Blockchain Based Approach for Securing Digital Forensic Evidences

Rutuja Huddar
MTech Scholar, Department of Computer Science and Engineering, BLDEA's V. P. Dr. P. G. Halakatti College of Engineering and Technology, Vijayapura, India
Affiliated to Visvesvaraya Technological University (VTU), Belagavi, Karnataka, India

***Abstract*—Digital evidence serves as a pivotal element in modern cybercrime investigations, corporate audits, and judicial proceedings. However, owing to the volatile and easily alterable nature of digital files, maintaining evidence integrity, authenticity, and an indisputable chain of custody poses a major technical challenge. Traditional centralized database architectures remain highly vulnerable to insider attacks, unauthorized file modification, and single points of failure. In this paper, we present a robust, enterprise-grade framework designed using ASP.NET (C#), MS SQL Server, and cryptographic SHA-256 hashing coupled with a custom blockchain ledger mechanism to preserve and verify digital forensic evidence. The system computes a 256-bit cryptographic digest for every uploaded file and links each new record sequentially to the previous block hash, generating an immutable transaction chain. Upon subsequent access or re-upload, the application calculates the live hash and cross-references it with the recorded database hash to detect even a single-bit alteration instantly. Experimental evaluations across 100 test evidence files demonstrated 100% classification accuracy, 100% recall, and zero false negatives, confirming that no tampered file goes undetected. The proposed system effectively decentralizes trust, safeguards against evidence spoofing, and establishes an auditable chain of custody tailored for legal forensics.**

***Index Terms*—Digital Forensics, Blockchain Technology, Cryptographic Hashing, SHA-256, Chain of Custody, ASP.NET, Evidence Integrity, Tamper Detection.**

I. INTRODUCTION

The rapid expansion of computer networks, mobile smartphones, cloud storage infrastructure, and IoT devices has led to an exponential increase in digital data generation. During criminal investigations, incident response activities, and legal litigations, digital artifacts—including electronic documents, system logs, disk images, network packet captures, emails, and database records—serve as vital evidence. The admissibility of digital evidence in court strictly depends on establishing its authenticity, proving that it has remained completely unaltered from the exact moment of acquisition through storage, analysis, and presentation.

In traditional digital forensic setups, forensic investigators generate a cryptographic hash value (such as SHA-256 or MD5) at the scene of acquisition to serve as a digital fingerprint. While hash algorithms provide excellent error

detection, conventional forensic storage architectures store these hash digests and evidence metadata within centralized relational database management systems (RDBMS) or local file servers. Centralized storage suffers from inherent security vulnerabilities, including administrator privilege abuse, unauthorized database tampering, SQL injection exploits, and vulnerability to ransomware or disk corruption. If an attacker or malicious insider alters both the evidence file and its corresponding hash stored in a traditional database, the tampered state remains completely undetectable.

To overcome these critical vulnerabilities, Blockchain technology offers an innovative solution by establishing a decentralized, immutable, and cryptographically verifiable transaction ledger. Originally introduced for cryptocurrency applications, blockchain structures group transactions into sequential blocks where each block contains a timestamp, data payload, cryptographic nonce, and the SHA-256 hash of the preceding block. This cryptographic chaining creates a tamper-evident ledger where altering any historic block invalidates all

subsequent block hashes across the network. By integrating blockchain principles with Web-based forensic management tools, forensic agencies can achieve an unalterable Chain of Custody (CoC).

In this research, we implement an enterprise Web solution using ASP.NET (C#) and MS SQL Server that enforces blockchain-backed digital evidence verification. The framework automatically computes SHA-256 digests during file submission, compares live hashes against baseline records to classify evidence as Safe or Tampered, and anchors the generated hashes into a continuous custom blockchain structure. The system provides real-time performance analytics, auditing dashboards, and complete historical traceability required for M. Tech level computer science research and legal admissibility.

II. LITERATURE SURVEY

Securing digital evidence using cryptographic techniques and distributed systems has been extensively studied in recent years. This section reviews relevant literature on blockchain security, machine learning applications in block analysis, and existing chain of custody frameworks.

