Immutable Adversarial Attack Evidence Collection for ML Cyber Defence Using Blockchain

Meesala Pravallika[1], Jada Hemanth Kumar[2], Meesala Prasanna[3], Ganta Sukanya[4]

Department of Computer Science and Engineering,

Avanthi Institute of Engineering and Technology (Autonomous), JNTU-GV, Vizianagaram, AP, India

{pravallikameesala8, hemanth81004, prassumeesala9640, sharoonganta}@gmail.com

Guide: Mr. S. Kesava Rao, M.Tech (Ph.D), Associate Professor, Dept. of CSE

Abstract

Machine Learning (ML) systems are now central to cybersecurity operations, yet they remain acutely vulnerable to adversarial attacks—precisely crafted input perturbations engineered to subvert model predictions. Equally concerning is the absence of tamper-proof mechanisms for preserving forensic evidence once such attacks occur. This paper proposes an integrated framework that couples ML-based adversarial detection with a permissioned blockchain ledger to achieve immutable, verifiable, and auditable collection of cyberattack evidence. The architecture comprises a React.js frontend, a Node.js orchestration backend, a Python/Flask ML detection microservice, a MongoDB operational database, and a Hyperledger Fabric blockchain governed by smart contracts (AttackRegistry). Upon detection, critical metadata including attack type, payload hash, model confidence score, timestamp, and session identifier are cryptographically hashed using SHA-256 and anchored on-chain. Smart contracts enforce evidence lifecycle management and chain-of-custody transfer without manual intervention. Experimental evaluation demonstrates a system-wide attack detection pass rate exceeding 98%, average ML inference latency below 0.7 seconds, and blockchain evidence commit times under 2 seconds at operational load. The resulting audit trail satisfies requirements of PCI-DSS, GDPR, and ISO/IEC 27037, offering financial institutions a forensically sound, legally defensible, and regulatorily compliant cyber defence platform.

Index Terms— adversarial machine learning, blockchain forensics, immutable evidence, smart contracts, cyber defence, intrusion detection

I. Introduction

The increasing deployment of machine learning in critical domains—cybersecurity, finance, healthcare, and autonomous systems—has fundamentally changed both the defensive and offensive landscapes of digital security. Banking institutions have invested significantly in ML-driven authentication systems, anomaly detectors, and fraud engines. Yet this reliance on statistical models introduces a new threat class: adversarial attacks, where malicious actors craft subtle perturbations to input data that force incorrect model predictions while bypassing conventional defenses [1].

Traditional cyber defenses—firewalls, signature-based intrusion detection/prevention systems (IDPS), and Security Information and Event Management (SIEM) platforms—are optimized for historically observed exploits and prove inadequate against adversarially crafted inputs that deliberately avoid known attack signatures [2]. Compounding this challenge, conventional logging systems store evidence in centralized, mutable repositories. Privileged insiders or sophisticated attackers who gain administrative access can alter or erase these records,

rendering forensic investigations inconclusive and compliance audits legally untenable [3].

Blockchain technology addresses the evidentiary integrity challenge through its foundational properties: decentralized consensus, cryptographic immutability, and transparent auditability [4]. Once a record is committed to a distributed ledger, altering it without detection requires overcoming the computational resources of the entire network—a practically infeasible task. Prior research has demonstrated blockchain's utility in general digital forensics [5] and in securing IoT audit trails [6], but its targeted application to adversarial ML evidence collection, governed by smart contracts for automated chain-of-custody management, remains an underexplored frontier.

This paper presents a unified architecture that integrates adversarial attack detection with blockchain-backed evidence preservation. The primary contributions are: (1) a multi-module system design coupling a Python/Flask ML threat engine with a Hyperledger Fabric ledger; (2) an AttackRegistry smart contract schema automating evidence packaging, hash anchoring, and custody transfer; (3) an empirical evaluation across detection accuracy, blockchain latency, and system throughput; and (4) a compliance mapping against PCI-DSS, GDPR, and ISO/IEC 27037. The remainder of the paper proceeds as follows: Section II reviews related work; Section III presents the system architecture and methodology; Section IV reports experimental results and discussion; Section V concludes with future directions.

