Online cloud resource pool provides cloud users with a multitude of appealing developments in highly sought-after online scalable storage services for them to eager new creative and investment business benefits. Today, the majority of cloud data security research focuses on ways to increase the correctness of outsourced data audits rather than internal and foreign foes who could hack a cloud server. Even though the data owner enviously envies the data auditing work of stored data to a trustworthy Third Party Auditor(TPA) to avoid communication as well as the computational overhead of data when outsourcing to a reputable Cloud Service Provider, TPA is unreliable in a realistic setting. In this chapter, we offer a basic model for an effective data integrity system based on the ZSS signature, which secures data privacy in cloud storage. Finally, the actual results of the performance of the proposed prototype of the system model were tested and indicated greater efficiency of the provided model in comparison to BLS signature.

3.1 Overview

Due to pay-per-services (IaaS, SaaS, PaaS), flexibility, decentralization, and long list of features, both academic and industrial fields are paying increased attention to the current internet-based era which is known as Cloud Computing. Users can now gain virtual access to cloud computing resources more easily by moving their data to online cloud storage and benefiting from lower cloud service costs without having to deal with the managerial difficulties of handling devices [113, 140]. In the modern era of technology, cloud computing has become a very common

practice for users to use it as a backup repository. In real-world scenarios, cloud computing poses a wide variety of security threats in terms of data privacy, data integrity, and data availability because it facilitates online shared data storage via unreliable cloud service providers [160, 161]. On the flip side of this observation is the possibility that a cloud service provider is actually a malicious attacker who is always trying to make a profit by either deleting data in order to make more space available in the cloud or intentionally sharing important data belonging to data owners with other parties in order to make a profit for their own company. Sometimes, CSPs are trustworthy; however, they may lose data coming from the side of the data owner either as a result of a lack of managerial capabilities [73, 162, 82, 163] or as a result of replacement attacks by authorised others in the same shared storage space [164]. Therefore, cloud service providers conceal this loss on purpose from the owners of the data in order to protect their professional reputation [82]. Because the owners of the data do not have physical control over the data that is outsourced, they are unable to determine who is responsible for improperly handling the data [164, 165, 166]. Previously, data owners used the traditional MAC scheme for data security, but it created a massive communication overhead. Due to the public key-based nature of HLA, a third-party auditor is able to verify the data integrity of outsourced information without the need for a duplicate copy [167, 74]. To ensure data correctness, data intactness, and auditing correctness, researchers are constantly working on the data integrity concept by improving data integrity verification techniques and by enhancing the data privacy-preserving techniques [113, 140, 168]. In data integrity auditing schemes, several researchers have used the provable data possession (PDP) method to shorten the amount of time needed to compute very long files and to shorten the amount of time needed to transfer a very large hash value [168]. Not supported by the PDP scheme are two crucial parameters of the auditing scheme: error localization and data recovery method. Since then, researchers have proposed the PoR method to address these issues [55]. The PoR method guarantees 100% retrieval probability of corrupted blocks of an original

data file and also facilitates dynamic data operations [55, 47]. However, the DO must always keep vital data for the data integrity test and the authorization test in order to pass. It makes the computation more expensive. To create signatures for data blocks and maintain DO data authorization in cloud storage[166], some researchers proposed a bilinear pairing concept. The BLS cryptographic signature was developed by a few studies and is the shortest signature currently in use [169]. This signature relies on a one-of-a-kind hash function that is probabilistic rather than deterministic. It's more work because it requires exponential and hash calculations. Therefore, in light of the above issues, we propose a public data auditing scheme based on ZSS signature signature-based outsourced data auditing strategy for ensuring data confidentiality and privacy while doing data integrity audits and it consumes less overhead computation than BLS signature. Taking the existing scenario into account, and in order to address the aforementioned difficulties, we offer an efficient public data auditing method based on ZSS signatures that maintains data confidentiality, privacy, auditing correctness, and resists collusion and forgery attacks. We describe in this research study a modified ZSS signature-based outsourced data auditing scheme that can protect data security and privacy while performing data integrity batch audits.

3.2 Proposed System Framework

The suggested data auditing architecture is based on a shared cloud resource environment and contains three entities: Cloud Customer(CC), Cloud Resource Pool(CRP), and Auditing Expertise(AE), as shown in Figure 3.1.

Cloud Customer(CC): CC hires storage services from CRP and uploads a considerable volume of data to the shared resource pool to achieve the goal of storing data remotely. CC could be a user inside an organization or an individual user. In this case, CC might be a person, a part of an organization, or even a server.

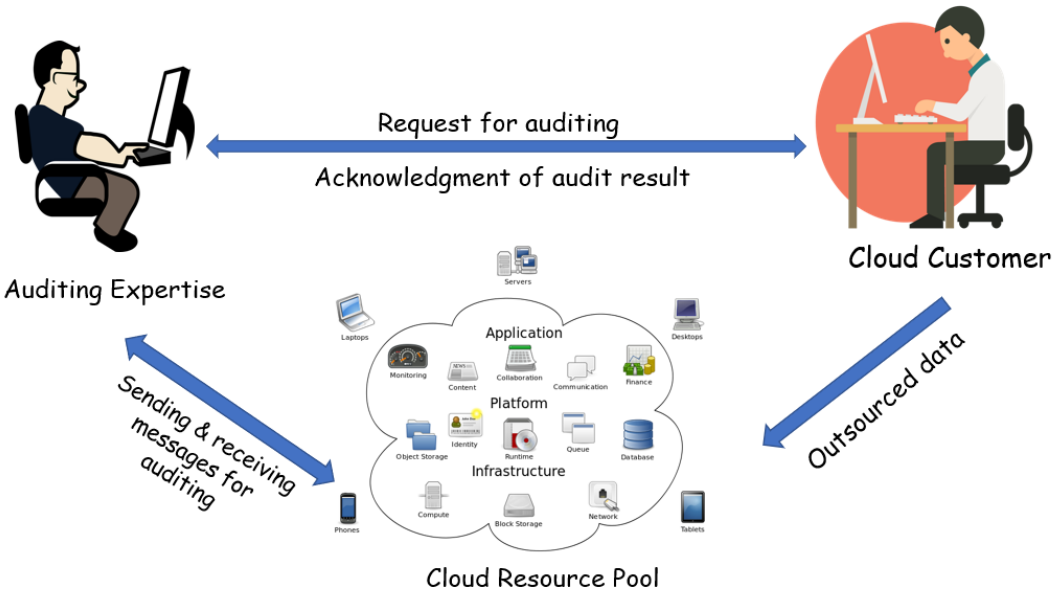

Figure 3.1: Proposed System Model

Cloud Resource Pool(CRP): Public and private cloud storage servers are combined to provide a shared cloud resource pool. The versatility of private and public cloud computing is combined with the efficiency of a regular dedicated server in new cloud storage server technology. CRP receives the outsourced data that CC has uploaded via wireless transmission and provides cloud computing and storage services to CC.

Auditing Expertise(AE): Apart from users and the cloud, AE is an independent third party. The AE is qualified and equipped to conduct cloud data analysis. The AE will create a challenge for the data auditing process after CC files an audit request to them. The AE will then forward the challenge to CRP. The CRP will give proof in accordance with the challenge information. The AE can generate an audit result after reviewing the proof. The AE can be used to audit user data in a cloud system, saving CC compute and storage resources while ensuring the verification process's integrity.

The major goals are to maintain the data integrity of CC's outsourced data while it is being stored on an insecure remote cloud server and to confirm the audit verification results given by AE on the CC end, as summarised above. To

earn money, CC generates data before uploading it to any remote cloud storage service. CRP is an independent company that supplies CC with storage services. CRP stores all CC's outsourced data in a resource pool, an online shared data storage. AE relieves CC of the responsibility of managing its data by confirming the correctness and integrity of outsourced data. We assume in this prototype system model that each entity has limited personal storage and processing power to process data.

- **Working Process**: CC first determines the maximum file size F to be w. The file F is then divided into chunk size variable chunks $B = \{B1, B2, ...Bj\}$, with a maximum size of w and $1 < l < cel(\frac{w}{d})$, $2 \leq d \leq k$. The proposed verification technique consists of six steps: parameter generation, message encryption, signature generation, challenge message generation, response message generation, and signature verification. Please note that our essential architecture is made up of six processes: keyGen, MsgEncrypt, SignGen, QGen, QRes, and SignVeri. The first three operations are located in the Block Data Processing phase, while the next three procedures are located in the Data auditing verification phase. Table 3.1 shows the nomenclature and descriptions used in our proposed model. A thorough description of the two phases as follows:

  i. Block Data Processing Phase

  - $KeyGen(g, r_1, r_2) \rightarrow (CC_{pub}, CC_{pri})$: CC produces a pair of public and private keys accordingly CC_{pub} and CC_{pri} by generating a group generator g_1 from G_1 using a bilinear map and two random numbers r_1, r_2 to generate a pair of private and public key CC_{pri} and CC_{pub} respectively where $g_1 \in G_1$ and $r_1, r_2 \in \mathbb{Z}_q$.
  - $MsgEncrypt(CC_{pri}, \text{M}=(\sum_{t=1}^{j}\sum_{i=1}^{k} B_t[M_i])]) \rightarrow EncryptMessage(M' = (\sum_{t=1}^{j}\sum_{i=1}^{k} B_t[M_i'])]$: CC breaks the original file F into j no. of variable-size blocks $B = \{B_1, B_2, ..., B_j\}$ and encrypts each block elements using the CC_{pri} where $t \in j$ in SHA 2 encry[tion environment

Table 3.1: Definitions of Notations

Notation	Meaning
G_1	Multiplicative Cyclic group with q prime order
r_1	Random no.s
r_2	Secret key
g_1	Group generator
Z_q	Ring of integer modulo q
e	Billinear paring map $e : G_1 \times G_1 = G_2$
F	Original File
ω	File Tag
$B = \{B_1, B_2, ...B_j\}$	Original blocks
$B_t[M_i]$	Block Elements of t^{th} block where $t \in j$,$i \in k$
$B_t[M_i']$	Encrypted Block Elements of t^{th} block
w	Maximum size of a filer
H	Cryptographic Hash Function: $H : \{0,1\}^* \rightarrow \{0,1\}^\lambda$
q	Prime number
k	Total size of a particular block
j	Total no. of block
B_τ	Particular challenged block no where $i \in \tau$
$S = \{S_1, S_2, .., S_j\}$	Signature of all blocks B_j
$B_t[M_i']$	Encrypted blocks where $t \in j$ and $i \in k$
ϕ	Collude value
CC_{pri}	Private key of CC
CC_{pub}	Public key of CC

.

– $SignGen : (B_t[M_i'], g_1, r_1) \rightarrow (S)$: Input file F blocks B and generator parameter g_1, random number r_1, output the signature of data block set $S = \{S_1, S_2, ..., S_j\}$ where

$$S_i = [1/H(B_t[M_j'] + r_1)].g_1 \tag{3.1}$$

CC generates a message as $detail\_msg = \{S, B_j', \omega, g_1, r_1\}$ and send it to CRP.

ii. Data Auditing Verification Phase

Few data blocks are questioned for often correctness of outsourced data over the stored original encrypted data blocks at the resource pool in block-level

auditing. In this case, CC sends a query message to AE for auditing, using $query\_msg = \{B_j, g_1, CC_{pub}, \omega\}$ for verification.

- $QGen : (query\_msg) \rightarrow chal\_pro(\omega, B_\tau)$:Input is an question as a message and output is $chal\_pro$ message and send it to CRP.
- $QRes : (\omega, \tau, S_{1,2,..,\tau}) \rightarrow chal\_res(\omega, sk, \gamma)$:When CRP receives a $chal\_pro$ message, it produces a $chal\_res$ message before sending it to AE where

$$\gamma = \sum_{t=1}^{\tau} r_1/[sk + g1 * H(B_t[M_i])] \tag{3.2}$$

  and $sk = r1 * g1$.

- $SignVer : (S_\tau, \gamma, sk, g1) \rightarrow Result(T/F)$:After receiving $chal_res$ message , AE verify the message as follows:

$$e(g1 * \sum_{t=1}^{\tau} S_t, sk/\gamma) = e(g_1, g_1) \tag{3.3}$$

  The output will be true if the above equation is true; otherwise, the output will be false. After verification, the verification result is sent to CC.

3.2.1 Designing Features

Desire design goals of outsourced data auditing scheme can be summarized as follow:

Public Auditing: According to our suggested public auditing approach, CC outsources critical data to CRP for storage purposes. Later, CC delegated the outsourced data auditing work to a third-party auditor known as AE, who is capable of doing similar activities. This model provides this feature like AE-based partial data verification. This technique, like [168], does not contain additional computing costs for data preprocessing performed by CC throughout the auditing process.

Auditing Correctness: The evidence of CRP can only pass the AE validation test if both CRP and AE are truthful and CRP follows the pre-defined data storage mechanism as described in [82].

Data Confidentiality: This crypto parameter ensures that original data must not disclose in front of either CRP or AE during the auditing process [169].

Unforgeability: If the LCSP properly maintains outsourced data at SCS, then only $chal\_res\_pro$ message can pass the verification test i.e. our system model can resist internal and external forgery attack [145].

Collision Resistance: If stored data at the resource pool is not colluded by CRP, then the signature verification test shows the data intactness [169].

3.3 Security Analysis

This section describes the security parameters of the proposed auditing scheme. The following concepts are introduced: auditing correctness, data confidentiality, data privacy, unforgeability, and collision resistance.

- Auditing Correctness: The evidence of CRP can only pass the AE validation test if both CRP and AE are truthful and CRP follows the pre-defined data storage mechanism. The auditing process is defined as follows using

Equations (3.2) and (3.3):

$$
\begin{aligned}
& e(g1 * \sum_{t=1}^{\tau} S_t, \frac{sk}{\gamma}) \\
& = e(g_1 * \frac{g_1}{H(B_1[M_i'])+r_1} + \frac{g_1}{H(B_2[M_i'])+r_1} + ... + \frac{g_1}{H(B_\tau[M_i'])+r_1}, \frac{sk}{\gamma}) \\
& = e(g_1 * \frac{g_1}{H(B_1[M_i'])+r_1} + \frac{g_1}{H(B_2[M_i'])+r_1} + ... + \frac{g_1}{H(B_\tau[M_i'])+r_1}, \\
& \qquad \frac{sk}{\frac{r_1}{r_1*g_1+g_1*H(B_1[M_1'])} + \frac{r_1}{r_1*g_1+g_1*H(B_2[M_1'])} + .. + \frac{r_1}{r_1*g_1+g_1*H(B_\tau[M_i'])}}) \\
& = e(g_1 * \frac{g_1}{H(B_1[M_i']+r_1} + \frac{g_1}{H(B_2[M_i']+r_1} + ... + \frac{g_1}{H(B_\tau[M_i']+r_1}, \\
& \qquad \frac{g_1}{\frac{1}{r_1*g_1+g_1*H(B_1[M_1']} + \frac{1}{r_1*g_1+g_1*H(B_2[M_1']} + .. + \frac{1}{r_1*g_1+g_1*H(B_\tau[M_i']}}) \\
& = e(g_1, g_1)(\frac{g_1}{H(B_1[M_i']+r_1} + \frac{g_1}{H(B_2[M_i']+r_1} + ... + \frac{g_1}{H(B_\tau[M_i']+r_1}, \\
& \qquad \frac{1}{\frac{1}{r_1*g_1+g_1*H(B_1[M_1']} + \frac{1}{r_1*g_1+g_1*H(B_2[M_1']} + .. + \frac{1}{r_1*g_1+g_1*H(B_\tau[M_i']}}) \\
& = e(g_1, g_1)
\end{aligned}
$$

- Data Privacy: For verification, AE verifies signatures of challenged blocks using the information from the *query_message* from CC and the *chal_res* message from CRP. The true beauty of this proposed paradigm is that AE checks the signature of encrypted block elements that are completely unknown to both AE and CRP. They never know the CCpri or the random number r2. r2 conceals CC's confidential information. As a result, our proposed strategy effectively protects CC's privacy.

- Data Confidentiality: The original block elements cannot be obtained by AE from the *chal_res* message. CRP stores *detail_msg* of all block data pieces from CC. AE is aware of the value of g1, but no one [CRP or AE] is aware of the value of r2. As a result, it is demonstrated that AE never discovers the original CC block elements, and data confidentiality is appropriately

Table 3.2: Comparison of Computational overhead

Entity	Public Auditing using BLS signature [170]	Our Proposed Auditing Model using ZSS signature
CC	$jHash_{g_1} + jMul_{g_1} + jExp_{Z_q} + jExp_{Z_q}$	$WEncrypt_{Z_q} + jhash_{Z_q} + Add_{Z_q} + Inv_{Z_q}$
CRP	$\tau MulExp_{g_1} + Hash_{Z_q} + Exp_{g_1} + CAdd_{Z_q} + (\tau + 1)Mul_{Z_q}$	$\tau(Mul_{Z_q} + Inv_{Z_q} + Add_{Z_q})$
AE	$\tau MulExp_{g_1} + \tau Hash_{g_1} + Hash_{Z_q} + 3Exp_{g_1} + 2Pair_{g_1,g_1} + Mul_{g_1} + Mul_{Z_q}$	$Pair_{g_1,g_1} + Mul_{Z_q} + Inv_{Z_q} + \tau Add_{Z_q}$

maintained on both the CRP and AE sides i.e.,

$$MsgEncrypt(g_1 * r_2, M = \sum_{t=1}^{j}\sum_{i=1}^{k} B_t(M_i)) = (\sum_{t=1}^{j}\sum_{i=1}^{k}(g_1 * r_2) \oplus B_t[M_i])) = (\sum_{i=1}^{n}\sum_{j=1}^{w} B_i[M'_j]))$$

- Unforgeability: The ZSS signature mechanism is used in this proposed approach to establish signatures on each block. AE verifies the signature S_τ of all requested blocks B_τ after receiving the $chal\_res$ message sent by CRP. Because of the security nature of the ZSS signature scheme, the integrity verification cannot pass the AE's audit test if the specific signature is not properly maintained by CRP at resource pool created by CC. As a result, this entire strategy is resistant to both internal and external forgery attempts. The correctness of a valid signature is defined by the bilinear

property as follows:

$$
\begin{aligned}
& e(g_1 * \sum_{t=1}^{\tau} S_t, g_1) \\
& = e(g_1 * \frac{g_1}{H(B_1[M_i'])+r_1} + \frac{g_1}{H(B_2[M_i'])+r_1} + ... + \frac{g_1}{H(B_\tau[M_i'])+r_1}, g_1) \\
& = e(g_1 * \frac{g_1}{H(B_1[M_i'])+r_1} + \frac{g_1}{H(B_2[M_i'])+r_1} + ... + \frac{g_1}{H(B_\tau[M_i])+r_1}, g_1) \\
& = e(g_1 * \frac{g_1}{H(B_1[M_i'])+r_1} + \frac{g_1}{H(B_2[M_i'])+r_1} + ... + \frac{g_1}{H(B_\tau[M_i'])+r_1}, g_1) \\
& = e(g_1, g_1)(\frac{g_1}{H(B_1[M_i'])+r_1} + \frac{g_1}{H(B_2[M_i'])+r_1} + ... + \frac{g_1}{H(B_\tau[M_i'])+r_1}) \\
& = e(g_1, g_1)(S_1, S_2, .., S_\tau)
\end{aligned}
$$

- Collision Resistance: The auditing verification phase of the challenge response message $chal\_res$ from CRP can only pass if CRP does not change the values of genuine block elements $B_t[M_i']$ with corrupted block elements $B_t[\phi_i]$.According to the collision resistance property of the hash function, $H(B_t[M_i']) \neq H(B_t[\phi_i'])$ and $e(H(B_t[M_i']).g_1 + CC_{pub}, S_t) \neq e(H(B_t[\phi_i']).g_1 + CC_{pub}, S_t)$.

3.4 Performance Analysis

The primary goal of this section of the study is to calculate the computational cost of signature generation with block numbers, signature verification with block numbers, and block size (KB). The test conditions are as follows: It has a 12-GB RAM configuration, an Intel(R) Core(TM) i5-9300H processor, and Windows 10. Here computational overhead is evaluated through Python programming language in the Jupiter notebook environment and Figure 3.2, Figure 3.3, and also Figure 3.4 are drawn in an Excel sheet. In this research study, just like in all other research papers [82, 164, 165, 167], we simply consider the time parameter while evaluating the performance of the suggested model. Computational and communication are all taken into account in relation to time.

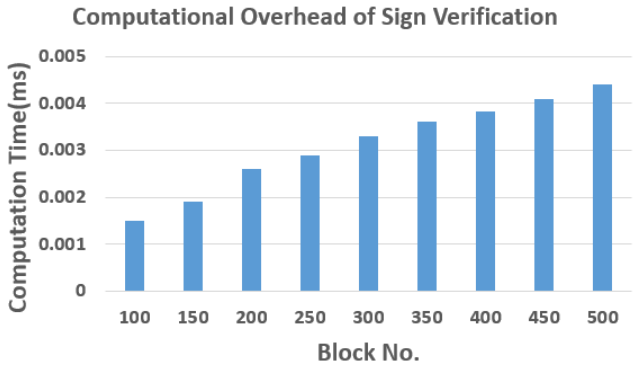

Figure 3.2: Computational Overhead of AE for Signature Verification

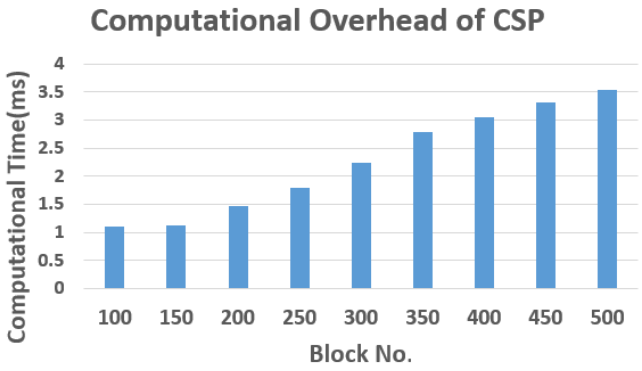

Figure 3.3: Computational Overhead of CRP for preparation of $chal\_res$ message

This section of the research focuses mostly on calculating the computational cost of signature creation time with multiple block numbers at the cloud customer side, signature verification time with multiple block numbers 100,150,200, 250,300,350,400,450, and 500 at the auditing expertise side in Figure 3.2 and Figure 3.3 consequently. Figure 3.4 also depicts the computational overhead of preparing the $chal\_res$ message by CRP.Table 3.2 summarises a comparison of computational overheads of CC,CRP, and AE that aids understanding of the proposed model's efficiency over BLS signature where Mul_{g_1} denotes multiplication operation in g_1, Inv_{Z_q} denotes inversion operation in Z_q,Mul_{Z_q} denotes multiplication operation in Z_q,$Hash_{g_1}$ is used to map Strings which are consistently assigned to the group G1, Exp_{g_1} denotes exponential operation, $Pair_{g_1,g_1}$ denotes pairing operation to verify the signatures, Add_{Z_q} denotes addition operation in Z_q.

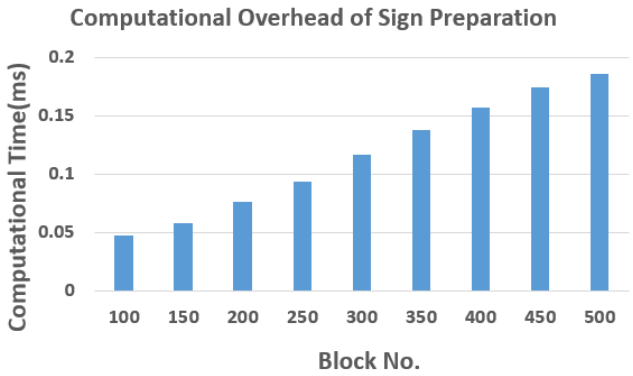

Figure 3.4: Computational Overhead of CC for Signature preaparation

In this case, we take into account three different scenarios: first, CC to CRP communication; second, CC to AE communication; and third, AE to CRP and CRP to AE communication. Similar to the other methods, CC sends a file to CRP to store their data in cloud storage, and at the same time, CC hires AE as an auditor to verify their outsourcer's data integrity. AE prepares a challenge message and sends it to the CRP; after the CSP produces proof of verification and sends it to PTPC, AE then validates the proof that the CRP sent. The first communication cost is O(j), where j is the total number of blocks requested for CRP storage. O(1) is the second communication overhead. A third claim of communication overhead ZSS signature of the τ number of the challenged blocks, $O(\tau)$.

3.5 Summary

The ZSS signature and original random number generation data encryption ensure outsourced data consistency in cloud computing at the resource pool, deterring collusion and forgery attempts. AE identifies and alerts CC of block corruption during auditing, maintaining data privacy from CRP and AE. Our research aims to enhance CC-side audit message verification, fortifying the auditing architecture to safeguard data integrity and privacy throughout the process.

Chapter 4

Acknowledgement Verification of Stored Data in Shared Cloud Resource Pool

Even if CU jealously envies the public data auditing task of stored data to a trustworthy third-party auditor to prevent communication costs as well as computing costs after data outsourcing. Practically speaking, neither TPA nor CSP can be trusted. Data may be deleted by CSPs in order to clear up storage space for financial gain or in order to protect their own reputation among cloud consumers by fabricating data loss instances. Since they do not have physical control over the outsourced data, cloud users are unable to identify the person claiming responsibility for unlawful data treatment. Internal security risks and risks to shared cloud storage still exist. On the other hand, TPA errs on occasion and acts deliberately or is the victim of other illegal activities. As a result, either the audit message might be spoofed or TPA could create a misleading audit message without performing any data verification work, in which case CSP would be up against it. In this chapter, we provide an innovative acknowledgment verification scheme for cloud storage services to resolve these challenging issues.

4.1 Overview

The storage and processing demand on smartphones, laptops, and other terminal devices have surged recently as a result of the amount of information. Some users store their data on the cloud to ease the storage burden on terminal devices to access cloud features [160, 171]. To lessen server load, some cloud service providers may choose to delete some rarely used data. Cloud data loss happens when data that has been deleted cannot be recovered. The cloud server instead

of the local device stores the data that users upload. It has become necessary to verify the accuracy of the data that users upload. Users gain a lot from using the cloud service since it enables them to outsource their data without having to pay expensive hardware and software maintenance costs. Users will lose physical control over their data if they cease storing it locally and instead shift it to the cloud [172]. Data integrity, or whether users' data is preserved on cloud servers, is one of their top security concerns. A cloud service provider may conceal data loss incidents to protect its reputation or delete infrequently used data to free up storage space, all the while maintaining its denial that any data loss has taken place. Additionally, a third-party opponent may falsify user data on cloud servers for commercial or political gain. Users need an effective and secure verification technique as a result to guarantee the accuracy of their input[72, 173].

Because either TPA or CSP may be a malicious entity in a shared cloud environment, we can identify a research gap in all proposed cloud frameworks where every cloud user must intentionally observe or attempt to suggest acknowledgment message verification results efficiently at the cloud user side while maintaining other quality aspects of cloud services through cloud service provider. There are likely opportunities to trick CUs using falsified audit notifications and false data integrity verification results. After numerous repetitions of verification, TPA may occasionally obtain the whole set of data or, using metadata, the original block position number from CSP [2, 163]. This gives TPA the option of using it for its own gain or of disclosing the true data to the public in front of a repudiation attacker or a malicious CSP. The conclusion that can be derived from the aforementioned studies is that in order to ensure ongoing openness between the CU and CSP, the CU must demonstrate that the data was properly saved through the use of verification techniques. In order to properly conduct all verification activities without revealing the original data to anyone, we may now add acknowledgment message verification approaches to the existing capabilities of cloud security services. The following are the primary contributions of the suggested model:

- This research work presented an efficient stub signature-based outsourced

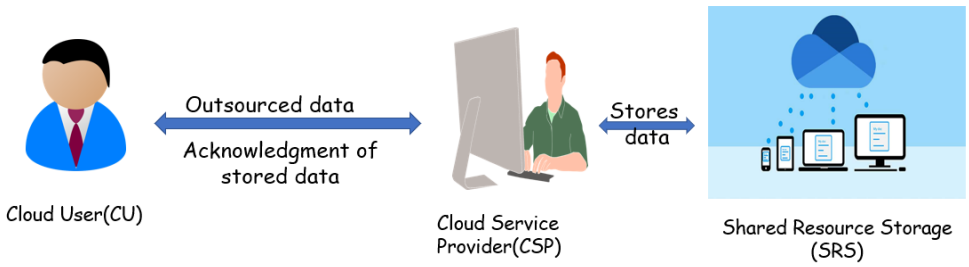

Figure 4.1: Proposed 'ACK' System Model

data acknowledgement verification technique capable of maintaining data security and privacy in a shared resource server.

- This research work presentd an encryption strategy for encrypting original block elements in order to protect data elements from both insider and external hostile assaults at cloud storage.
- This approach prevented forgery and replace attacks on cloud storage, as well as block collusion concerns.

4.2 Proposed System Model

The acknowledgment verification model based on shared cloud resource environment includes three entities: Cloud User, Cloud Service Provider and Shared Resource Storage which depict in Figure 4.1.

Cloud User(CU): For the goal of storing data remotely and allowing any-time access, CU hires storage services and uploads a significant amount of data to the storage server. CU may be a user inside an organization or an individual user. To validate the acknowledgement message received from CSP, CU has a fixed compute capacity.

Cloud Service Provider(CSP): Through wireless communication means, CSP receives the outsourced data that CU has uploaded, offers cloud computing and storage services to CU, and then sends an acknowledgment message that the data

has been stored.

