Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 5, 2026

ZTFORENSICS: ZERO TRUST POLICY ENFORCEMENT WITH TAMPER-EVIDENT FORENSIC EVIDENCE PACKAGING FOR HYBRID CLOUD ENVIRONMENTS

[*1]Ali Shan, [2]Ali Sufyan, [3]Muhammad Ahmad, [4]Hassaan Iqbal
[1-4]Department of Information and Communication Engineering, The Islamia University of Bahawalpur, Punjab, Pakistan
[*1]alishan.cyber@gmail.com

DOI:-

Keywords
Digital Forensics, Cloud Security, Hash Chain, Open Policy Agent, Forensic Readiness, Evidence Integrity, Hybrid Cloud, Access Control, Tamper-Evident Logging.

Article History
Received: 06 March 2026
Accepted: 09 April 2026
Published: 11 May 2026

Corresponding Author: * Ali Shan

Abstract
Hybrid cloud environment reveals a serious lack in security: perimeter-based models of access control provide one-shot authentication, with no continuous verification, nor evidence collection that is tamper-proof. If valid credentials are gained by an adversary, they can exfiltrate data, and then delete or manipulate audit logs, which can interfere with forensic accountability. Current Zero Trust Architecture (ZTA) solutions focus on enforcement and don't necessarily tie every access decision to a legally admissible, cryptographically bound forensic record. This paper introduces an integrated framework, called ZTForensics, which enforces Zero Trust policy decisions in real time, and generates hash-chained forensic evidence records for every access event using the SHA-256 hash function. Enforcement of the framework is done by a FastAPI gateway, Open Policy Agent (OPA) with Rego policies, Keycloak identity management, PostgreSQL evidence storage and MinIO object packaging. Seven contextual risk factors are applied to each request, and the outcome of the decision (allow, deny, or challenge) is recorded as an integrity-linked forensic record. Long-term evidence integrity is provided by RSA-signed anchors and a chain-verification endpoint. In a simulated banking environment, correct decisions are validated, tamper detection on several attack vectors is ensured with no false positives and the evidence bundle export is structured according to legal rules. The challenges of reactive forensic collection and proactive decision-making are bridged by ZTForensics, creating trustworthy forensic evidence within cloud-based application programming interfaces in cloud API environments.

I. Introduction

The fast transfer of enterprise services to the hybrid cloud platforms has fundamentally changed the threat landscape [1]. Conventional trust-but-verify security frameworks verify user identities at the network edge and trust that internal packet traffic is securely handled by default [2]. This assumption has been proven false: a malicious actor with valid credentials will look like a legitimate user and can access cloud APIs, steal sensitive information, and even alter or delete audit logs to hamper the investigation process [3].

With the move by organizations towards using microservices and distributed data lakes, conventional boundary-focused defense mechanisms have become inadequate. Furthermore, the rapid expansion of API endpoints generates a vast attack surface where static perimeter protection becomes ineffective. In case of a breach, audit logs prove important to incident responders in order to rebuild the attack vector. Nonetheless, when such logs are stored in unstable layers, and are not cryptographically bound, advanced attackers regu- larly overwrite or modify them, implementing an anti-forensic policy that invalidates the legal admissibility and recovery of such logs after the incident.

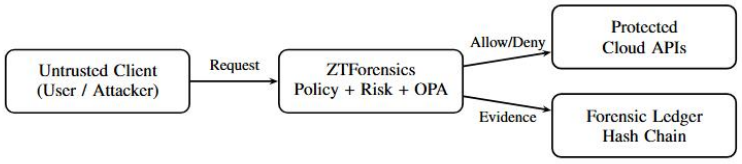

Fig. 1. Conceptual overview of ZTForensics

The challenge of cloud forensics is twofold [4]. First, the traditional method of logging is largely reactive -the evidence is gathered only after a compromise has been identified, and at this point, the critical artifacts can be corrupted or erased by the time the evidence is gathered and analyzed [5]. Second, integrity of logs is quite often not assured even in the presence of logs; without cryptographic binding logs can be easily altered and can be inadmissible in court cases [6]. With no reliable, tamper evident audit trail, organizations cannot answer the underlying forensic question: who did what, when, where and why? [7]

ZTA prevents implicit trust by ensuring that all requests are verified regardless of network origin, leading to a significant decrease in the blast radius of compromised credentials by enforcing continuous validation of all requests, no matter their network origin [8]. Nevertheless, the current implementations of ZTA are more concerned with access enforcement, they lack a systematic approach, of tying each enforcement decision with a forensically admissible, integrity-guaranteed evidence history [9].This, as shown in Fig.1, brings about a pressing necessity of a proactive paradigm shift. This paper presents ZTForensics, a framework that combines ZTA enforcement policies with proactive forensic evidence packaging. Our principal contributions are:

- Designing an integrated architecture that links access control decisions to forensic records at decision time, eliminating the post-incident evidence gap. Proposing a lightweight SHA-256 hash chain mechanism that prior approaches without the associated computational overhead [10].
- Implementing a multi-factor contextual risk engine that evaluates risk signals per request and supports dynamic allow/deny/challenge decisioning via OPA Rego policies.
- Providing RSA-signed daily anchor points and an ex-portable evidence bundle (ZIP) format to satisfy audit, compliance, and legal admissibility requirements.
- Performing experimental validation in a banking domain scenario demonstrating decision distribution correctness, hash chain integrity under tampering, and anchor signature verification.

II. Literature Review

A. Foundations of the Zero Trust Paradigm

Initiated by John Kindervag and officially standardized via the NIST SP 800-207 guidelines, the Zero Trust model eliminates the traditional concept of a safe internal network. It mandates

strict, continuous verification for every single access request, assuming absolutely no inherent reliability based on a user's physical or logical placement. [18]. The subsequent work has generalized ZTA. Hong et al [19] introduced a programmable data flow telemetry-based ZTA structure, SysFlow, at the system level. The model of access control based on the trust-score with sensitivity analysis presented by Jeong and Yang cite works on the real-life deployment of the model designed by them [20]. Cao et al. [21] investigated orchestrating surveyed AI-based automation or orchestration methods which revealed important bottlenecks in dynamic policy management.

A comprehensive multivocal review regarding the real- world deployment of ZTA was conducted by Itodo and Ozer [22], but Ghasemshirazi et al. [23] made a list of ZTA applications, challenges, and open opportunities. The changing environment towards dynamism demands constant monitoring because the isolated implementations deteriorate security posture with time. Recent research by Dhumal et al.

