International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 14, Issue 1, 2026

Efficient Auditing Scheme For Secure Data Storage In Fog-To-Cloud Computing

Samala Nilohitha[1], Mrs M Anusha[2]

[1]B.Tech Student, Department of Electronics and Computer Engineering, J.B. Institute of Engineering and Technology, Hyderabad, India.

[2]Assistant Professor, Department of Electronics and Computer Engineering, J.B. Institute of Engineering and Technology, Hyderabad, India

anusha.ecm@jbiet.edu.in

Abstract

Cloud computing has become the dominant platform for large-scale data storage and service delivery due to its scalability, flexibility, and cost efficiency. However, the growing reliance on third-party cloud service providers raises serious concerns regarding data integrity, privacy, and transparency. To address these issues, third-party auditors (TPAs) are commonly employed to verify the correctness of outsourced data without retrieving the entire content. At the same time, emerging fog-to-cloud computing architectures, driven by the rapid expansion of the Internet of Things (IoT), introduce new challenges because data is no longer managed solely by centralized cloud servers but also by intermediate fog nodes and mobile sinks.

In addition, public digital services such as government welfare schemes require secure and transparent mechanisms that allow citizens to submit applications and track their status without manipulation by intermediaries. This paper presents an integrated study of privacy-preserving cloud data auditing techniques and proposes a secure fog-to-cloud based service framework for public scheme management. The proposed approach enhances data integrity verification, supports user privacy, and improves transparency while reducing computational overhead compared with traditional public auditing methods that rely heavily on expensive cryptographic operations.

Keywords

Cloud storage, third-party auditing, privacy preservation, fog-to-cloud computing, data integrity, public service systems, IoT.

1. Introduction

The widespread adoption of cloud computing has transformed the way organizations and individuals store data and deploy applications. Cloud platforms enable users to access files and services from any location while offering large storage capacity, high scalability, and reduced infrastructure cost. Despite these advantages, data security remains a critical concern. Since data is physically stored and managed by cloud service providers, users lose direct control over their information.

To overcome this limitation, cryptographic auditing techniques are introduced to verify whether the cloud maintains data correctly. In many systems, third-party auditors (TPAs) are used to perform integrity checks on behalf of data owners. A major design requirement of such auditing systems is that the auditor should not learn the actual content of the data.

In parallel, the increasing number of IoT devices has motivated the transition from traditional cloud computing to fog-to-cloud computing, where computation and storage are partially performed at the network edge. Fog nodes and mobile sinks cooperate with cloud servers to provide low-latency services and local processing. However, this multi-layer architecture significantly complicates integrity auditing and trust management.

Furthermore, public digital services, especially government schemes and welfare programs, suffer from lack of transparency and accountability. In many cases, eligible users fail to receive benefits due to administrative inefficiencies or manual processing errors. A secure and transparent digital platform is therefore required to allow citizens to directly apply for schemes and continuously track their application status.

This paper focuses on cloud data auditing techniques with strong privacy protection and extends these concepts to fog-to-cloud environments. It also demonstrates how such auditing mechanisms can support a secure public service application system.

2. Background and Related Work

Early cloud auditing schemes primarily focus on verifying data integrity stored at a single cloud service provider. These methods commonly employ public-key cryptography and challenge–response protocols, allowing a TPA to check correctness without downloading the full dataset.

Several privacy-preserving techniques have been proposed to prevent data leakage to the auditor, such as random masking and homomorphic authentication tags. Although these schemes successfully reduce communication overhead, many of them rely on computationally expensive operations, including bilinear pairings and complex proof constructions.

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026

Recently, fog-to-cloud computing has emerged as a new paradigm to support latency-sensitive and data-intensive IoT applications. Unlike conventional cloud models, fog-to-cloud systems involve multiple participating entities, such as fog nodes, mobile sinks, and centralized cloud servers. Data is often distributed among these layers, making traditional auditing methods insufficient.
Existing public auditing designs for fog-to-cloud storage adopt cloud-oriented cryptographic techniques without considering the limited computing capabilities of fog nodes. As a result, the overall system becomes inefficient and difficult to scale.
At the same time, digital governance platforms are being deployed to deliver public services electronically. However, many of these systems lack strong cryptographic guarantees, especially in terms of integrity, traceability, and tamper resistance.