Shared Resource Storage(SRS): A shared resource storage is composed of both public and private cloud storage servers. New cloud storage server technology combines the versatility of private and public cloud computing with the efficiency of a standard dedicated server.

CU breaks a file into blocks in our architecture, encrypts each block with a secret parameter, and then saves the encrypted data on a SRS server. CSP creates and sends an acknowledgement message to CU after storing outsourced blocks in SRS. As a result, in order to ensure the accuracy and availability of the stored data, CU must perform message verification in the cloud storage server. To save money on storage, CU will not store the original data locally. Our proposed system simplifies verification while increasing efficiency. SRS stores data for numerous CUs in multiple copies. It must respond to the challenge as encrypted block components that the CU has begun for storage purposes and provide back proof in the form of an acknowledgement message to the CU.Here, the main goals are to protect the confidentiality of the data that CU has outsourced at the unstable shared cloud server and to validate the acceptance message that CSP gave to CU after the data was saved at SRS. We assume that each entity in this prototype system model has a modest quantity of personal storage and a modest amount of personal processing power to handle data.

4.2.1 Working Process

The maximum size of file F is first determined by CU to be n. Next, file F is divided into chunk size variable blocks $B = \{B_1, B_2, ...B_m\}$, with the maximum size of each block being w and $1 < l < cel(\frac{n}{d})$, $2 \leq d \leq w$. The five steps of the proposed acknowledgement verification technique are parameter generation, message encryption, sign generation, acknowledge generation, and acknowledge verification. Please take note that our fundamental architecture consists of six processes, which we refer to as keyGen, MsgEncrypt, SignGen, CSP_Key, Ack-

Gen, and AckVeri, respectively. The Block Data Processing phase is where the first three procedures are located, and the ACK Data Verification phase is where the next three processes are located. The nomenclature and descriptions utilized in our suggested model are shown in Table 4.1. The following is a detailed description of the two phases:

Table 4.1: Definitions of Notations

Notation	**Meaning**
g	Multiplicative Cyclic group with p prime order
Z_p	Ring of integer modulo p
F	Original File
n	Size of File F
ϕ	File Tag
r_1, r_2	Private parameters of Cu and CSP where $x, b \in Z_p$
CU_{pub}, CU_{pri}	Public & private key of CU
vk	Public parameter of CSP
$B = \{B_1, B_2, ...B_n\}$	Original blocks
$B[M_{ij}]$	Block Elements where $t \in j$,$i \in k$
$B[M'_{ij}]$	Encrypted Block Elements
H	A cryptographic hash function: $H : 0, 1* \leftarrow g$
σ	Secret message of CSP
m	Total block numbers(Block number depend on CU)
w	Dynamic Block Size
sk	Secret parameter of CU
$\Theta[M'_{ij}]$	Collude Block Elements

i. Block Data Processing Phase

- $Key\_Gen(g, r_1, r_2) \rightarrow (CU_{pub}, CU_{pri})$: Enter group generator g as a secret parameter and two random numbers r_1 and r_2 where $g \in \mathbb{Z}_p$ and p is a $\mathbb{Z}$'s prime order, output CU's public key, and private key CU_{pub}, CU_{pri} respectively.

- $MsgEncrypt(CU_{pri}, M = (\sum_{i=1}^{m} \sum_{j=1}^{w} B_i[M_j])]) \rightarrow EncryptMessage(M' =$

$(\sum_{i=1}^{m}\sum_{j=1}^{w} B_i[M'_j])]$: CU divides original file F into variable size blocks $B = \{B_1, B_2, ..., B_m\}$ and encrypts each block element using CU_{pri} key where $i \in m$.

- $SignGen : (B_i[M'_j], g, r_1) \rightarrow (S)$: Input blocks B of a file F and generator parameter g, random number r_1, output the signature of data block set $S = \{S_1, S_2, ..., S_m\}$ where

$$S_i = [1/H(B_i[M'_j] + r_1)].g \tag{4.1}$$

ii. ACK Data Verification Phase

CU prepares $info\_msg$ where $info\_msg = \{B', S, \phi, g, CU_{pub}\}$ and send it to CSP.The CSP should respond to the CU with verification of information stored at SRS after receiving the $info\_msg$ from the CU. The CSP carries out this phase.

- $CSP\_Key(g, CU_{pub}) \rightarrow vk$: The g and CU_{pub} are inputs and output is vk.
- $AckGen(B', g, CU_{pub} \rightarrow H(B'), \sigma)$:The stub, anonymity, and own security parameters are the inputs. CSP prepares σ where

$$\sigma = \frac{H[\sum_{i=1}^{m}\sum_{j=1}^{w} B_i[M'_j]}{g} + vk \tag{4.2}$$

  and send it to CU along with ϕ as an ack_msg.

- $AckVeri(H[\sum_{i=1}^{m}\sum_{j=1}^{w} B_i(M'_j)], g, CU_{pub}) \rightarrow Result(Y/N))$: After receiving the ack_msg, CU prepares sk and verifies the auditing of the message.CU performs verification proof as

$$e(H[\sum_{i=1}^{m}\sum_{j=1}^{w} B_i(M'_j)] * g + sk, \frac{1}{\sigma}) = e(g, g) \tag{4.3}$$

4.2.2 Design Goals

- Data Confidentiality: This encryption option ensures that original data in SRS must not be revealed to CSP or any other nefarious authorized entity during the acknowledgement verification procedure[174].
- Data Privacy: The verification process should not reveal the user's information to any adversary [82].
- Verification Correctness: The validation test of CU can only be passed by the proof of CSP if both CSP and CU are truthful and CSP and CU correctly adhere to the pre-defined procedure of data storing.
- Collusion Resistance: If both CSP and CU are true and accurately follow the pre-established protocol of data storing, the validation test of CU can only be passed by the proof of CSP [73].
- Unforgeability: Only the ack msg message can pass the verification test, demonstrating that our system model can fend off both internal and external forgery attempts assuming the CSP appropriately preserves outsourced data at SRS [82].

4.3 Security Analysis

This section describes the security parameters of the proposed verification scheme like verification correctness, privacy preservation, data confidentiality, and unforgeability. The concepts of collision resistance and replacement attack are also introduced. Here, Table 4.2 closes with a security factor comparison.

- Verification Correctness

  If CSP correctly records the block data component of $info\_msg$, the message is successfully confirmed by CU. Using the suggested verification process, the ack_msg verification process for m number of multiple challenged

data blocks is demonstrated here as follows using Equation (4.2) and Equation (4.3):

$$e(H[\sum_{i=1}^{m}\sum_{j=1}^{w}B_i(M'_j)] * g + sk, \frac{1}{\sigma})$$

$$= e(H[\sum_{i=1}^{m}\sum_{j=1}^{w}B_i(M'_j)] * g + CU_{pub} * g * g, \frac{1}{\frac{H[\sum_{i=1}^{m}\sum_{j=1}^{w}B_i(M'_j)]}{g} + vk})$$

$$= e(H[\sum_{i=1}^{m}\sum_{j=1}^{w}B_i(M'_j)] * g + CU_{pub} * g * g, \frac{g}{H[\sum_{i=1}^{m}\sum_{j=1}^{w}B_i(M'_j)] + g * vk}$$

$$= e(g,g)(H[\sum_{i=1}^{m}\sum_{j=1}^{w}B_i(M'_j)] + CU_{pub} * g, (H[\sum_{i=1}^{m}\sum_{j=1}^{w}B_i(M'_j)] + vk)^{-1})$$

$$= e(g,g)$$

Table 4.2: Comparison table of Security parameters

Scheme	Data Confidentiality	Correctness of $'ack\_msg'$	Collision & Replace Attack Resistance	Data Privacy	Unforgeability
[138]	√	×	√	√	√
[47]	√	×	×	√	×
[175]	√	×	×	√	×
[174]	√	×	×	√	√
[94]	√	×	×	√	×
Our Scheme	√	√	√	√	√

- Privacy Preservation

  This model's real beauty is that CSP maintains encrypted block elements,

which are completely unknown to CSP. They are unaware of the CU_{pri} and the random number r2 which conceals the private information of CU. As a result, our suggested strategy effectively protects CU's privacy.

- Data Confidentiality

  Original block elements from the $info\_msg$ message cannot be obtained by CSP where $info\_mssg$ contains encrypted block elements. Block data items from $info\_msg$ are arrived at CSP side and stored by CSP. $\sum_{i=1}^{n}\sum_{j=1}^{w} B_i(M'_j)$ are a secure version of the original blocks $B = \{B_1, B_2, ..., B_m\}$ that CU has encrypted through its private parameter CU_{pri} i.e. r_2

$$= CU_{pri}(\sum_{i=1}^{n}\sum_{j=1}^{w} B_i(M_j)) = r_2 * g(\sum_{i=1}^{n}\sum_{j=1}^{w} B_i(M_j)) = (\sum_{i=1}^{n}\sum_{j=1}^{w} B_i(M'_j))$$

  Except for CU, neither CSP nor any other malicious attackers are permitted to reveal the value of r_2. As a result, it is established that neither CSP nor any others improperly compromised data secrecy by learning the initial block parts of the CU.

- Unforgeability

  The inherent security of our proposed method prevents the verification process from passing the CU's verification test if block elements, which are generated by CSP at SRS, are not correctly stored there. Therefore, both internal and external forgeries attempts may be resisted by this entire plan. According to our suggested scheme, the accuracy of "ack msg" is as follows:

$$e(H[\sum_{i=1}^{m}\sum_{j=1}^{w} B_i(M'_j)] * g + sk, \frac{1}{\sigma}) = e(H[\sum_{i=1}^{m}\sum_{j=1}^{w} B_i(M'_j)] * g + vk * g, \frac{1}{\sigma})$$

$$= e(g, g)(H[\sum_{i=1}^{m}\sum_{j=1}^{w} B_i(M'_j)] + vk, \frac{1}{\sigma})$$

where $\sigma = \frac{H[\sum_{i=1}^{m}\sum_{j=1}^{w} B_i(M'_j)]}{g} + vk$ which is coming from Equation (4.2).

- Collusion Resistance & Replace attack

  Only if CSP doesn't swap out the values of valid block elements $M' = \sum_{i=1}^{m}\sum_{j=1}^{w} B_i(M'_j)$ for corrupted block elements $M'' = \sum_{i=1}^{m}\sum_{j=1}^{w} B_i(\theta'_j)$ can the verification step of the message 'ack msg' from CSP be passed. Because of its ignorance of r2, CSP is unable to change hash values. sigma will be wrong if CSP alters the elements of outsourced blocks and generates new hash values. Therefore, verification's outcome will be e(g,g). This describes the collision resistance property of the hash function. We can examine this property using Equation 4.3 as follows:

$$e(H[\sum_{i=1}^{m}\sum_{j=1}^{w} B(M'_{ij})] * g + sk, \frac{1}{\sigma}) \neq e(H[\sum_{i=1}^{m}\sum_{j=1}^{w} B(_{ij}')] * g + sk, \frac{1}{\sigma'})$$

$$\Rightarrow e(H[\sum_{i=1}^{m}\sum_{j=1}^{w} B(M'_{ij})] * g + sk, \frac{1}{\frac{H[\sum_{i=1}^{m}\sum_{j=1}^{w} B(M'_{ij})]}{g} + vk}) \neq e(H[\sum_{i=1}^{m}\sum_{j=1}^{w} B(M_{ij})] g + sk, \frac{1}{\frac{H[\sum_{i=1}^{m}\sum_{j=1}^{w} B(\theta_{ij})]}{g} + vk})$$

4.4 Performance Analysis

The main objective of this analysis portion is to determine the computational cost of signature generation with block numbers, signature verification with block numbers, and block size (KB). Additionally looked at in this instance was a comparative simulation study of dynamic data processing. The following are the test conditions: It has a 12-GB RAM configuration, an Intel(R) Core(TM) i5-9300H processor, and Windows 10 installed. This analysis part mainly focuses on determining the computational cost of acknowledgement creation time with multiple block numbers at CSP side, acknowledgement verification time with multiple block numbers, and block size at CU side. Figure 4.2 and Fig-

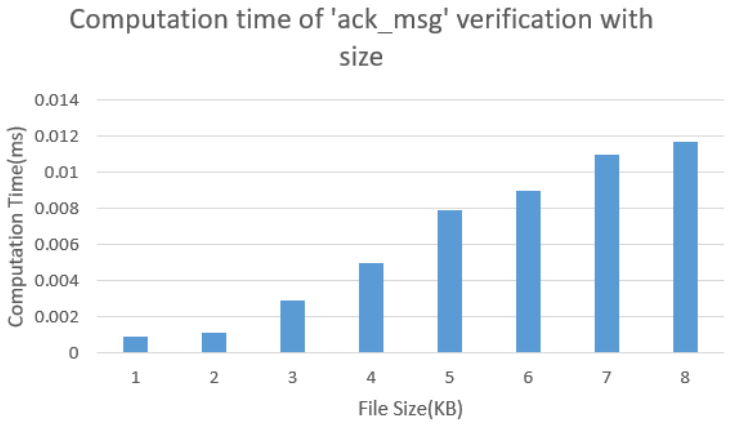

Figure 4.2: Verification Time with Block Size(KB)

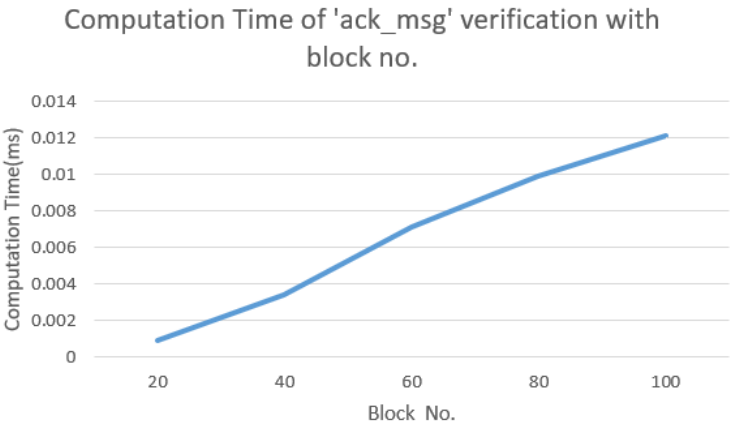

Figure 4.3: Verification Time with Block no.

ure 4.3 shows the suggested scheme's computational cost of signature creation time with different data blocks 20, 40, 60,80, and 100 and with block size from 1KB to 8 KB. The proposed scheme's average computation time is 0.0061 ms for ack_msg verification with size and 0.0066 ms for ack_msg verification for block numbers.Therefore, the computational cost of CU for $'ack\_msg'$ verification is $mHash + mAdd + Add_{Z_p} + 3Mul_{Z_p}$.

4.5 Summary

The accuracy of the acknowledgement message is achieved on the CU side in order to guarantee that the cloud user's complete file is correctly placed in cloud storage. Furthermore, this useful ACK paradigm shields CU's data privacy from hostile attackers like CSP and other authorised parties during verification in the SCS. It aids in fending off forgeability and replace attacks. Our next research project will entail extending this suggested model to evaluate data integrity for several blocks within a single CU and multiple CUs at the TPA's side while utilising the suggested signature.

Chapter 5

Stub Signature-Based Efficient Public Data Auditing System using Dynamic Procedures in Cloud Computing

Cloud data storage is popular for its convenience and scalability, but security concerns persist due to the risk of unauthorized access. To address this, a partial signature-based data auditing system is proposed. This system enhances privacy and accuracy while reducing computational costs by using cryptographic techniques like homomorphic encryption and hash functions. It ensures integrity checks on stored files, thwarting potential tampering attempts by external attackers or malicious insiders. The system supports dynamic operations on outsourced data and has shown effectiveness in real-world applications, as demonstrated by simulation outcomes. It offers data owners assurance that their data is secure and accurate, even in shared cloud storage environments. This solution aligns with the increasing demand for secure cloud solutions and offers a practical approach to mitigating security risks associated with cloud data storage. achieve the desired security qualities, according to the security analysis, and it is effective for real-world applications, as demonstrated by simulation outcomes of dynamic operations on various numbers of data blocks and sub-blocks.

5.1 Introduction

Cloud computing offers businesses diverse benefits like on-demand self-service and dynamic resource scalability. However, outsourcing data to the cloud raises concerns about maintaining data integrity. Various auditing methods, such as global

file integrity auditing, involve signing outsourced data blocks before uploading them to ensure their integrity and proper storage in the cloud, addressing security concerns.

Cloud computing has been visualized as the next-generation information technology (IT) for businesses due to a lengthy list of unmatched benefits in IT's historical past, including on-demand self-service, widespread network services, location-independent dynamic resources, quick resource stretchability, utilization pricing, and threat transmission [176]. Since using the services of cloud allows users to outsource their data without having to pay high hardware and software servicing fees, users benefit greatly from using it. However, if users move their data to the cloud and stop keeping it locally, they will no longer have physical control over it. It is challenging to guarantee the integrity of cloud data because hardware/software faults and human mistakes are unavoidable in the cloud. [172]. To examine whether the data saved in the cloud is unbroken or not and to evaluate whether the information is properly stored in the cloud, a variety of data auditing approaches have been presented [77]. In global file integrity auditing methods, data blocks must first be signed by the cloud user before being delivered to the cloud. The evidence for that reason is provided by such signatures. At this stage of the integrity inspection, these data blocks are genuinely present in the cloud. After that, the data owner uploads these data blocks and their matching signatures to the cloud. In many research works[169, 82, 174], cloud users divide their data into blocks, generating signatures for each block before uploading them along with the original data to shared cloud storage. However, the signature generation process can be time-consuming and raises concerns about data privacy. To address this, a proposed efficient public data auditing scheme using stub signatures ensures data confidentiality, privacy, auditing correctness, and resistance to attacks, supporting dynamic operations at the sub-block level.

The proposed research offers a partial signature-based data auditing method that, while maintaining data security, is more effective and computationally time-consuming for signature creation and verification than related schemes. The

following is a summary of our contributions:

- To validate the data integrity in shared cloud storage, we provide a public auditing mechanism through a TPA. The security requirements for maintaining the privacy of outsourced data stores in cloud storage are met by this auditing system, which also upholds data confidentiality, data correctness, auditing correctness, unambiguity, anonymity, etc. while the audit is being conducted.

- We present a privacy method to protect the original block elements from malicious insider and outsider assaults in cloud storage. This technique stops forgery, substitutes attacks on cloud storage, and resists block collusion issues.

- We extend an existing decentralized security model with our suggested data auditing scheme based on the partial signature.

- The linear data structure is used as the foundation for a cloud data dynamics mechanism. The suggested dynamics operation offers data addition, deletion, and update at the level of the multiple blocks and their sub-blocks.

The following is how this research project is organized: A brief explanation of digital and partial signatures is provided in Section 5.1 . Section 5.2 discusses the proposed system model's overview, design objectives, basic definitions, and dynamic operation. To demonstrate the superior efficacy of the suggested model, Section 5.3 briefly discusses security analysis and performance analysis using simulated results and compares the implemented result with research papers [169, 82, 177, 174]. The research study is concluded in the chapter's summary.

5.1.1 Digital Signature

Three procedures make up a digital signature scheme (DS), which has the following features.

Key Generation Algorithm (SKG): The public key and corresponding secret key are both returned as a pair (vk, sk) by the key generation algorithm SKG.
Signing Algorithm (SIG): The secret key sk and the message M are input into the signing algorithm SIG, which returns a signature S.
Signature Verification Function (SVF): A candidate signature, a message, and a public key are provided to the deterministic verification algorithm SVF, which returns either 1 or 0.

5.1.2 Partial Signature Scheme

Let's say a signer distributes his message to the audience before claiming ownership of it. The signer can calculate "stub", which keeps his identity private[178]. This "stub" is regarded as a partial signature in this case, and only the owner of the public key can verify the partial signature. It is a deterministic approach. Any signature produced by the signing algorithm is essentially a pair (σ, k) in a partial signature scheme $PS = \{PKG, PSIG, PVF\}$, which is a type of digital signature scheme. Here, σ is considered a stub, while k is considered a de-anonymizer. Its three security features are unambiguity, unforgeability, and anonymity. The algorithms for the partial signature scheme $PS = (PKG, PSIG, PVF)$, which was created from the Schnorr identification protocol using the splitting structure, are listed below in Table 5.1.

Table 5.1: Different Sub-parts of Partial Signature Algorithm

Algorithm PKG()	**Algorithm** $PSIG(sk, M)$	**Algorithm PVF** $(vk, M, (\sigma, \kappa))$
$a \leftarrow Z_p$;	$sk \leftarrow a$;	$If A \notin G$ and $\sigma \neq \kappa$ and $\kappa \notin Z_p$
$sk \leftarrow a$;	$B \leftarrow g^b$;	Then return 0;
$A \leftarrow g^a$;	$\sigma = H(A \parallel B \parallel M)$;	$B = g^{\kappa} * A^{\sigma}$;
$vk \leftarrow A$	$\kappa = b + \sigma * a \mod p$;	If $\sigma = H(A \parallel B \parallel M)$
$return(sk, vk)$	$return(\sigma, \kappa)$;	Then return 1;
		else
		return 0;

5.2 Proposed Outsourced Data Auditing

In this section, the system model depicted in Figure 5.1 is presented, encompassing the auditing process, system model overview, basic description, cloud storage security model example, design objectives, and dynamic operations on data blocks. The suggested system comprises Cloud User (CU), Cloud Service Provider (CSP), Public Third Party Checker (PTPC), and Remote Cloud Server (RCS)like [82, 177, 174, 179]. CU encrypts and stores file blocks on a hybrid cloud storage server (RCS). In the batch auditing scheme, CU engages PTPC to verify data block integrity. PTPC challenges CSP, which responds with a reply message verified by PTPC to ensure data integrity

5.2.1 System Model

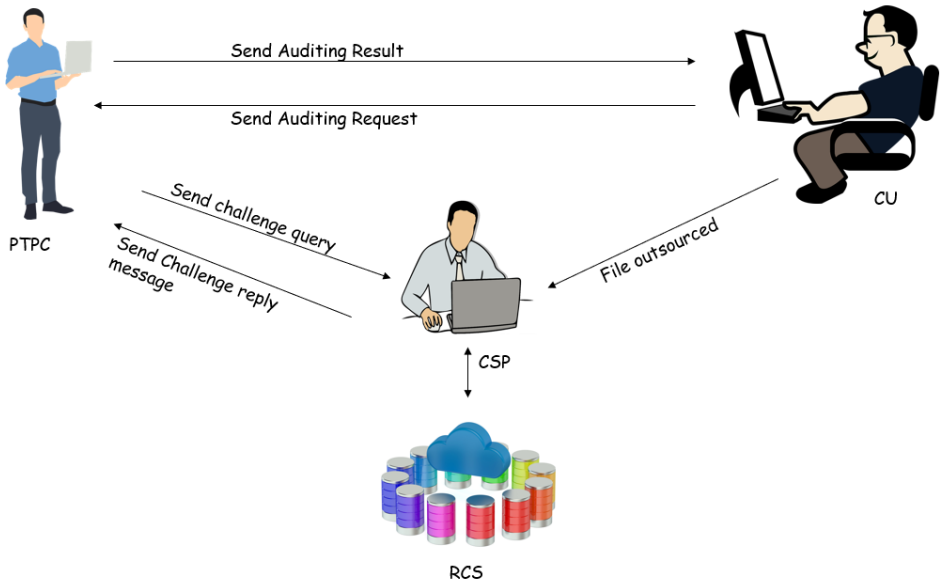

Figure 5.1: Proposed Auditing Model

- **Cloud User (CU):**CUs, or cloud users, upload personal or private data to cloud servers, utilizing various tools and networks for access. Their storage and computing capacities, though limited compared to the cloud, vary across terminals.

- **Cloud Service Provider (CSP):** A remote cloud server can be thought of as a storage and computational pool, giving CU access to an endless number of resources. Numerous redundant and distributed servers are leased by CSP, and these servers deliver various services to CU in accordance with their needs. CSP is honest but not entirely dependable; it is interested in a user's data, particularly sensitive data. CSP will carry out all actions in line with the security assurance protocol while it is active on introducing the system.

- **Public Third-Party Checker (PTPC):** The PTPC, acting independently from users and the cloud, verifies cloud data. Upon receiving an audit request from the CU, it creates a challenge for the CSP, which responds with proof. The PTPC then assesses the proof, providing an audit result. This process conserves CU resources while ensuring verification integrity.

- **Remote Cloud Server (RCS):** An amalgam of both public and private cloud storage servers makes up a Remote Cloud Server. A brand-new remote storage server technology provides both the efficiency of a traditional dedicated server and the adaptability of both private and public cloud computing.

Here, the main objectives, as they are summarised above, are to maintain the data integrity of CU's outsourced data while it is being stored on an insecure remote cloud server and to confirm the audit verification results produced by PTPC on the CU end. In order to make money, CU creates data before uploading it to any remote cloud storage. PTPC is an independent company that provides storage services to CUs. All outsourced data from CU is stored by CSP in RCS, an online shared data storage. By verifying the accuracy and integrity of data that is outsourced, PTPC relieves CU of the duty of managing its data. In this prototype system model, we assume that each entity has a little amount of personal storage and personal computational power to process data.

5.2.2 Design Goal

Desire design goals of the outsourced data auditing scheme can be summarized as follow:

- Data Confidentiality: This encryption option assures that original data must not be disclosed in RCS during the outsourced data integrity verification procedure to either CSP or any other malicious authorized entity[177].
- Public Auditing: A third-party auditor (PTPC) is added to facilitate public auditing in the data validation process based on user authorization and privacy protection. Instead of the user, PTPC requests the cloud storage server's data intactness verification and completes the verification [82].
- Auditing Correctness: It makes sure that only when CSP correctly stores outsourced data into cloud storage can the response message from the CSP side pass the verification trial of PTPC[72].
- Verification Correctness: The validation test of CU can only be passed by the proof of CSP if both CSP and CU are truthful and CSP and CU correctly adhere to the pre-defined procedure of data storing, such as [177, 174].
- Collusion Resistance: If CSP did not collude with the data stored at RCS, the auditing verification test demonstrates the accuracy of the data storage [140].
- Unforgeability: If the CSP correctly maintains outsourced data at RCS, only the $chal\_reply$ message can pass the verification test, proving that our system model can withstand both internal and external forgery attacks[82, 177].
- Unambiguity: Given a stub σ under sk' of the challenged blocks of her choosing, an adversary is prevented from creating κ, vk such that (σ, κ, vk) validates as a signature of m under vk by unambiguity[180, 178].

- Anonymity: It is not recommended to infer the creator of the stub's identity from the stub and message[178].

5.2.3 Overview of the Proposed Auditing Model

The overview of the proposed system model is described below and Table 5.2 describes the definition of all notations:

Table 5.2: Definitions of Notations

Notation	Meaning
g	Multiplicative Cyclic group G with p prime order
$\mathbb{Z}_p$	Ring of integer modulo p
F	Original File
l	Size of File F
ϕ	File Tag
a, b	Random parameter of CU as sk_1 and of CSP as sk_2 where $a, b \in Z_p$
x	Secret parameter of CU where $x \in Z$
vk_1, vk_2	Public parameter of CU as A and of CSP as B accordingly
$B = \{B_1, B_2, ...B_n\}$	Original blocks
$B_i[M_j]$	Block Elements where $i \in n$,$j \in w$
$B_i[M'_j]$	Encrypted Block Elements where $i \in n$,$j \in w$
H	A cryptographic hash function: $H : 0, 1* \leftarrow g$
$\{H(B_1[M'_j]), ..., H(B_n[M'_j])\}$	Hash value of all encrypted blocks
n	Total block numbers(Block number depend on CU
w	Dynamic Block Size
σ	Stub
κ	Anonymity
τ	list of challenged blocks
t	total no. of challenged blocks
$B_i[\theta_j]$	Collude Block Elements

- First, CU determines the maximum size of file F as l, then divides the file F into n no. of chunk size variable blocks $B = \{B_1, B_2, ...B_n\}$, with the

maximum size of each block being w and $1 < l < cel(\frac{l}{d})$, $2 \leq d \leq w$ i.e.each block size is w and it will vary for all blocks.

- CU selects a secret parameter x as sk from p at the start of the scheme. CU and CSP select random parameters a as sk_1 and b as sk_2 respectively where $a, b \in \mathbb{Z}$ with p prime order. CU prepares a public parameter vk_1 as g^a and CSP prepares a public parameter vk_2 as g^b. Here, p denotes $\mathbb{Z}$'s prime order. Here, g is a generator of G and G is a group of prime order p.

- Later each block elements $B_i[M_j]$, $i \in n$, $j \in w$ are encrypted by x to produce modified block elements $B_i[M'_j]$, and CU calculates hash values of each blocks like $\{H(B_1[M'_j]), H(B_2[M'_j]), .., H(B_n[M'_j])\}$, where $j \in w$.

- CU prepares a message $info\_msg$ which contains a file tag ϕ , generator g , encrypted block elements $B_i[M'_j]$ along with public parameter vk_1 and sends it to CSP for storing data.

- CU also sends $\{H(B_i[M'_j]), g, vk_1, \phi, n\}$ as a req_gen to PTPC for data integrity verification.

- Based of req_gen message PTPC generates a challenge message as $chal\_pro$ where it contains challenged block list τ, file tag ϕ and sends it to CSP.

- After verifying the file tag ϕ, CSP prepares stub σ and κ. Later CSP prepares a challenge-response message as a $chal\_reply = \{\phi, \sigma, \kappa, vk_2\}$ and sends it to PTPC to examine the verification test. After receiving $chal\_reply$ message, PTPC verifies the message to ensure the intactness of block elements in RCS, and the verification result will be sent to CU.