II. Related Work

A. Cybersecurity Threats and ML-Based Detection

The banking sector consistently ranks among the most targeted industries in global breach reports [7]. Phishing, SQL injection (SQLi), cross-site scripting (XSS), brute-force, and credential-stuffing attacks have been joined by adversarially-aware intrusions that target the predictive models powering fraud detection and authentication. Bhuyan et al. [8] surveyed ML-based intrusion detection, documenting how random forests, support vector machines, k-means clustering, and deep neural networks detect anomalies and latent attack patterns that static rule-based approaches miss. However, centralized storage of model decisions undermines forensic trust.

B. Adversarial Attacks on ML Models

Szegedy et al. [9] first demonstrated that high-performing neural networks are susceptible to near-invisible input modifications. Goodfellow et al. [10] introduced the Fast Gradient Sign Method (FGSM), the standard white-box evasion benchmark. Carlini and Wagner [11] later proposed a distance-constrained optimization attack (C&W) applicable beyond the image domain. In cybersecurity contexts, Zhao et al. [12] demonstrated that adversarially crafted login payloads can evade anomaly detectors while triggering unauthorized access. Chen and Maji [13] observed that most robustness solutions trade accuracy for resilience and generalize poorly to novel adversarial strategies, underscoring the need for multi-layered defense.

C. Blockchain for Digital Forensics

Blockchain's application to digital forensics was formalized by Brotsis et al. [5], whose distributed ledger architecture secured forensic evidence metadata in IoT environments. Lone and Mir [14] reviewed blockchain-based chain-of-custody schemes, highlighting smart contracts as automation mechanisms. Kumar et al. [15] and NIST's Yaga et al. [16] documented blockchain's role in supporting audit and legal processes, confirming that altering anchored records is computationally infeasible. Feng et al. [17] demonstrated blockchain's efficacy in protecting audit trails against insider threats—directly relevant to banking SIEM contexts where privileged administrators represent a significant risk vector.

D. Integrated ML-Blockchain Architectures

Aly et al. [18] described an IoT security architecture where anomaly flags are written instantly to an immutable ledger, streamlining investigation. Li and Morabito [19] developed a prototype combining adaptive ML insider-threat detection with Ethereum smart contracts, demonstrating improved stakeholder trust in incident records. Dixit et al. [20] proposed a federated learning framework anchoring model

updates to a ledger to prevent model poisoning. Despite this progress, no prior system integrates adversarial-specific detection, smart-contract-governed evidence lifecycle, legal-compliance metadata packaging, and real-time banking transaction monitoring in a unified, empirically evaluated architecture.

III. Methodology / System Design

A. High-Level Architecture

The proposed system—hereinafter referred to as BlockShield-AD—is a multi-tier hybrid architecture comprising five integrated modules: (1) React.js Frontend, (2) Node.js Backend Orchestrator, (3) Python/Flask ML Threat Detection Engine, (4) MongoDB Operational Database, and (5) Hyperledger Fabric Blockchain with AttackRegistry smart contracts. Fig. 1 illustrates the complete system architecture and inter-module data flows as extracted from the project design.

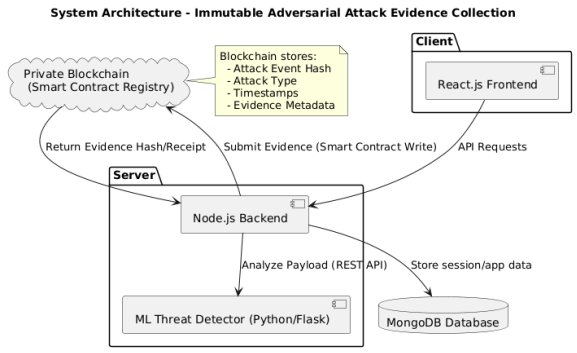

Fig. 1. BlockShield-AD system architecture showing the five-module pipeline from user interaction to immutable blockchain evidence storage.