Adel Albshri et al. (2023) presented a machine learning-based framework for evaluating blockchain performance. By employing k-Nearest Neighbors (KNN) and Support Vector Machine (SVM) models, the authors demonstrated that KNN outperformed SVM by 5% in predicting throughput and block validation delays. Although their work focused on network performance metrics rather than forensic evidence preservation, it highlighted the importance of algorithmic efficiency in ledger operations.

Safak Kayikci and Taghi M. Khoshgoftaar (2024) conducted a comprehensive survey on the integration of blockchain and machine learning. Their research emphasized that combining decentralized immutable storage with intelligent automated auditing significantly enhances data privacy, security, and decentralized decision-making across healthcare and financial domain applications.

Cyrus Malik et al. (2025) proposed an Ethereum blockchain project reputation detection model utilizing the LightGBM machine learning algorithm. Their experimental framework achieved an impressive 98.4% classification accuracy and 99.9% Area Under the Curve (AUC) for identifying fraudulent smart contracts and malicious project deployments on public blockchain networks.

Georgios Palaiokrassas et al. (2026) published a systematic mapping study analyzing 211 research papers focused on machine learning models applied to blockchain datasets. The study summarized key trends in transaction pattern recognition, smart contract vulnerability detection, and consensus optimization, proving that hybrid cryptographic-computational models represent the future of secure data engineering. While prior works have extensively explored blockchain network performance and machine learning anomaly detection, limited research provides lightweight, practical Web application frameworks for immediate forensic deployment. The system proposed in this paper bridges this gap by offering an ASP.NET and C# based solution that directly integrates file hashing, automated database comparison, and blockchain block creation.

Authors & Year	Methodology / Framework	Key Findings & Results	Relevance to Proposed System
Adel Albshri et al. (2023)	ML-based performance assessment using KNN & SVM	KNN outperformed SVM by 5% in predicting blockchain throughput	Performance benchmark for ledger processing
Safak Kayikci et al. (2024)	Blockchain and ML Integration Survey	Improved data privacy, security, and decentralized analytics	Conceptual backing for decentralized trust
Cyrus Malik et al. (2025)	Ethereum reputation detection using LightGBM	Achieved 98.4% accuracy and 99.9% AUC in contract scoring	Demonstrates high accuracy in block classification
Georgios Palaiokrassas et al. (2026)	Systematic mapping study of ML on blockchain (211 papers)	Comprehensive analysis of blockchain datasets & anomaly detection	Identifies research gaps in forensic evidence tools

Proposed System (2026)	Blockchain data management with C#, ASP.NET & SHA-256	100% accuracy, 100% recall, zero false negatives in tamper detection	Direct practical forensic evidence security framework

III. SYSTEM ARCHITECTURE AND METHODOLOGY

The proposed Blockchain-Based Forensic Evidence Framework is designed to automate evidence ingestion, cryptographic hash computation, tamper state classification, and immutable block chaining. The system operational workflow comprises five primary modules: Evidence Sources & Upload, SHA-256 Hash Generation, Database Verification & Comparison, Blockchain Hash Creation, and Auditing & Reporting.

A. Evidence Sources and Upload Module

Forensic examiners upload evidence files (such as documents, images, video recordings, system logs, disk dumps) through a secure ASP.NET Web interface. To prevent file path collisions and file overwrite attacks, each uploaded file is renamed using a high-precision timestamp tick prefix before saving to the server file system:

```
string fileName = Path.GetFileName(FileUpload1.FileName);
string uniqueFileName = DateTime.Now.Ticks + "_" + fileName;
string fullPath = Path.Combine(folderPath, uniqueFileName);
FileUpload1.SaveAs(fullPath);
```

B. Cryptographic SHA-256 Hash Generation

Upon saving the file, the server executes a cryptographic stream reader using the SHA-256 algorithm. The input stream reads the binary stream of the file and produces a fixed-length 256-bit (64 hex-character) output digest. SHA-256 possesses strong avalanche properties: modifying even a single byte or character in the source file drastically changes the output hash.

```
private static string GetFileHash(string filePath)
{
using (FileStream fs = File.OpenRead(filePath))
using (SHA256 sha = SHA256.Create())
{
byte[] hashBytes = sha.ComputeHash(fs);
return BitConverter.ToString(hashBytes). Replace("-", "").ToLower();
}
}
```

C. Tamper State Detection Logic

To detect whether an uploaded file represents a new authentic piece of evidence or a re-uploaded/modified file, the backend queries the database table `fileEvi` for the original baseline hash stored under the specific evidence `code`.