5.2.4 Blockchain Technology: Security in Cloud Storage

Our data integrity technique, which is represented in Figure 5.2, can be used in distributed blockchain technology to store information about user transactions

involving digital currencies like bitcoin [155]. Decentralization, autonomy, openness, information modification, and anonymity are features of the underlying technology and infrastructure of this blockchain. All seven layers make up an analogue computer network, and they are how the blockchain system is divided into these layers depicted in Figure 5.2. This blockchain architecture offered a data integrity method that made use of recoverable proof (POR) and proven data holding (PDP) technologies in the consensus layer. It is based on challenge-response in distributed cloud storage systems and data access. The challenge-proof response-based integrity verification mechanism includes both verifiers and responders. The verifier in this case is a TPA, and the responder is a cloud service provider.

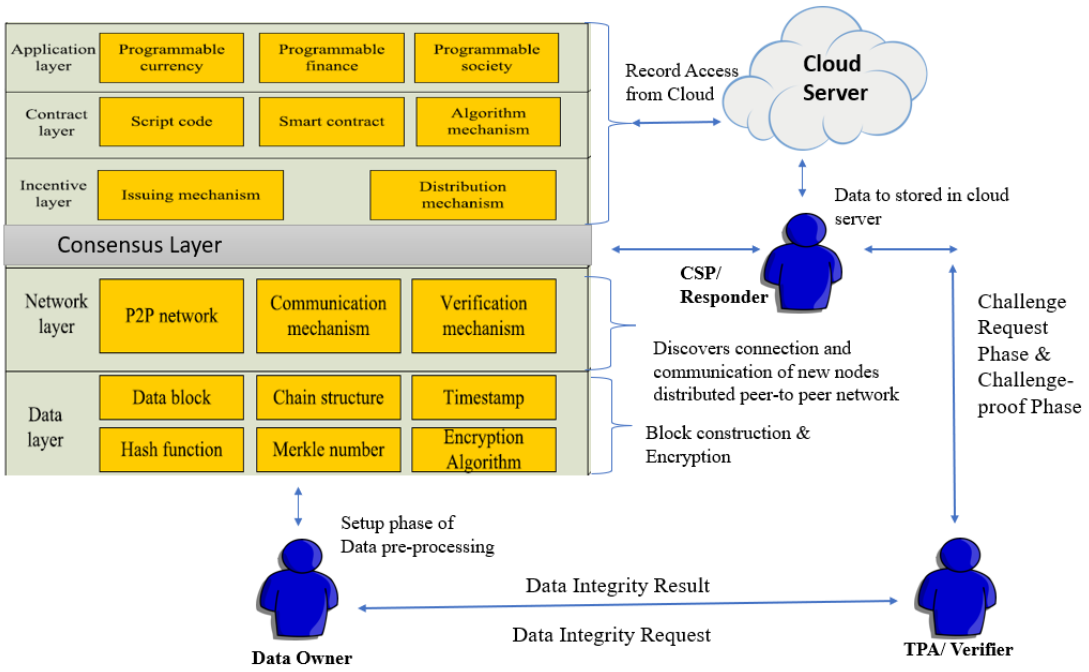

Figure 5.2: Blockchain Infrastructure using Proposed Auditing Model

The challenge-response-based data integrity verification scheme's workflow is divided into the following three stages: The initial step in the setup procedure is for CU to preprocess the data files (by blocking them, creating different tag information, etc.) before sending the data owner's data to the cloud storage server.

Stage 2 of the challenge: TPA generates the relevant challenge data following its own requirements and transmits it to the cloud server. 3. Check Proof stage: CSP produces the appropriate response data following a predetermined protocol and provides it to TPA. Later, TPA calculates using the information from the returned answer to see if the data is accurate.

5.2.5 Basic Signature Verification Scheme

In this research study, we use chunk-sized variable blocks to ignore the problem of a boundary shifting problem [181]. The parameter generation phase, the sign generating phase, and the signature verification phase make up the three phases of the suggested data integrity verification technique. Note that we refer to the six steps in our basic architecture as keyGen, MsgEncrypt, RequestGen, ChalGen, SignGen, and SignVeri, respectively. Our suggested audit approach has been broken down into two parts in this section: Block Data Processing and Block Data Auditing which is depicted in Figure 5.3. The stages will be introduced in depth. The first three steps are under the Block Data Processing phase and the next three steps are under the Block Data Auditing phase.

a) Block Data Processing Stage

$KeyGen() \rightarrow Para(sk, vk)$**:** DO generates a group generator g_1 from G_1 using a bilinear map and two random numbers r_1, r_2 to produce a pair of public and private keys (DO_{pub}, DO_{pri}) where $g_1 \in G_1$ and $r_1, r_2 \in Z_q^*$.DO also prepares a file tag ω for file identification.CU selects a random parameter x from p at the start of the scheme. CU and CSP randomly select private parameters an as sk_1 and b as sk_2 respectively where $a, b \in \mathbb{Z}_p$. CU prepares a public parameter vk_1 as g^a and CSP prepares a public parameter vk_2 as g^b. Here, p denotes $\mathbb{Z}$'s prime order. Here, g is a G's generator and G is a p's group of prime order. CU also

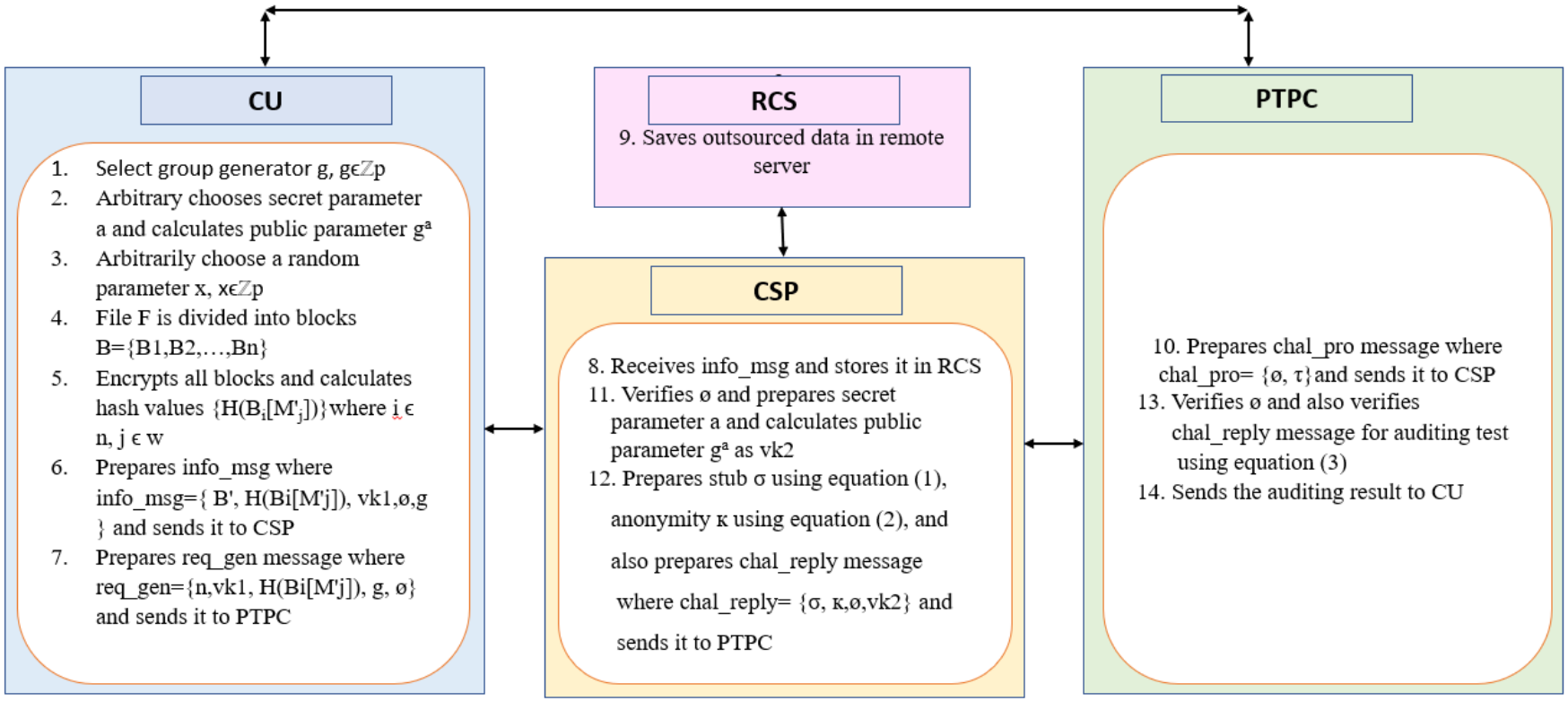

Figure 5.3: Process of Data Auditing at Block Level

prepares ϕ from the combination of A, B, and $\mathbb{Z}_l$ as file identifiers.

$MsgEncrypt(x, B_i[M_j]) \rightarrow CipherMessage(B' = \{B_1[M'_j], B_2[M'_j], ..., B_n[M'_j]\})$**:** CU executes this phase because it generates an encrypted version for each data block. The random parameter and the data blocks are the phase's inputs, and its output is an encrypted set containing all of the data blocks' cipher versions. CU sends the information message $info\_msg = \{B', H(B_i[M'_j]), vk_1, \phi\}$ to CSP where $H(B_i[M'_j])$ is a set of randomized hash values of splitting construction of Schnorr algorithm [178] based on general Md5 or SHA hash family. After receiving $info\_msg$, CSP verifies ϕ and stores it in RCS.

$RequestGen() \rightarrow req\_gen(H(B_i[M'_j]), g, n, vk_1, \phi)$**:** CU executes this phase because it generates a requested message req_gen and sends it to PTPC for public auditing of data blocks where $req\_gen = \{H(B_i[M'_j]), g, vk_1, \phi, n\}$.

b) Block Data Auditing Stage

- $ChalGen(n, \phi) \rightarrow chal\_pro(\tau, \phi, t)$**:**PTPC executes this phase for preparing $chal\_pro$ message to inspect the intactness of CU's data, and sends it to CSP.

- $SignGen(vk_1, vk_2, \tau, B', H(B[M'_{ij}]) \rightarrow stubGen(\sigma, \kappa)$**:**At first, CSP verifies ϕ and CSP should respond to the PTPC with verification of information stored at RCS after receiving the $chal\_pro$ message from the PTPC. CSP carries out this phase. The stub and anonymity are the outputs. CSP prepares stub σ where

$$\sigma = H[\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[M'_j]||vk_1||vk_2)].a^t \tag{5.1}$$

and also prepares the value of anonymity κ where

$$\kappa = (b.H(\sum_{i=1}^{t}\sum_{j=1}^{w} B_i(M'_j) + \sigma.a) + t \tag{5.2}$$

Here, the concatenation operation || is used. CSP prepares challenge reply message $chal\_reply$ where $chal\_reply = \{\sigma, \kappa, vk_2, \phi\}$ to PTPC.

- $SignVeri(\sigma, \tau, H(B_i[M'_j]), t, vk_1, vk_2, g) \rightarrow Result(Y/N)$**:** The PTPC will confirm the accuracy of the stored data blocks through $chal\_reply$ message once it has been received from the CSP. From the value of ϕ, PTPC can identify the CU's file. This phase's inputs are $\{\sigma, \kappa, H(B_i[M'_j]), t, vk_1, vk_2, g\}$, and its output is the auditing verification result for challenged data blocks. PTPC already has hash values of all blocks received from CU, no need to worry about the size of each block, and simply performs verification proof as

$$g^{\kappa} = vk_1^{\sigma} + vk_2^{H[\sum_{i=1}^{\tau}\sum_{j=1}^{w} B_i(M'_j)]} \tag{5.3}$$

and t should be 0.

5.2.6 Dynamic operations at Sub-Block level

CU outsources data to the remote cloud for dynamic operations, especially for frequently modified data. A hash-chaining structure simplifies dynamic operations, allowing multiple block elements in the same position. Hash table chaining handles collisions, with elements inserted into appropriate slots. A hash function determines storage slots for efficient retrieval. A search method confirms element presence, leveraging each slot's reduced components. This enhances search efficiency, making the process more effective for CU's dynamic data operations in the cloud, despite not physically owning the data.

We developed the file structure in this research work using a Python dictionary, an associative data type that allows us to store key-data pairs. The key is handled as a set of block numbers when looking for the associated data values, or block elements. This concept is frequently described as a map. The following defines the abstract data type for maps. The structure is a group of associations between a key and a data value that is not ordered. Because each key on a map

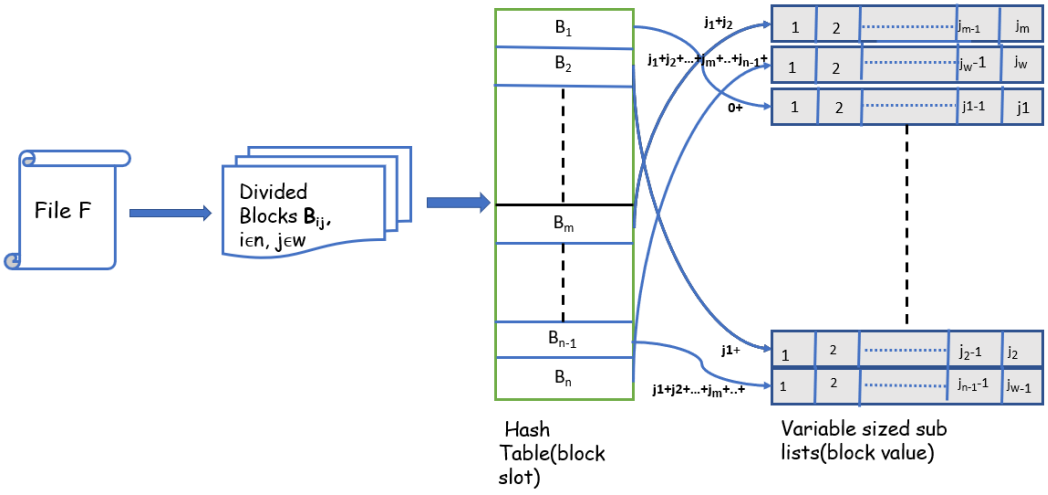

Figure 5.4: The Structure of File using Hash Table

is distinct, there is a one-to-many link between a key and many values.

We build a hash table that implements the map abstract data type using two lists. A parallel list called block value will contain the block elements in such a way that every block no. contains the dynamic size of the sub-list while a list called block slot will hold the block numbers depicted in Figure 5.4.

When we look up a key, the related block elements will be stored in the location that matches the block slot. 2nd list contains all sub-lists collectively. Assume that the first sub-list has a total of j_1 entries and allows spaces in the list from position 1 to j_1; the second sub-list will begin from position $(j_1 + 1)$ to j_2; and so on. Consequently, total file size $n = j_1 + j_2 + ... + j_m + ... + j_w - 1 + j_w$. The locations of requested block elements must first be found in the sub-lists before being deleted in our dynamic operations if we wish to update multiple subblock elements. Finally, we need to add the new block's elements to all of the original block's locations. We use block slots and sub-lists to help with deletion operations by utilizing the location of the block element. When deleting a subblock's elements, if one cloud user desires to remove a block's elements, the appropriate sublist automatically resizes itself and removes the block's element. Find the place in the related sublist if a cloud user wishes to insert a block element at a specific location in the sublist. Shift the element from this location to the

last position in a temporary list and add a new element at the precise location, then append all the shifted items to the sublist after the specified location. The pseudo-code of our proposed method has been shown in Algorithm 5.1, Algorithm 5.2, and Algorithm 5.3. For an explanation of algorithms, we used *uu* as List of Block no.s, up as Block index, and *val* as the updated Sub-Block element for Block.

Here, we assume that the hash table's load factor μ is low and that the likelihood of collisions is low. We suggested chaining in this article in terms of the dynamic size of sub-lists to prevent problems with collision resolution in sub-lists. Normal hashing would demand constant time $O(1)$for searching in the best-case scenario. $(1+\mu/2)$ is the typical number of comparisons if the search is successful, and just μ comparisons if it failed.

We have mentioned of pseudo process of all dynamic operations in Algorithms 5.1, 5.2, and 5.3 accordingly. In order to adjust the sub-block components as well as the sub-block position range, CU first builds a list of the block numbers we want to change, arranging them in ascending order. This list is labeled a "uu" list and an "up" list, respectively. A separate list called a "val list for list of elements" must be prepared by the central unit (CU) for update and insert operations. CSP does not require an additional list for removal operations. CU sends lists to CSP based on operations. After receiving it, CSP scans $info\_msg$, locates the mentioned block numbers of the 'uu' list, and detaches all mentioned blocks from their hash value because hash values need to be altered after performing any operation on the sub-block components of mentioned blocks. Later, CSP determines the length of each block indicated in the "uu" list. For the insert operation, CSP stores the sub-block elements of the mentioned z1 blocks from the up position to the last position in a temp list and inserts the sub-block elements from its positions using a val list after concatenating the first part of z1 with the temp list as described in *Algorithm 5.3*. When performing a delete operation, CSP removes z1 block's sub-block elements from the locations specified in the 'up' list described in *Algorithm 5.2*. For update operations, CSP uses the 'val'

list described in *Algorithm 5.1* to update the sub-block elements of the z1 block from the places specified in 'up' list. After completing any requested operation on the CU side, CSP updates the hash value of the relevant blocks in $info\_msg$ along with the updated version of those blocks. Later, all new hash values are sent to CU for additional data integrity verification in accordance with the 'uu' list.

Algorithm 5.1: Process of Update Operations at Sub Block Level

Input: uu,up,val, $[uu \neq 0 \vee up \geq 0 \vee val]$
Output:Update Sub-Block's element out

Initialisation :

1: **for** $i = 0$ to n **do**
2: **if** $i = uu[i]$ **then**
3: $x_1 = list1[i]$
4: $y_1 \rightarrow x_1.split(",")$
5: $z_1 \rightarrow y1[:-1]$
6: **else**
7: print("Block no not found")
8: **end if**
9: **for** $j = 0$ to n_1 **do**
10: **if** $j = up[j]$ **then**
11: $z_1[up] = val[0]$
12: $val.pop(0)$
13: **else**
14: print("Sub -Block location not found")
15: **end if**
16: **end for**
17: **end for**

Algorithm 5.2: Process of Delete Operations at Sub Block Level

Input: uu,up, $[uu \neq 0 \vee up \geq 0]$

Output:Update Sub-Block's element out

Initialisation :

1: **for** $i = 0$ to n **do**
2: **if** $i = uu[i]$ **then**
3: $x_1 = list1[i]$
4: $y_1 \rightarrow x_1.split(",")$
5: $z_1 \rightarrow y1[:-1]$
6: **else**
7: print("Block no not found")
8: **end if**
9: **for** $j = 0$ to n_1 **do**
10: **if** $j = up[j]$ **then**
11: $z_1.pop(up)$
12: **else**
13: print("Sub -Block location not found")
14: **end if**
15: **end for**
16: **end for**

Algorithm 5.3: Process of Insertion Operations at Sub Block Level

Input: uu,up,val $[uu \neq 0 \vee up \geq 0 \vee val \neq 0]$

Output:Update Sub-Block's element out

Initialisation :

1: **for** $i = 0$ to n **do**
2: **if** $i = uu[i]$ **then**
3: $x_1 = list1[i]$
4: $y_1 \rightarrow x_1.split(",")$
5: $z_1 \rightarrow y1[:-1]$

```
    else
        print("Block no not found")
    end if
    for j = 0 to n_1 do
        if j = up[j] then
            temp = z_1[up : n_1]
            z_1[up] = val[0]
            val.pop(0)
            z_1.append(temp)
        else
            print("Sub -Block location not found")
        end if
    end for
end for
```

5.3 Analysis of Proposed Model

The suggested auditing scheme's security parameters are detailed in this section. Data confidentiality, unforgeability, collusion resistance, replica attack resistance, unambiguity, and anonymity are then introduced. Table 5.3 summarizes a comparison of security features which helps to understand the efficiency of the proposed model than others.

5.3.1 Security Analysis

We assess the proposed scheme's security by examining its fulfillment of the security guarantee given in Section 5.2.2: data confidentiality, auditing correctness, collision resistance, unforgeability, unambiguity, and anonymity property.

- **Data Confidentiality** Original block elements from the *chal_reply* mes-

Table 5.3: Comparison table of Security parameters

Scheme	Data Confidentiality	Auditing Correctness	Collision resistance	Public Auditing	Unforgeability	Unambiguity	Anonymity
[172]	×	√	×	√	√	×	×
[82]	×	√	×	√	√	×	×
[169]	√	√	×	√	√	×	×
[177]	×	×	√	√	×	×	×
[111]	×	√	×	√	√	×	×
[140]	√	√	√	√	√	×	×
[171]	√	√	√	√	√	×	×
[182]	√	√	√	√	√	×	×
[183]	√	√	√	√	√	×	×
Our Scheme	√	√	√	√	√	√	√

sage cannot be obtained by $PTPC$ where $chal\_reply = \{\kappa, \sigma, vk_2, \phi\}$. Block data items from $info\_msg$ are arrived at CSP side and stored by CSP. $\sum_{i=1}^{n}\sum_{j=1}^{w} B_i[M'_j])$ are a secure version of the original blocks B that CU has encrypted through its private parameter x i.e.

$$\begin{aligned} MsgEncrypt(x, M) &= \sum_{i=1}^{n}\sum_{j=1}^{w} B_i(M_j)) \\ &= (\sum_{i=1}^{n}\sum_{j=1}^{w} x \oplus B_i[M_j])) \\ &= (\sum_{i=1}^{n}\sum_{j=1}^{w} B_i[M'_j])) \end{aligned} \tag{5.4}$$

Here, $\oplus$ is a modification operation performed by x with each and every block element. Except for CU, neither CSP nor $PTPC$ are permitted to reveal the value of x. As a result, it is established that neither CSP nor any other adversaries in RCS improperly compromised data secrecy by learning the initial block parts of the CU.

- **Collision Resistance:** The auditing verification step of the challenge response message $chal\_reply$ from CSP may only be passed if CSP doesn't

modify the values of valid block elements $\sum_{i=1}^{t}\sum_{j=1}^{w} B_i(M'_j)$ for corrupted block elements $\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[\theta_j]$. Due to a lack of knowledge regarding x, CSP is unable to alter hash values. Stub σ and anonymity κ will be incorrect if CSP modifies the elements of challenged blocks and generates new hash values. Therefore, the verification's outcome will be false. This is the hash function's characteristic of collision resistance. From Equation(5.3), we can see that

$$\begin{aligned} & vk_1^{\sigma} + vk_2^{H(\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[M'_j])} \neq vk_1^{\sigma'} + vk_2^{H(\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[M'_j])} \\ \Longrightarrow\ & g^{a*\sigma} + g^{b*H(\sum_{i=1}^{\tau}\sum_{j=1}^{w} B_i[M'_j])} \neq g^{a*\sigma'} + g^{b*H(\sum_{i=1}^{\tau}\sum_{j=1}^{w} B_i[M_j])} \\ \Longrightarrow\ & g^{a*H(\sum_{i=1}^{n}\sum_{j=1}^{w} B_i[M'_j]||vk_1||vk_2).a^i} + g^{b*H(\sum_{i=1}^{\tau}\sum_{j=1}^{w} B_i[M'_j])} \\ & \neq g^{a*H(\sum_{i=1}^{\tau}\sum_{j=1}^{w} B_i[\theta_j]||vk_1||vk_2).a^i} + g^{b*H(\sum_{i=1}^{n}\sum_{j=1}^{w} B_i[M'_j])} \end{aligned} \tag{5.5}$$

Hence, $L.H.S \neq R.H.S$ Our suggested strategy can thereby prevent collisions.

- **Unforgeability:**

  Only the intactness of all challenged block components is proven by PTPC's verification test if CSP properly saves all block elements of CU's outsourced data(We need to demonstrate that a malicious authorized attacker cannot falsify proof (σ, κ) to deceive the PTPC, and a malicious CSP cannot falsify outsourced data for the verification test.).

  A hostile authorized attacker cannot fabricate evidence (σ, κ) by tricking CSP since every time the PTPC delivers a challenge to CSP as *chal_pro* message, τ is included in the *chal_pro* message. The CSP will generate evidence (σ, κ) depending on τ from the PTPC, as shown in Equation (5.1 & 5.2), where τ varies by data shard.

  Even if we suppose that the malicious CSP forges (σ', κ') from the prior proof, namely $(\sigma, \kappa) \neq (\sigma', \kappa')$, the PTPC verification process in Equation (5.3) demonstrates that the PTPC must validate τ with *req_gen* message, which also contains tau value. As a result, the (σ', κ') cannot hold Equation

(5.3).

$$\sigma' = H[\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[\theta_j]||vk_1||vk_2)].a^t \tag{5.6}$$

Malicious CSP can prepare κ' using Equation(3.4)

$$\kappa' = (b.H(\sum_{i=1}^{t}\sum_{j=1}^{w} B_i(\theta_j) + \sigma.a) + t \tag{5.7}$$

Another scenario is when an untrustworthy CSP attempts to trick the verifier by swapping out the disputed data block$\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[\theta_j]$ with another data block $\sum_{i=1}^{n}\sum_{j=1}^{w} B_i[M'_j]$ and preparing hash values for collude blocks when the former data blocks are broken.

So,using Equation(5.4) and Equation(5.5), Equation (5.3) can be represented as:

$$g^{\kappa'} = vk_1^{\sigma'} + vk_2^{H[\sum_{i=1}^{\tau}\sum_{j=1}^{w} B_i(M'_j)]} \tag{5.8}$$

As a result, $H(\sum_{i=1}^{n}\sum_{j=1}^{w} B_i[M'_j] = H(\sum_{i=1}^{n}\sum_{j=1}^{w} B_i[\theta_j]$. However, due to the hash function's anti-collision characteristic, $H(\sum_{i=1}^{n}\sum_{j=1}^{w} B_i[\theta_j]$ cannot be equal to $H(\sum_{i=1}^{n}\sum_{j=1}^{w} B_i[M'_j]$. As a result, making Equation (5.6) hold is impossible, and The verification process cannot accept the proof from CSP.

- **Unambiguity:**

  An internal adversary cannot show itself to be a genuine CSP even if it knows information about vk_2 and σ of a genuine CSP(We must show that an enemy never confirms themselves as a genuine identity due to a lack of information about CU's sk_1, CSP's sk_2 and prepares the incorrect κ').

  Assume an internal adversary I_{vk_1} is there, notices CSP's stub and anonymity, and discovers his victory. He then makes the decision to assert the winning offer as his personal by sending in both his personal public verification key vk_2 and the deanonymize k' made up of the CSP's stub of challenged block

elements under vk_2', which he can compute because he already knows the public key vk_2 and stub σ. However, due to a lack of information on sk_1 of CU, I_{vk_1} prepares above mentioned Equation(5.5) κ' using Equation(5.2) as follows:

$$\kappa' = (b'.H(\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[M_j'] + \sigma.a') + t \tag{5.9}$$

According to Equations (3.4),(3.5), and (3.6), PTPC verifies the auditing test as follows:

$$g^{\kappa'} = vk_1^{\sigma} + vk_2^{H(\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[M_j'])} \tag{5.10}$$

From req_gen message, PTPC can calculate the value of a where $req\_gen = \{H(B_i[M_j'], g, n, vk_1, \phi\}$. Now, if we elaborate Equation(3.3) as follows:

$$\begin{aligned} g^{\kappa'} &= vk_1^{\sigma} + vk_2^{H(\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[M_j'])} \\ &\implies g^{b'.H(\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[M_j']+\sigma.a')+t} \\ &= g^{a*H[\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[M_j']||vk_1||vk_2).a^i} + g^{b*H(\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[M_j'])} \\ &\implies g^{b'.H(\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[M_j']+a'.H[\sum_{i=1}^{n}\sum_{j=1}^{w} B_i[M_j']||vk_1||vk_2).a^i+t} \\ &= g^{a*H[\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[M_j']||vk_1||vk_2).a^i} + g^{b*H(\sum_{i=1}^{t}\sum_{j=1}^{w} B_i[M_j'])} \end{aligned}$$

if $b' \neq b$ and $a' \neq a$, then t never would be 0 and adversary I_A violates the auditing criteria of PTPC.

- **Anonymity:**

  The stub σ and anonymity κ cannot be used to identify the stub-creator. To further explain, we assume that the external and internal adversaries have little bit priori knowledge regarding the signer's prospective verification key, such as that it belongs to some prime set Z of keys of p order, where identities are connected to verification keys like vk_1 and vk_2 for CU and CSP. This set, in the worst situation, contains enormous verification keys. The adversary has little advantage over guessing when determining which

one among all keys the stub was produced given knowledge of these keys, a stub created under one of them, and also the message.

5.3.2 Performance Analysis:

Finding out the computational cost of signature generation with block numbers, signature affirmation with block numbers, and block size(kilobytes(KB)) is the primary goal of this analysis part. Here, comparative simulation analysis of dynamic data processes was also examined. These are the experimental settings: The processor is an Intel(R) Core(TM) i5-9300H, the RAM is 12 GB, and Windows 10 is running on it. Simulation is done in Google Colab.