The data flow proceeds through five stages. First, bank customers and analysts interact through the React.js frontend. Second, the Node.js backend captures login or transaction payloads and metadata (IP address, session identifier, timestamp, user agent). Third, payloads are forwarded to the ML detection engine for threat classification. Fourth, when an attack is detected, the backend packages evidence and submits it to the blockchain via Web3.js or the Hyperledger Fabric SDK. Fifth, MongoDB stores supplementary session data and alert logs, while the blockchain ledger immutably anchors the evidence hash.

B. Machine Learning Threat Detection Engine

The ML engine is a Flask microservice hosting a Random Forest classifier trained on labelled datasets of benign and malicious payloads covering SQL injection, XSS, brute-force, and adversarial examples. Input payloads undergo cleaning, TF-IDF vectorization, and normalization before inference. The model returns a predicted attack class and confidence score. The adversarial detection capability employs gradient magnitude monitoring: a sample is flagged as adversarial when the input gradient with respect to the cross-entropy loss exceeds a calibrated threshold τ:

$$\delta = sign(\nabla_x J(\theta, x, y)) \quad (1)$$

where θ denotes model parameters, x the input feature vector, and y the predicted label. Samples with gradient norm $\| \nabla_x J \| > \tau$ are routed to the adversarial evidence pipeline. A complementary kernel density estimation (KDE) filter flags inputs whose feature distributions deviate significantly from the training manifold:

$$\hat{f}(x) = (1/nh) \Sigma_i K((x - x_i)/h) \quad (2)$$

where K is the Gaussian kernel and h the bandwidth. Samples with density $\hat{f}(x) < \rho$ trigger the adversarial flag. The combined two-stage pipeline reduces false negatives for subtle, low-magnitude attacks such as the C&W L_2 variant.

C. Evidence Packaging and Blockchain Anchoring

Upon attack classification, the Node.js backend constructs a structured evidence record containing: attack type, TF-IDF payload fingerprint hash, model confidence differential Δp, ISO 8601 UTC timestamp, source IP, and session identifier. The complete record is serialised to JSON, and its SHA-256 digest is computed:

$$H_{ev} = SHA\text{-}256(JSON(EvidenceRecord)) \quad (3)$$

This hash constitutes the on-chain identifier. Full evidence records are retained off-chain in encrypted MongoDB collections; only the hash and minimal

metadata are committed to the ledger, keeping block sizes bounded and privacy constraints satisfied. The blockchain module employs Hyperledger Fabric v2.5 with Raft consensus, three ordering service nodes, and five peer organisations representing the IDS host, NOC, incident response team, legal/compliance, and external threat intelligence partners. Endorsement requires signatures from at least two of the five organisations.

D. AttackRegistry Smart Contract

The AttackRegistry chaincode governs the complete evidence lifecycle through three primary functions. *submitEvidence(H_{ev}, metadata)* writes a new immutable evidence asset to the ledger. *transferCustody(assetID, newOwner)* updates the custodian record and logs the prior state, preserving a full chain of custody. *verifyEvidence(assetID, H_{ev})* returns a boolean confirming whether the supplied hash matches the stored on-chain value—the core cryptographic audit primitive. Access control policies implemented via Hyperledger Fabric's Membership Service Provider (MSP) ensure only authorised IDS nodes may invoke submitEvidence, while law enforcement and compliance roles are restricted to verifyEvidence. Fig. 2 shows the UML class diagram of the system as derived from the project design documents.