3. System Model

The proposed framework consists of the following main entities:

1. **User (Citizen or Data Owner):**
   Submits data to the system and applies for public schemes. The user can verify the status of submitted requests.
2. **Fog Node:**
   Acts as an intermediate processing unit close to users and IoT devices. It temporarily stores and pre-processes data before forwarding it to the cloud.
3. **Cloud Service Provider (CSP):**
   Stores long-term data and manages centralized services.
4. **Third-Party Auditor (TPA):**
   Performs integrity verification of data stored across fog and cloud layers.

4.1 System Administrator:
Manages public scheme information, reviews applications, and updates approval or rejection status.

Architecture Of Cloud Service Provider

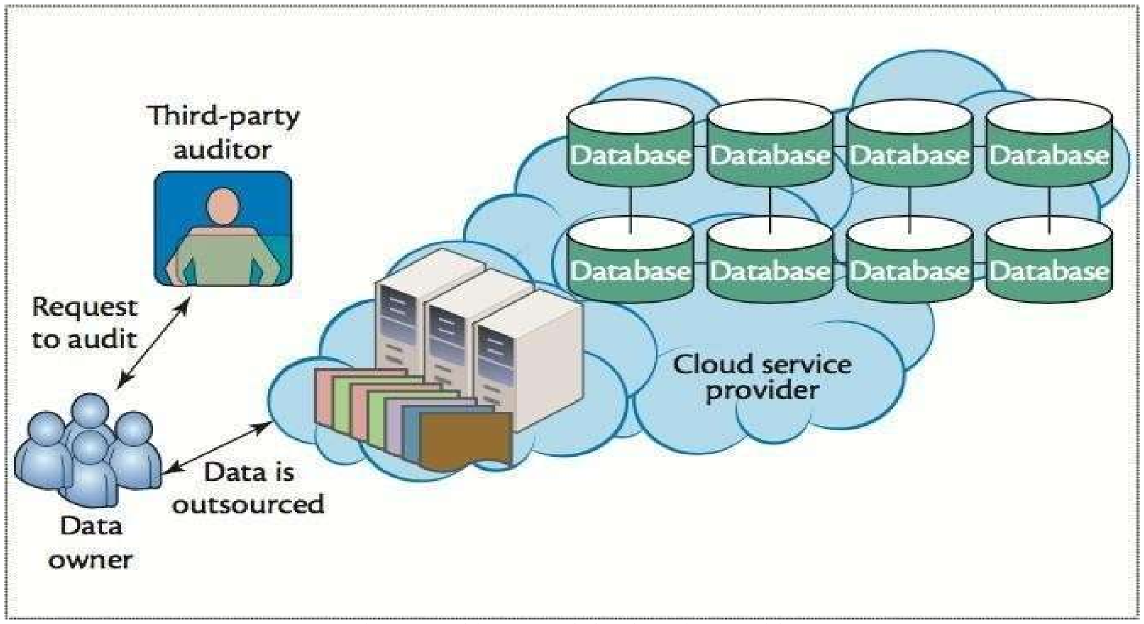


Fig 4.1 Architecture of cloud service provider

Auditor Client: The Auditor Client is the entity that outsources its data to the CSP for storage or processing and seeks the services of the TPA to audit the security and integrity of its data.

- Third-Party Auditor: The TPA is responsible for auditing the security and integrity of the data stored or processed by the CSP. The TPA performs audits by accessing the data stored at the CSP and verifying that it meets the security and integrity requirements.
- CSP: The CSP provides the storage or processing services to the Auditor Client. The CSP is responsible for maintaining the security and integrity of the data stored or processed on its infrastructure.
- Secure Channel: The Secure Channel is the communication channel established between the TPA and the CSP to ensure that the data being audited is not tampered with or compromised.
- Audit Logs: Audit Logs are the records of all activities performed on the data stored or processed by the CSP. The TPA uses the Audit Logs to verify the integrity and security.