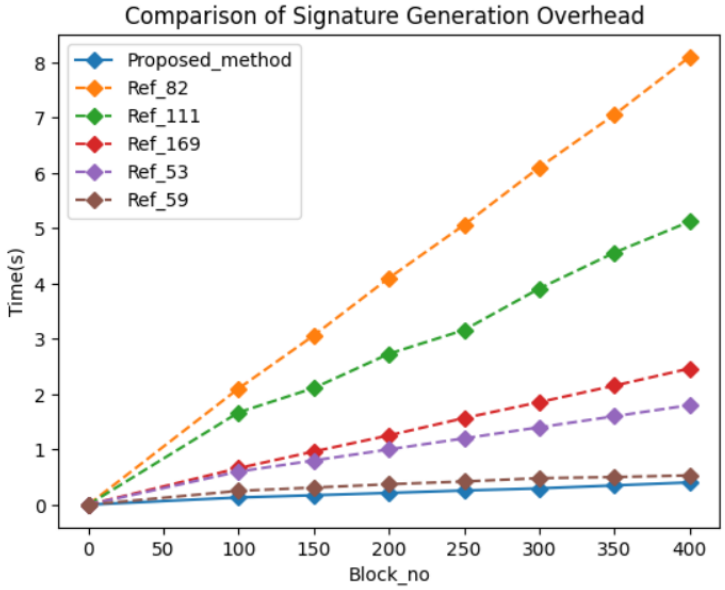

Figure 5.5: Signature Generation Overhead with Block no

Research Scenario

As per the research requirements, any research needs a research scenario to propose the research model. So, this research focuses on the given problem statement in terms of cloud storage security. The proposed research focuses on given input research parameters as per the proposed machine learning model requirements

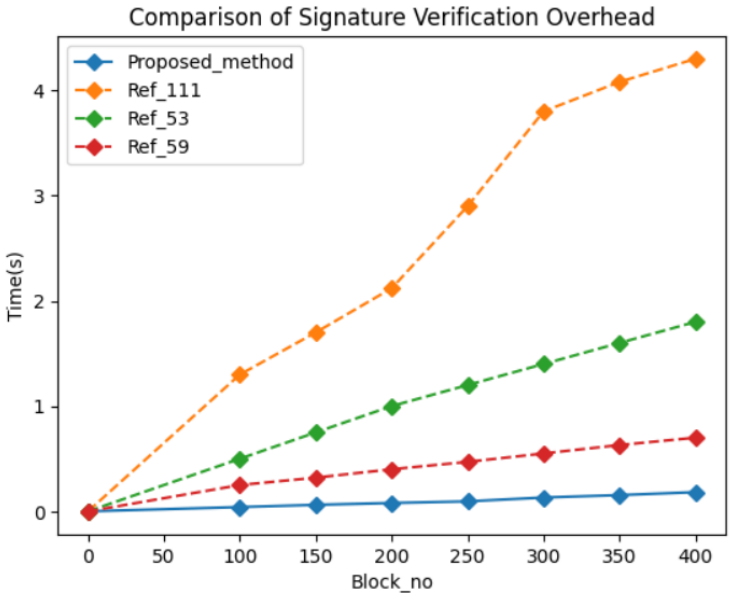

Figure 5.6: Signature Verification Overhead with Block no

Table 5.4: Sample values for Signature generation and signature Verification

Block_no	**Time for Signature Generation**	**Time for Signature Verification**
100	0.1305	.66164
150	0.1705	.9623
200	0.2128	1.2539
250	0.2567	1.567
300	0.2947	1.8548
350	0.3487	2.154
400	0.4026	2.4633

such as *Block_no*,*Sub_Block_no*, Time, etc. It analyzes the performance of the proposed model, research requires some data sets. The sample of data sets is described in Table 5.4 and Table 5.5.

Computational Overhead

This analysis part mainly focuses on determining the computational cost of signature creation time with multiple block numbers, signature verification time with multiple block numbers. A comparison of three aspects of the suggested method with earlier research [169, 82, 111, 140, 52, 53, 59] was later briefly de-

Table 5.5: Sample values for all Dynamic operation

Block_no	**Sub Block no**	**Time for Insert Operation**	**Time for Update Operation**	**Time for Delete Operation**
50	50	0.012	0.008	0.0129
	100	0.0209	0.0349	0.0199
	200	0.0277	0.0447	0.0369

scribed, as well as evidence of the suggested work's effectiveness. Figure 5.5 displays the computational cost of signature generation time with various data blocks 100,150,200,250,300,350,400 for the proposed scheme and the research papers [169, 82, 111, 53, 59].

In terms of computation complexity, we compare our partial signature-based auditing scheme to the BLS signature-based scheme[111], the ZSS signature-based scheme [169, 52, 53], and the algebraic signature-based scheme[169, 59] in terms of computational overhead for signature generation and verification, signature's type, and the size of the signature, among other things. Table 5.6 is defined to show the meaning of all crypto operations in order to establish unambiguous comparisons.

Our technique is more efficient since it does not use pairing operations like the other signatures stated in [169, 82, 52, 53]. Furthermore, unlike the BLS signature, a partial signature does not require a particular hash function. It can make use of a widely used SHA family hash function, like the ZSS signature.

Pairing process and exponential process are typically regarded as high-cost procedures. It is already proven in the article [184]that the cost of an exponential operation is smaller than the cost of a pairing operation (for both curved and non-curved). Regrettably, the attribute-based signature demanded more exponential operations than alternative approaches.

In papers [169, 82, 111, 52, 53], cloud users have the responsibility to prepare signature for all blocks. For all the above-mentioned research papers, a file F is broken up into n no. of blocks of varying sizes where $B = \{B1, B2, ..., Bn\}$ and

Table 5.6: Notations in Computational overhead

Notation of computational operation	**Meaning**
$Pair$	Bilinear pairing operation
Mul	Multiplication Operation
Inv	Inverse operation
$Hash$	Hash operation
Exp	Exponential operation
$Encrpt$	Symmetric encryption operation
Add	Addition operation

the ZSS signature, BLS signature, and identity-based signature are used at each block level. In the sign generation phase, scheme [169] spends n(1Hash + 1Mul + 2Exp)for the signature generation, scheme [127] spends nHash+ nMul+ nInv for the signature generation, scheme [52] spends n(1Exp + 1Inv + 1Mul+1Add + 3Hash) for the signature generation, scheme [53] spends nHash+ nMul+ 2 nAdd + n Inv for the signature generation, scheme [59] spends n(2Exp+1Mul+1Hash) for the signature generation respectively.

In Table 5.7, a comparison of the computational overhead of existing research works with proposed research work is depicted. Unlike other mentioned schemes, CSP is responsible for preparing stub signatures of challenged blocks in our proposed scheme, because our proposed signature scheme prepares signatures for challenged blocks for which a request is made from the PTPC side, whereas all other mentioned schemes prepare signatures for all blocks. As a result of this approach, our suggested signature method has less computational overhead than previous signature-based schemes in general. We further assume that CSP necessitates (4+t) addition operations, 3 multiplication operations,1 hash operation, and 1 exponential operation. Because there is no original version, it enhances the level of data privacy and security in RCS for outsourced *data*. As a result, the computational cost of the CSP to prepare a stub signature as a challenge answer for a public auditing test is $(4+t)Add+3Mul+1Exp+1Hash$. Bilinear pairing is required for BLS, ZSS, and algebraic signature-based data auditing. A

research paper[169] requires four pairing operations, a paper [82, 111] requires two pairing operations, a paper [52] requires n pairing operations, and a data auditing verification technique [53]requires one pairing operation. Because of the partial signature notion, our suggested data auditing approach does not involve any pairing operations. [59] requires (t+1) Exp operation, but our proposed approach requires only one Exp operation.

Figure 5.5 depicts the computational cost of signature creation time for the proposed approach and the research publications[169, 82, 111, 53, 59] with various data blocks 100,150,200,250,300,350,400. Figure 5.6 depicts the computational cost of signature verification time for the proposed approach and the research publications[111, 53, 59] with various data blocks 100,150,200,250,300,350,400. The average computing time of the suggested technique is 0.2595 seconds, whereas the average sign generation time in [169, 82, 111, 53, 59] research works was 1.5595, 5.0842 3.3228, 1.00, 0.4085 seconds respectively. The average computation times for sign production with three system models differ by 1.30 seconds, 4.824 seconds, 3.063 seconds, 0.7405 seconds, and 0.1490 seconds, respectively.

Communicational Overhead

The research paper [169] shows BLS signature-based data auditing and [82] shows data auditing verification based on ZSS signature. Both ZSS and BLS signature performs bilinear pairing operations. The BLS signature process, which has an additional communication cost of around 960 bits, provides the basis for the scheme[169] and the ZSS signature mechanism which is about 720 bits [82]. Our proposed signature is a Schnorr signature and it has no overhead when compared to the base entropy scheme. Based on the Schnorr protocol[185], we create a modified partial signature method for a 240-bit full signature using an 80-bit stub and a 160-bit de-anonymizer.

In the article [178], it is already proved that the cost of exponential operation

Table 5.7: Comparison of Computational Overhead

Ref.	**Signature Scheme**	**Size of Signature(bits**	**Overhead of Signature preparation**	**Overhead of Signature Verification**
[169]	Algebric	Na	$n(Hash + Mul + 2Exp)$	$4Pair+2Mul+2(t-1)Add+2Exp+(t+1)Exp+(t+1)Mul+tHash$
[82]	ZSS	720	$nHash + nMul + +nInv$	$Mul + 2Pair + Add$
[111]	BLS	960	$n(Hash + Mul_2Exp)$	$tMulExp + t2Hash + 3Exp + 2Pair + 2Mul$
[182]	ZSS	260	$n(1Exp + 1Inv + 1Mul + 1Add + 3Hash)$	$n(1Add + 2Hash + Pair)$
[183]	ZSS	480	$nHash + nMul + 2nAdd + nInv$	$1Pair + 2Mul(2tAdd + 1) + 2Add$
[146]	Algebric	NA	$n(2Exp + 1Mul + 1Hash)$	$(t + 1)Exp + 1Mul$
Proposed Method	Partial	240	$(t+4)Add+3Mul+1Hash + 1Exp$	$3Exp + (t + 1)Add$

is less than the cost of pairing operation (for both curved and without curved). It has already been demonstrated in a research study [82] that as a signature technique based on the ZSS requires less time to sign documents than a BLS's signature scheme as the number of data blocks increases.

The research paper [111, 59] shows data integrity verification based on BLS signature [82, 52, 53] shows data auditing verification based on ZSS signature and [169] shows data auditing verification based on algebraic signature. Both ZSS and BLS signature performs bilinear pairing operations. The extra communication overhead of the scheme is based on the BLS signature mechanism which is about 960 bits [169] and the ZSS signature mechanism which is about 720 bits [82]. Our proposed signature is a Schnorr signature and it has no overhead when compared to the base entropy scheme. Based on the Schnorr protocol[178], we prepare a modified partial signature scheme with an 80-bit stub and a 160-bit de-anonymizer

Table 5.8: Comparison of Communication overhead

Reference	**CU to CSP**	**CU to PTPC**	**CSP to PTPC**
[182]	O(n)	O(1)	O(n)
[183]	O(n)	O(1)	O(t)
[146]	O(n)	O(1)	O(t)
[186]	zero	O(n)	O(t)
Our's	O(n)	O(1)	O(t)

for a 240-bit full signature. It has already been demonstrated in a research study [82] that as the number of data blocks rises, the signature time required for a signature scheme based on the ZSS is shorter than that required for a signature scheme based on the BLS. The extra communication overhead caused by the CU in the proposed system based on the partial signature technique is mostly the $info\_message$ value uploaded to the CSP, which is approximately 80 bits. The extra communication cost for the proof of $chal\_pro$ message is approximately 240 bits, which is forwarded by the CSP to the PTPC as $chal\_reply$ message.As a result, the additional communication overhead generated by the proposed scheme is around 320 bits, which is less than the communication overhead required by the ZSS and BLS-based signature mechanisms. Here, we consider three scenarios like 1st is CU to CSP communication, 2nd is CU to PTPC communication, and 3rd is PTPC to CSP and CSP to PTPC communication which depict in Table 5.8. Like the other methods[182, 183, 146, 186], CU sends a file to CSP to store their data in cloud storage, and simultaneously CU hires PTPC as an auditor to check their outsourced's data integrity, where PTPC prepares a challenge message and sends it to the CSP and PTPC then validates the proof that the CSP sent after it produces evidence of verification and transmits it to PTPC. First communication overhead is O(n) where n is the total blocks requested to store in CSP like [182, 183, 146]. 2nd communication overhead is O(1) like [182, 183, 146] and 3rd communication overhead claims $O(t)$ for stub signature of t no. of challenged blocks like [183, 146, 186]. In [186], there is not a single communication happening between the user and CSP.

Simulation overhead of dynamic data operations:

In a manner similar to [78], we carry out each operation three times in accordance with the various data blocks.50 data blocks are available. We simulate the three separate data block operations with varying numbers of sub-blocks 50, 100, and 200, respectively, in order to show the regular pattern of the time cost in Figure 5.7, Figure 5.8, and Figure 5.9 respectively. which displays the computational time required for the data sub-block update, delete, and, insertion simulation. We have already mentioned of pseudo process of all dynamic operations in Algorithm 5.1, 5.2, and 5.3 accordingly.

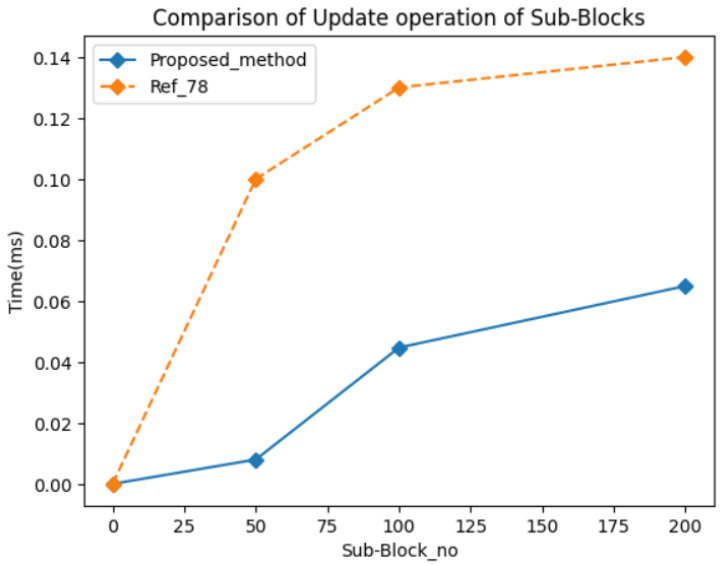

Figure 5.7: Comparison of Update operation with Block no.

We have considered here 50 blocks and for this we have taken the simulation result of 50 sub-blocks, 100 sub-blocks and 200 sub-blocks respectively for all dynamic operations. From this figure, we can see that every time a sub-block update, delete and, insertion time will increase for the growing no. of sub-blocks. Our proposed model takes 0.039ms for update operation,0.019ms for delete operation, and 0.020 ms for insertion operation while [78] takes 0.092 ms for update operation,0.073 ms for delete operation, and 0.046 ms for insertion operation which are less than [78] averagely.

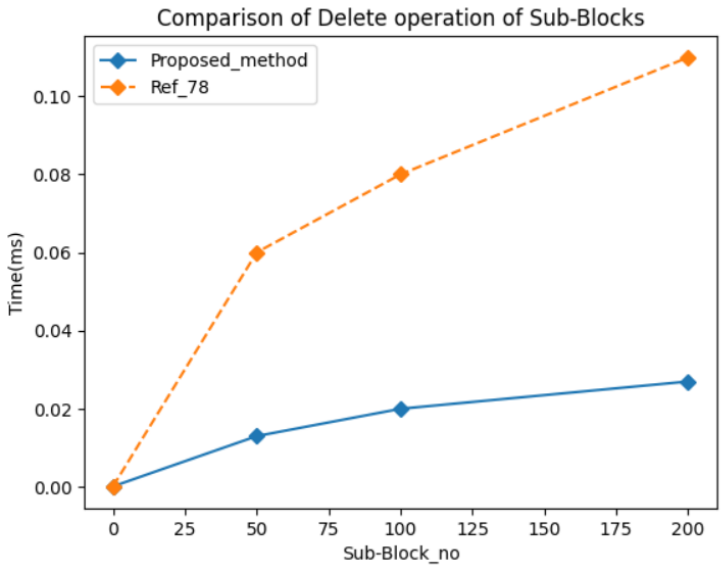

Figure 5.8: Comparison of Delete operation with Block no.

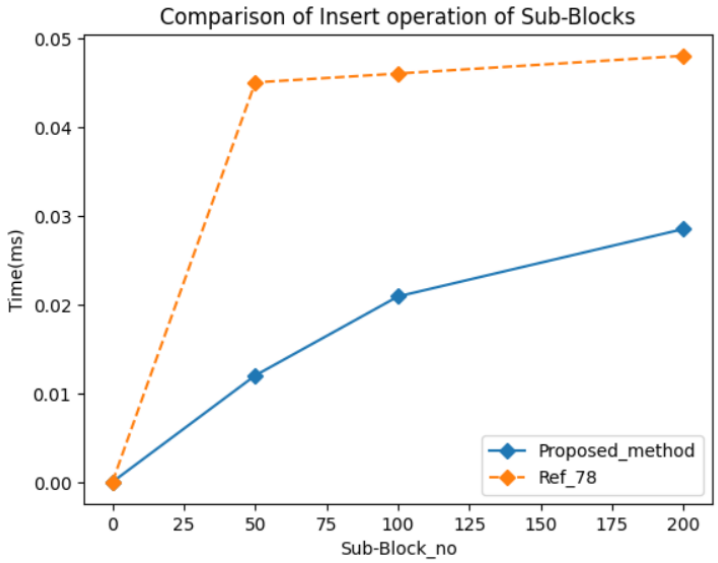

Figure 5.9: Comparison of Insert operation with Block no.

5.4 Summary

By assessing accuracy at the block level, dynamic operations are carried out to safeguard the integrity of outsourced data on shared cloud storage. By employing original data encryption technology that generates random numbers and a partial signature, RCS ensures the consistency of the outsourced data for CU. It helps deter attacks involving replaceability and forgeability. Furthermore, during

auditing, PTPC can identify block corruption and alert CU to it. Furthermore, while completing the audit, this user-friendly auditing architecture safeguards CU's data privacy from all parties, including CSP and PTPC.Furthermore, the aforementioned experimental results demonstrate that the dynamic data update operation is a superior solution than the current ones, as it is executed while preserving public auditing on the same with lower computational cost. We plan to extend our auditing model for CU-side audit message verification and error localization in our subsequent research study.

Chapter 6

ZSS Signature-based Audit Message Verification Process for Cloud Data Integrity

The focus of cloud data security research has shifted towards improving data audit accuracy to prevent hacking by internal and external adversaries, rather than just relying on Third Party Auditors (TPAs). However, TPAs may not always be trustworthy, posing risks of fake audit results or collusion with attackers. To address these challenges, a model for an effective data integrity and audit message verification system based on ZSS signature is proposed. This system enhances data privacy in cloud storage and employs fake data recovery to retaliate against attackers. Additionally, the proposed model is adapted for a Blockchain-based Medical system, enabling patients to verify the integrity of their medical records in shared cloud storage through a third-party auditor. Experimental results demonstrate the efficiency of the proposed model compared to existing auditing methods.

6.1 Overview

Cloud Computing has garnered significant attention in both academic and industry domains due to its flexible pay-per-service model (IaaS, SaaS, PaaS), decentralization, and an array of features such as online services, cost-effectiveness, easy maintenance, and network accessibility. Users can now conveniently access cloud resources by migrating their data to online storage without managing physical devices[113, 140]. This transition underscores cloud computing's widespread adoption across various sectors, serving as a backup repository for personal or

business data to maximize profits and reduce the burden of maintaining data on physical machines[82].

Cloud computing, despite its benefits, presents numerous security threats such as data privacy breaches, integrity issues, and confidentiality risks[169, 187, 188, 57]. Cloud service providers (CSPs) may act maliciously, deleting data or sharing it without consent to gain profit. Even honest CSPs can lose data due to managerial issues[82, 165] or insider attacks[189], concealing losses to protect their reputation[165]. Data owners lack physical control over outsourced data, making it challenging to identify breaches[145, 167]. Traditional MAC schemes incurred high communication overhead, leading to the adoption of public auditing schemes. However, these schemes assume the Third Party Auditor (TPA) is always honest, which isn't guaranteed[145, 190]. A malicious TPA, colluding with a CSP, could produce biased challenge-proof messages, falsely assuring data integrity [72, 164]. To address this, data owners should verify audit messages and recover original data from fabricated versions when necessary, ensuring the integrity and security of outsourced data

Researchers are focused on enhancing data integrity verification and privacy-preserving techniques to ensure data correctness[73, 82, 113, 140, 162, 191], intactness, and auditing accuracy. Provable data possession (PDP) techniques have been used to expedite computation for lengthy files and reduce hash value transfer time [50, 54, 55, 192, 193], but lack error localization and data recovery support. Proof of Retrievability (PoR) techniques[58] address these issues by ensuring complete retrieval of corrupted blocks and supporting dynamic data operations[95, 101], but require the data owner (DO) to maintain key information, increasing computational costs. Bilinear pairing concepts and BLS cryptographic signatures have been proposed to enhance data authorization and signature efficiency[162, 72], but have drawbacks such as probabilistic nature and computational overhead. To overcome these limitations, the ZSS signature scheme has been introduced, supporting privacy protection, auditing correctness, and resistance against attacks. However, data confidentiality is not guaranteed, and

DO cannot verify the correctness of audit response messages from Third Party Auditors (TPAs). To address these challenges, an effective public data auditing scheme using ZSS signatures is proposed[82], ensuring confidentiality, privacy, auditing correctness, resistance against attacks, message verification, error localization, data recovery, and dynamic data updates at the block level.

This research introduces an efficient short signature-based outsourced data auditing scheme, ensuring security and data confidentiality during integrity checks. Unlike [72, 82, 145], it proposes a public auditing scheme through a Third Party Auditor (TPA) to maintain data privacy and integrity in shared cloud storage, preventing insider and outsider attacks. The scheme modifies original block elements to prevent forgery and replacement attacks, resisting collusion issues. It incurs less encryption overhead than AES encryption[60] and introduces a modified ZSS signature method for generating signatures of multiple blocks[53], reducing computational time of BLS[162]which has been proved by research paper [82]. The protocol includes a method for TPA to prepare challenge messages and for Cloud Service Providers (CSP) to respond, ensuring audit message verification on the Data Owner (DO) side to detect collusion or biased messages. Error localization at the block index level and encrypted data recovery using modular techniques are introduced, along with dynamic data updating methods for error blocks. Performance analysis demonstrates the computational efficiency of the proposed protocol.

In this research work, we introduce an effective short signature-based outsourced data auditing scheme that is capable to maintain security and data confidentiality during data integrity auditing. The DO side audit message verification is likewise ensured by this system. This section outlines the main contributions of the suggested model and demonstrates how it differs from other research work in the manners described below.:

- We propose a public auditing scheme through a TPA to check data integrity in shared cloud storage. This auditing scheme fulfills the security criteria

to upkeep the privacy of outsourced data stores in cloud storage and maintain data confidentiality, data correctness, auditing correctness, etc. during auditing time.

- We introduce a way to maintain the data element's originality from insider as well as outsider malicious attacks at cloud storage. This method prevents forgery and replaces attack on cloud storage and resist collusion problems of blocks.

- We establish a ZSS signature method to produce signatures of all encrypted blocks which takes less computational time than generating the BLS signature of [133] which has been proved by research paper [128].

- Like [60, 151, 65], we introduce here a challenge message preparation method by RTPA and challenge-response message by LCSP which take less computational time than [60, 151, 65] and will discuss in the performance analysis section. Unlike [82, 60, 151, 65, 64, 34], this scheme ensures an audit message verification test at the DO side to check whether TPA colludes with the verification result or TPA produces a biased message without verifying data integrity unlike.

- Unlike [82, 60, 151, 65, 64, 34, 62], We introduce the error localization method of public auditing at the block index level and also introduce the encrypted data recovery method by modular technique at the DO side.

The organization of this research work is as given follow: Section 6.2 shows a small description of bilinear pairing, ZSS signature and Core Deffie Hellman Key Exchange Protocol Section 6.3 discuss the detailed system model, design goals, working flow of the auditing scheme, basic definitions and stages of auditing scheme, audit message verification scheme, data recovery scheme,two-layer security model, an example of proposed auditing model Section 6.4 briefly discuss security analysis and performance analysis with simulated value and compare

the implemented result with research paper [82]to prove the better efficiency of proposed model Section 6.5 conclude the research work.

6.2 Preliminaries

6.2.1 Bilinear Pairing

Here this research work describes the bilinear pairing concept[149] that is used in the proposed scheme. Let two cyclic groups G_1 and G_1' generate two group generators accordingly g_1 and g_1'. Both cyclic groups are additive in nature. The order of both generators is q where q is a prime no. Another group G_2 is a multiplicative cyclic group with the same q order.

Let $e : G_1 \times G_1' \rightarrow G_2$, be a bilinear map with given below properties:

Bilinearity : $< xg_1, yg_1' >= (g_1, g_1')^{(x,y)}$, where $\forall g_1 \in G_1$ and $\forall g_1' \in G_1'$.

Non-degeneracy: There exist g_1, g_1' elements of G_1 and G_1', respectively, such that $< g_1, g_1' > \neq 1$.

Computability: An efficient algorithm is present here to compute $< g_1, g_1' >$ where $\forall g_1 \in G_1$ and $\forall g_1' \in G_1'$.

Therefore, g_1 is generator of G_1 and g_1' is generator of G_1', then $< g_1, g_1' >$ is a generator of G_2.

6.2.2 ZSS Signature

Bilinear pairing-based ZSS signature is first proposed Zhang et al.[183]. Here, four algorithms parameters generation P_{gen}, key generation k_{gen}, signature generation S_gen and signature verification S_{ver} are introduced in ZSS public key cryptographic signature [82]. It is defined as:

P_{gen}: A set of public parameters are e, G_1, G_2, g_1, q, H.

K_{gen}: K is a positive integer number, randomly selected to compute $P_{pub} = k * g1$. Here P_{pub} is a public key and K is a secret key where $k \in Z_q^*$.

S_{gen}: If S is a ZSS signature and m is a message, then

$$S = 1/(H(m) + K).g_1 \tag{6.1}$$

S_{ver}: Given a message m, signature S and public key P_{pub}, verify if

$$e(H(m) \cdot g_1)P_{pub}, S) = e(g_1, g_1) \tag{6.2}$$

6.2.3 Core Deffie-Hellman Key Exchange protocol

Two entities Alice A and Bob B are allowed to obtain a shared secret key over a common transmission channel. At first both A and B accord on a big prime no. p from a cyclic group $\hat{\text{G}}$ calmed the set $Z_p^* = 1, 2, \ldots., p-1$ with multiplication modulo p and a generator $\bar{g}$, $2 \leq \bar{g} \leq p-2$, that produces a large ordered set.The key exchange protocol is given below [194]:

- A and B randomly select x and y from the set $1, 2, ..., p-2$.
- A dispatches $\bar{g}^x$ to B and B dispatches $\bar{g}^y$ to A.
- Finally, A and B obtain shared secret key $\bar{g}^{xy}$.

6.3 Proposed Data Integrity Scheme

In this section, the proposed system model has been given which is depicted in Figure 6.1. Further in this section, auditing of the proposed scheme with a basic definition, audit message verification scheme, Data recovery scheme, security model, and design goals are described.

6.3.1 System Model

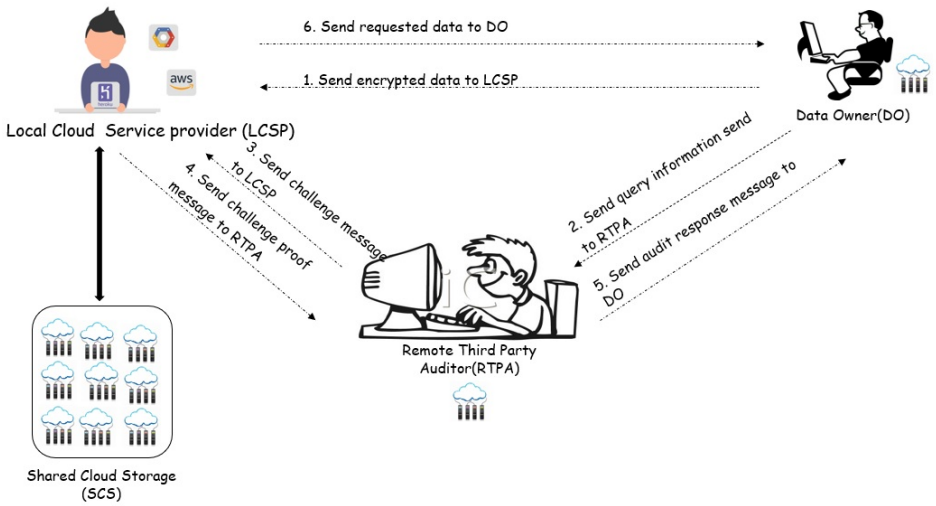

Figure 6.1: Proposed Auditing Model

Figure 6.1 demonstrates the proposed system model for the desired framework. The unabridged proposed system is composed of four entities as follows: Data Owner (DO), Local Cloud Service Provider (LCSP), Remote Third-Party Auditor (RTPA), and Shared Cloud Storage (SCS) .

- **Data Owner (DO):** DO i.e. cloud users have online data storage requirements and outsourced crucial data to LCSP for saving in storage. DO can access their data through any mobile device or any PC using a network setup.DO have limited computational capacity and also storage capacity to prepare signature for their crucial data to maintain data confidentiality and to verify audit message which comes from LCSP. DO also have the capacity to recover original data from colluded data if needed.

- **Local Cloud Service Provider (LCSP):** LCSP is a third-party organization offering storage services to DO. This entity is responsible to maintain the privacy of DO's outsourced data.