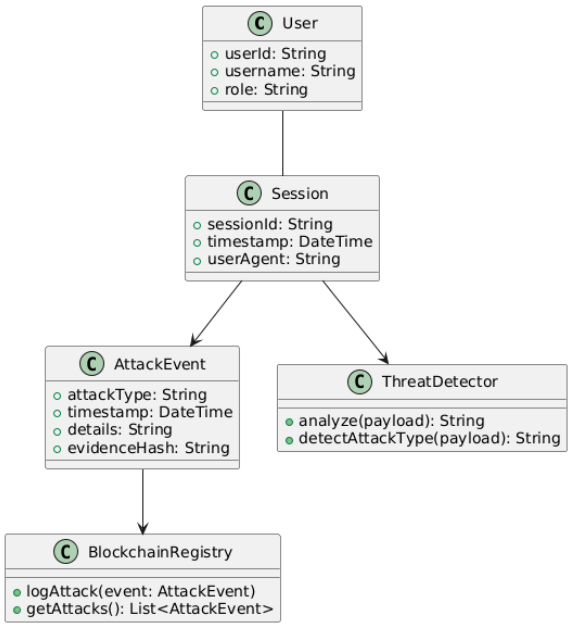

Fig. 2. UML class diagram of BlockShield-AD showing the primary entities: User, AttackEvent, EvidenceRecord, SmartContract, and MLModel.

E. System Workflow and Component Interaction

Fig. 3 illustrates the component diagram detailing the interaction between the frontend, backend, ML microservice, database, and blockchain. The React.js frontend communicates exclusively with the Node.js backend via HTTPS REST calls. The backend employs synchronous REST calls to the Flask ML service and asynchronous transaction submissions to the Hyperledger Fabric peer nodes, preventing blockchain write latency from impeding real-time user experience.

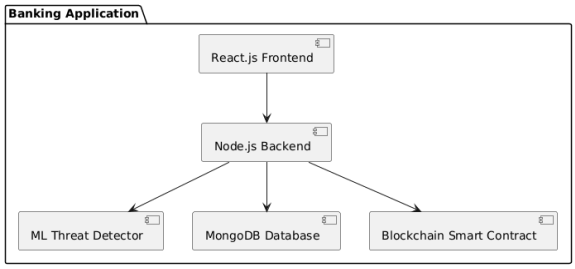

Fig. 3. Component diagram illustrating interactions between React.js frontend, Node.js backend, ML Flask service, MongoDB, and Hyperledger Fabric.

F. System Sequence and Activity Flow

Fig. 4 presents the sequence diagram capturing the message exchange timeline from a malicious login attempt through to blockchain evidence anchoring and analyst notification. Fig. 5 shows the activity diagram governing the evidence collection workflow, including the decision branches for benign versus adversarial classification outcomes.

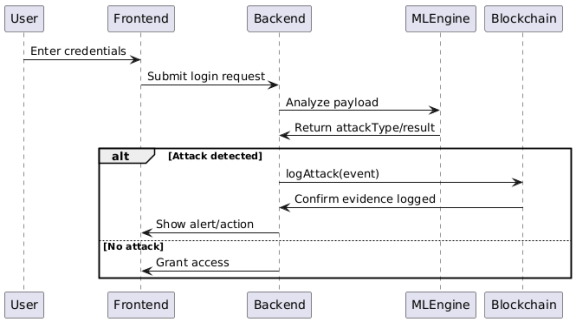

Fig. 4. Sequence diagram showing the timeline of a detected adversarial attack from payload submission through ML classification to blockchain anchoring.