[24] suggested ZTA to secure the cloud-based endpoint of critical infrastructure, focusing on the posture of the device and Phiayura and Teerakanok [25] proposed a comprehensive migration framework for transitioning enterprise systems to Zero Trust Architecture. Additionally, Alim et al. [26] analyzed ZTA deployments across multi-cloud infrastructures, highlighting significant challenges related to scalability and latency are apparent when implementing policies in distributed boundaries.

B. Cloud and Digital Forensics

Multi-tenancy, data volatility, jurisdictional ambiguity and lack of access to infrastructure to the investigator complicate digital investigations in the cloud. Because cloud service vendors remove hardware layers, the traditional block-level acquisition of **forensic** is virtually infeasible. Alenezi et al. tested the cloud forensic readiness frameworks with the help of expert reviews, determining the starting requirements of proactive evidence gathering before an incident. Kebande and Venter [28] suggested an advanced forensic readiness framework for the cloud that integrates event reconstruction techniques. Wu et al. [11] proposed CETS, a cloud evidence tracing architecture designed for extensive public cloud platforms; CETS is a better system than the existing one because it produces better results on post-incident collection, but since it is a passive system, the evidence may be lost before the retrieval. Alnajjar et al. [12] suggested a two-level forensic preparedness framework of Industry 4.0, but with a strong emphasis on the IoT edge nodes, Santillan-Lima et al. [29] performed a systematic review of the evidence preservation risks and legal admissibility criteria in cloud computing In another study, MohanRaj [30] examined the encryption approaches to cloud systems digital forensic preparedness and found that there is a necessity to have secure and isolated evidence vaults.

C. Tamper-Evident Logging and Evidence Integrity The integrity of forensic logs has been an ongoing problem and this problem increases as much as the number of transactions. Zhao et al. [13] introduced Nitro, a high performance audit logging system, implemented in eBPF, which can produce provable tamper evidence, while not losing data, but it would need deep access to the kernel, which is not usually available in Serverless cloud functions.

[27] Numerous blockchain-based approaches have been explored to address trust deficits in forensic logging: Hanif [14] showed how blockchain could be used for verify integrity of the custody chain in digital forensics; Miller and Singh[31] applied blockchain-based evidence integrity verification; Adewumi [32] integrated AI with Hyperledger Fabric for automated chain-of-custody management; and Pourvahab and Ekbatanifard [33] proposed a blockchain/SDN architecture for IaaS cloud forensics. AlKhanafseh and Surakhi [15] combined LSTM-based steganography with blockchain for evidence preservation. However, the consensus overhead of Byzantine Fault Tolerant protocols makes blockchain entirely unsuitable for inline API authorization. ZTForensics deliberately avoids blockchain overhead, instead employing lightweight SHA-256 hash chain, which provides equivalent tamper detectability with a fraction of the deployment cost.

D. Zero Trust and Forensics Integration

Neale et al. [16] conceptualized the argument of

Zero Trust Digital Forensics as they believe that removing implicit trust in the investigation of the forensics automatically reduces the risk of tampering with evidence. Inukonda et al. [17] suggested ZETA, which is a conceptual system of Zero Trust and forensic preparedness of perimeter-less networks but ZETA has not been completely implemented in the form of a prototype and empirically tested. Both works are further advanced by ZTForensics who provides a working, testable implementation with gateway enforcement, policy decision making using Open Policy Agent and verifying cryptographic evidence.

Table I: ***Comparative Literature Review***

Ref	Approach	Key Limitation	Mechanism	Our Contribution
[11]	Post-incident evidence tracing	Reactive; evidence may be lost	Basic Hash	Decision-time proactive evidence creation
[12]	Two-tier forensic readiness	Domain-specific (IoT/Industry 4.0)	Standard Encryption	Domain-agnostic hybrid cloud API focus
[13]	High-performance eBPF auditing	Dependent on OS kernel support	Provable Audits	Application-layer API gateway enforcement
[14]	Blockchain Chain of Custody	Consensus overhead latency	Distributed Ledger	Edge-level hash chain without consensus
[15]	Blockchain + Steganography	High deployment complexity	LSTM + Blockchain	Lightweight SHA-256 hash chain integrity
[16]	Conceptual ZT forensics	No implementation artifacts	Conceptual Framework	Deployable architecture with measurable outputs
[17]	ZETA perimeter-less readiness	No working operational prototype	Conceptual Framework	Full implementation with enforcement & verification

E. Access Control, Authentication and Risk Scoring

Mostafa et al. [34] came up with a multi-factor, multi-layer authentication system with built-in intrusion detection to the cloud environment. In their article, Mahto and Singh [35] showed how reverse proxies and mTLS can be used to enforce access control across trust boundaries. Arif et al. [36] surveyed various cloud-native security techniques focused on enhancing privacy and establishing trust. Furthermore, Ali et al. [37] systematically reviewed the specific privacy and security challenges inherent to hybrid and multi-cloud configurations and revealed lack of consistency in policy implementation as the main threat. This is taken care of in ZTForensics whereby all policy decision making is centralized in one Open Policy Agent, which minimizes policy drift. Table I is a summary of the differences between ZTForensics and the work that is the most similar. Compared to traditional logging methods, ZTForensics is much more integrity and tamper-resistant and forensic-ready as shown in a concise manner in Fig. 2.

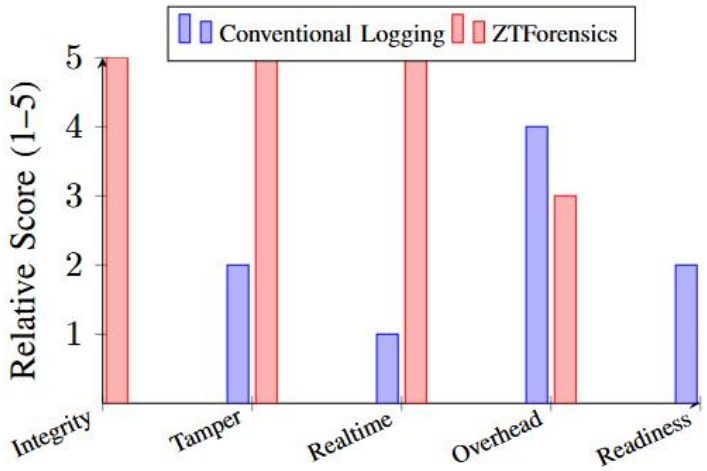

Fig. 2. Qualitative comparison of conventional logging versus ZTForensics across key security and forensic readiness dimensions.