- **Remote Third-Party Auditor (RTPA):** In the proposed public auditing model, RTPA has the expertise and ability to verify the data intactness of

DO's outsourced data which have been stored in Shared Cloud Storage. Shared Cloud Storage (SCS) It is a virtual shared pool that is proven to DO for storage resource usages purpose via LCSP in terms of the lease.

The primary goals are condensed here to keep up data integrity of outsourced data of DO through RTPA at the insecure local cloud server-side and also to verify the audit verification result generated by RTPA on the DO side. The data recovery method is established here to get back the original form of fabricated data. DO produces data prior to uploading at any local cloud storage to acquire financial profit. LCSP is a third-party organization offering storage services to data owners. SCS is an online shared data storage where LCSP stores all outsourced data coming from DO. RTPA exempts the burden of management of data of DO by checking the correctness and intactness of outsourced data. In this prototype system model, we assume that every entity has the personal computational capacity to process data and personal storage a little bit.

6.3.2 Design Goal

Desire design goals of the outsourced data auditing scheme can be summarized as follow:

- Public Auditing: Our proposed public auditing scheme looks like DO outsources crucial data to LCSP for storing purposes. Later DO delegates the outsourced data auditing task to a third-party auditor known as RTPA who has the capability to do similar types of tasks. This model should support partial data verification via RTPA. Like [74], this scheme does not include additional computational overhead for data preprocessing done by DO in the auditing process.
- Auditing Correctness: The proof of LCSP can pass the validation test of RTPA only if LCSP and RTPA are being honest and LCSP, DO properly follow the pre-defined process of data storing like [73].

- Data Confidentiality: This crypto parameter ensures that original data must not disclose in front of either LCSP or RTPA during the auditing process[78].

- Unforgeability: If the LCSP properly maintains outsourced data at SCS, then only $chal\_res\_pro$ message can pass the verification test i.e. our system model can resist internal and external forgery attack [145].

- Collision Resistance: If stored data at SCS is not colluded by LCSP, then the signature verification test shows the data intactness [74]. Error localization at block's index level In block-level auditing, it helps to find out error blocks of a file in cloud storage during verification time[73].

6.3.3 Working flow of the scheme

At first, DO calculates the maximum size of file F as n and puts it into an array $A[n-1]$. DO divides the file F into chunk size variable blocks $B = \{B_1, B_2, ...B_j\}$ where the maximum size of each block is k and $1 < n < cel(n/d)$, $2 \leq d \leq k$.

DO generates public and private key pairs (DO_{pub}, DO_{pri}) using two random no.s r_1, r_2, and group generator g_1 with prime order q where r_1, $r_2 \in Z_q^*$.

Later each block elements $B_t[M_i]$, $t \in j$, $i \in k$ are encrypted by DO_{pri} to produce fabricated block elements $B_t[M_i']$, and DO calculates mean value $\mu = \{\mu_1, \mu_2, ..., \mu_j\}$, and mod_mean value for each block. DO also calculates the values of τ known as substitute data, where $\tau = \{0, 1, ..., n-1\}$ using g_1 to recover colluded block elements whenever needed.

DO maintains a table T which contains some important information related to file F , where $T = \{\omega, r_1, r_2, g_1, j, \tau, \mu, d_\mu\}$. After that,DO generates signature $S = \{S_1, S_2, ..., S_j\}$, for each block B_j.

DO prepares a message $info\_msg$ which contains a file tag ω, public key DO_{pub}, encrypted block elements $Bj[M'i]$ along with signatures S and sends it to

Table 6.1: Definitions of Notations

Notation	Meaning
G_1	Multiplicative Cyclic group with q prime order
r_1	Random no.s
r_2	Secret key
Z_q	Ring of integer modulo q
e	Billinear paring map $e : G_1 \times G_1 = G_2$
F	Original File
ω	File Tag
$\{B_1, B_2, ...B_j\}$	Original blocks
$B_t[M_i]$	Block Elements of t^{th} block where $t \in j$,$i \in k$
$B_t[M_i']$	Encrypted Block Elements of t^{th} block
$n-1$	Maximum size of a filer
H	Hash Function: $H : \{0,1\}^* \rightarrow \{0,1\}^\lambda$
q	Prime number
k	Total size of a particular block
j	Total no. of block
μ	Mean value of each block
d_μ	Mod_div value of each block element
B_x	Particular challenged block no where $x \in j$
S_x	Signature of challenged block B_x where $x \in j$
B_x'	Encrypted block no.
τ	Substitute values
ψ	Collude value
$\oplus$	Operation for Collude Block elements
DO_{pri}	Private key and public key of DO
C_{pri}	Private key and public key of LCSP
R_{pri}	Private key and public key of RTPA

LCSP for storing data.

DO also sends j, DO_{pub}, and g_1 as a query message namely req_msg to RTPA for data integrity verification. RTPA prepares public and private key pair using the Diffie-hellman key exchange algorithm, prepares a challenge message for x no. of challenged blocks, and sends it to LCSP.

After verification of the challenge message, LCSP prepares a challenge-response message and sends it to RTPA. The encrypted message contains mainly elements of requested blocks $B_t'[M_i']$, a verification message.

After the decryption of the challenge-response message, RTPA verifies this

message which comes from LCSP.

If the verification result is true that means data intactness is properly maintained in SCS. Later, RTPA prepares an audit result message and sends it to the DO. DO verify the message through the previously stored values from Table T to assure whether the audit verification is done by RTPA successfully or not. After verification, DO uses substitute data for original block elements recovery purposes if block elements are fabricated by attackers.

6.3.4 Basic Definition of the Scheme

Our proposed auditing scheme consists of seven phases: key produce phase, data encryption, and substitute generation phase, signature produce phase, challenge message produce phase, challenge-response message produce phase, and auditing phase. In our proposed auditing scheme, we use Key_pro, $Data\_encrypt\_subs$, $Sign\_pro$, $Chal\_msg\_pro$, $Chal\_res\_pro$, $Audit\_proof$ to represent the seven phases respectively. Table 6.1 shows the notation with descriptions used in our proposed model. The seven phases are elaborately described as follows:

$Key\_pro(g_1, r_1, r_2) \rightarrow (DO_{pub}, DO_{pri})$: Enter group generator g_1 as a secret parameter and two random numbers r_1 and r_2 , output data owner's public key, and private key DO_{pub}, DO_{pri} respectively.

$Data\_encrypt\_subs(B, DO_{pri}, g_1) \rightarrow (B', d\_\mu, \tau, \mu)$:Input all blocks B and private key DO_{pri} , output encrypted data blocks $B' = \{B'_1, B'_2, ..., B'_j\}$, mean data μ of each block B_i where $i \in j$, substitute value τ and mod mean value d_μ of each data block. In order to build an encrypted version for each data block based on SHA2 encryption, DO performs this phase.

$Sign\_pro(B_t[M'_i], g_1, r_1) \rightarrow (S)$: Input all blocks B' of a file F and generator parameter g_1, random number r_1, output the signature of data block set $S = \{S_1, S_2, ..., S_j\}$.

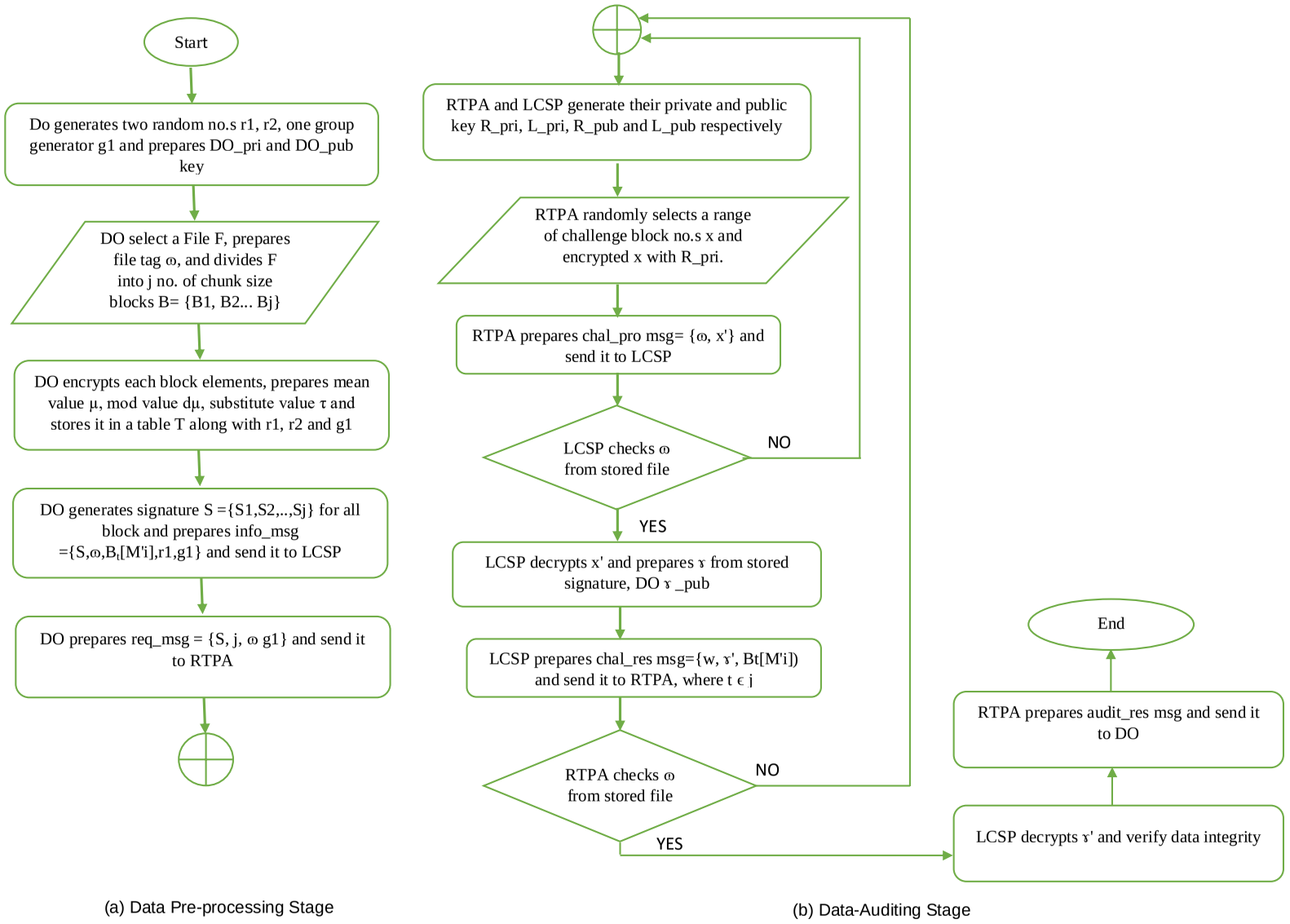

Figure 6.2: Process of Data Auditing at Block Level

$chal\_gen(R_{pri}, x, B'_t[M'_i]) \rightarrow Chal\_msg\_pro(B'_t[M''_i], \omega)$: It takes as inputs x no. of challenged multiple blocks B'_t where $t \in x$,$x \in j$ and private key of RTPA R_{pri} ,output as a set of encrypted blocks $B'_t[M''_i]$, $t \in x$. RTPA prepares $chal\_msg\_pro$ as x, $B'_t[M''_i]$ where $t \in x$, file tag ω, RTPA's public key R_{pub} for identification of file.

$Chal\_msg\_pro(B'_t[M''_i], x, R_{pub}, \omega) \rightarrow Chal\_res\_pro\ (B'_t[M''_i]), \gamma, \omega, C_{pub})$: It takes as inputs $chal\_msg\_pro$, output a generated challenge response message $Chal\_res\_pro$. It returns challenged all block elements $B'_t[M''_i]$ encrypted by the private key of LCSP C_{pri} and verification message γ where $x \in j$.

$Audit\_proof(chal\_res\_pro, g_1, DO_{pub}) \rightarrow (True, False)$: It takes $Chal\_res\_pro$ message as an input, outputs the auditing results True or False. If the auditing passes successfully, the output is True, otherwise False.

In this section, our proposed audit model has been described as two stages: data pre-processing stage and data auditing stage. Figure 6.2 depicts here a flowchart of two stages. A detailed introduction of the two stages will be described as follows :

a) Data Pre-processing-Stage

- **Key Produce Phase**: DO generates a group generator g_1 from G_1 using a bilinear map and two random numbers r_1, r_2 to produce a pair of public and private keys (DO_{pub}, DO_{pri}) where $g_1 \in G_1$ and $r_1, r_2 \in Z_q^*$.DO also prepares a file tag ω for file identification.
- **Data Encryption Phase**: DO divides original file F into variable size blocks $B = \{B1, B2, ...Bj\}$ and encrypts each block element $B_t[M_i]$ using DO_{pri} key where $t \in j$.In order to avoid the issue of a boundary-shifting problem, we use chunk-sized variable blocks in this research paper [181].
- **Substitute Generation**: DO generates substitute elements τ for all block

elements using g_1 and calculates μ,d_μ for each block.

- **Signature Produce Phase**: DO computes ZSS signature $S = \{S_1, S_2, ..., S_j\}$ for each block B_t using g_1 and r_1 where and $t \in j$.

$$S_j = [1/(H[B_t(M_i')]) + r_1].g_1 \tag{6.3}$$

or,

$$S_j = [1/(H(B_t[DO_{pri}(M_i)]) + r_1].g_1 \tag{6.4}$$

Algorithm 6.1 explains here the process of the signature generation scheme.DO prepares a $info\_msg = \{B_t[M_i'], S, \omega, g_1, DO_{pub}\}$ and send it to LCSP. DO maintains a table $T = \{\omega, r_1, r_2, g_1, j, \tau\}$ for future references.DO also prepares a query message $req\_msg = \{\omega, g_1, j, S\}$ and sends it to RTPA for data integrity verification of outsourced data.

Algorithm 6.1: Process of Signature Generation Method

Input:$g_1, r_1, r_2, B_t[M_i']$

Output:Signature $S = \{S_1, S_2, .., S_j\}$ out

Initialisation :

1: **for** $t = 1$ to j **do**
2: **for** $i = 1$ to k **do**
3: Determine the hash values for each block B_t.
4: Add random number r_1 to every hash value for j blocks individually.
5: Calculates $(H[B_t(M_i')] + r_1)^{-1}$ for j blocks individually.
6: Multiplies g_1 with $(H[B_t(M_i')] + r_1)^{-1}$ to prepare signatures S for all j blocks.
7: **end for**
8: **end for**

b) Data-Auditing-Stage: In block-level auditing, selected range of blocks is challenged for frequently outsourced data accurateness over the stored original

encrypted data blocks at SCS. Here, RTPA prepares $chal\_msg\_pro$ based on req_msg and sends it to LCSP. LCSP and RTPA randomly select two numbers $\bar{x}$ and $\bar{y}$ consequently where $\bar{x}, \bar{y} \in \{1, 2, ..., q-2\}$ .LCSP produces $g^{\bar{x}*r_1}$ as a public key C_{pub} and RTPA produces $g^{\bar{y}*r_1}$ as a public key R_{pub}.Both entity generates same private key or secret exchange key $g^{\bar{x}\bar{y}}$ for C_{pri} and R_{pri} through Deffie-Hellman Key Exchange method.

- **Challenge Message Produce Phase**: Every challenge message generation time, RTPA will generate a new R_{pub} key. Using R_{pub}, RTPA generates a private key R_{pri}. After generation R_{pri}, RTPA prepares a challenge message $chal\_msg\_pro = \{R_{pri}[B_t], \omega\}$ i.e. $\{B_t''$,x, ω $\}$ for x no. of challenged blocks and sends it to LCSP.Here challenged block no.s B_t'' are encrypted by R_{pri}.

- **Challenge-Response Produce Phase**: After receiving $chal\_msg\_pro$, LCSP decrypts the message using $C_{pri}$and verify file tag ω. After verification, RTPA prepares a $chal\_res\_pro$ message and sends it to RTPA where

$$\begin{aligned} chal\_res\_pro = \{\gamma, C_{pub}(B_t'[M_i']), \omega, sk\} \\ \text{i.e.}\{\gamma, B_t'[M_j''], \omega, sk\} \end{aligned} \tag{6.5}$$

  where

$$\gamma = \sum_{t=1}^{\tau} r_1/[sk + g1 * H(B_t[M_i])] \tag{6.6}$$

  and $sk = r_1 * g_1$

- **Auditing Phase**: The RTPA will confirm the accuracy of the stored data blocks through $chal\_res\_pro$ message once it has been received from the LCSP. From the value of ω, RTPA can identify the DO's file. This phase's inputs are $\{\gamma, sk, g_1, S_t\}$, and its output is the auditing verification result for challenged data blocks. RTPA already has signature values of all blocks

received from DO, no need to worry about the size of each block, and simply performs verification proof as

$$e(g1 * \sum_{t=1}^{x} S_t, \frac{sk}{\gamma}) = e(g_1, g_1) \tag{6.7}$$

If the above equation is true, the output will be true, otherwise, the output will be false.

Table 6.2: Comparison table of Security parameters

Scheme	DC	AC	CRAR	PP	UF	ELC
[140]	×	√	×	√	√	×
[73]	×	√	×	√	√	√
[82]	×	√	×	√	×	×
[78]	√	√	×	√	×	×
[164]	√	√	√	√	√	×
[165]	×	√	√	√	×	×
[169]	√	√	×	√	×	×
[143]	√	√	√	√	×	×
[183]	√	√	×	√	×	×
[146]	×	√	×	√	√	×
[144]	√	√	×	√	×	×
Proposed Model	√	√	√	√	√	√

DC: Data Confidentiality; AC: Auditing Correctness; CRAR: Collision & Replace Attack Resistance; PP: Privacy-Preserving; UF: Unforgeability ; ELC: Error Localization & correcting data at block's index level

6.3.5 Proposed Two-Layer Security Model

Our data integrity model hires the concept of the core DH model[194] with the criteria of $g^{\bar{x}\bar{y}} \neq 1$ for resisting degenerate message attacks in the shared communication channel between LCSP and RTPA. It provides two-layer security from outsider adversary attacks during auditing time and insider forgery attacks at SCS.

Outer-layer-Adversary attacks in shared channel describes as an adversary $\mathcal{A}$ can interrupt a message during transmission to obtain $g^{\bar{x}}$ and $g^{\bar{y}}$ from A and B consequently and generates modified $g^{x'}$ and $g^{y'}$. So, A gets collude $\bar{g}^{y'}$ and B also gets collude $\bar{g}^{x'}$ and both prepare shared secret key $g^{x'y'}$. A sends encrypt message $g^{xy'}(m)$ to B. but during transmission, $\mathcal{A}$ modifies the message as $g^{x'y}(m')$ because $\mathcal{A}$ knows the value of $\bar{x}$, $\bar{y}$, x' and y'. In our proposed model LCSP as A and RTPA as B. LCSP and RTPA select random numbers from the set $\{1, 2, ..., p-2\}$ in such a way that $\bar{x} \mod r_1 = g$ and $\bar{y} \mod r_1 = g$. LCSP sends $g^{\bar{x}*r_1}$ to RTPA and RTPA sends $g^{\bar{y}*r_1}$ to LCSP.Therefore, LCSP and RTPA are capable to ensure whether $\bar{g}^{y'} = g^y$ and $\bar{g}^{x'} = g^x$ or not. Hence, before message transmission, both LCSP and RTPA can easily detect any adversary $\mathcal{A}$ attacks using key exchange values.

Inner Layer-In this security model, the r2 value remains unknown to all entities except the Data Owner (DO). Encrypted versions of the original block elements are stored in Shared Cloud Storage (SCS), and the Remote Third Party Auditor (RTPA) tests the auditing of these encrypted elements, which are also unknown to all, including the Cloud Service Provider (CSP), RTPA, and adversaries. Any changes in block elements cannot affect the original encrypted elements. RTPA can detect colluded blocks during auditing, allowing the DO to recover actual block elements at the index level after audit message verification, as described in the data recovery section.

6.3.6 Audit Message Verification Scheme

RTPA sends $audit\_res$ message to DO to check the deceivableness of RTPA whether the auditing result is true or not. DO checks parameters of the message which contains the status result(Y/N), x no. of challenged blocks B'_t, $t \in x$ and block elements $B'_t[\bar{M}_i]$ encrypted by R_{pri} where $i \in k$.If the status is 'Y' then DO decrypts $B_t[\bar{M}_i]$ using secret exchange key and checks the value of $\frac{B_t[\sum_{i=1}^{k}(M_i)]}{k}$ with previously stored μ_t and the value of $\frac{B_t[\sum_{i=1}^{k}(M_i)] \mod g_1}{i}$ with d_{μ_t} where $t \in x$.

If both values are the same, DO confirms that RTPA actually performs the verification test of challenged block B_t,$t \in x$ and RTPA does not produce a biased message without verifying the challenged data block otherwise data is colluded by some malicious attacks, and DO needs to apply data recovery method to reform original encrypted data elements. If the status is 'N' then again DO needs to apply the data recovery method at block index level i, $i \in k$, to get back the original encrypted version of challenged data block elements. The audit message verification scheme is explained in Algorithm 6.2 step by step.

Algorithm 6.2: Process of Audit Message Verification Method

Input:$status, x, B_t[\bar{M}_i'], g_1, \mu, k, i, DO_{pri}$ in

Output: Audit Message Verification Result (Y/N) out

Initialisation :

After verifying the $chal\_res\_pro$ message, the RTPA transmits $audit\_res$ message to the DO.
if $(status = 'Y')$ **then**
  Decrypts challenged block elements $B_t[\bar{M}_i']$ by secret exchange key based on core Deffie-Hellman protocol, $t \in x$.
  if $(status = 'Y')$ **then**
    for $t = 1$ to x **do**
      for $i = 1$ to k **do**
        if $(\frac{B_t[\sum_{i=1}^{k}(M_i)]}{k} = \mu_t)$ **then**
          Check Mod_div value of each block element
          if $(\frac{B_t[\sum_{i=1}^{k}(M_i)] \mod g_1}{i} = d_{\mu_t})$ **then**
            $audit\_res$ message verification is true.
          else
            $audit\_res$ message verification is false.
          end if
        else
          $audit\_res$ message verification is false.
        end if

17: **end for**

18: **end for**

19: **else**

20: $status = `N'$

21: **end if**

22: **end if**

6.3.7 Data Recovery and Dynamic Data Update

In an audit message verification initially, DO compares the average mean value and average mean mod value with precomputed means values which are stored on the DO side. If the verification passes the test then DO assumes that data are properly maintained at SCS otherwise DO notifies LCSP with manipulated block no.s B'_t, $t \in x$ and encrypted version of original data block elements $B_t[M'_i]$.

DO stores the values of τ, $\tau = \{1, 2, \ldots., n-1\}$, for all block elements previously at their own storage which is calculated by:

$$(B'_t[M'_i] + \tau_i) \mod g_1 = 0 \tag{6.8}$$

Here M'_i is an encrypted version of the original data M_i. DO can get the encrypted version of the original data from challenged malicious data block $B'_t[M'_i]$ by

$$(B_t \bar{M}'_i \oplus \psi - \tau_p) \mod g_1 = 0, p \in (n-1) \tag{6.9}$$

where $\oplus$ is an operation performed in this equation and $\tau = (\tau_1, \tau_2, ..., \tau_{n-1})$.

Therefore, from the above equation, DO can calculate the original encrypted data block elements at the index level. Later DO notifies LCSP with the file tag, challenged block no. along with original block elements. LCSP then verifies the consistency of maintained block elements at SCS and replaces stored block elements with new block elements. Thus, our proposed framework recovers original

block elements at the block level as well as the block index level from malicious modification of the block. Therefore, our proposed auditing model supports dynamic update operation while needed.

6.3.8 Auditing Model for Block-chain Based Medical Systeml

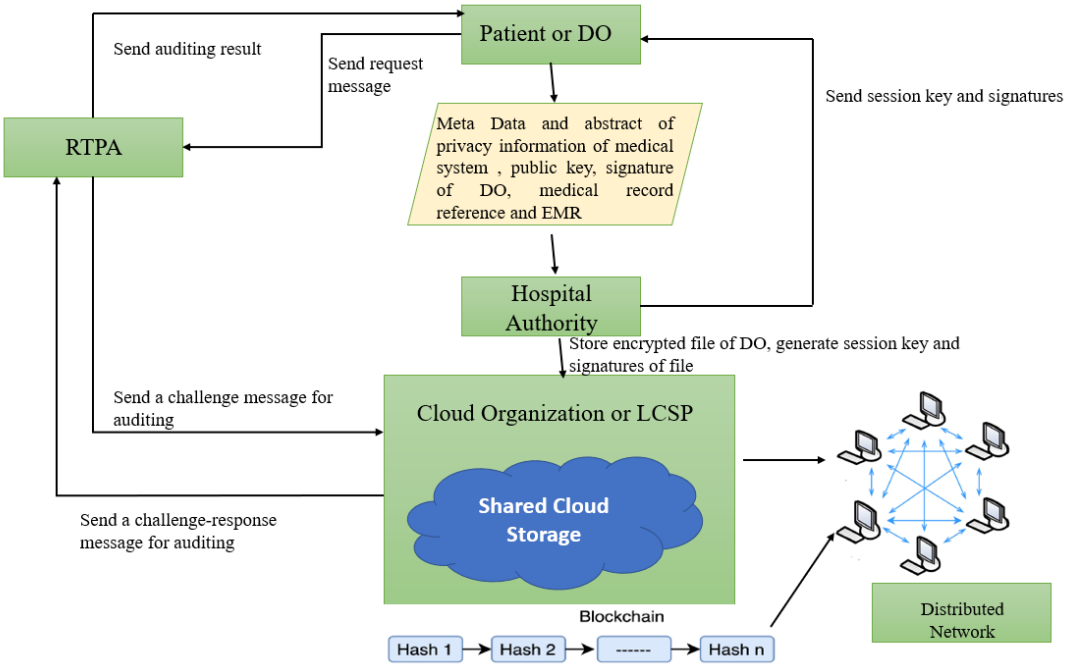

Figure 6.3: Blockchain technology-based Medical System using Proposed Auditing Scheme

Here, we discuss a blockchain-based medical system [195] where we consider our suggested auditing method through a third-party auditor to ensure the accuracy of patient information in a distributed network shown in Figure 6.3. Generally, only the MPI metadata and summary, the information provider's session key, signature, and a reference to the patient record in the cloud database are retained locally due to the general medical system's low storage capacity. All other Electronic Medical record(EMR) data is stored in a cloud database that is hosted by a separate cloud service provider. This architecture consists of mainly four entities namely Patient or DO, Hospital Authority, LCSP or Cloud organi-

zation along with cloud storage and RTPA. Patients send all medical information to the Hospital Authority, next they split it into blocks, encrypted all information, and prepare signatures for all blocks. Hospital Authority also generates a session key for the patient to access his her medical record and send it to LCSP along with signatures and block numbers. The patient treats the session key as a file tag and sends it to RTPA for auditing because blockchain technology ensures the protection of medical data but there is no guarantee about the accuracy of medical information in third-party cloud storage while using a distributed network via untrusty LCSP. Data that is kept in the cloud generates a hash value, which is also stored on the blockchain. If an opponent modifies the data in such a circumstance, it may be easily traceable. The blockchain is also used to store all data events in the form of transactions, so whoever manipulates or modifies the data does so with the knowledge that the information will be recorded and traceable. If a Third-party cloud organization will be a malicious entity, then data corruption will not be manageable because LCSP is responsible to prepare the hash values of all blocks. Using our auditing scheme via a thirdparty auditor, a patient can check the intactness of their medical information and rectify also from collude block.

6.4 Analysis of Proposed Model

In this section, the security parameters of the proposed auditing scheme are discussed. Firstly, auditing correctness is introduced, after that data confidentiality, unforgeability, batch auditing, privacy preservation, collusion resistance, replica attack resistance, error localization at the block level, and data correction are presented one by one. Lastly, a comparison of security parameters is presented in Table 6.2.

6.4.1 Security Analysis

- Data Confidentiality: The originality of DO's outsourced block elements must not be revealed to LCSP or any other nefarious authorized entity in SCS during the outsourced data integrity verification procedure. We must demonstrate that LCSP and any other attackers are unable to decode the block elements of DO's outsourced data in SCS in their original form.

  RTPA cannot get original block elements from $chal\_res\_pro$ message where $chal\_res\_pro = \{\gamma, B_t[M_i''], \omega, C_{pub})\}$, $x \in j$. LCSP stores block data elements from $info\_msg$ come from DO, where $info\_msg = \{B_t[M_i'], S, \omega, g_1, D_{pub}\}$. $B_t[M_i']$ are encrypted version of each block $B_t[M_i]$ which is encrypted by DO_{pri}. i.e.,

$$MsgEncrypt(Do_{pri}, M) = \sum_{t=1}^{j}\sum_{i=1}^{k} B_t(M_i)) \quad = (r2 * g1) \oplus (\sum_{t=1}^{j}\sum_{i=1}^{k} B_t[M_i]))$$
$$= (\sum_{t=1}^{j}\sum_{i=1}^{k} B_t[M_i'])$$

  RTPA knows the value of g_1 but no one [neither LCSP nor RTPA] cannot disclose the value of r_2 except DO. Hence, it is proved that RTPA never discovers the original block elements of DO, and data confidentiality is properly maintained at both LCSP and RTPA sides.