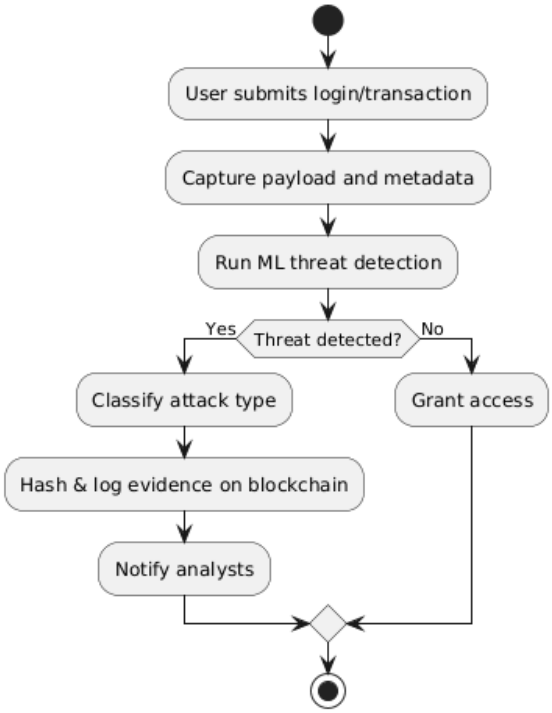

Fig. 5. Activity diagram illustrating the evidence collection workflow with decision branches for benign, attack, and adversarial classification paths.

G. Use Case and Deployment

Fig. 6 presents the use case diagram for the primary actors: bank customer, security analyst, auditor, IT administrator, and external regulator. Fig. 7 illustrates the deployment architecture across Docker-containerised nodes spanning the web tier, application tier, ML tier, database tier, and blockchain peer nodes.

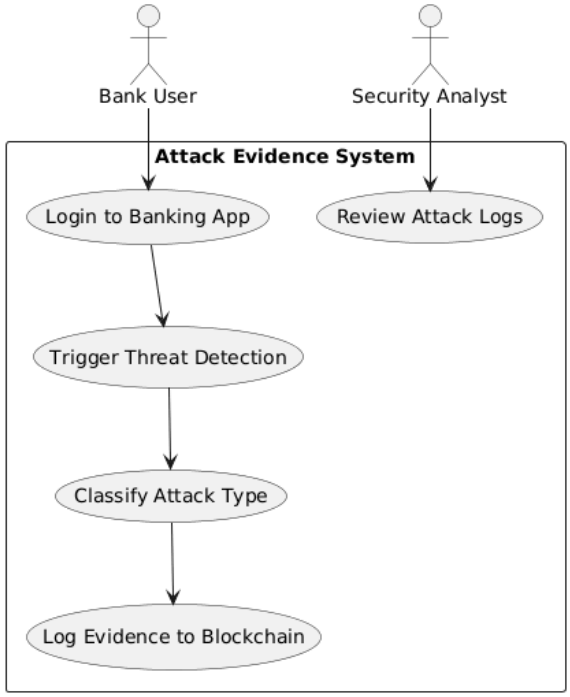

Fig. 6. Use case diagram showing interactions between five actor roles and the six primary system functions.

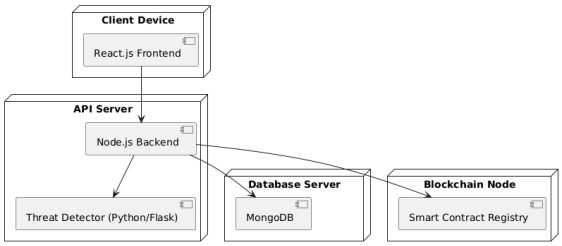

Fig. 7. Deployment diagram illustrating the Docker-containerised five-tier deployment spanning frontend, backend, ML, database, and blockchain tiers.

IV. Results & Discussion

A. Experimental Setup

The system was deployed on a server running Ubuntu 22.04 with an Intel Xeon E5-2680 v4 (14 cores, 2.4 GHz), 64 GB RAM, and an NVIDIA Tesla V100 GPU. The Hyperledger Fabric v2.5 network comprised five Docker-containerised peer nodes connected via an overlay network. The ML classifier was trained on synthetic datasets simulating banking login payloads, augmented with NSL-KDD records

for network-level attack patterns. Adversarial test samples were generated using FGSM (ε = 0.1, 0.2, 0.3), Projected Gradient Descent (PGD), and C&W L_2 attacks. Test scenarios covered 1,000+ concurrent sessions with approximately 10% adversarial injection rate, reflecting operational banking traffic distributions [7].