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026

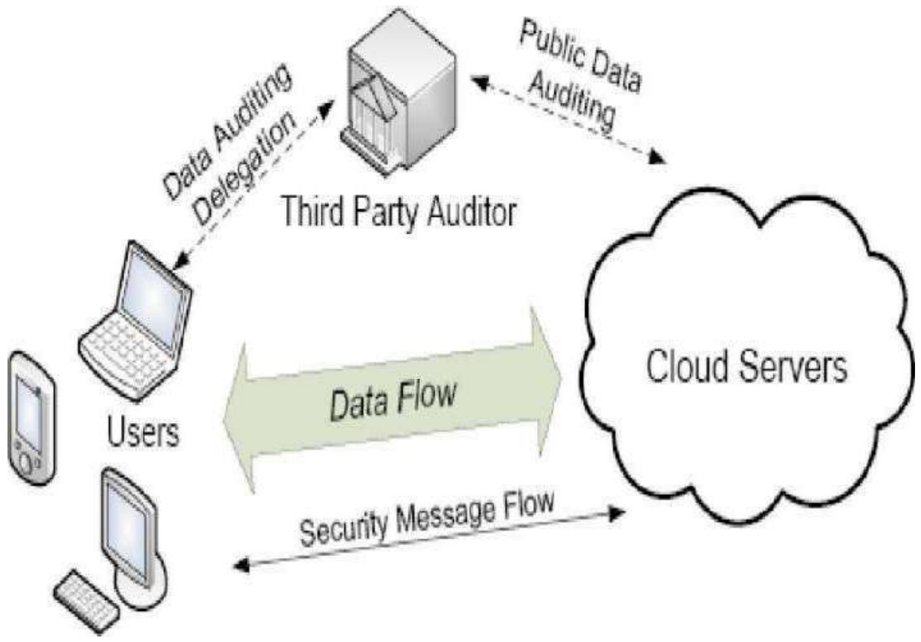


Fig 4.2 Data flow diagram of Third party auditor

Data Auditing Delegation: Data auditing delegation refers to the process of delegating the task of auditing the data stored on cloud servers to a third-party auditor (TPA) by the data owner or data user. This delegation is done to ensure that the data stored on the cloud is secure, confidential, and free from any unauthorized access or tampering. The TPA verifies the data integrity and security by performing various auditing operations on the data, such as verification of the data hash, comparison of the data copies, and analysis of the data logs. By delegating the auditing task to a TPA, the data owner or user can focus on their core business or personal activities, while the TPA takes care of the data security and integrity.

System Modules

The proposed third-party auditing based cloud and fog–cloud storage system is organized into a set of functional modules in order to improve maintainability, scalability, and security. Each module performs a clearly defined role in the overall auditing and data management workflow.

Admin Module

The administrator module is responsible for the overall supervision of the cloud storage environment.

The administrator is authorized to:

- view the list of registered users,
- monitor the data stored in the cloud storage area, and
- observe system activities related to file storage and auditing requests.

This module acts as a centralized management component and supports accountability and administrative control without directly accessing user data contents.

Third Party Auditor (TPA) Module

The Third Party Auditor module performs independent verification of data integrity on behalf of users.

Its main functions include:

- verifying whether any stored data blocks have been altered,
- detecting unauthorized modifications at the server side, and
- notifying the corresponding user when data inconsistency or integrity violation is detected.

The TPA does not download the complete data files. Instead, it validates integrity using metadata and challenge–response mechanisms, thereby preserving both privacy and efficiency.

User Module

The user module provides the interface through which legitimate users interact with the system.

The primary operations supported by this module are:

- user registration and authentication,
- secure login using user credentials, and
- uploading files and data to the cloud storage through the auditing framework.

Each uploaded file is associated with integrity metadata, enabling subsequent verification by the TPA.

Block Verification Module

The block verification module allows users to verify whether their uploaded data blocks remain intact.

Using this module, the user can:

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026

- request integrity verification for a specific file or data block, and
- confirm whether any modification has occurred at the cloud or server level.

The verification result is produced by the TPA based on cryptographic metadata, ensuring that users are informed about the correctness of their stored data without directly accessing internal storage structures.