III. System Architecture

ZTForensics is a multi-layered and microservice-based architecture that is implemented using Docker Compose. The entire system flow is shown in Fig.3 . It has six main layers in the architecture, which is designed to thwart the lateral flows of threats, and to provide strict logging. (1)Client/Dashboard Layer. End-users will get to interrelate through a browser-based portal. A Flask dashboard is used to monitor security in real-time and a forensic audit view, which is visible to authorized investigators. This is a decoupled presentation layer that makes sure that the clients do not have access to underlying forensic databases. (2)Identity Layer (Keycloak + JWT). Keycloak is the identity provider with OpenID Connect (OIDC). Keycloak JWTs include user identity and role statements, signed with the RS256 algorithm. FastAPI gateway authenticates token signature, issuer and role attributes on each individual request, adhering strictly to the principle of Zero Trust Architecture that identities have to be explicitly verified in each transaction. The window of opportunity of token replay attacks is drastically reduced by a token revocation list and a short lifetime (e.g. 5 minutes), (3)FastAPI Enforcement Gateway. The central Policy Enforcement Point is a centralized reverse proxy. All inbound requests are captured and broken down into a formatted JSON context triple: (subject, action, resource, metadata). This is a tuple, which is sent one by one to the risk engine and Open Policy Agent. This layer is the only way to secure resources, and it totally removes proximity vulnerabilities of trust, as found in the legacy hybrid clouds environment [38], [39]. Any request that bypasses this gateway is extremely limited in Docker overlay network, (4)Risk Scoring Engine. A multi-factor, inline risk scorer evaluates seven contextual signals per request in real time Table II. It calculates a normalized composite risk score using a weighted sum of signal severities. The resulting score is forwarded to Open Policy Agent as part of the decision context,(5) Open Policy Agent. Policies are authored in Rego, a declarative query language designed for fine-grained authorization. Policies express three potential decision outcomes: allow, deny, or challenge. Open Policy Agent evaluates the incoming context and risk score against the active policy set and returns a structured decision object [38], (6)Evidence and Storage Layer. The structured forensic records based on the decision of the Open Policy Agent are stored in PostgreSQL, binary evidence bundles and anchor artifacts are stored in MinIO. All forensic records are linked together with each other using SHA-256 as defined in IV.

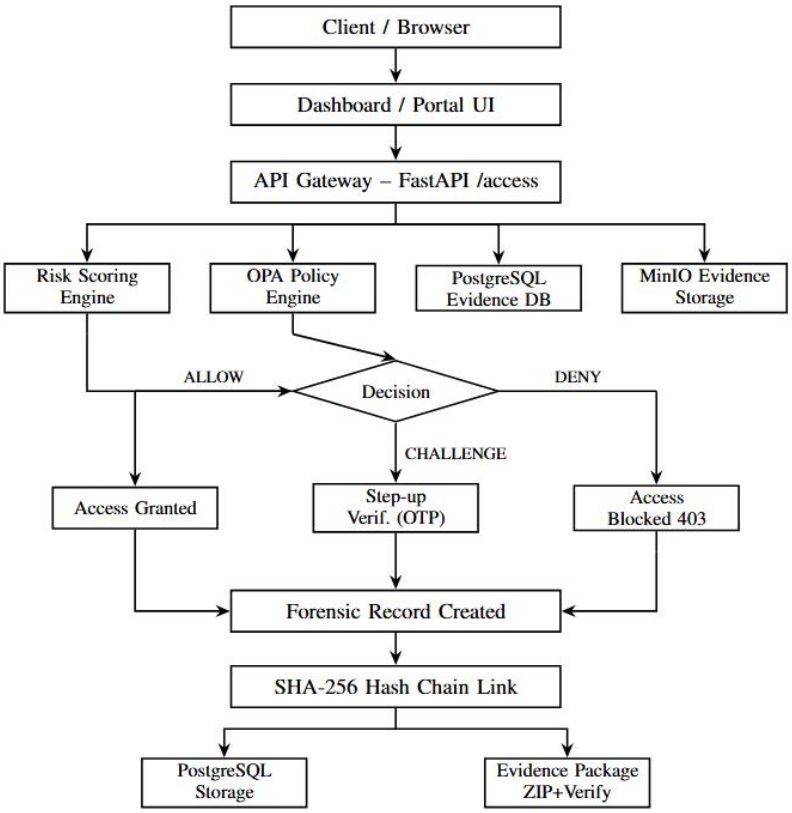

Fig. 3. ZTForensics System Architecture

IV. Implementation Details

A.Technology Stack

ZTForensics itself is implemented solely using strong open- source components: FastAPI (Python 3.11) as the high-performance enforcement gateway; OIDC/JWT identity brokering with Keycloak 26.1.4; Open Policy Agent v0.68.0 as the detached authorization engine to execute Rego policies; the audit dashboard with Flask; PostgreSQL 16.4 for structured, relational forensic storage; MinIO for S3-compatible object storage; and Docker Compose for multi-service orchestration. All internal services only interact on an isolated, internal Docker network overlay, with only the gateway and dashboard open to the outside world.

Table II: ***Multi-Factor Risk Signals Evaluated per Request***

#	Signal	Description
1	IP Reputation	Known malicious or anomalous source IP
2	User-Agent	Suspicious or automated client fingerprint
3	Time Anomaly	Access outside normal operating hours
4	Geolocation	Unexpected geographic origin
5	Action Sensitivity	Criticality class of requested operation
6	Resource Sensitivity	Classification of targeted resource
7	Failed Attempts	Repeated authentication or access failures in session

B. Cryptographic Hash Chain Schema

The structured forensic record is generated by each access decision. In order to ensure immutability the system forms an internal hash chain when it is created. The schema has: timestamp, subject identity, subject role, requested action, targeted resource, the seven values of risk input all evaluated, the final composite risk score, the explicit Open

Policy Agent decision, the headers of the HTTP request (origin IP, user-agent, headers), a cryptographic pointer previous_hash, and the newly computed current hash value. The hash computation is performed as follows:

$$Hn = SHA\text{-}256 (serialize(Rn) || Hn-1) \quad (1)$$

where Rn is the complete, canonicalized JSON record payload of decision n, Hn−1 is the hash of the immediately preceding record in the database sequence, and denotes direct string concatenation. The genesis record (n = 0) uses a securely generated, fixed initialization vector (IV) as H−1.