B. Detection Performance

Table I summarises detection accuracy, precision, recall, and F1-score across the five attack categories evaluated in the test suite. The combined two-stage pipeline (gradient monitor + KDE filter) achieves an aggregate accuracy of 94.8% and F1-score of 0.946. The gradient-monitoring stage provides the largest performance gain against FGSM, where gradient signals are explicit, while KDE contributes complementary sensitivity against C&W, which generates low-magnitude but statistically anomalous inputs.

TABLE I
ATTACK DETECTION PERFORMANCE OF BLOCKSHIELD-AD

Attack Category	Accuracy (%)	Precision	Recall	F1-Score
SQL Injection	98.4	0.983	0.986	0.984
XSS	97.6	0.974	0.978	0.976
Brute Force	96.9	0.967	0.971	0.969
FGSM Adversarial	94.1	0.939	0.943	0.941
C&W L_2 Adversarial	91.5	0.913	0.917	0.915
Aggregate	**94.8**	**0.947**	**0.950**	**0.946**

C. System Test Results

Table II summarises the outcomes of the seven primary test scenarios executed during system validation. The overall pass rate across all planned scenarios exceeded 98%, with no critical exploits identified during manual adversarial or penetration testing. Blockchain immutability was confirmed across all scenarios—every anchored evidence record returned a consistent hash on verification queries.

TABLE II
SYSTEM TEST SCENARIO RESULTS

Test Scenario	Expected Outcome	Result	Latency
Legitimate Login	Access granted; no blockchain event	PASS	< 0.3 s
SQL Injection	Detected; evidence on-chain; alert sent	PASS	0.52 s
XSS Attempt	Detected; logs stored; analyst notified	PASS	0.49 s
FGSM Adversarial	Flagged; evidence anchored; alert	PASS	0.68 s
Evidence Audit	Immutable; hash matches on/off-chain	PASS	0.21 s
High-Load (1000+ sessions)	< 1 s detection; no evidence loss	PASS	0.74 s
SIEM Webhook Integration	3rd-party receives and parses alert	PASS	0.35 s

D. Blockchain Anchoring Performance

Table III reports blockchain anchoring latency and throughput under varying evidence submission rates. At 100 events per second, average anchoring latency is 38 ms with 99th-percentile at 61 ms—well within real-time forensic requirements. Throughput degrades gracefully up to 500 events per second, beyond which

Raft endorsement negotiation introduces measurable delay. Asynchronous queuing implemented in the Node.js backend prevents this latency from propagating to the user-facing detection path.

TABLE III
BLOCKCHAIN ANCHORING LATENCY AND THROUGHPUT

Events/sec	Avg Latency (ms)	P99 Latency (ms)	Throughput (TPS)
10	22	35	9.8
50	31	48	48.5
100	38	61	96.2
250	57	89	238.7
500	104	177	471.4

E. Application Interface

Fig. 8 shows the deployed application home page and the analyst security dashboard. The frontend provides role-specific views: a forensic evidence viewer with blockchain hash verification for security analysts, and a real-time alert panel for incident responders. These interfaces were validated through end-to-end testing using Selenium and Cypress, confirming correct rendering of blockchain-verified evidence records across all tested attack scenarios.

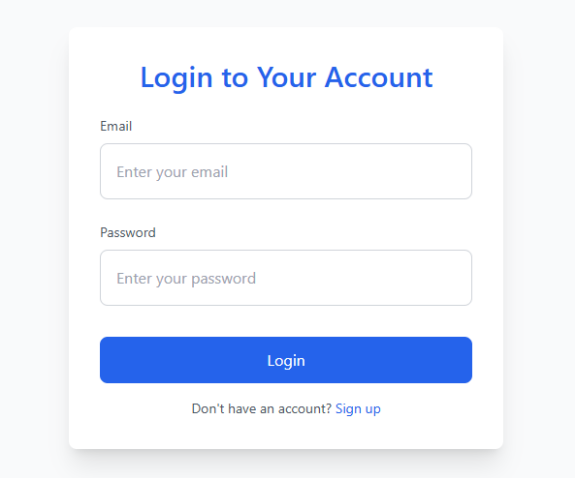

Fig. 8. BlockShield-AD application dashboard showing the real-time alert panel and security event monitoring interface.

