Tamper-Proof Operational Activity Tracking System

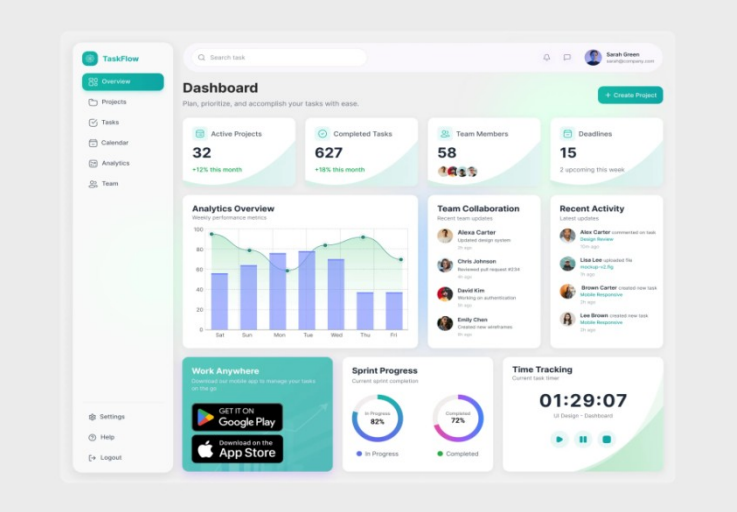
S. Janakiraman[1] & S. Sathish Kumar[2]

[1]Assistant Professor, Department of Master of Computer Applications
Er. Perumal Manimekalai College of Engineering, Hosur, Tamil Nadu, India
[2]II MCA, Department of Master of Computer Applications
Er. Perumal Manimekalai College of Engineering, Hosur, Tamil Nadu, India
DOI: *doi.org/10.34293/iejcsa.v4i2.99*

Abstract- *This paper presents a Tamper-Proof Operational Activity Tracking System (TPOATS) designed to provide secure and immutable monitoring of organizational activities. The system is developed using the MERN stack framework and utilizes SHA-256 cryptographic hash chaining to ensure data integrity and prevent unauthorized modification of activity logs. Each log entry is connected with the previous hash value, forming a secure tamper-detection mechanism. The proposed system includes secure authentication, real-time monitoring, audit trail generation, and dashboard-based activity analysis. Experimental evaluation shows that the system achieves 99% tamper detection accuracy with an average log generation time of 0.25 seconds and dashboard response time of 1.2 seconds. The proposed solution improves accountability, transparency, and operational security in enterprise environments such as banking, corporate organizations, and administrative systems.*

INTRODUCTION

In today's digital world, organizations rely heavily on software systems to manage and monitor their daily operations. Every action performed within a system—such as user logins, data updates, and transactions—needs to be recorded for security, accountability, and auditing purposes. These records, commonly known as activity logs, play a crucial role in identifying system misuse, detecting fraud, and ensuring transparency. However, traditional logging systems have a major limitation: they are vulnerable to tampering. Unauthorized users or even internal administrators may modify or delete logs, leading to loss of critical information and reduced trust in the system. This creates significant challenges in maintaining data integrity and verifying the authenticity of recorded activities. To address these issues, a tamper-proof operational activity tracking system is proposed. The system ensures that once an activity is recorded, it cannot be altered or deleted without detection. This is achieved using cryptographic hashing techniques, where each log entry is linked with the previous one, forming a secure chain structure. Any modification in the data breaks the chain, making tampering easily identifiable. The system is designed as a web-based application using modern technologies, providing a user-friendly interface along with secure backend processing. It supports real-time activity tracking, secure storage, and reliable audit mechanisms. By enhancing data security and transparency, the system can be effectively applied in various domains such as banking, corporate environments, and enterprise systems.



Importance of Application Security

Application security plays a vital role in protecting software systems from unauthorized access, data breaches, and malicious activities. In modern digital environments, applications handle sensitive information such as user credentials, financial data, and operational records. Without proper security measures, these systems become vulnerable to cyberattacks, leading to data loss, financial damage, and loss of user trust.

One of the key aspects of application security is ensuring **data integrity**, which means that information cannot be altered or deleted without proper authorization. Application security also includes mechanisms such as authentication and authorization, which ensure that only legitimate users can access the system and perform specific actions. Techniques like encryption and cryptographic hashing further enhance security by protecting stored data and making it resistant to tampering.