C. Tamper Verification Endpoint

Authorized auditors have a special cryptographic verification endpoint, called /forensics/verify-chain. Upon invocation, the endpoint reads all the entries in the log sequence in PostgreSQL, computes all the digests iteratively starting with the genesis block by using Eq.(1), and strictly compares the calculated digests to the stored values. When integrity confirmation is made on an unbroken integrity body, a reply of endpoint is: {"status": "valid", "records_checked": N"}; when a broken cryptographic link (mismatched hash) is detected, the endpoint stops, and instantly responds with the exact primary key identifier of the first broken record. This actual isolation enables forensic investigators to identify the exact millisecond, when an attacker tried to alter the audit trail [40].

D. RSA Anchor Signing for Daily Evidence Binding

Although hash chain ensure that it is impossible to modify the records without detection, they can be truncated (all records deleted) entirely (e.g. the last X records). In order to counteract this, an anchored current full log set is set to a scheduled daily cron job. The system then uses the PKCS#1 v1.5 padding to sign a 2048-bit RSA private key with a canonicalized Merkle root equivalent (a concatenated hash of all records generated that day), and stores the root hash and the DER-encoded signature in a safe place in MinIO. Independent verification just repeats the computation of the root hash over the batch of the day, authenticates the RSA signature using the matching public key and verifies that the anchor is the same as the database state. This gives out-of-band integrity ensuring that the logs are present and in order on that date and time, cryptographically.

E. Evidence Bundle Export Architecture

To obtain the technical logging and legal requirements in line, the system is characterized with a special export module. It packs a gated ZIP package containing only six critical artifacts: evidence.json and evidence.csv (with all the structured forensic records to be parsable by various tools), chain_verify.json (the automated hash chain verification report), anchor_verify.json (the RSA anchor verification report), summary.json (aggregated session metadata and anomaly statistics), and fundamentally, manifest.json. SHA-256 digests of all the files included in the ZIP are in the manifest, thus creating a verifiable chain of custody of the bundle itself. This strict format is directly based on ISO/IEC 27037 best-practice standards of digital evidence packaging, and is highly legally admissible.

F. Step-Up Authentication (OTP Challenge) Logic

When Open Policy Agent evaluates a borderline risk score for a high-risk transaction (e.g., an administrative API call from an unknown IP), the gateway initiates a Time-Based One-Time Password (TOTP) verification flow. The consecutive OTP failures are proactively monitored per session and feedback into the risk engine in the form of the signal, which is the 'Failed Attempts'. This feedback mechanism quickly raises the composite risk score of future requests, and eventually results in a hard denying of that request and the end of the session. The dynamic design offers the essential defence-in- depth, countering session hijacking attack even after initial authentication successfully.

G. Contextual Anomaly Detection

In addition to the risk scoring in the form of static values, inline temporal anomaly module compares every request with recent historical baselines. It detects: multiple denials from the same JWT identity, burst-rate traffic (i.e., more than k = 50 requests in a sliding window W = 10 seconds), traffic of unusual high volume on certain subnets of the IP address, and impossible-travel paths (where the geographic dispersion of two consecutive requests is physically impossible with the time elapsed). The anomalies identified by this module are typed in the forensic record payload

and prominently displayed on the real-time audit dashboard.

V. Security Analysis

A. Zero Trust Compliance and Formal Threat Model

ZTForensics fully meets the three core Zero Trust Architecture principles as outlined by NIST SP 800-207: Verify explicitly all incoming API requests are strongly authenticated through their JWT signature and assessed contextually by Open Policy Agent, irrespective of the originating network location; Use least privilege Open Policy Agent Rego policies strictly enforce role-scoped permissions based on specific API resources; and Assume breach the continuous risk engine and anomaly detector continuously monitors for compromise indicators originating from already authenticated internal sessions.

In order to formally certify the strength of the architecture, we apply the STRIDE threat modeling technique. We are assuming an adversary Dolev-Yao that has the capability to intercept, alter and inject messages into the network and have compromised valid user credentials.

Table III: *STRIDE Threat Model and Mitigation Strategies*

STRIDE Threat	Adversary Action	ZTForensics Mitigation
Spoofing	Attacker captures JWT and replays identity.	Continuous Contextual Risk Engine & OTP Step-up
Tampering	Deletion or alteration of PostgreSQL database.	SHA-256 Cryptographic Hash Chain
Repudiation	Malicious admin denies unauthorized activity.	Immutable, Signed Ledger blocks repudiation
Information Disclosure	Network sniffing of API context.	Forced TLS 1.3 & Keycloak Identity Masking
Denial of Service	API gateway flooding.	Rate-limiting inside Risk Engine evaluation
Elevation of Privilege	Modifying roles in DB.	Open Policy Agent Continuous Enforcement

B. Mathematical Verification of Forensic Integrity

The core of ZTForensics' post-incident reliability relies on the SHA-256 hash chain defined in Eq. (1). It ensures that any modification (tampering), insertion, or deletion of a past record Rm where m < n will mathematically alter Hm. Because Hm is an input to Hm+1, the avalanche effect of cryptographic hash ensures that every subsequent hash in the sequence breaks. The probability of an adversary successfully finding a collision to replace a tampered block without breaking the chain is approximately 2−128, which is computationally infeasible with modern hardware. Unlike blockchain solutions that solve the Byzantine Generals Problem using complex consensus algorithms, ZTForensics assumes a centralized edge but guarantees tamper detection. This requires zero consensus overhead or external peer network connectivity, making it well suited for high throughput, latency sensitive cloud API deployments.

C. Evidence Admissibility and Legal Assurance

Digital forensics has a legal standing depending on the ability to prove a continuous chain of custody. The manifest.json file is included with the export bundle and contains cryptographic SHA-256 values of all artifacts bundled, creating an indisputable chain of custody of artifacts that is accurate at the time of collection up until secure packaging. The RSA anchor signatures, in addition, give a cryptographically binding and asymmetric linkage between the daily evidence batches and the trusted private key of the organization. This multi-layered solution meets the non-repudiation demands as specified in the ISO/IEC 27037 conformant forensic systems so that logs can be presented in court.