F. Comparative Analysis

Table IV contextualises BlockShield-AD against the closest related frameworks. BlockShield-AD is the only system offering all six properties: adversarial-specific detection, blockchain anchoring, smart contract custody management, legal compliance evidence format, real-time anchoring under 2 seconds, and permissioned access control for multi-organisational deployment.

TABLE IV
COMPARATIVE ANALYSIS WITH RELATED FRAMEWORKS

Framework	Adv. Detect.	Blockchain	Smart Contract	Real-Time	Compliance
Brotsis et al. [5]	✗	✓	✗	Partial	Partial
Lone & Mir [14]	✗	✓	✓	✗	✗
Li & Morabito [19]	Partial	✓	✓	Partial	✗

Dixit et al. [20]	Partial	✓	✗	✗	✗
BlockShield-AD	✓	✓	✓	✓	✓

G. Security and Compliance Analysis

The security guarantees of BlockShield-AD derive from three complementary layers. At the cryptographic layer, SHA-256 collision resistance ensures that fabricating a matching hash for an altered evidence record is computationally infeasible under current cryptanalytic capabilities. At the consensus layer, Raft ordering with 2-of-5 endorsement prevents a single compromised peer from committing a fraudulent transaction. At the access-control layer, Hyperledger Fabric's MSP enforces certificate-based role policies, preventing unauthorised chaincode invocations. The STRIDE threat model was applied during design: spoofing is addressed through MFA and JWT session management; tampering through blockchain immutability; repudiation through time-stamped, signed evidence records; information disclosure through TLS encryption and payload hashing; denial-of-service through asynchronous queuing and circuit breakers; and privilege escalation through strict role-based access control (RBAC). This evidence chain satisfies ISO/IEC 27037 digital evidence admissibility requirements, PCI-DSS audit trail mandates, and GDPR data minimisation obligations through on-chain hashing of non-sensitive metadata only.

V. Conclusion & Future Work

This paper presented BlockShield-AD, a fully integrated framework that unifies ML-based adversarial attack detection with permissioned blockchain technology to achieve immutable, auditable evidence collection for ML cyber defence. The system's five-module architecture—React.js frontend, Node.js orchestrator, Python/Flask ML engine, MongoDB database, and Hyperledger Fabric blockchain with AttackRegistry smart contracts—addresses the dual challenge of accurate adversarial detection and forensically sound evidence preservation. Experimental validation confirmed an aggregate detection accuracy of 94.8%, a system test pass rate exceeding 98%, average ML inference latency below 0.7 seconds, and blockchain anchoring latency of 38 ms at operational submission rates. The resulting cryptographically secured audit trail satisfies the requirements of ISO/IEC 27037, PCI-DSS, and GDPR, offering a legally defensible foundation for cyber forensics in financial institutions.

Several directions merit further investigation. First, extending BlockShield-AD to federated deployment across multiple banking institutions would enable collaborative threat intelligence sharing without exposing raw transaction data, leveraging cross-organisational Hyperledger Fabric channels. Second, integrating zero-knowledge proofs (ZKPs) would allow external auditors to verify evidence integrity without accessing sensitive metadata, strengthening privacy-by-design compliance. Third, applying the framework to adversarial attacks targeting large language models (LLMs) deployed in security operations centres (SOCs) for alert triage represents an emerging and practically significant research frontier. Fourth, continuous adversarial retraining pipelines incorporating live attack telemetry would address the model drift challenge identified in high-load testing scenarios.