Block Insertion Module

The block insertion module supports dynamic data operations.

Through this module, users are able to:

- insert new data blocks into an already stored file, and
- update the associated integrity information accordingly.

This functionality enables efficient support for dynamic cloud data without requiring complete re-generation of verification metadata for the entire file.

Block Deletion Module

The block deletion module enables users to securely remove selected data blocks from the cloud storage.

This module ensures that:

- deleted blocks are properly reflected in the verification metadata, and
- future auditing processes correctly consider the updated structure of the stored data.

By supporting controlled block deletion, the system maintains consistency between stored data and auditing information.

Unified Modeling Language (UML)

Unified Modeling Language (UML) is adopted to describe the structure and behavior of the proposed auditing system in a standardized and visual manner. UML allows software engineers to express both functional and architectural aspects of the system using well-defined modeling notations governed by syntactic and semantic rules.

The proposed system is modeled using five complementary views, each representing a different perspective of the system.

User Model View

The user model view represents the system from the end-user's perspective.

This view focuses on:

- how users interact with the system,
- the sequence of actions involved in registration, authentication, file upload, and verification requests, and
- the overall usage scenarios supported by the system.

The user model is mainly captured using use-case representations that describe the expected behavior of the system as observed by external actors.

Structural Model View

The structural model view represents the internal organization of the system.

This view describes:

- the main software components such as user interface, auditing services, storage services, and administrative services, and
- the static relationships between data structures and processing components.

The structural model emphasizes the arrangement of classes, modules, and interfaces required to support cloud storage, auditing operations, and user management.

Behavioral Model View

The behavioral model view describes the dynamic behavior of the system.

It illustrates:

- the interactions among users, the TPA, and the cloud server,
- the execution flow of operations such as file upload, block insertion, block deletion, and integrity verification, and
- the coordination among different modules during auditing requests.

This view captures how system components collaborate over time to fulfill functional requirements.

Implementation Model View

The implementation model view describes how the structural and behavioral designs are transformed into deployable software components.

This view represents:

- the mapping of software modules to program units,
- the organization of packages and executable components, and
- the realization of auditing, storage, and verification services in the target programming environment.

It reflects the actual construction of the system as it is to be implemented.

Environmental Model View

The environmental model view represents the operational context of the proposed system.

This view describes:

- the deployment of the client system, TPA system, and cloud server system,
- the communication infrastructure used to connect these components, and
- the execution environment in which the system operates, such as heterogeneous operating systems and network platforms.

This view is essential for understanding how the system functions within a distributed and networked environment.

UML Modeling Domains

UML modeling in the proposed system is divided into two major domains:

- **UML analysis modeling**, which concentrates on the user model view and structural model view in order to capture requirements and system structure.
- **UML design modeling**, which focuses on behavioral, implementation, and environmental model views to support detailed system design and deployment planning.

- TPA that is responsible for auditing the data stored on the cloud server to ensure its security and integrity.

Component Diagram Of Client And Server

This component diagram contains three components that are Server, TPA, Client and. Server will perform operations like it maintains client details & session information stores details & files and generate graphs. Client will perform operations like registration, login, upload files, download files, verify documents, add blocks, delete blocks. TPA will perform operations like take file size, divide file into blocks, maintain metadata information, send response and verification message. And this diagram shows the actions performed by these components.

- Client Interface: This component represents the user interface through which the client interacts with the system. It may include features such as a file browser, login screen, and upload/download buttons.
- Client Application: This component represents the application that runs on the client side and manages the interactions between the client interface and the cloud server.
  It may include features such as encryption/decryption modules, communication protocols, and access control modules.
- Server Application: This component represents the application that runs on the server side and manages the interactions between the cloud server and the client. It may include features such as storage management, audit management, and access control management.
- Database: This component represents the database system that stores the data and metadata related to the client's files on the cloud server. It may include features such as backup and recovery, data access control, and scalability.
- Third-Party Auditor (TPA): This component represents the TPA that performs the auditing of the client's data stored on the cloud server. It may include features such as auditing algorithms, signature verification, and logging.