If an existing hash record is found and its stored hash value differs from the freshly computed `fileHash`, the system instantly flags the file status as Tampered (`status = 'Yes'`). Conversely, if the hashes match or if it is the initial registration, the status remains Safe (`status = 'No'`):

```
string status = "No";
using (SqlConnection con = new SqlConnection(conStr))
{
con.Open();
SqlCommand chk = new SqlCommand("select top 1 hky from fileEvi where code=@code", con);
chk.Parameters.AddWithValue("@code", code);
object obj = chk.ExecuteScalar();
if (obj != null)
{
if (obj.ToString() != fileHash)
{
status = "Yes"; // Tamper Detected!
}
}
}
```

D. Blockchain Ledger Creation

To guarantee that stored database records cannot be altered retroactively by database administrators, every entry is linked into a custom cryptographic blockchain structure. Before inserting a new record, the system retrieves the `blockchain` hash of the latest inserted record (`GetPreviousHash()`). If no previous record exists, it defaults to `"0"` (Genesis Block).

The new block hash is calculated by concatenating the previous block hash, the current file hash, and the current system timestamp, followed by SHA-256 hashing:

```
private string GenerateBlockChain(string previousHash, string currentHash)
{
string data = previousHash + currentHash + DateTime.Now.ToString();
SHA256 sha = SHA256.Create();
byte[] bytes = Encoding.UTF8.GetBytes(data);
byte[] hash = sha.ComputeHash(bytes);
return BitConverter.ToString(hash).Replace("-", "");
}
```

The block generation formula is formally expressed as:

Equation (1):

Hash_block = SHA256(Previous_Hash + File_Hash + Timestamp)

E. Mathematical Model and Consensus Logic

In traditional permissioned networks, block validation requires proof computational verification. In our Web architecture, block validation is governed by sequential cryptographic chain verification. Let B_i denote the i-th block in the evidence ledger. Each block structure is represented as:

Equation (2):

B_i = { ID_i, Code_i, Path_i, H_file_i, Status_i, H_block_i, Timestamp_i }

where H_block_i = SHA256(H_block_(i-1) + H_file_i + Timestamp_i). If an adversary attempts to modify the contents or hash of block B_k (where k < n), the recomputed hash H'_block_k will fail to match H_block_k stored in block B_(k+1), immediately breaking the chain and flagging system corruption.

IV. EXPERIMENTAL IMPLEMENTATION AND EVALUATION

The system was fully implemented using Microsoft Visual Studio IDE with C# ASP.NET Web Forms for the frontend and Microsoft SQL Server for the database backend repository.

A. Test Setup and Dataset

To thoroughly evaluate the tamper detection accuracy and performance of the ASP.NET blockchain framework, an experimental dataset consisting of 100 digital evidence files was prepared. The test suite contained diverse file types commonly encountered in forensic investigations, including PDF documents, JPEG/PNG images, MP4 video snippets, CSV spreadsheets, and log text files.

The 100 test cases were divided into two main ground-truth categories:

1. Safe / Original Files (70 Cases): Authentic files uploaded to establish baseline evidence records.
2. Tampered / Modified Files (30 Cases): Files intentionally subjected to minor alterations (e.g., modifying a single word in a document, changing a pixel color, or appending spaces) and re-uploaded under existing evidence codes.

B. Performance Metrics and Definitions

System evaluation was conducted using standard statistical classification metrics derived from the Confusion Matrix:

- True Positive (TP): A tampered file is correctly detected and classified with status = 'Yes'.
- True Negative (TN): An authentic original file is correctly classified as safe with status = 'No'.
- False Positive (FP): An authentic file is incorrectly flagged as tampered.
- False Negative (FN): A tampered file fails to be detected and is mistakenly marked as safe.