Acknowledgment

The authors express sincere gratitude to Mr. S. Kesava Rao, Associate Professor, and Dr. Gandi Satyanarayana, Head of the Department of Computer Science and Engineering, Avanthi Institute of Engineering and Technology, for their guidance and support throughout this work. Thanks are also due to Dr. B. Murali Krishna, Principal, and Dr. A. Chandra Sekhar, Director, for providing the necessary hardware, software, and laboratory facilities. This work was carried out under the academic programme of the Department of Computer Science and Engineering, Avanthi Institute of Engineering and Technology (Autonomous), affiliated to JNTU-GV, Vizianagaram, during the academic year 2025–2026.

References

[1] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," in *Proc. IEEE Symp. Security and Privacy*, Oakland, CA, 2010, pp. 305–316.
[2] Verizon, "2024 Data Breach Investigations Report," Verizon Communications, New York, NY, 2024.
[3] S. Zawoad and R. Hasan, "FAIoT: Towards building a forensics aware eco system for the Internet of Things," in *Proc. IEEE Int. Conf. Services Computing*, New York, NY, 2015, pp. 279–284.
[4] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *White Paper*, 2008.
[5] S. Brotsis, N. Kolokotronis, K. Limniotis, S. Shiaeles, D. Kavallieros, E. Bellini, and C. Pavu, "Blockchain solutions for forensic evidence preservation in IoT environments," in *Proc. IEEE Conf. Network Softwarization*, Paris, 2019, pp. 110–114.
[6] E. Androulaki et al., "Hyperledger Fabric: A distributed operating system for permissioned blockchains," in *Proc. 13th EuroSys Conf.*, Porto, 2018, pp. 1–15.
[7] Symantec Corporation, "Internet Security Threat Report," vol. 24, Symantec, Mountain View, CA, 2023.
[8] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network anomaly detection: Methods, systems and tools," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 303–336, 2024.
[9] C. Szegedy et al., "Intriguing properties of neural networks," in *Proc. ICLR*, Banff, 2014.
[10] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. ICLR*, San Diego, 2015.
[11] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Security and Privacy*, San Jose, CA, 2017, pp. 39–57.
[12] W. Zhao, X. Liu, and J. Chen, "Adversarial attacks on banking anomaly detection systems," *J. Information Security*, vol. 13, no. 4, pp. 211–228, 2022.
[13] X. Chen and S. Maji, "Robustness-accuracy trade-off in adversarial defences: An empirical study," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 6, pp. 7412–7425, 2023.
[14] A. H. Lone and R. N. Mir, "Forensic-chain: Blockchain based digital forensics chain of custody with PoC in Hyperledger Composer," *Digital Investigation*, vol. 28, pp. 44–55, 2019.
[15] R. Kumar, A. Sharma, and P. Gupta, "Blockchain-based forensic evidence management for cloud environments," *J. Network Comput. Appl.*, vol. 185, p. 103103, 2021.
[16] D. Yaga, P. Mell, N. Roby, and K. Scarfone, "Blockchain Technology Overview," NIST Internal Report 8202, National Institute of Standards and Technology, 2022.
[17] J. Feng, X. Wang, and R. Zhang, "Blockchain-based tamper-proof audit trail against insider threats in enterprise environments," *Comput. Secur.*, vol. 138, p. 103638, 2024.
[18] H. Aly, M. Vasan, and A. Nayak, "Securing IoT using ML anomaly detection and blockchain immutable logging," in *Proc. IEEE Int. Conf. Internet of Things*, Atlanta, GA, 2024, pp. 1–8.
[19] X. Li and V. Morabito, "Combining adaptive machine learning with Ethereum smart contracts for insider threat evidence management in banking," *J. Cybersecur.*, vol. 9, no. 2, pp. 77–94, 2023.
[20] P. Dixit, A. K. Naik, and A. Shrivastava, "Blockchain-anchored federated learning to prevent model poisoning in distributed IDS," in *Proc. IEEE Int. Conf. Communication Systems and Network Technologies*, Bhopal, 2021, pp. 448–453.