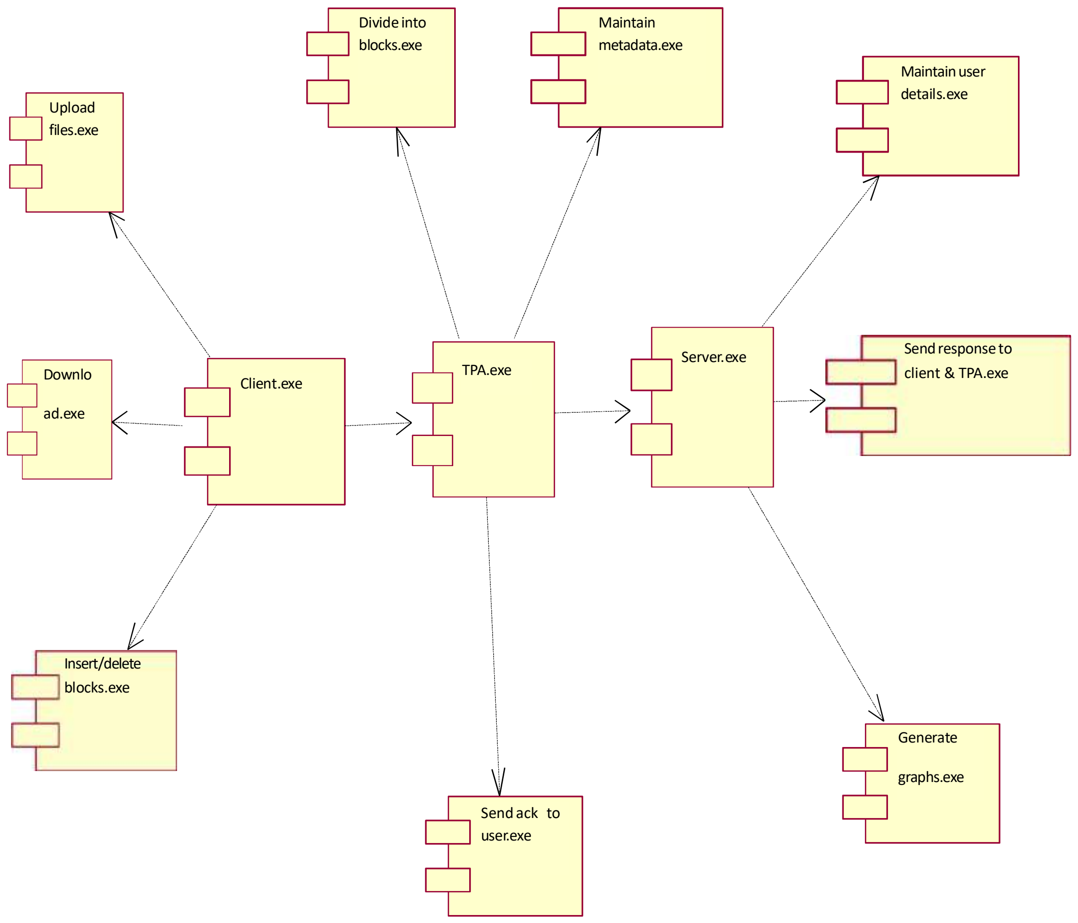


Fig 4.6 Component diagram of client and server

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 14, Issue 1, 2026

Activity Diagram Of Server And Client

This activity diagram contains three activities that are Server, TPA, Client and.This diagram shows the flow of control between these activities.

- Uploading Data:
- The client selects a file to upload.
- The client encrypts the file and sends it to the server.
- The server receives the file, stores it, and updates the database with the file's metadata.
- The server sends an acknowledgement to the client.
- Downloading Data:
- The client selects a file to download.
- The client sends a request to the server for the file.
- The client receives the file and stores it locally.
- Deleting Data:
- The client selects a file to delete.
- The client sends a request to the server to delete the file.