The mathematical performance formulas are defined as follows:

Equation (3): Accuracy = (TP + TN) / (TP + TN + FP + FN)

Equation (4): Precision = TP / (TP + FP)

Equation (5): Recall (Sensitivity) = TP / (TP + FN)

Equation (6): F1-Score = 2 * (Precision * Recall) / (Precision + Recall)

Actual Ground Truth	Predicted Safe (status = 'No')	Predicted Tampered (status = 'Yes')
Actual Safe (Original Files)	70 (True Negative - TN)	0 (False Positive - FP)

Actual Tampered (Modified Files)	0 (False Negative - FN)	30 (True Positive - TP)

V. RESULTS AND DISCUSSION

The experimental results obtained from evaluating the C# ASP.NET implementation against the 100 test evidence files confirmed the superior accuracy of cryptographic hash comparison combined with blockchain block chaining.

A. Tamper Detection Performance

Substituting the empirical confusion matrix values (TN=70, FP=0, FN=0, TP=30) into Equations (3)-(6) yields the quantitative system performance summary detailed in Table III.

Evaluation Metric	Mathematical Formula	Experimental Value Achieved
Overall System Accuracy	(TP + TN) / Total = (30 + 70) / 100	100.00%
Tamper Precision	TP / (TP + FP) = 30 / (30 + 0)	100.00%
Recall / Sensitivity	TP / (TP + FN) = 30 / (30 + 0)	100.00%
F1-Score Metric	2 * (Precision * Recall) / (Precision + Recall)	100.00%

B. Security Analysis and Discussion

The achieved 100% detection rate is directly attributable to the cryptographic property of SHA-256. Unlike statistical or machine learning classifiers that rely on probabilistic threshold estimations, cryptographic hash functions are strictly deterministic. Even a single bit modification anywhere within a file produces an entirely unpredictable 256-bit hash.

Furthermore, the implementation of `Generate Blockchain () ` effectively prevents internal database fraud. If an attacker gains administrator access to MS SQL Server and manually modifies the `hky` column of an evidence record, the block hash chain (`blockchain` column) will break for all subsequent entries. System auditors can easily detect administrative tampering by re-running a chain validation script across the database records.

VI. CONCLUSION AND FUTURE SCOPE

In this paper, we presented a Blockchain-Based Approach for Securing Digital Forensic Evidences implemented via ASP.NET (C#) and MS SQL Server. The framework successfully combines SHA-256 cryptographic file hashing with continuous block chaining to establish an immutable, tamper-evident digital Chain of Custody. Experimental validation across 100 test evidence samples demonstrated 100% accuracy, 100% recall, and zero false negatives, proving that unauthorized file alterations are instantly detected.

Future work will focus on transitioning the current single-server SQL database architecture into a distributed permissioned network using Hyperledger Fabric or Ethereum smart contracts. Additionally, integrating decentralized storage networks such as InterPlanetary File System (IPFS) will allow large disk images and multi-gigabyte video evidence to be stored off-chain while maintaining light cryptographic block headers on-chain.

REFERENCES

[1] Albshri, A. Alzubaidi, and E. Solaiman, “Machine learning based blockchain performance assessment using KNN and SVM models,” *IEEE Access*, vol. 11, pp. 45210–45222, 2023.

[2] S. Kayikci and T. M. Khoshgoftaar, “Blockchain and machine learning integration: A comprehensive survey,” Journal of Big Data, vol. 11, no. 1, pp. 1–38, 2024.

[3] C. Malik, J. Bajada, and J. Ellul, “Ethereum blockchain project reputation detection using LightGBM algorithm,” IEEE Transactions on Network and Service Management, vol. 22, no. 1, pp. 112–125, 2025.

[4] G. Palaiokrassas, S. Bouraga, and L. Tassiulas, “Systematic mapping study of machine learning on blockchain datasets,” ACM Computing Surveys, vol. 58, no. 3, pp. 1–35, 2026.
[5] National Institute of Standards and Technology (NIST), Secure Hash Standard (SHS), FIPS PUB 180-4. Gaithersburg, MD, USA: Information Technology Laboratory, 2015.
[6] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” Decentralized Business Review, 2008.
[7] M. B. Yassein, S. Aljawarneh, and E. Qawasmeh, “Securing digital forensics chain of custody using blockchain technology,” in Proc. IEEE Int. Conf. Computer Science and Engineering (CSE), 2020, pp. 145–150.
[8] K. R. Choo and D. Quick, “Digital forensic intelligence: Data analytics and blockchain integration,” *IEEE Security & Privacy*, vol. 18, no. 5, pp. 22–30, 2020.