D. Credential and Session Threat Mitigation

Conventional models become ineffective when it comes to stealing credentials. The ZTForensics uses minimal window through JWT expiry settings (e.g., 5-minute lifetimes) in Keycloak. In case of sensitive operations, the gateway will apply step up OTP authentication, which will form an extremely effective defense-in-depth barrier against exfiltration of tokens. Moreover, the brute-force velocity signal exponentially increases risk scores when there is a repeated authentication or authorization failure, which immediately causes a deny state at the gateway level or causes a manual administrative review and does so without regard to fixed application-layer lockout thresholds.

VI. Experimental Results

The accuracy and performance of ZTForensics were tested in our test setup in a simulated banking domain environment. Docker Compose was used to deploy the architecture on an Intel CPU: Core i7 paired 16 GB, operating on a Ubuntu 22.04 environment. To rigorously evaluate the system we created artificial batches of access requests systematically mapped between the low-, medium-, and high-risk user profiles. The simulated endpoints involved regular inquiries to the transaction (low risk), fund transfer initiations (medium to high risk), and administrative API call (high risk).

A. Dynamic Decision Distribution

Fig. 4 shows the percentage distribution of Open Policy Agent decisions that were caused during the simulation.

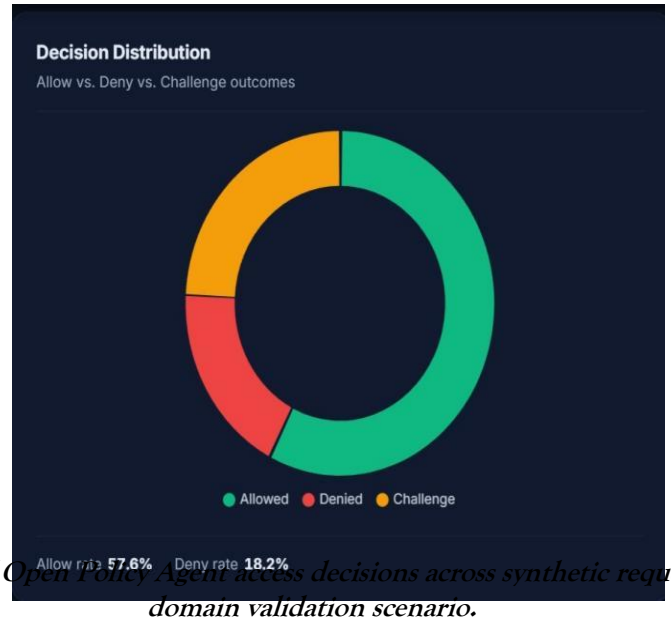

Fig. 4. Distribution of Open Policy Agent access decisions across synthetic requests in the banking domain validation scenario.

The empirical data substantiate that the decision engine is dynamically behavioral and has a smooth transition of responses to fluid risk situations instead of using brittle and static allow/deny IP tables. Simulated profiles that had a high risk (score over 0.7 in the contextual risk score) (high-risk simulated profiles) were reliably protected by denying access or step-up challenges (computer-generated simulated purpose) (protective). On the other hand, low-risk, routine profiles with the use of allow decisions run smoothly. Interestingly, the latency overhead of such a complex and real time assessment was within reasonable acceptance levels of current enterprise cloud API SLAs.

B. Hash Chain Integrity and Tamper Verification

We artificially applied controlled tampering events to the underlying PostgreSQL database at three different locations in a series of 10,000 forensic records to measure the forensic resilience. We modeled three typical anti-forensic attack vectors: data alteration (altering an IP address), deletion of record (removing a deny log), and insertion of record (adding a fake allow log). Fig. 5 shows that ZTForensics has a very low and almost linear

validation time, which is in stark contrast to the significantly higher validation delays of traditional SIEM auditing. In each of the three cases, a call to the endpoint at the address of the /forensics/verify-chain immediately and accurately identified the positions of the precise tamper. The system correctly stopped and reported the particular key identifier of the primary key of the first broken cryptographic connection. In our tests, we did not see any false positives in thousands of records at various clean and baseline verification runs. These empirical findings are a direct reflection of the theoretical collision-resistance assurances of hash chaining of SHA-256