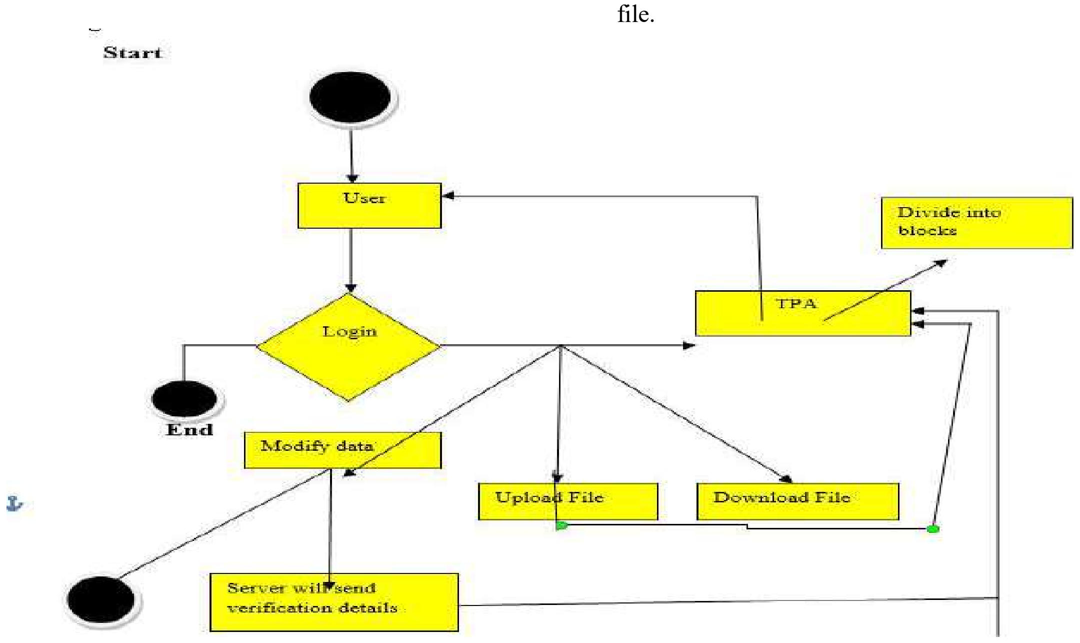


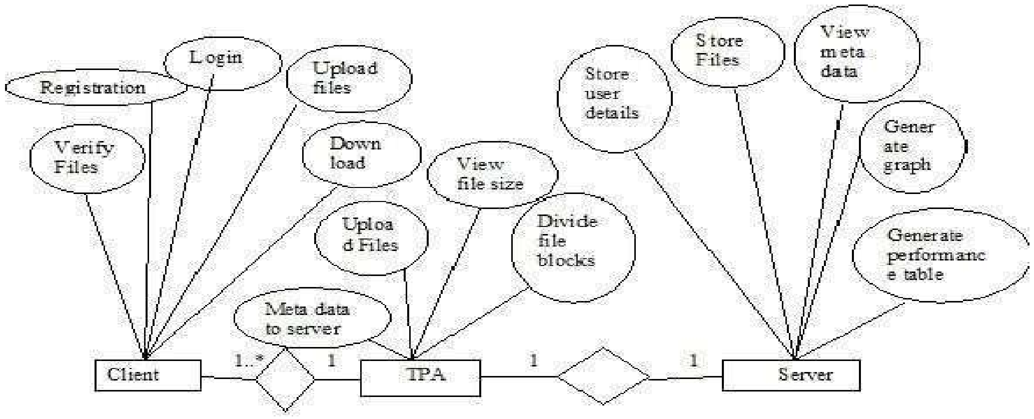
Fig 4.7 Activity diagram of server and client

Er-Diagrams Of Client , Tpa And Server

This ER-Diagram contains three entities that are Server, TPA, Client and. Serverwill perform operations like it maintains client details & session information stores details & files and generate graphs. Client will perform operations like registration, login upload files, download files, verify documents, add blocks, delete blocks. TPA will perform operations like take file size, divide file into blocks, maintain metadata information, send response and verification message. And this diagram shows the relationship between these entities.