- Unforgeability: A hostile authorized attacker cannot fabricate evidence (γ) by tricking LCSP since every time the RTPA delivers a challenge to LCSP as $chal\_pro$ message, ω is included in the $chal\_pro$ message. The LCSP will generate evidence γ depending on x from the RTPA. The RTPA verification procedure shows that the RTPA must validate *omega* with req_gen message, which also contains *omega* value, even if we assume that the malicious LCSP forges γ' from the preceding proof, specifically $\gamma \neq \gamma'$. Therefore,

Equation(5.6) cannot be held by the γ'.

$$\gamma' = \sum_{t=1}^{\tau} r_1/[sk + g1 * H(B_t[M_i''])] \tag{6.10}$$

Another example is when an unreliable LCSP tries to deceive RTPA by replacing the disputed data block $\sum_{t=1}^{x}\sum_{i=1}^{k} B_t[\theta_i]$ with another data block $\sum_{t=1}^{x}\sum_{i=1}^{k} B_t[M_i']$ and setting up hash values for collusive blocks when the former data blocks are corrupted. So, Equation (5.7) can be represented as:

$$e(g1 * \sum_{t=1}^{x} S_t, \frac{sk}{\gamma'}) = e(g_1, g_1) \tag{6.11}$$

As a result, $H(\sum_{t=1}^{j}\sum_{i=1}^{k} B_t[M_i'] = H(\sum_{t=1}^{j}\sum_{i=1}^{k} B_t[\theta_i]$. However, due to the hash function's anti-collision characteristic, $H(\sum_{t=1}^{j}\sum_{i=1}^{k} B_t[\theta_i]$ cannot be equal to $H(\sum_{t=1}^{j}\sum_{i=1}^{k} B_t[M_i']$. As a result, making Equation (5.10) hold is impossible, and The verification process cannot accept the proof from LCSP.

- Privacy Preservation: In this proposed model, RTPA verifies γ with signatures S_t of encrypted block elements which are totally unknown to both RTPA as well as LCSP. They never know the DO_{pri} and also the random no. r_2. The confidential information of DO is hidden by r_2. Therefore, our proposed model well maintains DO's privacy.

- Collision Resistance & Replica Attack Resistance: If the LCSP does not collude block elements of outsourced data that originated from DO, only the LCSP passes the RTPA auditing test. We must demonstrate that the LCSP cannot produce a legitimate response to the RTPA's auditing test unless it is true and does not alter any block components of the DO's outsourced data.

  The challenge-response message $chal\_res\_pro$ from LCSP can pass only the auditing verification phase only if LCSP doesn't replace the values of

legitimate block elements $B_t[M_i']$ with corrupted block elements $B_t[\theta_i]$ where $t \in j$. LCSP cannot change the signature hash value of S_t due to a lack of information about r_1. Hence, $H(M_i') \neq H(\theta_i)$ and $e(g1 * \sum_{t=1}^{x} S_t, sk/\gamma) \neq e(g1 * \sum_{t=1}^{x} S_t', sk/\gamma')$ because of property of collision resistance of hash function. The γ' produced by LCSP never matches with S_t' of original block elements $B_t[M_i']$.

L.H.S

$$
\begin{aligned}
& e(g_1 * \sum_{t=1}^{x} S_t, \frac{sk}{\gamma}) \\
& = e(g_1 * \frac{g_1}{H(B_1[M_i']) + r_1} + \frac{g_1}{H(B_2[M_i']) + r_1} + ... + \frac{g_1}{H(B_x[M_i']) + r_1}, \frac{sk}{\gamma}) \\
& = e(g_1 * \sum_{t=1}^{x} S_t, \frac{sk}{\frac{r_1}{r_1 * g_1 + g_1 * H(B_1[M_1'])} + \frac{r_1}{r_1 * g_1 + g_1 * H(B_2[M_1'])} + .. + \frac{r_1}{r_1 * g_1 + g_1 * H(B_x[M_i'])}}) \\
& = e(g_1 * \frac{g_1}{H(B_1[M_i'] + r_1} + \frac{g_1}{H(B_2[M_i'] + r_1} + ... + \frac{g_1}{H(B_x[M_i'] + r_1}, \\
& \qquad \frac{g_1}{\frac{1}{r_1 * g_1 + g_1 * H(B_1[M_1'])} + \frac{1}{r_1 * g_1 + g_1 * H(B_2[M_1'])} + .. + \frac{1}{r_1 * g_1 + g_1 * H(B_x[M_i'])}}) \\
& = e(g_1, g_1)(\frac{g_1}{H(B_1[M_i'] + r_1} + \frac{g_1}{H(B_2[M_i'] + r_1} + ... + \frac{g_1}{H(B_x[M_i'] + r_1}, \\
& \qquad \frac{1}{\frac{1}{r_1 * g_1 + g_1 * H(B_1[M_1'])} + \frac{1}{r_1 * g_1 + g_1 * H(B_2[M_1'])} + .. + \frac{1}{r_1 * g_1 + g_1 * H(B_x[M_i'])}}) \\
& = e(g_1, g_1)
\end{aligned}
$$

R.H.S

$$
\begin{aligned}
& e(g_1 * \sum_{t=1}^{x} S_t, \frac{sk}{\gamma'}) \\
& = e(g_1 * \frac{g_1}{H(B_1[M_i']) + r_1} + \frac{g_1}{H(B_2[M_i']) + r_1} + ... + \frac{g_1}{H(B_x[M_i']) + r_1}, \frac{sk}{\gamma'}) \\
& = e(g_1 * \sum_{t=1}^{x} S_t, \frac{sk}{\frac{r_1}{r_1 * g_1 + g_1 * H(B_1[\theta_i])} + \frac{r_1}{r_1 * g_1 + g_1 * H(B_2[\theta_i])} + .. + \frac{r_1}{r_1 * g_1 + g_1 * H(B_x[\theta_i])}}) \\
& = e(g_1 * \frac{g_1}{H(B_1[M_i'] + r_1} + \frac{g_1}{H(B_2[M_i'] + r_1} + ... + \frac{g_1}{H(B_x[M_i'] + r_1}, \\
& \qquad \frac{g_1}{\frac{1}{r_1 * g_1 + g_1 * H(B_1[\theta_i]} + \frac{1}{r_1 * g_1 + g_1 * H(B_2[\theta_i]} + .. + \frac{1}{r_1 * g_1 + g_1 * H(B_x[\theta_i]}}) \\
& = e(g_1, g_1)(\frac{g_1}{H(B_1[M_i'] + r_1} + \frac{g_1}{H(B_2[M_i'] + r_1} + ... + \frac{g_1}{H(B_x[M_i'] + r_1}, \\
& \qquad \frac{1}{\frac{1}{r_1 * g_1 + g_1 * H(B_1[\theta_i]} + \frac{1}{r_1 * g_1 + g_1 * H(B_2[\theta_i]} + .. + \frac{1}{r_1 * g_1 + g_1 * H(B_x[\theta_i]}})
\end{aligned}
$$

Hence, $L.H.S \neq R.H.S$. Thus, our proposed protocol can resist collision occurrence and replace attacks.

- Auditing Correctness: If the LCSP passes the verification stage, it must contain all required data. We must show that the LCSP is unable to produce an accurate response for the RTPA without the outsourced data being faithfully stored.

  The $chal\_res\_pro$ message is successfully verified by RTPA if only if LCSP properly stores all block data elements of info_msg where $chal\_res\_pro = \{\gamma, B_t[M_i''], x, \omega\}$, $x \in j$, $i \in k$.The signature verification process is proven here using the ZSS signature verification process using (5.6) and (5.7) no. Equations as follows:

$$
\begin{aligned}
& e(g1 * \sum_{t=1}^{x} S_t, \frac{sk}{\gamma}) \\
& = e(g_1 * \frac{g_1}{H(B_1[M_i']) + r_1} + \frac{g_1}{H(B_2[M_i']) + r_1} + ... + \frac{g_1}{H(B_x[M_i']) + r_1}, \frac{sk}{\gamma}) \\
& = e(g_1 * \frac{g_1}{H(B_1[M_i']) + r_1} + \frac{g_1}{H(B_2[M_i']) + r_1} + ... + \frac{g_1}{H(B_x[M_i']) + r_1}, \\
& \qquad \frac{sk}{\frac{r_1}{r_1 * g_1 + g_1 * H(B_1[M_1'])} + \frac{r_1}{r_1 * g_1 + g_1 * H(B_2[M_1'])} + .. + \frac{r_1}{r_1 * g_1 + g_1 * H(B_x[M_i'])}}) \\
& = e(g_1 * \frac{g_1}{H(B_1[M_i'] + r_1} + \frac{g_1}{H(B_2[M_i'] + r_1} + ... + \frac{g_1}{H(B_x[M_i'] + r_1}, \\
& \qquad \frac{g_1}{\frac{1}{r_1 * g_1 + g_1 * H(B_1[M_1']} + \frac{1}{r_1 * g_1 + g_1 * H(B_2[M_1']} + .. + \frac{1}{r_1 * g_1 + g_1 * H(B_x[M_i']}}) \\
& = e(g_1, g_1)(\frac{g_1}{H(B_1[M_i'] + r_1} + \frac{g_1}{H(B_2[M_i'] + r_1} + ... + \frac{g_1}{H(B_x[M_i'] + r_1}, \\
& \qquad \frac{1}{\frac{1}{r_1 * g_1 + g_1 * H(B_1[M_1']} + \frac{1}{r_1 * g_1 + g_1 * H(B_2[M_1']} + .. + \frac{1}{r_1 * g_1 + g_1 * H(B_x[M_i']}}) \\
& = e(g_1, g_1)
\end{aligned}
$$

This will be satisfied if $B_t[M_i'] = R_{pri}\{B_t[M_i'']\}$ will be true where $t \in x$. Therefore, it is proved that RTPA can only audit successfully $chal\_res\_pro$ message returned by LCSP when LCSP correctly stores block data elements in SCS.

- Error Localization : If DO correctly processes all block elements for Mod_div values, it will be able to localize block index level errors and recover them during audit message verification time.

  When DO receives an $audit\_msg$ from RTPA, DO will first determine whether the audit message is valid by determining whether the status is yes or no. If the status is "no," DO will later check each block element of all challenged blocks against previously stored values in Table T as follows:

$$
\frac{B_t[\sum_{i=1}^{k}(M_i)] \mod g_1}{i} = d_{\mu_{ti}} \tag{6.12}
$$

where $t \in x$ and $i \in k$.If the aforementioned equation is false, DO will be able to identify the colluded block with colluding block element at the block index level. Then DO implement our suggested data recovery technique using Equation 6.1 and Equation 6.2 without bringing up the matter of collusions in front of either LCSP or RTPA. When DO recovers colluding block elements, it encrypts with its own private key and transmits them to LCSP for block element modification at the index level. As a result, our auditing model maintains both data confidentiality and data privacy.

6.4.2 Performance Analysis:

Finding out the computational cost of signature generation with block numbers, signature verification with block numbers, and audit message verification with block size(kilobytes(KB)) is one primary goal of this analysis part. The communication cost of DO, LCSP, and RTPA is another goal of this analysis. Here, comparative simulation analysis of $chal\_res\_pro$ message generation was also examined. These are the experimental settings: The processor is an Intel(R) Core(TM) i5-9300H, the RAM is 12 GB, and Windows 10 is running on it. Simulation is done in CloudSim.

Research Scenario

As per the research requirements, any research needs a research scenario to propose the research model. So, this research focuses on the given problem statement in terms of cloud storage security. The proposed research focuses on given input research parameters as per the proposed machine learning model requirements such as $Block\_no$,$Block\_size$, Time, etc. It analysis the performance of the proposed model, and research requires some data sets. The sample of data sets is described in given Table 6.4.

Computational Overhead

This analysis section mainly focuses on finding out the computational overhead of encryption time of all block elements, signature generation time, $chal\_res\_pro$ generating time, signature verification time, and audit message verification time. Here experimental environments are: Intel(R) Core (TM) i5-9300H CPU 2.40GHz 2.40 GHz, RAM size is 12 GB and the operating system is Windows 10. The effectiveness of the suggested work was later demonstrated, along with a comparison of the proposed approach with past research works [82, 143, 146, 195]. Figure 6.4 shows the computational cost of signature generation time for the proposed system and the research publications [82, 143, 146, 195] with various data blocks 100, 150, 200, 250, 300, 350, and 400.

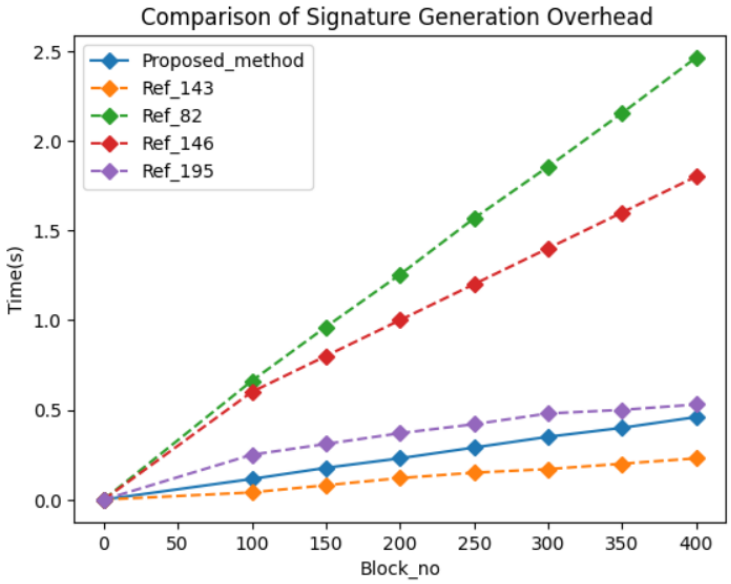

Figure 6.4: Comparison of Signature Generation Time

In terms of computation complexity, we compare our ZSS-based auditing scheme to the other ZSS signature-based schemes [82, 143, 195], and the algebraic signature-based scheme [146] in terms of computational overhead for signature generation and verification, encryption overhead with [143] and overhead of challenge proof message generation with [143]. Table 6.3 is defined to show the meaning of all crypto operations in order to establish unambiguous comparisons and Table 6.5

Table 6.3: Notations for Computational overhead

Notation of computational operation	**Meaning**
$Pair$	Bilinear pairing operation
Mul	Multiplication Operation
Inv	Inverse operation
$Hash$	Hash operation
Exp	Exponential operation
$Encrpt$	Symmetric encryption operation
$Decrypt$	Symmetric decryption operation
Add	Addition operation

is defined to show comparisons of the computational overhead of the proposed method with previous research works [82, 143, 146, 195].

In Figure 6.4, a comparison of the computational overhead of signature generation time is shown for our proposed scheme with research papers [82, 143, 146, 195]. The average computational time for signature generation of proposed work is 3.038s and in [82, 143, 146, 195] research work 1.36,0.129,1.05,0.357s times have been taken for sign generation. The difference between average computational times for sign generation in the above-mentioned schemes [82, 146, 195] are 1.107s, 0.797s, and 0.104s which are minimum. Only [143] has less computational overhead for signature generation than our proposed work due to encryption operation. For all research works, a file F is divided into j no. of blocks with variable sizes where B={B1, B2,. . . , Bj} and signature is applied on each block level. In our proposed auditing method, blocks are encrypted by SHA encryption whereas [143] done block element encryption by AES algorithm. Therefore, from Figure 6.5, we conclude that our proposed method takes less encryption overhead than [143].

Hence, from Equation(6.3), we assume that $jHash$ computational operations require to calculate hash values for j blocks, $jMul$ multiplication operations for j blocks, $jAdd$ addition operation for $j$blocks and $jInv$ inverse operations for j

Table 6.4: Sample values for Signature Generation, Signature Verification, and Audit Message Verification

$Block\_no$	**Time for Signature Generation(s)**	**Time for Signature Verification(s)**	**Time for $chal\_res\_pro$ Message verfication(s)**
100	0.155	0.14	0.26
150	0.177	0.21	0.38
200	0.23	0.28	0.50
250	0.29	0.35	0.62
300	0.35	0.41	0.77
350	0.40	0.48	0.94
400	0.46	0.55	1.11

blocks are required for the signature time of j blocks.

In our proposed model DO encrypts all block values with their own non-sharable private key DO_{pri}. Though the extra $nEncrpt$ computational operations are required for the signature generation scheme, it enhances the data privacy security level of outsourced data in SCS due to the lack of an original version of it. Therefore, the computational cost for signature generation of DO is $nEncrpt+jHash+jMul+jAdd+jInv$. On the other hand, from Figure 6.6, we can conclude that [82, 143, 146, 195] require 2.52s, 0.35s 1.03s, and 0.41s for average signature verification overhead where our proposed method requires 0.30 s averagely.
Research papers [82, 144, 195] require two pairing operations, and another paper [62] requires j pairing operations. Our suggested data auditing approach requires only one pair operation like [151]. As per the proposed model's assumption, RTPA prepares a challenge message known as $chal\_msg\_pro$ includes encrypted block no.s x. LCSP performs a decryption operation using L_{pri} on $chal\_msg\_pro$, gets block no.s, and prepares encrypted challenge response message chal_res_pro by C_{pri}. For preparing this message, LCSP has some computational cost and this is $xAdd+Mul+Inv$. Figure 6.7 shows here the comparison of LCSP's computational time of our proposed model and the Data possession verification model of [143]. In this figure, the size. of blocks has been increased with an interval of 50 and time

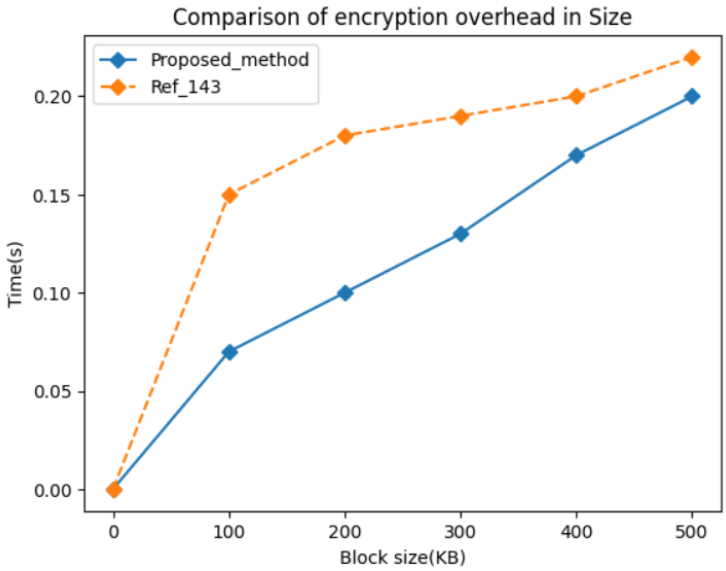

Figure 6.5: Comparison of Encryption Overhead

is defined herein as seconds. The difference between the average computational time for $chal\_res\_pro$ message preparation in both schemes is .30 s because our proposed scheme consumes less time than[82]. Therefore, from the depiction of Figure 6.7, we can say that our proposed model has more efficiency than [143] in terms of LCSP's computational time to prepare a challenge-response message which needs to verify for the data auditing test by RTPA in public auditing.

In our proposed model, audit message verification is an important contribution that is not defined in [60, 62, 151, 65, 34] research works. Here, Figure 6.8 depicts audit message verification through DO through block no.s.

Communicational Overhead

Here, the communication overhead of $info\_msg$, $chal\_msg\_pro$, $chal\_res\_pro$, $audit\_proof$ message have been measured. Figure 6.9 depicts the total communication overhead of DO, LCSP, and RTPA. It shows LCSP's communication overhead as the processing of $info\_msg$ at SCS and processing of $chal\_msg\_pro$ message sent by RTPA. Figure 6.9 also depicts RTPA's communication overhead as the processing of req_msg sent by Do and processing of $chal\_res\_pro$ message

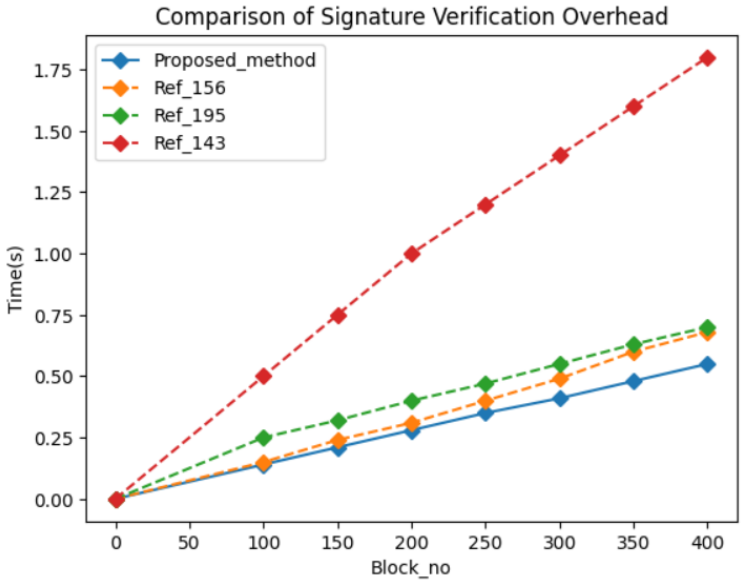

Figure 6.6: Comparison of Signature Verification Overhead

sent by LCSP. For the sample test, here, we consider different no. blocks with 50 sectors to measure the processing time of LCSP. The communication cost spent by LCSP for this $info\_msg$ is approximately $(|\omega|+|g_1|+|r_1|+|r_2|+j(|h(M')|+|k|))$, where $|\omega|$ is the size of file tag,$|g_1|$ is the length of a group generator,$|r_1|$ and $|r_2|$ are the size of random keys,$|h(M')|$ is the length of hash values of per blocks, $|k|$ is a length of each encrypted block sent by DO to LCSP and the length of $info\_msg$ is nearly similar to [82]. RTPA is incurred the cost of communication overhead is $|j|+|g_1|+|r_1|+j|S|$ which is send by DO as req_msg. Here, |S| is the size of the signature per block. The communication overhead of $chal\_msg\_pro$ message is $|\omega|+|x|$. The cost is the total length of $chal\_res\_pro$ message is$|\gamma|+|\omega|+x|k|$ which is sent by LCSP to RTPA. Therefore total communication overhead of LCSP is $2|\omega|+|g_1|+|r_1|+j|h(M')|+j|k|+|x|$, where $t \in j$ and for RTPA, communication overhead is $2|\omega|+|j|+|g_1|+|r_1|+j|s|+|\gamma|+|B'_t[M'_i]|$ where $t \in x$.

In our proposed model, audit message verification is an important contribution that is not defined in [82, 143, 144, 146, 101]. The communicational cost incurred by DO to process audit_proof message is $x|k|+|c|$ where $x|k|$ is the total length of challenging blocks and $|c|$ is the length of the status value.

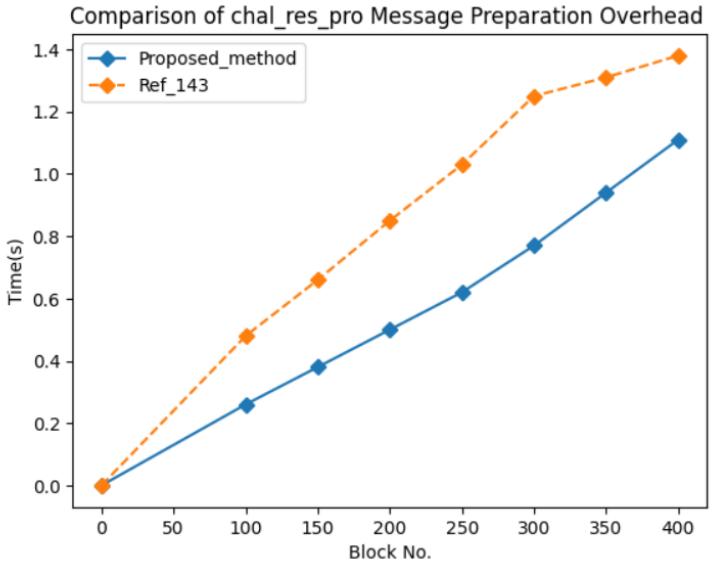

Figure 6.7: Comparision of chal_res_pro Message Generation Overhead

Here, we take into account three different scenarios, including DO to LCSP communication, DO to RTPC communication, RTPC to LCSP communication, and LCSP to RTPC communication. Similar to the other methods [82, 143, 144, 146, 101], DO sends a file to LCSP to store their data in cloud storage, and at the same time, DO hires RTPC as an auditor to verify the accuracy of the data they have outsourced. RTPC prepares a challenge message and sends it to LCSP; RTPC then validates the proof that LCSP sent after it produces evidence of verification and sends it to RTPC. In our proposed model, First communication overhead is O(j), where j is the total number of blocks that have been requested for LCSP storage, as in [82, 143, 144, 146, 101] depicted in Table 6.6. Like [82, 143, 195, 144, 101], the second communication overhead is O(1). and third communication overhead claims O(x) for x number of blocked challenges, such as [82, 144, 143, 101].

Storage Overhead

The ZSS signatures of j data blocks are computed as $S_t = [1/(H(M'_t)) + r_1].g_1$, $t \in j$. DO divides the file F into chunk size variable blocks $B = \{B_1, B_2, ...B_j\}$

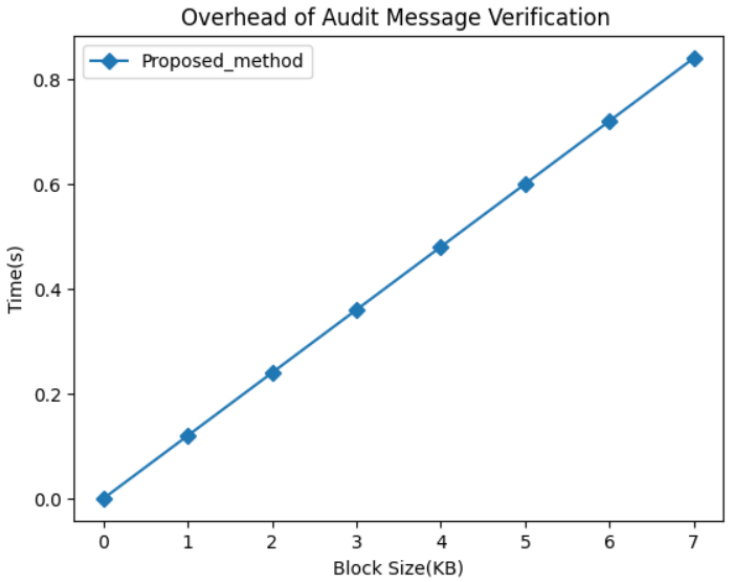

Figure 6.8: Computational Overhead of Audit Message Verification

where the maximum size of each block is k and and $1 < n < cel(n/d)$, $2 \leq d \leq k$. Here, k denotes the total no. of block sectors per block. Each block generates one signature. If a chunk-sized data block is divided into k sectors, and applies ZSS signature generation technique, exclusively one signature will be generated for a block. Therefore, $1/k$ storage overhead is required per block instead of k during sign generation time. j no. of signature along with j block elements are stored at SCS through LCSP. Hence total storage overhead of LCSP is O(j). Do stores group generator g_1, two random numbers r_1,r_2, a set of substitute value τ, the mean value of each block μ and average mode value of each block d_μ, thus storage overhead of DO is O(n). RTPA stores public key DO_{pub}, group generator g_1, and total block no. j for performing outsourced data auditing processes partially to check data intactness. Hence, the storage overhead of RTPA is $O(1)$. If RTPA performs m times auditing test, the storage overhead will be $O(m)$.

Table 6.5: Comparison of Computational overhead

Entity	[143]	[146]	[144]	[183]	[111]	[82]	Our Proposed Model
DO	$j(Hash+Mul+2Exp)+j*KEncrypt$	$j(2Exp+1Mul+1Hash)$	$4jExp+jHash+jMul$	$nHash+nMul+2nAdd+nInv$	$j(Exp+Inv+Mul+Add+3Hash)$	$jHash+jMul+jAdd+jInv$	$(j*k)Encrypt+jhash+jAdd+jInv$
LCSP	NA	$2MulExp$	$xMul+xExp$	$jHash+2Add+Mul+Inv+4Mul$	$j(2Mul+Hash)$	$jHash+2Add+Mul+Inv+4Mul$	$xAdd+Mul+Inv$
RTPA	$xMul+xHash+2xExp+Pair$	$(x+1)Exp+1Mul$	$xExp+2Pair+xHash+xMul$	$1Pair+2Mul(2tAdd+1)+2Add$	$j(Add+2Hash+Pair)$	$Mul+Add+2Pair$	$Pair+Mul+xAdd+Inv$
DO (Audit MSG Verify)	NA	NA	NA	NA	NA	NA	$(K*x)Decrypt+2Mod$

6.5 Summary

In conclusion, block level information is required to preserve data integrity on shared cloud storage, therefore the auditing procedure in this research study concentrates on correctness at the block level, message verification at the DO side, and minimising error localisation at the block index level. ZSS signature and original data modification technique through random number generation ensure consistency of outsourced data for DO at SCS While RTPA can compute if data blocks are colluded or not and DO can restore the colluded data blocks with sustainable values at DO's device, this procedure is helpful in replacing attacks. Data privacy is enforced by this innovative data auditing architecture from all data imports, including CSP and RTPA. For public audit decision-making on the same reduced computational and communication for improved solutions based on

Table 6.6: Comparison of Communication overhead

Reference	**Type of Signature**	**DO to LCSP**	**LCSP to RTPC**	**RTPC to DO**
[82]	ZSS	O(j)	O(x)	O(1)
[182]	ZSS	O(j)	O(j)	O(1)
[183]	ZSS	O(j)	O(x)	O(1)
[143]	BLS	O(j)	O(x)	O(1)
[146]	Algebric	O(j)	O(x)	O(1)
Proposed Method	ZSS	O(j)	O(x)	O(x)

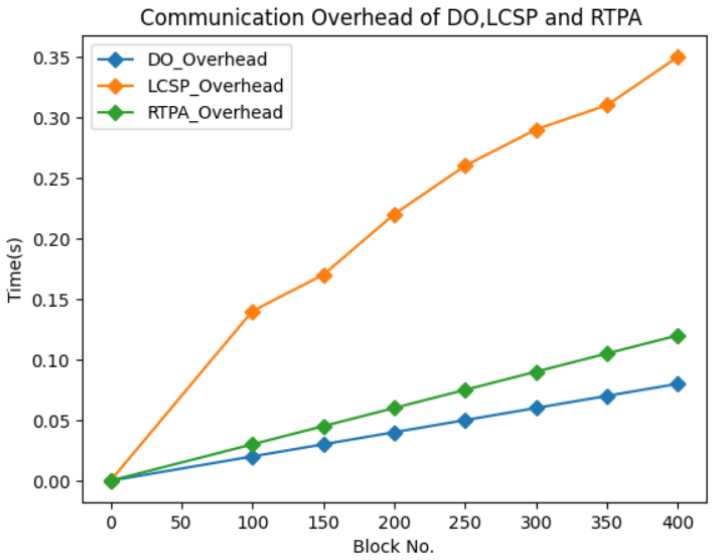

Figure 6.9: Communication Overhead of DO, LCSP and RTPA

experimental results, the dynamic data audit operational outcomes are extremely helpful. Future work will concentrate on improving the verification paradigm, with a particular emphasis on multiple block data integrity, computation, and testing at both the DO and TPA sides.