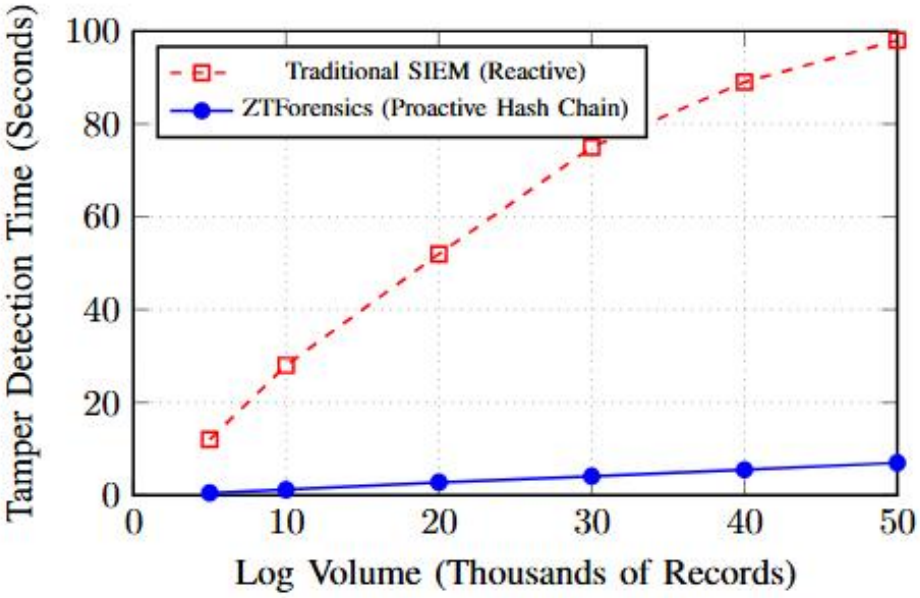

Fig. 5. Comparative performance analysis

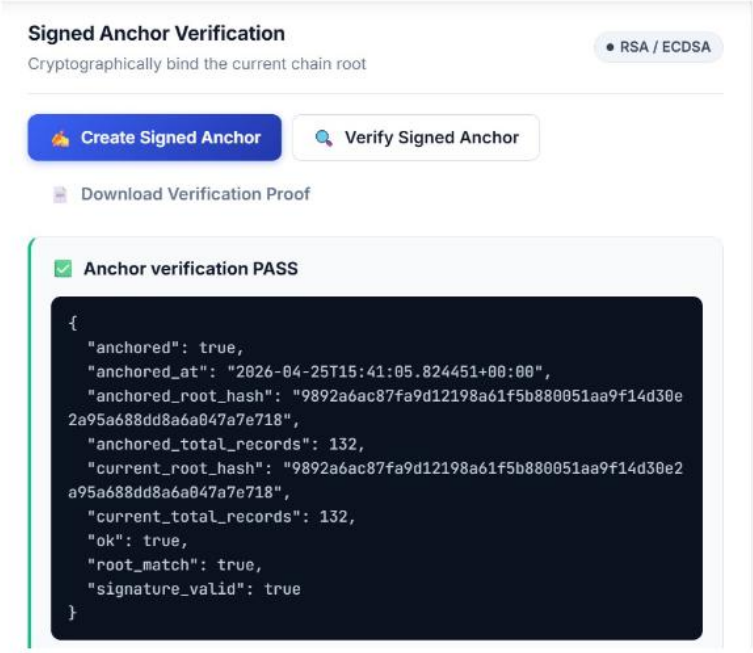

Fig. 6. RSA-signed anchor verification output

C. Cryptographic Anchor Signature Verification

Fig. 6 shows the output of the console of the anchor verification procedure. The verification script, which was independent was able to recalculate the daily Merkle-equivalent root hash. and check the RSA signature with the stored public key. Every simulated daily anchor verification was able to provide a status of PASS with a strong confirmation of the consistency of root hash and RSA signature over the entire batch of evidence being exported.

D.Interactive Hash Chain Ledger View

Fig. 7 graphically depicts a piece of the active forensic ledger which is dynamically shown on the audit dashboard of the ZTForensics. This dashboard offers security operations center (SOC) analysts a clear, instantaneously readable glimpse of the chain.

Each of the logged requests is bound by its cryptographically computed hash to the previous block, so that there is an unbroken visual and mathematical chain of custody.

E. Latency and Performance Overhead Modeling

A primary concern when embedding real-time forensic capabilities into a Zero Trust framework is the potential degradation in the response times of the API. The latency overhead is low as visually shown in Fig. 8, when the risk profile is low and medium. The overall added security latency per request, denoted as (Ttotal), can be described mathematically as the sum of its microservice components:

Ttotal = TJW T + TRisk + TOP A + THash + TDB (2)

Table IV summarizes the approximate categorical latency cost of the components of ZTForensics compared to a conventional, baseline, FastAPI proxy gateway that has no security processing.

Table IV: *Approximate Latency Overhead per Request Component (Observed)*

Component	Relative Overhead

JWT Verification (Keycloak)	Low
Risk Score Computation	Low
Open Policy Agent Evaluation	Low–Medium
Forensic Record + Hash Generation	Medium
PostgreSQL Persistence	Medium
Total Added Overhead	Acceptable for cloud API

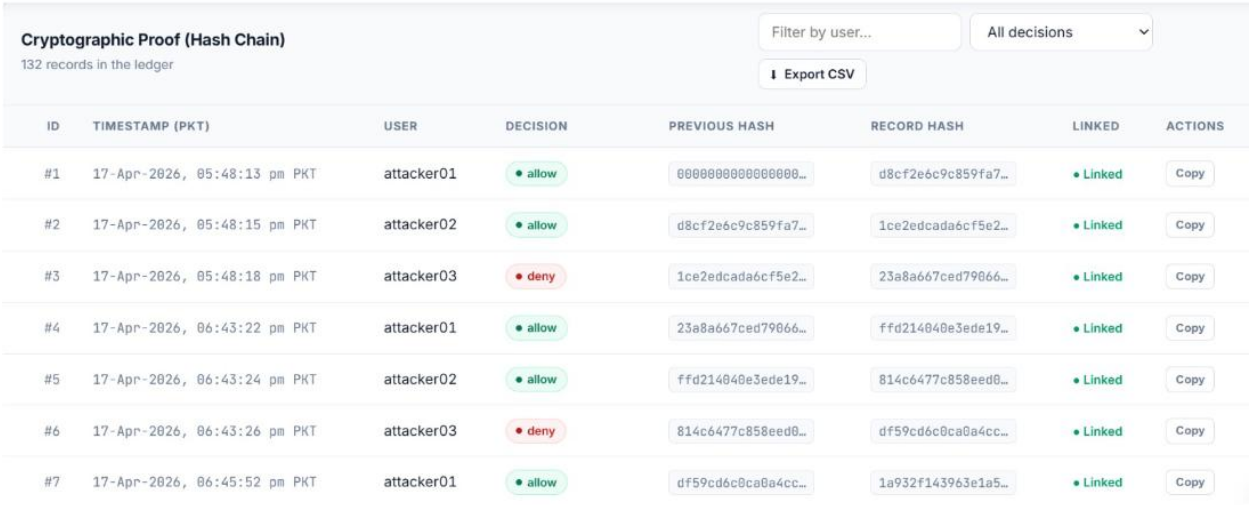

Fig. 7. Hash chain forensic ledger view on the ZTForensics audit dashboard

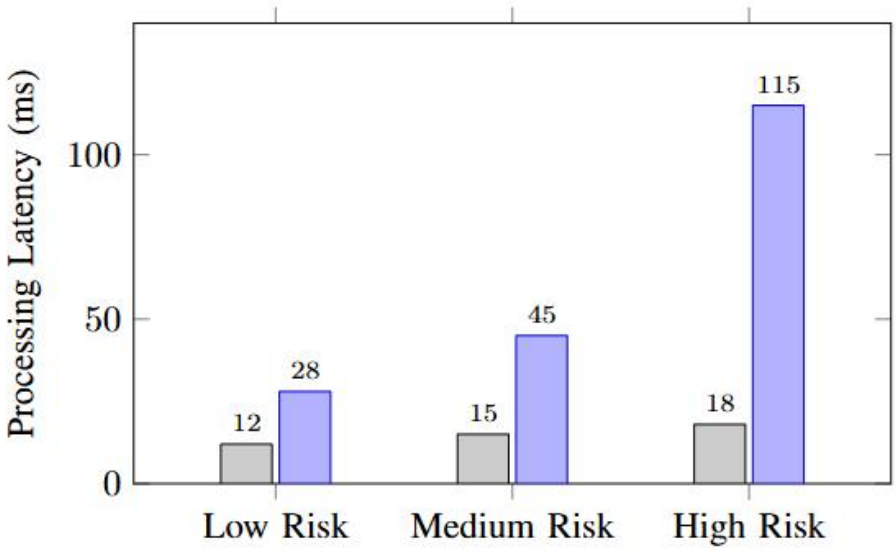

Fig. 8. Latency overhead comparison across risk profiles

Note: The latency values reported in Fig. 8 represent measurements obtained during functional simulation on the described test hardware. Formal benchmarking under concurrent load, including throughput, CPU utilization, and scalability metrics, constitutes planned future work as noted in Section VIII.