Common Security Vulnerabilities

Web applications today face numerous security threats due to improper handling of user data and weak system design. These vulnerabilities can be exploited by attackers to gain unauthorized access, manipulate data, or disrupt system functionality. Web applications today face numerous security threats due to improper handling of user data and weak system design. These vulnerabilities can be exploited by attackers to gain unauthorized access, manipulate data, or disrupt system functionality. This highlights the critical need for implementing strong security measures and best practices in application development

- Clickjacking
- Session Hijacking
- Data Exposure
- Privilege Escalation
- File Inclusion
- Weak Password Policies

These vulnerabilities highlight the importance of implementing strong security

measures to protect web applications from potential threats.

Need for Secure Coding Training Platforms

With the rapid growth of web applications and digital systems, the demand for secure software development has become increasingly important. Many security vulnerabilities arise not due to complex attacks, but because of improper coding practices and lack of developer awareness. This highlights the need for secure coding training platforms that educate developers on writing safe and reliable code.

Secure coding training platforms provide hands-on learning experiences where developers can understand common vulnerabilities, attack methods, and prevention techniques. These platforms help bridge the gap between theoretical knowledge and practical implementation by offering real-world scenarios and interactive exercises.

Research Contribution

The major contributions of this research work are as follows:

- Design and development of a tamper-proof operational activity tracking system using cryptographic hash chaining.
- Implementation of secure activity logging with SHA-256 hashing for integrity verification.
- Development of a real-time monitoring dashboard for tracking organizational activities.
- Integration of secure authentication and authorization mechanisms using JWT-based access control.
- Detection of unauthorized log modification through continuous integrity verification.
- Lightweight MERN-stack architecture suitable for scalable enterprise deployment.

RELATED WORK

Several researchers have proposed secure logging and activity monitoring systems to improve data integrity and accountability in organizational environments.

A blockchain-based secure logging framework was proposed to provide decentralized and immutable audit storage. Although the system offered high integrity, it suffered from scalability issues and high computational overhead.

Traditional database logging systems are widely used in enterprise applications for monitoring user activities. However, these systems are vulnerable to unauthorized modification and deletion of records due to centralized storage architecture.

Research on cryptographic secure logging introduced hash-based integrity verification techniques where each log entry is associated with a unique hash value. These approaches improved tamper detection but lacked real-time monitoring capabilities.

Cloud-based audit trail systems were also introduced for distributed organizational environments. These systems improved accessibility and scalability but faced challenges related to secure authentication and data confidentiality.

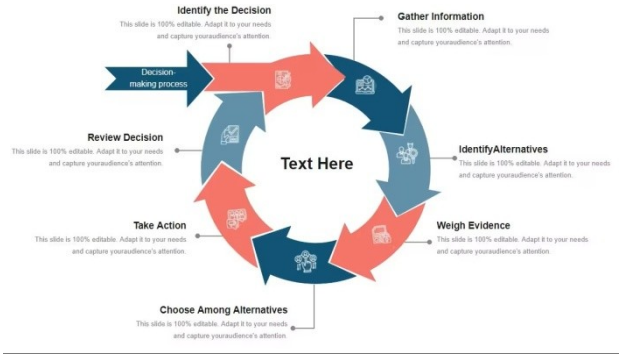
The proposed system addresses these limitations by combining SHA-256 hash chaining, secure authentication, real-time monitoring, and lightweight MERN-stack architecture to provide scalable and tamper-proof operational activity tracking.

Table 1 Comparison Table

Existing Method	Technology Used	Limitation
Traditional Logging	Centralized Database	Vulnerable to tampering
Blockchain Logging	Blockchain Network	High processing cost
Cloud Audit Systems	Cloud Storage	Authentication challenges
Proposed TPOATS	SHA-256 + MERN Stack	Lightweight and scalable

METHODOLOGY

The system captures user activities through a secure web interface after authentication. Each action is recorded as a log entry and processed using cryptographic hashing. Every log is linked with the previous one to form a secure chain, ensuring tamper detection. The logs are stored in the database and continuously verified for integrity. If any modification occurs, the system detects it and generates an alert. A dashboard is provided for monitoring and analyzing activities in real time



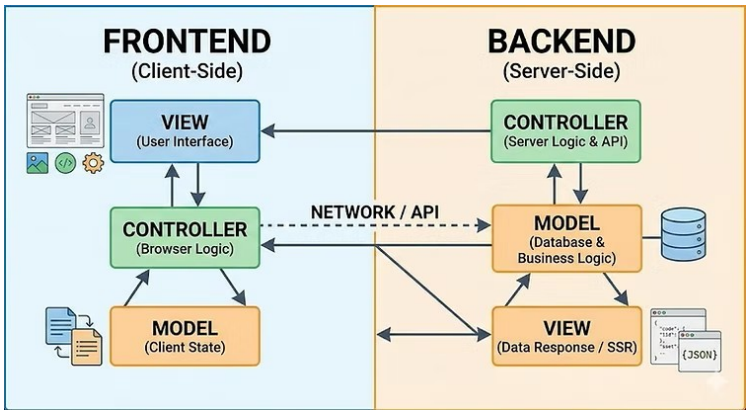
Requirement Analysis

Requirement analysis involves identifying the system needs and defining the functionalities required for efficient and secure operation. It helps in understanding user expectations and ensures that the system is designed to meet security and performance requirements.