Chapter 7

Conclusion AND Future Scope

With the continuously enlarging popularity of attractive and optimized cost-based cloud ser- vices, it is inconvenient to make sure data owners the intactness of outsourced data in cloud storage environments has become a disaster security challenge. In our research work, we have tried to highlight several issues and the corresponding solution approaches for cloud data integrity which will provide a visualization as well as clear directions to researchers. The current state of the art in this mentioned research field will provide extra milestones in several areas like cloud-based sensitive health care, secured financial service, managing social media flat-forms, etc.

Here, we have discussed phases of data integrity, characteristics of data integrity scheme, classification of data integrity strategy based on the type of proposal, nature of data and type of verification schemes, and desired design challenges of data integrity strategy based on performance overhead. Throughout the course of this study, we have investigated issues in physical cloud storage and attacks on storage-level services along with mitigating solutions.

Firstly, we outlined the research problem and built the foundation for the following chapters as we started our investigation. The journey started with the introduction of concepts such as data recovery, cloud data integrity classification, and cloud data security for cloud users. Chapter 1, introduction section provided a strong framework for our investigation by highlighting the critical role that symptoms, data integrity verification systems, and cloud data security challenges play. After that, we read through the literature to learn about the most recent methods and strategies used in cloud-outsourced data integrity verification.
Chapter 2 included a thorough assessment of the literature, a comparison of methods in the area of third-party auditing schemes for cloud data, and an analysis of

pertinent research and current state-of-the-art methodologies.

In Chapter 3, to ensure that the cloud user's entire file is properly stored in cloud storage, the accuracy of the acknowledgment message is achieved on CU side. Additionally, this practical ACK paradigm protects CU's data privacy during verification in the SCS from hostile attackers like CSP and other authorized parties. It helps to resist replace attacks and forgeability attacks.

In Chapter 4, We have also discussed the use of ZSS signature and original data encryption method based on random number generation to ensure the consistency of outsourced data for CC at cloud resource pool. It helps to prevent forgeability attacks, and resist collusion. Furthermore, AE can detect and inform CC of block corruption during auditing. Furthermore, while completing the audit, this user-friendly auditing architecture secures the privacy of CC's data from all entities, including CRP and AE.

In Chapter 5, dynamic operations are accomplished to protect the integrity of outsourced data on shared cloud storage by auditing correctness at the block level. Consistency of outsourced data for CU at RCS is ensured through the use of a partial signature and original data encryption technology using random number generation. It aids in thwarting replace and forgeability attacks. Additionally, PTPC is capable of detecting block corruption during auditing and notifying CU. Additionally, this user-friendly auditing architecture protects the privacy of CU's data from all entities, including CSP and PTPC, while conducting the audit. Additionally, the dynamic data update operation is carried out while maintaining public auditing on the same with reduced computational cost and has already been shown through the aforementioned experimental results to be a better solution than the current options.

In Chapter 6, we have discussed the Block level of information that is required to preserve data integrity on shared cloud storage, therefore the auditing procedure in this research study concentrates on correctness at the block level, message verification at the DO side, and minimizing error localization at the block index level. ZSS signature and original data modification technique through random

number generation ensure consistency of outsourced data for DO at SCS While RTPA can compute if data blocks are colluded or not and DO can restore the colluded data blocks with sustainable values at DO's device, this procedure is helpful in replacing attacks. Data privacy is enforced by this innovative data auditing architecture from all data imports, including CSP and RTPA. For public audit decision-making on the same reduced computational and communication for improved solutions based on experimental results, the dynamic data audit operational outcomes are extremely helpful.

7.1 Summary of Our Research Contributions

The main contributions of the thesis can be summarized as follows:

Survey of literature works and find out objectives: We have read a succinct analysis of the body of research on cloud data integrity verification schemes. This chapter provided a basis for comprehending the different approaches, methodologies, and a comparative study of the cloud storage data integrity strategy for projected design methods with constraints by reviewing previous research. The process of literature survey helps to find out the research gap and our thesis objectives.

Development of Public Auditing Model based on Stub Signature: In order to maintain data security, avoid forgery attacks, and provide collision resistance, the proposed study proposed a partial signature-based data auditing technique that is both effective and computationally time-consuming for the production and verification of signatures.

Development of Acknowledgement Verification Model: It provided a modification method for encrypting the original block elements in order to protect data elements in cloud storage from both internal and external hostile attacks. In Chapter 4, a successful public data auditing method based on ZSS signatures was shown. This method protects against collusion and forgery attacks while guaranteeing data confidentiality and privacy. This approach tackles the previously

described difficulties while taking into consideration the existing situation.

Development of Public Auditing Model based on ZSS Signature: In order to maintain data security and confidentiality while doing data integrity audits, a quick signature-based outsourced data auditing system is proposed. Furthermore, this approach ensures DO audit message verification. This method offers a way to defend cloud storage's unique data element against damaging internal and external attacks. This method prevents attacks and forgeries on cloud storage and withstands block collision problems.

Development of Audit Message Verification Model based on ZSS Signature: It introduced the use of challenge-response messages from cloud service providers and a way for preparing challenge messages by remote third-party auditors. This approach ensures an audit message verification test at the cloud user side and utilised optimal computational resources. This chapter also introduced the cloud user-side modular encrypted data recovery method and presented the block index level public auditing error localization approach.

7.2 Future Research Directions

As further research work, we are discussing here the future direction of the data integrity scheme to enlarge the scope of cloud data security for research process continuity. New emerging trends in data integrity schemes are listed below.

a) **Blockchain data-based integrity** : Blockchain technology is decentralized, peer-peer technology. It supports scalable and distributed environments in which all the data are treated as transparent blocks that contain cryptographic hash information of the previous block, and timestamps to resist any alteration of a single data block without modifying all the subsequent linked blocks. This feature of this technology improves the performance of cloud storage and maintains the trust of data owners by increasing data privacy through the Merkle tree concept. In [155], a distributed virtual

agent model is proposed through mobile agent technology to maintain the reliability of cloud data and to ensure trust verification of cloud data via multi-tenant. In [156], a blockchain-based generic framework is proposed to increase the security of the provenance data in cloud storage which is important for accessing log information of cloud data securely. In [157, 158, 159], all research works have the same intention of using blockchain technology to enhance data privacy and maintain data integrity in cloud storage.

b) **Data integrity in fog computing** : Generally, privacy protection schemes are able to resist completely insider attacks in cloud storage. In [196], a fog computing-based TLS framework is proposed to maintain the privacy of data in Fog servers. The extension part of cloud computing is fog computing which was firstly introduced in 2011[197]. The three advantages of fog computing are towering real-time, low latency, and broader range geographical distribution which is embedded with cloud computing to ensure the privacy of data in fog servers which is a powerful supplement to maintain data privacy preservation in cloud storage.

c) **Distributed Machine Learning Oriented Data Integrity** : In artificial intelligence, maintaining the integrity of training data in the distributed machine learning environment is a rapidly growing challenge due to network forge attacks. In [146], a distributed machine learning-oriented data integrity verification scheme (DML-DIV) is introduced to assure training data intactness and to secure the training data model. PDP sampling auditing algorithm is adopted here to resist tampering attacks and forge attacks. Discrete logarithm problem (DLP) is introduced in the DML-DIV scheme to ensure privacy preservation of training data during TPA's challenge verification time. To reduce key escrow problem and certificate cost, identity-based cryptography and key generation technology are proposed here.

d) **Data Integrity in Edge Computing** : Edge computing is an exten-

sional part of distributed computing. Cache data integrity is a new concept in edge computing developed based on cloud computing which serves optimized data retrieval latency on edge servers and gives centralized problems of cloud storage server. Edge data integrity(EDI) concept is first proposed to effectively handle auditing of vendor apps' cache data on edge servers which is a challenging issue in dynamic, distributed, and volatile edge environments described In [198]. Research work proposed here EDI-V model using variable Merkle hash tree (VMHT) structure to maintain cache data auditing on a large scale server through generating integrity of replica data of it. In [199], the EDI-S model is introduced to verify the integrity of edge data and to localize the corrupted data on edge servers by generating digital signatures of each edge's replica.

REFERENCES

[1] R. Buyya, J. Broberg, and A. M. Goscinski, *Cloud computing: Principles and paradigms.* John Wiley & Sons, 2010.

[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, *et al.*, "Above the clouds: A berkeley view of cloud computing," tech. rep., Technical Report UCB/EECS-2009-28, EECS Department, University of California . . . , 2009.

[3] P. Mell, T. Grance, *et al.*, "The nist definition of cloud computing," 2011.

[4] P. Mell, T. Grance, *et al.*, "The nist definition of cloud computing," 2011.

[5] Y. Sun, J. Zhang, Y. Xiong, and G. Zhu, "Data security and privacy in cloud computing," *International Journal of Distributed Sensor Networks*, vol. 10, no. 7, p. 190903, 2014.

[6] M. R. Sutradhar, N. Sultana, H. Dey, and H. Arif, "A new version of kerberos authentication protocol using ecc and threshold cryptography for cloud security," in *2018 Joint 7th International Conference on Informatics, Electronics & Vision (ICIEV) and 2018 2nd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, pp. 239–244, IEEE, 2018.

[7] S. C. Patel, R. S. Singh, and S. Jaiswal, "Secure and privacy enhanced authentication framework for cloud computing," in *2015 2nd International Conference on Electronics and Communication Systems (ICECS)*, pp. 1631–1634, IEEE, 2015.

[8] H. Hong, Z. Sun, and Y. Xia, "Achieving secure and fine-grained data authentication in cloud computing using attribute based proxy signature," in *2017 4th International Conference on Information Science and Control Engineering (ICISCE)*, pp. 130–134, IEEE, 2017.

[9] W. Wang, L. Ren, L. Chen, and Y. Ding, "Intrusion detection and security calculation in industrial cloud storage based on an improved dynamic immune algorithm," *Information Sciences*, vol. 501, pp. 543–557, 2019.

[10] Q. Yan, F. R. Yu, Q. Gong, and J. Li, "Software-defined networking (sdn) and distributed denial of service (ddos) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE communications surveys & tutorials*, vol. 18, no. 1, pp. 602–622, 2015.

[11] S. Dong, K. Abbas, and R. Jain, "A survey on distributed denial of service (ddos) attacks in sdn and cloud computing environments," *IEEE Access*, vol. 7, pp. 80813–80828, 2019.

[12] C. Thirumallai, M. S. Mekala, V. Perumal, P. Rizwan, and A. H. Gandomi, "Machine learning inspired phishing detection (pd) for efficient classification and secure storage distribution (ssd) for cloud-iot application," in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 202–210, IEEE, 2020.

[13] B. F. Mary and D. G. Amalarethinam, "Data security enhancement in public cloud storage using data obfuscation and steganography," in *2017 World Congress on Computing and Communication Technologies (WCCCT)*, pp. 181–184, IEEE, 2017.

[14] I. Nakouri, M. Hamdi, and T.-H. Kim, "A new biometric-based security framework for cloud storage," in *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 390–395, IEEE, 2017.

[15] A. Bakas, H.-V. Dang, A. Michalas, and A. Zalitko, "The cloud we share: Access control on symmetrically encrypted data in untrusted clouds," *IEEE Access*, vol. 8, pp. 210462–210477, 2020.

[16] A. Meddeb-Makhlouf, F. Zarai, *et al.*, "Distributed firewall and controller for mobile cloud computing," in *2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA)*, pp. 1–9, IEEE.

[17] Y. Fu, M. H. Au, R. Du, H. Hu, and D. Li, "Cloud password shield: A secure cloud-based firewall against ddos on authentication servers," in *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, pp. 1209–1210, IEEE, 2020.

[18] C. Zeidler and M. R. Asghar, "Authstore: Password-based authentication and encrypted data storage in untrusted environments," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pp. 996–1001, IEEE, 2018.

[19] E. Erdem and M. T. Sandıkkaya, "Otpaas—one time password as a service," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 3, pp. 743–756, 2018.

[20] D. Chandramohan, T. Vengattaraman, D. Rajaguru, R. Baskaran, and P. Dhavachelvan, "Emppc-an evolutionary model based privacy preserving technique for cloud digital data storage," in *2013 3rd IEEE International Advance Computing Conference (IACC)*, pp. 89–95, IEEE, 2013.

[21] A. N. Rukavitsyn, K. A. Borisenko, I. I. Holod, and A. V. Shorov, "The method of ensuring confidentiality and integrity data in cloud computing," in *2017 XX IEEE International Conference on Soft Computing and Measurements (SCM)*, pp. 272–274, IEEE, 2017.

[22] Y. Chen, L. Li, and Z. Chen, "An approach to verifying data integrity for cloud storage," in *2017 13th International Conference on Computational Intelligence and Security (CIS)*, pp. 582–585, IEEE, 2017.

[23] S. Alneyadi, E. Sithirasenan, and V. Muthukkumarasamy, "A survey on data leakage prevention systems," *Journal of Network and Computer Applications*, vol. 62, pp. 137–152.

[24] C. B. Guevara Maldonado, "Data leakage detection using dynamic data structure and classification techniques," 2015.

[25] N. Rakotondravony, B. Taubmann, W. Mandarawi, E. Weishäupl, P. Xu, B. Kolosnjaji, M. Protsenko, H. De Meer, and H. P. Reiser, "Classifying malware attacks in iaas cloud environments," *Journal of Cloud Computing*, vol. 6, no. 1, pp. 1–12, 2017.

[26] D. Perez-Botero, J. Szefer, and R. B. Lee, "Characterizing hypervisor vulnerabilities in cloud computing servers," in *Proceedings of the 2013 international workshop on Security in cloud computing*, pp. 3–10, ACM, 2013.

[27] C. Tunc, S. Hariri, M. Merzouki, C. Mahmoudi, F. J. De Vaulx, J. Chbili, R. Bohn, and A. Battou, "Cloud security automation framework," in *2017 IEEE 2nd International Workshops on Foundations and Applications of Self Systems*, pp. 307–312, IEEE, 2017.

[28] K. Maithili, V. Vinothkumar, and P. Latha, "Analyzing the security mechanisms to prevent unauthorized access in cloud and network security," *Journal of Computational and Theoretical Nanoscience*, vol. 15, no. 6-7, pp. 2059–2063, 2018.

[29] T. S. Somasundaram, V. Prabha, and M. Arumugam, "Scalability issues in cloud computing," in *2012 Fourth International Conference on Advanced Computing (ICoAC)*, pp. 1–5, IEEE, 2012.

[30] A. Yousafzai, A. Gani, R. M. Noor, M. Sookhak, H. Talebian, M. Shiraz, and M. K. Khan, "Cloud resource allocation schemes: review, taxonomy, and opportunities," *Knowledge and Information Systems*, vol. 50, no. 2, pp. 347–381, 2017.

[31] A. Mahajan and S. Sharma, "The malicious insiders threat in the cloud," *International Journal of Engineering Research and General Science*, vol. 3, no. 2, pp. 245–256, 2015.

[32] X. Liao, S. Alrwais, K. Yuan, L. Xing, X. Wang, S. Hao, and R. Beyah, "Cloud repository as a malicious service: challenge, identification and implication," *Cybersecurity*, vol. 1, no. 1, pp. 1–18, 2018.

[33] A. Singh and K. Chatterjee, "Cloud security issues and challenges: A survey," *Journal of Network and Computer Applications*, vol. 79, pp. 88–115, 2017.

[34] F. Zafar, A. Khan, S. U. R. Malik, M. Ahmed, A. Anjum, M. I. Khan, N. Javed, M. Alam, and F. Jamil, "A survey of cloud computing data integrity schemes: Design challenges, taxonomy and future trends," *Computers & Security*, vol. 65, pp. 29–49, 2017.

[35] E. Daniel, S. Durga, and S. Seetha, "Panoramic view of cloud storage security attacks: an insight and security approaches," in *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, pp. 1029–1034, IEEE, 2019.

[36] Q. Yan, F. R. Yu, Q. Gong, and J. Li, "Software-defined networking (sdn) and distributed denial of service (ddos) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE communications surveys & tutorials*, vol. 18, no. 1, pp. 602–622, 2015.

[37] B. K. Devi and T. Subbulakshmi, "Ddos attack detection and mitigation techniques in cloud computing environment," in *2017 International Conference on Intelligent Sustainable Systems (ICISS)*, pp. 512–517, IEEE, 2017.

[38] Z. M. Yusop and J. Abawajy, "Analysis of insiders attack mitigation strategies," *Procedia-Social and Behavioral Sciences*, vol. 129, pp. 581–591, 2014.

[39] M. A. Khan, "A survey of security issues for cloud computing," *Journal of network and computer applications*, vol. 71, pp. 11–29, 2016.

[40] H. Song, J. Li, and H. Li, "A cloud secure storage mechanism based on data dispersion and encryption," *IEEE Access*, vol. 9, pp. 63745–63751, 2021.

[41] Y. Zhang, J. Yu, R. Hao, C. Wang, and K. Ren, "Enabling efficient user revocation in identity-based cloud storage auditing for shared big data," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 3, pp. 608–619, 2020.

[42] C. Zuo, J. Shao, J. K. Liu, G. Wei, and Y. Ling, "Fine-grained two-factor protection mechanism for data sharing in cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 1, pp. 186–196, 2018.

[43] H. Cui, R. H. Deng, Y. Li, and G. Wu, "Attribute-based storage supporting secure deduplication of encrypted data in cloud," *IEEE Transactions on Big Data*, vol. 5, no. 3, pp. 330–342, 2019.

[44] S. Sun, H. Ma, Z. Song, and R. Zhang, "Webcloud: Web-based cloud storage for secure data sharing across platforms," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 3, pp. 1871–1884, 2022.

[45] K. Cheng, L. Wang, Y. Shen, H. Wang, Y. Wang, X. Jiang, and H. Zhong, "Secure kk-nn query on encrypted cloud data with multiple keys," *IEEE Transactions on Big Data*, vol. 7, no. 4, pp. 689–702, 2021.

[46] Y. Dong, L. Sun, D. Liu, M. Feng, and T. Miao, "A survey on data integrity checking in cloud," in *2018 1st International Cognitive Cities Conference (IC3)*, pp. 109–113, IEEE, 2018.

[47] G. Bian, Y. Fu, B. Shao, and F. Zhang, "Data integrity audit based on data blinding for cloud and fog environment," *IEEE Access*, vol. 10, pp. 39743–39751, 2022.

[48] A. Iqbal and H. Saham, "Data integrity issues in cloud servers," *International Journal of Computer Science Issues (IJCSI)*, vol. 11, no. 3, p. 118, 2014.

[49] G. Caronni and M. Waldvogel, "Establishing trust in distributed storage providers," in *Proceedings Third International Conference on Peer-to-Peer Computing (P2P2003)*, pp. 128–133, IEEE, 2003.

[50] S. OGISO, M. MOHRI, and Y. SHIRAISHI, "Transparent provable data possession scheme for cloud storage," in *2020 International Symposium on Networks, Computers and Communications (ISNCC)*, pp. 1–5, IEEE, 2020.

[51] L. González-Manzano and A. Orfila, "An efficient confidentiality-preserving proof of ownership for deduplication," *Journal of Network and Computer Applications*, vol. 50, pp. 49–59, 2015.

[52] C.-M. Yu, C.-Y. Chen, and H.-C. Chao, "Proof of ownership in deduplicated cloud storage with mobile device efficiency," *IEEE network*, vol. 29, no. 2, pp. 51–55, 2015.

[53] R. Di Pietro and A. Sorniotti, "Boosting efficiency and security in proof of ownership for deduplication," in *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, pp. 81–82, ACM, 2012.

[54] R. Masood, N. Pandey, and Q. Rana, "Dht-pdp: A distributed hash table based provable data possession mechanism in cloud storage," in *2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO)*, pp. 275–279, IEEE, 2020.

[55] G. Bian and J. Chang, "Certificateless provable data possession protocol for the multiple copies and clouds case," *IEEE Access*, vol. 8, pp. 102958–102970, 2020.

[56] X. Zhang, X. Wang, D. Gu, J. Xue, and W. Tang, "Conditional anonymous certificateless public auditing scheme supporting data dynamics for cloud storage systems," *IEEE Transactions on Network and Service Management*, vol. 19, no. 4, pp. 5333–5347, 2022.

[57] J. Li, H. Yan, and Y. Zhang, "Certificateless public integrity checking of group shared data on cloud storage," *IEEE Transactions on Services Computing*, vol. 14, no. 1, pp. 71–81, 2021.

[58] A. Juels and B. S. Kaliski Jr, "Pors: Proofs of retrievability for large files," in *Proceedings of the 14th ACM conference on Computer and communications security*, pp. 584–597, ACM, 2007.

[59] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pp. 169–178, ACM, 2009.

[60] S. Y. Enoch, J. B. Hong, and D. S. Kim, "Time independent security analysis for dynamic networks using graphical security models," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pp. 588–595, IEEE, 2018.

[61] S. Kumar, S. K. Singh, A. K. Singh, S. Tiwari, and R. S. Singh, "Privacy preserving security using biometrics in cloud computing," *Multimedia Tools and Applications*, vol. 77, no. 9, pp. 11017–11039, 2018.

[62] P. Sirohi and A. Agarwal, "Cloud computing data storage security framework relating to data integrity, privacy and trust," in *2015 1st international conference on next generation computing technologies (NGCT)*, pp. 115–118, IEEE, 2015.

[63] D. Prasad, B. R. Singh, M. Akuthota, and M. Sangeetha, "An etiquette approach for public audit and preserve data at cloud," *International Journal of Computer Trends and Technology (IJCTT) volume*, vol. 16, 2014.

[64] B. Skibitzki, "How zebra technologies manages security & risk using security command center," 2021.

[65] C. B. Tan, M. H. A. Hijazi, Y. Lim, and A. Gani, "A survey on proof of retrievability for cloud data integrity and availability: Cloud storage state-of-the-art, issues, solutions and future trends," *Journal of Network and Computer Applications*, vol. 110, pp. 75–86, 2018.

[66] S. R. Pujar, S. S. Chaudhari, and R. Aparna, "Survey on data integrity and verification for cloud storage," in *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1–7, IEEE, 2020.

[67] B. Wang, B. Li, and H. Li, "Oruta: Privacy-preserving public auditing for shared data in the cloud," *IEEE transactions on cloud computing*, vol. 2, no. 1, pp. 43–56, 2014.

[68] T. Indhumathil, N. Aarthy, V. D. Devi, and V. Samyuktha, "Third-party auditing for cloud service providers in multicloud environment," in *2017 Third International Conference on Science Technology Engineering & Management (ICONSTEM)*, pp. 347–352, IEEE, 2017.

[69] S. Mohanty, P. K. Pattnaik, and R. Kumar, "Confidentiality preserving auditing for cloud computing environment," in *2018 International Conference on Research in Intelligent and Computing in Engineering (RICE)*, pp. 1–4, IEEE, 2018.

[70] T. Subha and S. Jayashri, "Efficient privacy preserving integrity checking model for cloud data storage security," in *2016 Eighth International Conference on Advanced Computing (ICoAC)*, pp. 55–60, IEEE, 2017.

[71] S. Hiremath and S. Kunte, "A novel data auditing approach to achieve data privacy and data integrity in cloud computing," in *2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, pp. 306–310, IEEE, 2017.

[72] Y. Zhang, C. Xu, H. Li, and X. Liang, "Cryptographic public verification of data integrity for cloud storage systems," *IEEE Cloud Computing*, vol. 3, no. 5, pp. 44–52, 2016.

[73] M. Thangavel and P. Varalakshmi, "Enabling ternary hash tree based integrity verification for secure cloud data storage," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 12, pp. 2351–2362, 2019.

[74] W. Shen, J. Qin, J. Yu, R. Hao, and J. Hu, "Enabling identity-based integrity auditing and data sharing with sensitive information hiding for secure cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 2, pp. 331–346, 2018.

[75] P. Singh and S. K. Saroj, "A secure data dynamics and public auditing scheme for cloud storage," in *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pp. 695–700, IEEE, 2020.

[76] J. Ni, Y. Yu, Y. Mu, and Q. Xia, "On the security of an efficient dynamic auditing protocol in cloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 10, pp. 2760–2761, 2013.

[77] B. Shao, G. Bian, Y. Wang, S. Su, and C. Guo, "Dynamic data integrity auditing method supporting privacy protection in vehicular cloud environment," *IEEE Access*, vol. 6, pp. 43785–43797, 2018.

[78] J. Shen, D. Liu, D. He, X. Huang, and Y. Xiang, "Algebraic signatures-based data integrity auditing for efficient data dynamics in cloud computing," *IEEE Transactions on Sustainable Computing*, vol. 5, no. 2, pp. 161–173, 2017.

[79] B. Wang, H. Li, X. Liu, F. Li, and X. Li, "Efficient public verification on the integrity of multi-owner data in the cloud," *Journal of Communications and Networks*, vol. 16, no. 6, pp. 592–599, 2014.

[80] Y. Yu, Y. Li, B. Yang, W. Susilo, G. Yang, and J. Bai, "Attribute-based cloud data integrity auditing for secure outsourced storage," *IEEE Transactions on Emerging Topics in Computing*, vol. 8, no. 2, pp. 377–390, 2017.

[81] Y. Zhang, J. Yu, R. Hao, C. Wang, and K. Ren, "Enabling efficient user revocation in identity-based cloud storage auditing for shared big data," *IEEE Transactions on Dependable and Secure computing*, vol. 17, no. 3, pp. 608–619, 2018.

[82] H. Zhu, Y. Yuan, Y. Chen, Y. Zha, W. Xi, B. Jia, and Y. Xin, "A secure and efficient data integrity verification scheme for cloud-iot based on short signature," *IEEE Access*, vol. 7, pp. 90036–90044, 2019.

[83] M. Sookhak, A. Gani, H. Talebian, A. Akhunzada, S. U. Khan, R. Buyya, and A. Y. Zomaya, "Remote data auditing in cloud computing environments: a survey, taxonomy, and open issues," *ACM Computing Surveys (CSUR)*, vol. 47, no. 4, pp. 1–34, 2015.

[84] H. Wang, D. He, and S. Tang, "Identity-based proxy-oriented data uploading and remote data integrity checking in public cloud," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 6, pp. 1165–1176, 2016.

[85] A. S. Thakur and P. Gupta, "Framework to improve data integrity in multi cloud environment," 2014.

[86] C. Zhang, Y. Xu, Y. Hu, J. Wu, J. Ren, and Y. Zhang, "A blockchain-based multi-cloud storage data auditing scheme to locate faults," *IEEE Transactions on Cloud Computing*, 2021.

[87] T. Subha and S. Jayashri, "Data integrity verification in hybrid cloud using ttpa," in *Networks and communications (NetCom2013)*, pp. 149–159, Springer, 2014.

[88] J. Mao, Y. Zhang, P. Li, T. Li, Q. Wu, and J. Liu, "A position-aware merkle tree for dynamic cloud data integrity verification," *Soft Computing*, vol. 21, no. 8, pp. 2151–2164, 2017.

[89] S. Han, S. Liu, K. Chen, and D. Gu, "Proofs of retrievability based on mrd codes," in *International Conference on Information Security Practice and Experience*, pp. 330–345, Springer, 2014.

[90] N. Kaaniche, E. El Moustaine, and M. Laurent, "A novel zero-knowledge scheme for proof of data possession in cloud storage applications," in *2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pp. 522–531, IEEE, 2014.

[91] W. I. Khedr, H. M. Khater, and E. R. Mohamed, "Cryptographic accumulator-based scheme for critical data integrity verification in cloud storage," *IEEE Access*, vol. 7, pp. 65635–65651, 2019.

[92] T. S. Khatri and G. Jethava, "Improving dynamic data integrity verification in cloud computing," in *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, pp. 1–6, IEEE, 2013.

[93] H. Wang, "Proxy provable data possession in public clouds," *IEEE Transactions on Services Computing*, vol. 6, no. 4, pp. 551–559, 2012.

[94] F. Apolinário, M. Pardal, and M. Correia, "S-audit: Efficient data integrity verification for cloud storage," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing and Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pp. 465–474, IEEE, 2018.

[95] Y. Li, A. Fu, Y. Yu, and G. Zhang, "Ipor: An efficient ida-based proof of retrievability scheme for cloud storage systems," in *2017 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2017.