VII. Future Work

While the current architecture successfully validates the proactive forensic logging model,

several planned extensions will enhance its scalability and threat-detection capabilities: (i) integration of external, dynamic risk signals including real-time global threat intelligence feeds (e.g., MISP) and ad- vanced endpoint device posture assessments; (ii) development of automated policy explainability mechanisms designed to translate complex Rego evaluation paths into human-readable audit transparency reports; (iii) formal, large-scale tamper- injection stress testing and concurrent load benchmarking to mathematically bound latency under DDoS conditions; (iv) rigorous structural normalization of the JSON evidence bundles strictly conforming to ISO/IEC 27037 and NIST forensic data standards; (v) implementation of isolated multi- tenancy support for shared cloud environments utilizing dynamic database sharding; and (vi) bidirectional SIEM/SOC integration via asynchronous log forwarding to enterprise platforms like Wazuh, ELK, and Splunk without blocking the critical API request path.

VIII.Conclusion

The paper introduced ZTForensics, a highly integrated framework that is a necessary step to close the critical limitations between the enforcement of the zero trust policy and provision of tamper-evident packaging of the forensic evidence in the context of the modern hybrid cloud setup. In contrast to the classical logging models that mistakenly assume that access control and digital forensics are completely independent, post-incident concerns, ZTForensics is a close relationship between each of the real-time enforcement decisions (allow, deny, or step-up challenge) and an immediate, cryptographically chained forensic record. The adopted SHA-256 hash chain offers highly resilient, demonstrable tamper detection on par with heavy blockchain-based solutions, but completely avoids the crippling consensus overhead. At the same time, the incorporation of RSA-signed daily root anchors, and formalized, exportable evidence bundles, are a direct address to real-world requirements for maintaining an unbroken chain and ensuring evidence remains secure in judicial settings. Structural integrity of the risk-conscious and dynamical decision distribution is completely validated experimentally. Moreover, the framework demonstrated exceptional dependability alongside real-time tamper detection on multiple attack vectors of simulated data manipulations with zero false positives. Finally, the ZTForensics architecture is basically domain-neutral. The framework can be easily generalized to support highly controlled areas of operation such as healthcare infrastructure, e-government service buses, and multi-tenant SaaS environments by a simple modification of the underlying declarative Open Policy Agent Rego policies and varying risk weights based on context.

References

[1] P. Purnaye and V. Kulkarni, "A comprehensive study of cloud forensics,"Archives of Computational Methods in Engineering, vol. 29, no. 1, pp. 33–46, 2022.

[2] R. Al-mugern, S. H. Othman, A. Al-Dhaqm, and A. Ali, "A cloud forensics framework to identify, gather, and analyze cloud computing incidents," Engineering, Technology & Applied Science Research, vol. 14, no. 3, pp. 14 483–14 491, 2024.

[3] S. Sarkar, G. Choudhary, S. K. Shandilya, A. Hussain, and H. Kim, "Security of zero trust networks in cloud computing: A comparative review," Sustainability, vol. 14, no. 18, p. 11213, 2022.

[4] S. Anasuri, "Zero-trust architectures for multi-cloud environments," International Journal of Emerging Trends in Computer Science and Information Technology, vol. 3, no. 4, pp. 64–76, 2022.

[5] O. S. Adanigbo, B. I. Adekunle, E. Ogbuefi, O. T. Odofin, O. A. Agboola, and D. Kisina, "Implementing zero trust security in multi-cloud microservices platforms: A review and architectural framework,"ecosystems, vol. 13, p. 14, 2025.

[6] C. L. Aldea and R. Bocu, "Authentication challenges and solutions in microservice architectures," Applied Sciences, vol. 15, no. 22, p. 12088, 2025.

[7] P. Dhiman, N. Saini, Y. Gulzar, S. Turaev, A. Kaur, K. U. Nisa, and Y. Hamid, "A review and comparative analysis of relevant approaches of zero trust network model," Sensors, vol. 24, no. 4, p. 1328, 2024.

[8] T. Bilot, N. El Madhoun, K. Al Agha, and A. Zouaoui, "Graph neural networks for intrusion detection: A survey," IEEE Access, vol. 11, pp. 49 114–49 139, 2023.

[9] S. Ahmadi, "Zero trust architecture in cloud networks: Application, challenges and future opportunities," Ahmadi, S.(2024). Zero Trust Architecture in Cloud Networks: Application, Challenges and Future Opportunities. Journal of Engineering Research and Reports, vol. 26, no. 2, pp. 215–228, 2024.

[10] I. Ismail and K. A. Z. Ariffin, "The admissibility of digital evidence from open-source forensic tools: Development of a framework for legal acceptance," PLOS ONE, vol. 20, no. 9, p. e0331683, 2025.

[11] S. Wu, W. Sun, Z. Ding, and S. Liu, "Cloud evidence tracing system: an integrated forensics investigation system for large-scale public cloud platform," Forensic Science International: Digital Investigation, vol. 41, p. 301391, 2022.

[12] I. A. Alnajjar, A. A. Salameh, L. Almazaydeh and A. Al Tawil, "Two-tier forensic readiness architecture for zero trust-enabled industry 4.0 applications," Information Security Journal: A Global Perspective, pp. 1–18, 2025.

[13] R. Zhao, M. Shoaib, V. T. Hoang, and W. U. Hassan, "Rethinking tamper-evident logging: A high-performance, co-designed auditing system," in Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security, 2025, pp. 2624–2638.