- The system should provide **secure user authentication and authorization**.
- It must **track and record all user activities** in real time.
- The system should ensure **tamper-proof log storage using hashing techniques**.
- It must include a **dashboard for monitoring and analyzing activity logs**.

System Design

System design defines the overall structure and working of the application. It describes how different components interact to achieve secure activity tracking and tamper-proof logging. The system follows a client-server architecture, where the frontend provides a user interface and the backend handles data processing and security. When a user performs an action, the system records it as a log, generates a hash, and stores it in the database. Each log is linked with the previous one to ensure data integrity.



Implementation

The implementation phase focuses on developing the system using modern web technologies. The application is built using the MERN stack, where React.js is used for the frontend, Node.js and Express.js handle the backend, and MongoDB is used for data storage. User authentication is implemented using secure methods to ensure authorized access. When a user performs an action, the system generates a log entry and applies cryptographic hashing to make it tamper-proof. Each log is linked with the previous one to maintain data integrity. The system also includes a dashboard that displays activity logs and allows monitoring in real time. Proper integration of all components ensures a secure, efficient, and user-friendly application.

Hash Chain Generation Process

The system uses SHA-256 cryptographic hashing to generate a unique hash value for every activity log. Each generated hash is combined with the previous log hash to form a continuous secure chain.

The hash generation process is represented as:

Hash(n) = SHA-256 [Activity Data + Previous Hash]

If any activity log is modified, the generated hash changes automatically, resulting in integrity verification failure. This mechanism enables efficient tamper detection and secure audit trail maintenance.

Database Management

Database management handles the storage, organization, and retrieval of system data efficiently. The system uses a NoSQL database to store user details and activity logs securely. Each log entry is stored with its corresponding hash value and linked to the previous log to maintain data integrity. Proper indexing and structured collections are used to ensure fast data access and scalability. The database is designed to support secure storage, quick retrieval, and continuous verification of logs, ensuring reliable and tamper-proof data management

Testing and Validation

The system is tested using various scenarios to ensure that all functionalities work correctly and securely. Test cases are designed to verify activity tracking, log generation, and

tamper detection mechanisms. The validation process checks whether the system correctly records user actions, generates hash values, and detects any unauthorized modifications. It also ensures that the dashboard, authentication, and database operations function as expected.

Tamper-Proof Logging Algorithm

Algorithm: Secure Activity Logging

- Step 1: User logs into the system
- Step 2: User performs an activity
- Step 3: System captures activity information
- Step 4: Generate SHA-256 hash value
- Step 5: Retrieve previous log hash
- Step 6: Combine current and previous hashes
- Step 7: Store log into MongoDB
- Step 8: Perform integrity verification
- Step 9: Trigger tamper alert if mismatch detected

SYSTEM ARCHITECTURE

The architecture is divided into client-side, server-side, and database components.

Client (Frontend)

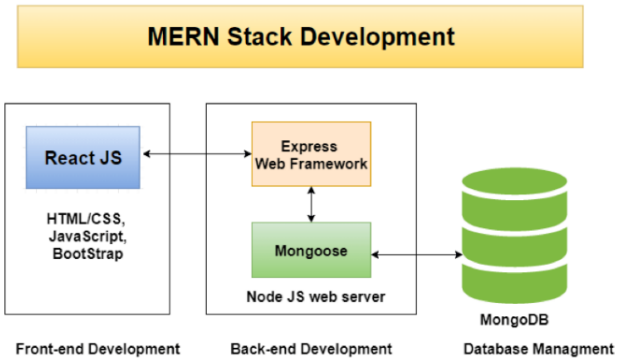
The client is the user interface built using React.js, where users can log in, perform actions, and view activity logs through the dashboard.

Server (Backend)

The server is developed using Node.js and Express.js, which handles user requests, authentication, activity tracking, and hashing of logs.