[96] H. Shacham and B. Waters, "Compact proofs of retrievability," in *International conference on the theory and application of cryptology and information security*, pp. 90–107, Springer, 2008.

[97] C. C. Erway, A. Küpçü, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," *ACM Transactions on Information and System Security (TISSEC)*, vol. 17, no. 4, pp. 1–29, 2015.

[98] C. Liu, J. Chen, L. T. Yang, X. Zhang, C. Yang, R. Ranjan, and R. Kotagiri, "Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 9, pp. 2234–2244, 2013.

[99] D. He, N. Kumar, H. Wang, L. Wang, and K.-K. R. Choo, "Privacy-preserving certificateless provable data possession scheme for big data storage on cloud," *Applied Mathematics and Computation*, vol. 314, pp. 31–43, 2017.

[100] B. Wang, B. Li, H. Li, and F. Li, "Certificateless public auditing for data integrity in the cloud," in *2013 IEEE conference on communications and network security (CNS)*, pp. 136–144, IEEE, 2013.

[101] A. Fu, Y. Li, S. Yu, Y. Yu, and G. Zhang, "Dipor: An ida-based dynamic proof of retrievability scheme for cloud storage systems," *Journal of Network and Computer Applications*, vol. 104, pp. 97–106, 2018.

[102] J. Xu and E.-C. Chang, "Towards efficient proofs of retrievability," in *Proceedings of the 7th ACM symposium on information, computer and communications security*, pp. 79–80, 2012.

[103] Y. Lu and F. Hu, "Secure dynamic big graph data: Scalable, low-cost remote data integrity checking," *IEEE Access*, vol. 7, pp. 12888–12900, 2019.

[104] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in *Proceedings of the 4th international conference on Security and privacy in communication netowrks*, pp. 1–10, ACM, 2008.

[105] 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, 2015.

[106] D. He, S. Zeadally, and L. Wu, "Certificateless public auditing scheme for cloud-assisted wireless body area networks," *IEEE Systems Journal*, vol. 12, no. 1, pp. 64–73, 2015.

[107] M. S. Yoosuf and R. Anitha, "Lduap: lightweight dual auditing protocol to verify data integrity in cloud storage servers," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–19, 2021.

[108] Y. Lu and F. Hu, "Secure dynamic big graph data: Scalable, low-cost remote data integrity checking," *IEEE Access*, vol. 7, pp. 12888–12900, 2019.

[109] H. Tian, F. Nan, C.-C. Chang, Y. Huang, J. Lu, and Y. Du, "Privacy-preserving public auditing for secure data storage in fog-to-cloud computing," *Journal of Network and Computer Applications*, vol. 127, pp. 59–69, 2019.

[110] A. P. Singh and S. K. Pasupuleti, "Optimized public auditing and data dynamics for data storage security in cloud computing," *Procedia Computer Science*, vol. 93, pp. 751–759, 2016.

[111] C. Wang, S. S. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for secure cloud storage," *IEEE transactions on computers*, vol. 62, no. 2, pp. 362–375, 2011.

[112] Y. Zhang, C. Xu, X. Lin, and X. S. Shen, "Blockchain-based public integrity verification for cloud storage against procrastinating auditors," *IEEE Transactions on Cloud Computing*, 2019.

[113] A. P. Singh and S. K. Pasupuleti, "Optimized public auditing and data dynamics for data storage security in cloud computing," *Procedia Computer Science*, vol. 93, pp. 751–759, 2016.

[114] Z. Xu, L. Wu, M. K. Khan, K.-K. R. Choo, and D. He, "A secure and efficient public auditing scheme using rsa algorithm for cloud storage," *The Journal of Supercomputing*, vol. 73, no. 12, pp. 5285–5309, 2017.

[115] 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.

[116] J. Xu and E.-C. Chang, "Towards efficient proofs of retrievability," in *Proceedings of the 7th ACM symposium on information, computer and communications security*, pp. 79–80, ASIA-CCS, 2012.

[117] H. Shacham and B. Waters, "Compact proofs of retrievability," in *International conference on the theory and application of cryptology and information security*, pp. 90–107, Springer, 2008.

[118] A. Mohammed and D. Vasumathi, "Locality parameters for privacy preserving protocol and detection of malicious third-party auditors in cloud computing," in *International Conference on Intelligent Computing and Communication*, pp. 67–76, Springer, 2019.

[119] A. Majumdar, S. S. Roy, and R. Dasgupta, "Job migration policy in a structured cloud framework," in *2017 International Conference on Computational Science and Computational Intelligence (CSCI)*, pp. 1529–1534, IEEE, 2017.

[120] M. Carroll, A. Van Der Merwe, and P. Kotze, "Secure cloud computing: Benefits, risks and controls," in *2011 Information Security for South Africa*, pp. 1–9, IEEE, 2011.

[121] Q. Zhang, S. Wang, D. Zhang, J. Wang, and Y. Zhang, "Time and attribute based dual access control and data integrity verifiable scheme in cloud computing applications," *IEEE Access*, vol. 7, pp. 137594–137607, 2019.

[122] K. M. Khan and M. Shaheen, "Data obfuscation for privacy and confidentiality in cloud computing," in *2015 IEEE International Conference on Software Quality, Reliability and Security-Companion*, pp. 195–196, IEEE, 2015.

[123] S. Kaushik and C. Gandhi, "Multi-level trust agreement in cloud environment," in *2019 Twelfth International Conference on Contemporary Computing (IC3)*, pp. 1–5, IEEE, 2019.

[124] D. Gonzales, J. M. Kaplan, E. Saltzman, Z. Winkelman, and D. Woods, "Cloud-trust—a security assessment model for infrastructure as a service (iaas) clouds," *IEEE Transactions on Cloud Computing*, vol. 5, no. 3, pp. 523–536, 2015.

[125] Y. Yang, R. Liu, Y. Chen, T. Li, and Y. Tang, "Normal cloud model-based algorithm for multi-attribute trusted cloud service selection," *IEEE Access*, vol. 6, pp. 37644–37652, 2018.

[126] V. C. Emeakaroha, K. Fatema, L. van der Werff, P. Healy, T. Lynn, and J. P. Morrison, "A trust label system for communicating trust in cloud services," *IEEE Transactions on Services Computing*, vol. 10, no. 5, pp. 689–700, 2016.

[127] V. Anuradha and D. Sumathi, "A survey on resource allocation strategies in cloud computing," in *International Conference on Information Communication and Embedded Systems (ICICES2014)*, pp. 1–7, IEEE, 2014.

[128] N. Dimitri, "Pricing cloud iaas computing services," *Journal of Cloud Computing*, vol. 9, no. 1, pp. 1–11, 2020.

[129] L. Klosterboer, *ITIL capacity management*. Pearson Education, 2011.

[130] N. C. Luong, P. Wang, D. Niyato, Y. Wen, and Z. Han, "Resource management in cloud networking using economic analysis and pricing models: A survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 954–1001, 2017.

[131] M. Rouse, "What is capacity planning? - definition fromwhatis.com," 2016.

[132] P. Goswami, S. S. Roy, and R. Dasgupta, "Design of an architectural framework for providing quality cloud services," in *International Conference on Grid, Cloud, & Cluster Computing*, pp. 17–23, American Council on Science and Education (ACSE), 2017.

[133] C. Wu, R. Buyya, and K. Ramamohanarao, "Cloud pricing models: Taxonomy, survey, and interdisciplinary challenges," *ACM Computing Surveys (CSUR)*, vol. 52, no. 6, pp. 1–36, 2019.

[134] S. Wu and Y. Zhang, "Efficient verification of data possession in cloud computing," in *2016 4th International Conference on Cloud Computing and Intelligence Systems (CCIS)*, pp. 424–428, IEEE, 2016.

[135] H. Zhang, F. Lou, H. Wang, and Z. Tian, "Research on data protection based on encrypted attribute access control in cloud computing," in *2018 5th International Conference on Information Science and Control Engineering (ICISCE)*, pp. 450–453, IEEE, 2018.

[136] A. Vineela, N. Kasiviswanath, and C. S. Bindu, "Data integrity auditing scheme for preserving security in cloud based big data," in *2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 609–613, IEEE, 2022.

[137] Y. Wang, Z. Chen, K. Wang, and Z. Yang, "Education cloud data integrity verification based on mapping-trie tree," in *2019 International Conference on Machine Learning, Big Data and Business Intelligence (MLBDBI)*, pp. 155–158, IEEE, 2019.

[138] T. Zhao, Z.-y. Zhang, and L.-l. Zhang, "A novel cloud data integrity verification scheme based on dynamic array multi-branch tree," in *2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP)*, pp. 377–380, IEEE, 2021.

[139] S. A. Ali, Y. Justindhas, M. Lakshmanan, P. H. Kumar, and P. Baskaran, "Secured cloud data outsourcing model using two party integrity scheme," in *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pp. 509–512, IEEE, 2022.

[140] W. Shen, J. Qin, J. Yu, R. Hao, J. Hu, and J. Ma, "Data integrity auditing without private key storage for secure cloud storage," *IEEE Transactions on Cloud Computing*, vol. 9, no. 4, pp. 1408–1421, 2019.

[141] H. Duan, Y. Du, L. Zheng, C. Wang, M. H. Au, and Q. Wang, "Towards practical auditing of dynamic data in decentralized storage," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 1, pp. 708–723, 2023.

[142] A. Sasikumar, L. Ravi, K. Kotecha, A. Abraham, M. Devarajan, and S. Vairavasundaram, "A secure big data storage framework based on blockchain consensus mechanism with flexible finality," *IEEE Access*, vol. 11, pp. 56712–56725, 2023.

[143] B. A. Jalil, T. M. Hasan, G. S. Mahmood, and H. N. Abed, "A secure and efficient public auditing system of cloud storage based on bls signature and automatic blocker protocol," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 7, pp. 4008–4021, 2022.

[144] R. Li, H. Yang, X. A. Wang, Z. Yi, and K. Niu, "Improved public auditing system of cloud storage based on bls signature," *Security and Communication Networks*, vol. 2022, 2022.

[145] H. Yan and W. Gui, "Efficient identity-based public integrity auditing of shared data in cloud storage with user privacy preserving," *IEEE Access*, vol. 9, pp. 45822–45831, 2021.

[146] X.-P. Zhao and R. Jiang, "Distributed machine learning oriented data integrity verification scheme in cloud computing environment," *IEEE Access*, vol. 8, pp. 26372–26384, 2020.

[147] Y. Yu, M. H. Au, G. Ateniese, X. Huang, W. Susilo, Y. Dai, and G. Min, "Identity-based remote data integrity checking with perfect data privacy preserving for cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 4, pp. 767–778, 2016.

[148] Y. Li, Y. Yu, B. Yang, G. Min, and H. Wu, "Privacy preserving cloud data auditing with efficient key update," *Future Generation Computer Systems*, vol. 78, pp. 789–798, 2018.

[149] N. Garg, S. Bawa, and N. Kumar, "An efficient data integrity auditing protocol for cloud computing," *Future Generation Computer Systems*, vol. 109, pp. 306–316, 2020.

[150] W. Shen, J. Qin, J. Yu, R. Hao, J. Hu, and J. Ma, "Data integrity auditing without private key storage for secure cloud storage," *IEEE Transactions on Cloud Computing*, vol. 9, no. 4, pp. 1408–1421, 2021.

[151] A. Li, Y. Chen, Z. Yan, X. Zhou, and S. Shimizu, "A survey on integrity auditing for data storage in the cloud: from single copy to multiple replicas," *IEEE Transactions on Big Data*, 2020.

[152] C. Zhang, Y. Xu, Y. Hu, J. Wu, J. Ren, and Y. Zhang, "A blockchain-based multi-cloud storage data auditing scheme to locate faults," *IEEE Transactions on Cloud Computing*, vol. 10, no. 4, pp. 2252–2263, 2022.

[153] Y. Miao, Q. Huang, M. Xiao, and H. Li, "Decentralized and privacy-preserving public auditing for cloud storage based on blockchain," *IEEE Access*, vol. 8, pp. 139813–139826, 2020.

[154] H. Cui, Z. Wan, X. Wei, S. Nepal, and X. Yi, "Pay as you decrypt: Decryption outsourcing for functional encryption using blockchain," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3227–3238, 2020.

[155] P. Wei, D. Wang, Y. Zhao, S. K. S. Tyagi, and N. Kumar, "Blockchain data-based cloud data integrity protection mechanism," *Future Generation Computer Systems*, vol. 102, pp. 902–911, 2020.

[156] E. B. Sifah, Q. Xia, K. O.-B. O. Agyekum, H. Xia, A. Smahi, and J. Gao, "A blockchain approach to ensuring provenance to outsourced cloud data in a sharing ecosystem," *IEEE Systems Journal*, 2021.

[157] P. Huang, K. Fan, H. Yang, K. Zhang, H. Li, and Y. Yang, "A collaborative auditing blockchain for trustworthy data integrity in cloud storage system," *IEEE Access*, vol. 8, pp. 94780–94794, 2020.

[158] R. Pise and S. Patil, "Enhancing security of data in cloud storage using decentralized blockchain," in *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, pp. 161–167, IEEE, 2021.

[159] P. Sharma, R. Jindal, and M. D. Borah, "Blockchain-based integrity protection system for cloud storage," in *2019 4th Technology Innovation Management and Engineering Science International Conference (TIMES-iCON)*, pp. 1–5, IEEE, 2019.

[160] S. Singhal and A. Sharma, "A job scheduling algorithm based on rock hyrax optimization in cloud computing," *Computing*, vol. 103, no. 9, pp. 2115–2142, 2021.

[161] S. Sciberras and J. G. Vella, "Investigating movement detection in unedited camera footage," in *Advances in Computing and Data Sciences: 4th International Conference, ICACDS 2020, Valletta, Malta, April 24–25, 2020, Revised Selected Papers 4*, pp. 362–371, Springer, 2020.

[162] N. Garg, S. Bawa, and N. Kumar, "An efficient data integrity auditing protocol for cloud computing," *Future Generation Computer Systems*, vol. 109, pp. 306–316, 2020.

[163] S. Gokulakrishnan and J. Gnanasekar, "Data integrity and recovery management in cloud systems," in *2020 Fourth International Conference on Inventive Systems and Control (ICISC)*, pp. 645–648, IEEE, 2020.

[164] L. Rao, H. Zhang, and T. Tu, "Dynamic outsourced auditing services for cloud storage based on batch-leaves-authenticated merkle hash tree," *IEEE Transactions on Services Computing*, vol. 13, no. 3, pp. 451–463, 2017.

[165] H. Yu, X. Lu, and Z. Pan, "An authorized public auditing scheme for dynamic big data storage in cloud computing," *IEEE Access*, vol. 8, pp. 151465–151473, 2020.

[166] S. Debnath, M. V. Nunsanga, and B. Bhuyan, "Study and scope of signcryption for cloud data access control," in *Advances in Computer, Communication and Control: Proceedings of ETES 2018*, pp. 113–126, Springer, 2019.

[167] Y. Zhu, G.-J. Ahn, H. Hu, S. S. Yau, H. G. An, and C.-J. Hu, "Dynamic audit services for outsourced storages in clouds," *IEEE transactions on services computing*, vol. 6, no. 2, pp. 227–238, 2011.

[168] J. Zhang, R. Lu, B. Wang, and X. A. Wang, "Comments on "privacy-preserving public auditing protocol for regenerating-code-based cloud storage"," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1288–1289, 2020.

[169] W. Shen, J. Qin, J. Yu, R. Hao, and J. Hu, "Enabling identity-based integrity auditing and data sharing with sensitive information hiding for secure cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 2, pp. 331–346, 2018.

[170] S. S. W. Q. R. K. L. W. Wang, C., "Privacy-preserving public auditing for secure cloud storage," *IEEE Transactions on Computers*, vol. 62, no. 2, pp. 7362–375, 2013.

[171] S. Singhal and A. Sharma, "Mutative bfo-based scheduling algorithm for cloud environment," in *Proceedings of International Conference on Communication and Artificial Intelligence*, pp. 589–599, Springer, 2021.

[172] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," *IEEE Internet computing*, vol. 16, no. 1, pp. 69–73, 2012.

[173] S. Singhal and A. Sharma, "Mutative aco based load balancing in cloud computing.," *Engineering Letters*, vol. 29, no. 4, 2021.

[174] M. Tian, Y. Zhang, Y. Zhu, L. Wang, and Y. Xiang, "Divrs: Data integrity verification based on ring signature in cloud storage," *Computers & Security*, p. 103002, 2022.

[175] S. Sushma, P. Srilatha, and G. Srinivas, "Efficient integrity checking and secured data sharing in cloud," in *2022 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)*, pp. 1–4, IEEE, 2022.

[176] P. Mell and T. Grance, "Draft nist working definition of cloud computing,‖ referenced on june. 3rd, 2009 online at http://csrc. nist. gov/groups," *SNS/cloud-computing/index. html*, 2009.

[177] P. Goswami, N. Faujdar, S. Debnath, A. K. Khan, and G. Singh, "Zss signature-based audit message verification process for cloud data integrity," *IEEE Access*, 2023.

[178] M. Bellare and S. Duan, "Partial signatures and their applications," *Cryptology ePrint Archive*, 2009.

[179] S. K. Pasupuleti, "Privacy-preserving public auditing and data dynamics for secure cloud storage based on exact regenerated code," in *Research Anthology on Privatizing and Securing Data*, pp. 1003–1022, IGI Global, 2021.

[180] S. Debnath, B. Bhuyan, and A. K. Saha, "Privacy preserved secured outsourced cloud data access control scheme with efficient multi-authority attribute based signcryption," *Multiagent and Grid Systems*, vol. 16, no. 4, pp. 409–432, 2020.

[181] A. Bhalerao and A. Pawar, "A survey: On data deduplication for efficiently utilizing cloud storage for big data backups," in *2017 international conference on trends in electronics and informatics (ICEI)*, pp. 933–938, IEEE, 2017.

[182] E. N. Witanto and S.-G. Lee, "Cloud storage data verification using signcryption scheme," *Applied Sciences*, vol. 12, no. 17, p. 8602, 2022.

[183] Y. Bai, Z. Zhou, X. Luo, X. Wang, F. Liu, and Y. Xu, "A cloud data integrity verification scheme based on blockchain," in *2021 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/IOP/SCI)*, pp. 357–363, IEEE, 2021.

[184] F. Benhamouda, G. Couteau, D. Pointcheval, and H. Wee, "Implicit zero-knowledge arguments and applications to the malicious setting," in *Annual Cryptology Conference*, pp. 107–129, Springer, 2015.

[185] S. S. Roy, C. Garai, and R. Dasgupta, "Performance analysis of parallel cbar in mapreduce environment," in *2015 International Conference on Computing, Communication and Security (ICCCS)*, pp. 1–7, IEEE, 2015.

[186] K. Akshay and B. Muniyal, "Dynamic list based data integrity verification in cloud environment," *Journal of Cyber Security and Mobility*, pp. 433–460, 2022.

[187] S. Debnath and B. Bhuyan, "Efficient and scalable outsourced data access control with user revocation in cloud: a comprehensive study," *Multiagent and Grid Systems*, vol. 14, no. 4, pp. 383–401, 2018.

[188] H. Yan, J. Li, and Y. Zhang, "Remote data checking with a designated verifier in cloud storage," *IEEE Systems Journal*, vol. 14, no. 2, pp. 1788–1797, 2020.

[189] K. Yang and X. Jia, "An efficient and secure dynamic auditing protocol for data storage in cloud computing," *IEEE transactions on parallel and distributed systems*, vol. 24, no. 9, pp. 1717–1726, 2012.

[190] D. Magalhaes, R. N. Calheiros, R. Buyya, and D. G. Gomes, "Workload modeling for resource usage analysis and simulation in cloud computing," *Computers & Electrical Engineering*, vol. 47, pp. 69–81, 2015.

[191] J. Li, H. Yan, and Y. Zhang, "Identity-based privacy preserving remote data integrity checking for cloud storage," *IEEE Systems Journal*, vol. 15, no. 1, pp. 577–585, 2021.

[192] J. Chang, B. Shao, Y. Ji, and G. Bian, "Efficient identity-based provable multi-copy data possession in multi-cloud storage, revisited," *IEEE Communications Letters*, vol. 24, no. 12, pp. 2723–2727, 2020.

[193] Y. Deswarte, J.-J. Quisquater, and A. Saïdane, "Remote integrity checking," in *Working conference on integrity and internal control in information systems*, pp. 1–11, Springer, 2003.

[194] J.-F. Raymond and A. Stiglic, "Security issues in the diffie-hellman key agreement protocol," *IEEE Transactions on Information Theory*, vol. 22, pp. 1–17, 2000.

[195] H. Wu, A. D. Dwivedi, and G. Srivastava, "Security and privacy of patient information in medical systems based on blockchain technology," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 17, no. 2s, pp. 1–17, 2021.

[196] T. Wang, J. Zhou, X. Chen, G. Wang, A. Liu, and Y. Liu, "A three-layer privacy preserving cloud storage scheme based on computational intelligence in fog computing," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 3–12, 2018.

[197] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pp. 13–16, 2012.

[198] B. Li, Q. He, F. Chen, H. Jin, Y. Xiang, and Y. Yang, "Auditing cache data integrity in the edge computing environment," *IEEE Transactions on Parallel and Distributed Systems*, 2020.

[199] B. Li, Q. He, F. Chen, H. Jin, Y. Xiang, and Y. Yang, "Inspecting edge data integrity with aggregated signature in distributed edge computing environment," *IEEE Transactions on Cloud Computing*, 2021.

BIO-DATA OF THE CANDIDATE

Name of Candidate	: Paromita Goswami
Date of Birth	: 21/01/1989
Phone	: 9064937690
email	: mcanbu.16@gmail.com,paromita.goswami@gla.ac.in
Permanent Address	: D/o Late Bijan Kumar Goswami P-47,Avas Vikas, Rudrapur, Uttarakhand Pin: 263153
Married	: Yes
Educational Details	
(a) MCA	: University of North Bengal, West Bengal
(b) M.Tech	: NITTTR, Kolkata, West Bengal
(c) Ph.D Course Work	: Mizoram University
Present Occupation	: Assistant Professor, : Department of Computer Engineering & Application
Organization	: GLA University

LIST OF PAPERS/PATENT BASED ON THESIS

Journals:

1. P. Goswami, S. Debnath, A. K. Khan and G. Singh. "ZSS Signature-Based Audit Message Verification Process for Cloud Data Integrity." *IEEE Access,* vol. 11, pp. 145485-145502, 2023, doi: 10.1109/ACCESS.2023.3343841 (SCI, IF-3.9)

2. P. Goswami, S. Debnath, A. K. Khan and G. Singh. "Stub Signature-Based Efficient Public Data Auditing System using Dynamic Procedures in Cloud Computing." *IEEE Access,* vol. 12, pp. 58502-58518,2024, doi: 10.1109/ACCESS.2024.3389076 (SCI, IF- 3.9).

3. P. Goswami, S. Debnath, A. K. Khan and G. Singh. "Investigation on Storage Level Data Integrity Strategies in Cloud Computing: Classification, Security obstructions, Challenges and Vulnerability", in *Journal of Cloud Computing*, vol.13, pp.-45,2024, doi: 10.1186/s13677-024-00605-z (SCI, IF-4.4)

Conference:

1. P. Goswami, V. Sharma, S. Debnath and A. K. Khan. "Acknowledgement Verification of Stored Data in Shared Cloud Resource Pool." In *2023 6th International Conference on Information Systems and Computer Networks (ISCON),* pp. 1-6. IEEE, Mathura, 2023.

2. P. Goswami, M. Gogoi, S. Debnath and A. K. Khan. "Signature-based Batch Auditing Verification in Cloud resource pool." In *International Conference of Advanced Computing, Machine Learning, Robotics and Internet Technologies (pp. 181-193)* Springer Nature Switzerland. 2023.

PARTICULARS OF THE CANDIDATE

NAME OF CANDIDATE	: Paromita Goswami
DEGREE	: Ph.D
DEPARTMENT	: COMPUTER ENGINEERING
TITLE OF THE THESIS	: DESIGN AND ANALYSYS OF EFFICIENT PROTOCOLS FOR DATA SECURITY IN CLOUD COMPUTING
DATE OF ADMISSION	: 06.11.2020
APPROVAL OF RESEARCH PROPOSAL	
1. DRC	:
2. BOS	:
3. SCHOOL BOARD	:
MZU REGISTRATION NO.	: 2006580
PH.D REGISTRATION NO.	: MZU/Ph.D/1655 of 06.11.2020
EXTENSION	: NA

(Dr.V.D.AMBETH KUMAR)

Head

Department of Computer Engineering

ABSTRACT

DESIGN AND ANALYSIS OF EFFICIENT PROTOCOLS FOR DATA SECURITY IN CLOUD COMPUTING

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

PAROMITA GOSWAMI

MZU REGN NO : 2006580
PH. D. REGN NO : MZU/PH.D/1655 OF 06.11.2020

DEPARTMENT OF COMPUTER ENGINEERING
SCHOOL OF ENGINEERING AND TECHNOLOGY
MAY, 2024

ABSTRACT

Cloud computing offers convenient access to a shared pool of configurable computing resources, reducing costs and management efforts for organizations. With the increasing digitization of processes, large amounts of data are generated, requiring storage for processing and future reference. Cloud data outsourcing provides a solution, relieving organizations of the need to set up their storage infrastructure and allowing them to scale storage according to their needs, thus minimizing costs. However, despite the benefits, some organizations are hesitant to outsource data to the cloud due to security concerns, as the security of outsourced data is managed by third-party cloud administrators. This reluctance stems from uncertainties about data privacy, integrity, and the risk of unauthorized access. Consequently, ensuring robust security measures and transparent policies for data handling by cloud service providers is crucial for fostering trust and encouraging wider adoption of cloud data outsourcing among organizations.

In the literature survey, the research challenges and framework are explored, introducing key concepts such as cloud data protection, integrity verification, and data recovery. It highlights the significance of symptoms, verification methods, and security challenges in ensuring cloud data integrity. The chapter provides a thorough review of recent techniques and auditing schemes, offering a comprehensive analysis of the existing literature in this domain.

The existing research papers focus on proposing systematic methodologies for ensuring data integrity verification in cloud computing environments, accompanied by security analyses and assessments of computational or cost overheads for both cloud users and Cloud Service Providers (CSP). If the verification result proves true on the Data owner/ Third Party Auditors(TPA)side, it indicates that the data has not been modified on the CSP side. However, a gap exists in these papers regarding the evaluation of CSP trustworthiness in cases where the verification result is false. While researchers have explored various trust models, they primarily focus on evaluating CSP trustworthiness at the time of service acquisition. There is a lack of consideration for assessing CSP trustworthiness post-verification failure, especially in scenarios where external attackers may have tampered with data during transmission. Addressing this gap would require the development of trust parameters specifically designed to evaluate CSP trustworthiness following verification failures, allowing cloud users to make informed decisions about ongoing trust in their CSPs.While each research work has focused on enhancing specific aspects of cloud services such as data privacy and access control mechanisms, there is a need to also consider the provision of essential services like acknowledgment of stored data in cloud storage to the respective data owners. Like any other service provider, the success of a cloud service provider depends not only on improving these aspects but also on offering a comprehensive and satisfactory set of services. Implementing an acknowledgment service for stored data in cloud storage would ensure that data owners receive confirmation of successful storage, enhancing transparency and trust in the cloud service. In the existing research on cloud frameworks, there is a notable gap regarding the oversight of TPAs during the audit message verification process. While TPAs play a crucial role in verifying data integrity and confidentiality, there is a risk of malicious activity or influence from attackers. This presents a challenge as TPAs may fabricate audit messages or manipulate verification results for personal gain. Additionally, TPAs may access sensitive data or block

positions from the CSP and exploit it. To address this gap, it is imperative to implement robust audit message verification techniques at the data owner's end. This ensures continuous monitoring of TPA activities and maintains the trustworthiness of TPAs. By enhancing audit message verification processes, cloud users can have confidence in the integrity and confidentiality of their data, mitigating the risk of manipulation or unauthorized access.

In the first approach, our main focus was developing a model for a ZSS signature-based efficient data integrity system that protects user privacy when storing data in the cloud. Ultimately, actual performance testing of the suggested system model prototype revealed that the offered model was more efficient than the BLS signature.

In the second approach, we address these difficult problems by offering a novel acknowledgment verification methodology for cloud storage services. This study offered an encryption technique for encrypting original block elements at cloud storage to safeguard data elements against hostile attacks from the outside as well as from within. This method avoided worries about collaboration and stopped assaults on cloud storage that involved forgeries and replacements.

A partial signature-based data auditing system is proposed in the next approach to address security concerns in cloud data storage. This system utilizes cryptographic techniques like homomorphic encryption and hash functions to enhance privacy, and accuracy, and reduce computational costs. It ensures integrity checks on stored files, preventing tampering by external attackers or malicious insiders. Supporting dynamic operations on outsourced data, the system proves effective in real-world applications, as evidenced by simulation outcomes. Assuring data owners of data security and accuracy in shared cloud storage environments aligns with the growing demand for secure cloud solutions. This approach provides a practical means of mitigating security risks associated with cloud data storage while achieving desired security qualities, as demonstrated in security analyses and real-world simulations.

In the last approach, we emphasized a model for an effective data integrity and audit message verification system, utilizing ZSS signatures proposed. This system bolsters data privacy in cloud storage and employs fake data recovery to counteract attackers. Moreover, the model is adapted for a Blockchain-based Medical system, empowering patients to verify their medical records' integrity in shared cloud storage through a TPA.