- Client:
- The client entity represents the user who wants to store their data on the server.
- The client entity may have attributes such as client ID, username, password, email, and phone number.
- Server:
- The server entity represents the cloud server where the client's data is stored.
- The server entity may have attributes such as server ID, server name, IP address, and storage capacity.
- TPA:
- The TPA entity represents the third-party auditor who provides auditing and access control services to the client.
- The TPA entity may have attributes such as TPA ID, TPA name, and TPA public key.
- File:
- The file entity represents the client's data stored on the server.

- The file entity may have attributes such as file ID, file name, file size, and file type.



Results And Discussion

The proposed cloud data auditing system demonstrates secure and efficient verification of outsourced data. The system employs homomorphic linear authenticators and random masking techniques to ensure that the third-party auditor is unable to infer any information about the actual data content during the auditing process.
The auditing mechanism effectively reduces the burden on cloud users by eliminating the need to perform repeated integrity checks locally. The system also supports multi-user batch auditing, enabling the auditor to process multiple verification requests simultaneously, thereby improving overall efficiency.
The experimental evaluation confirms that the auditing protocol preserves data privacy while maintaining high verification accuracy.

Discussion

Performance evaluation was conducted using a Linux-based experimental platform with an Intel Core i5 processor and 8 GB of memory. Cryptographic operations were simulated using a pairing-based cryptography library.
The experimental setup considered file sizes of up to 20 MB and evaluated performance over increasing numbers of data blocks. The results demonstrate that the computational cost of signature generation and verification increases linearly with the number of data blocks. This confirms that the proposed design scales well for large datasets.
Batch auditing significantly reduces verification overhead when multiple users submit auditing requests concurrently. The results further indicate that the proposed auditing framework is suitable for fog-to-cloud environments and resource-constrained systems, such as IoT platforms.

CONCLUSION

This work presented a secure and efficient public cloud data auditing framework designed for fog-to-cloud and IoT-oriented environments. The proposed approach enables privacy-preserving verification of outsourced data using homomorphic authentication techniques combined with randomized masking.
By separating verification responsibility from data ownership, the framework reduces the computational burden on users and improves trust in cloud storage services. The extension of the auditing protocol to a multi-user and batch auditing environment further enhances system scalability.
Security analysis confirms that the scheme preserves confidentiality of stored data and resists common integrity attacks. Performance evaluation demonstrates that the proposed method is more efficient than conventional auditing approaches, especially for resource-limited devices and large-scale deployments
.

References

[1] C. Wang, Q. Wang, K. Ren and W. Lou, "Privacy-Preserving Public Auditing for Storage Security in Cloud Computing," *Proceedings of IEEE INFOCOM 2010*, San Diego, CA, USA, March 2010, pp. 1–9.
[2] P. Mell and T. Grance, *The NIST Definition of Cloud Computing (Draft)*, National Institute of Standards and Technology (NIST), Special Publication, June 2009.
Available:
http://csrc.nist.gov/groups/SNS/cloudcomputing/
[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica and M. Zaharia, "Above the Clouds: A Berkeley View of Cloud Computing," Technical Report No. UCB/EECS-2009-28, University of California, Berkeley, Feb. 2009.
[4] Cloud Security Alliance, *Top Threats to Cloud Computing*, Cloud Security Alliance, 2010.
Available: https://www.cloudsecurityalliance.org

ISSN 2347–3657

Volume 14, Issue 1, 2026

[5] M. Arrington, "Gmail Disaster: Reports of Mass Email Deletions," *TechCrunch*, Dec. 2006. Available: http://www.techcrunch.com/2006/12/28/gmail-disaster-reportsof-mass-email-deletions/
[6] R. Pressman, *Software Engineering: A Practitioner's Approach*, 7th ed., McGraw-Hill, New York, USA, 2010.
[7] Y. Shiran, *JavaScript Programming*, 2008.
[8] S. Holzner, *HTML Black Book (HTML 4)*, Coriolis Group Books, USA, 2001.
[9] P. Moss and R. Patel, *Java Database Programming with JDBC*, O'Reilly Media, USA, 2003.
[10] S. W. Chatterjee et al., *Professional J2EE Development*, Wrox Press, USA, 2002.
[11] N. Todd, *Java Server Pages*, Prentice Hall, USA, 2001.