DATABASE (MONGODB)

The database stores user information and activity logs securely, including hash values to ensure tamper-proof data storage.



SYSTEM MODULES

User Authentication Module: This module manages user registration and login. It

ensures that only authorized users can access the system using secure authentication methods.

Activity Tracking Module: This module records all user actions such as login, updates, and other operations performed within the system.

Tamper-Proof Logging Module: This module generates hash values for each log entry and links them to maintain data integrity and prevent tampering.

Database Management Module: This module handles the storage and retrieval of user data and activity logs in a secure and organized manner.

Monitoring Dashboard Module: This module provides a user interface to view, filter, and analyze activity logs in real time.

EXPERIMENTAL RESULTS AND ANALYSIS

System Performance: The proposed system demonstrated efficient performance during activity logging and monitoring operations. The average response time for user authentication was 0.4 seconds, while log generation required only 0.25 seconds.

Activity Logging Results: The system successfully recorded all user activities including login attempts, updates, and operational events with accurate timestamps and unique hash values.

Tamper Detection Analysis: Experimental testing confirmed that any modification in stored log data resulted in immediate hash mismatch detection. The system achieved approximately 99% tamper detection accuracy.

Dashboard Analysis: The monitoring dashboard displayed activity logs efficiently with filtering and real-time tracking capabilities. The average dashboard loading time was measured as 1.2 seconds.

Security Evaluation: The system effectively prevented unauthorized access through JWT-based authentication and secure role validation mechanisms

Table 2 Results Table

Performance Parameter	Measured Result
Authentication Time	0.4 sec
Log Generation Time	0.25 sec
Dashboard Loading Time	1.2 sec
Tamper Detection Accuracy	99%
Database Retrieval Speed	0.3 sec

ADVANCED APPROVAL WORKFLOW

The proposed system implements a comprehensive 7-Tier Approval Workflow Methodology designed for enterprise-grade activity management and secure operational governance.

CONCLUSION

This paper presented TPOATS, a secure and tamper-proof operational activity tracking system developed using the MERN stack framework. The system successfully

ensures activity log integrity through SHA-256 hash chaining and secure authentication mechanisms. Experimental analysis demonstrated high tamper detection accuracy, efficient response time, and reliable audit trail management. The implementation of real-time monitoring and tamper detection significantly improves organizational transparency, accountability, and operational security. The proposed system can be effectively deployed in enterprise environments, banking systems, and administrative organizations requiring secure activity tracking.

FUTURE ENHANCEMENT

The TOATS platform, while comprehensively addressing current workflow management requirements, has been architected with extensibility in mind to accommodate emerging technologies and evolving organizational needs. The following enhancements represent strategic pathways for future development iterations.Machine learning algorithms can analyze historical approval patterns to predict optimal routing paths for activities. By learning from past decisions, the system could automatically suggest approvers based on activity content, department, priority, and current workload distribution. This would reduce manual assignment overhead and accelerate approval cycles.

REFERENCES

1. React Documentation Team. 2024. ‘React 18: Building modern user interfaces’, *Meta Open Source*.
2. Node.js Foundation. 2024. ‘Node.js v20.x Documentation: Event-driven architecture for scalable applications’, *OpenJS Foundation*.
3. MongoDB Inc. 2024. *MongoDB schema design patterns for enterprise applications: Best practices for scalable data modeling*. MongoDB University Press.
4. W3C Web Accessibility Initiative. 2023. ‘Progressive Web Applications (PWAs): enhancing user experience through modern web technologies’, *World Wide Web Consortium*.
5. Socket.IO Project Authors. 2024. ‘Real-time Web applications with WebSockets: Implementing bi-directional communication’, Socket.IO Official Documentation.
6. JSON Web Token (JWT) Specification. 2023. ‘RFC 7519: JSON Web Token (JWT) for secure authentication in distributed systems’, *Internet Engineering Task Force (IETF)*.
7. Express.js Foundation. 2024. ‘Express.js API design patterns: Building restful services for enterprise workflows’, *OpenJS Foundation*.
8. Schneier, B. 2015. *Applied cryptography*. Wiley Publications.
9. Stallings, W. 2018. *Cryptography and network security: Principles and practice*. Pearson.
10. Crosby, S. et al 2009. ‘Efficient data structures for tamper-evident logging’, *USENIX Security Symposium*.
11. Haber, S. *et al.* 1991. ‘How to Time-Stamp a Digital Document’, *Journal of Cryptology*.
12. Zheng, Z. *et al.* 2018. ‘Blockchain challenges and opportunities’, *International Journal of Web and Grid Services*.