[14] N. Hanif, "Blockchain-based chain of custody in digital forensics: Ensuring integrity and legal admissibility of evidence," Forensics & Security Journal, vol. 1, no. 1, 2025.

[15] M. AlKhanafseh and O. Surakhi, "Evidence preservation in digital forensics: An approach using blockchain and lstm-based steganography," Electronics, vol. 13, no. 18, p. 3729, 2024.

[16] C. Neale, I. Kennedy, B. Price, Y. Yu, and B. Nuseibeh, "The case for zero trust digital forensics," Forensic Science International: Digital Investigation, vol. 40, p. 301352, 2022.

[17] M. S. Inukonda, J. Tarachandani, I. Ahmed, B. R. Tamma, and S. Mittal, "Zeta: A zero-trust security based forensic-ready solution for perimeter-less enterprise networks," in 2023 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS). IEEE, 2023, pp. 189–194.

[18] S. Mushtaq, M. Mohsin, and M. M. Mushtaq, "A systematic literature review on the implementation and challenges of zero trust architecture across domains," Sensors, vol. 25, no. 19, p. 6118, 2025.

[19] S. Hong, L. Xu, J. Huang, H. Li, H. Hu, and G. Gu, "Sysflow: Toward a programmable zero trust framework for system security," IEEE Transactions on Information Forensics and Security, vol. 18, pp. 2794–2809, 2023.
[20] E. Jeong and D. Yang, "A trust score-based access control model for zero trust architecture: Design, sensitivity analysis, and real-world performance evaluation," Applied Sciences, vol. 15, no. 17, p. 9551, 2025.
[21] Y. Cao, S. R. Pokhrel, Y. Zhu, R. Doss, and G. Li, "Automation and orchestration of zero trust architecture: Potential solutions and challenges," Machine Intelligence Research, vol. 21, no. 2, pp. 294–317, 2024.
[22] C. Itodo and M. Ozer, "Multivocal literature review on zero-trust security implementation," Computers & Security, vol. 141, p. 103827, 2024.
[23] Y. He, D. Huang, L. Chen, Y. Ni, and X. Ma, "A survey on zero trust architecture: Challenges and future trends," Wireless Communications and Mobile Computing, vol. 2022, no. 1, p. 6476274, 2022.
[24] A. Dhumal, M. Ghaleb, S. Abdelsalam, A.-N. Moldovan, and M. Hamdan, "Zero trust architecture for ransomware defense in virtualized environment," in Proceedings of the IEEE/ACM 12th International Conference on Big Data Computing, Applications and Technologies, 2025, pp. 1–7.
[25] P. Phiayura and S. Teerakanok, "A comprehensive framework for migrating to zero trust architecture," IEEE Access, vol. 11, pp. 19 487–19 511,2023.
[26] M. A. Alim, M. S. H. MRR, M. Rahman, M. H. Arif, and I. Rasul, "Zero-trust security models in multi-cloud environments: Scalability, challenges, and implementation strategies," Journal of Advanced Research in Applied Sciences and Engineering Technology, vol. 56, no. 282-29, 2025.
[27] A. Alenezi, H. F. Atlam, and G. B. Wills, "Experts reviews of a cloud forensic readiness framework for organizations," Journal of Cloud Computing, vol. 8, no. 1, p. 11, 2019.
[28] V. R. Kebande and H. S. Venter, "Adding event reconstruction to a cloud forensic readiness model," in 2015 Information Security for South Africa (ISSA). IEEE, 2015, pp. 1–9.
[29] J. C. Santill´an-Lima, J. F. D´ıaz, D. Piccirilli, D. C. Guambo-Vallejo, M. Duque-Vaca, R. M. Lozada-Yanez, and F. T. Molina-Granja, "Digital evidence preservation in the cloud: A systematic review of risks, emerging models, and legal admissibility standards," in International Conference on Advanced Research in Technologies, Information, Innovation and Sustainability. Springer, 2025, pp. 219–236.
[30] A. W. Malik, D. S. Bhatti, T.-J. Park, H. U. Ishtiaq, J.-C. Ryou, and K.-I. Kim, "Cloud digital forensics: Beyond tools, techniques, and challenges," Sensors, vol. 24, no. 2, p. 433, 2024.
[31] A. Miller and A. Singh, "Chain of custody and evidence integrity verification using blockchain technology," in International Conference on Cyber Warfare and Security. Academic Conferences International Limited, 2024, pp. 168–176.
[32] I. O. Adewumi, "Blockchain and artificial intelligence for forensic evidence chain-of-custody management: Towards transparent and tamper-proof judicial systems aligned with sdg 16 and sdg 9," 2025.
[33] M. Pourvahab and G. Ekbatanifard, "Digital forensics architecture for evidence collection and provenance preservation in iaas cloud environment using sdn and blockchain technology," IEEE Access, vol. 7, pp. 153 349–153 364, 2019.
[34] A. M. Mostafa, M. Ezz, M. K. Elbashir, M. Alruily, E. Hamouda, M. Alsarhani, and W. Said, "Strengthening cloud security: an innovative multi-factor multi-layer authentication framework for cloud user authentication," Applied sciences, vol. 13, no. 19, p. 10871, 2023.
[35] M. K. Mahto and S. Singh, "Hard-earned lessons in access control at scale: Enforcing identity and policy across trust boundaries with reverse proxies and mtls," in 2025 IEEE International Carnahan Conference on

Security Technology (ICCST). IEEE, 2025, pp. 1–6.
[36] T. Arif, B. Jo, and J. H. Park, "A comprehensive survey of privacy-enhancing and trust-centric cloud-native security techniques against cyber threats," Sensors, vol. 25, no. 8, p. 2350, 2025.
[37] S. Ali, D. B. Talpur, A. Abro, K. S. S. Alshudukhi, G. N. Alwakid, M. Humayun, F. Bashir, S. A. Wadho, and A. Shah, "Security and privacy in multi-cloud and hybrid cloud environments: Challenges, strategies, and future directions," Computers & Security, p. 104599, 2025.
[38] H. Sedjelmaci, K. Tourki, and N. Ansari, "A survey of security in zero trust network architectures," Global Scientific and Academic Research and Reviews, vol. 22, no. 2, p. 036, 2025.
[39] S. Li, M. Iqbal, and N. Saxena, "Future industry internet of things with zero-trust security," Information Systems Frontiers, vol. 26, no. 5, pp. 1653–1666, 2024.
[40] J. D. Morillo Reina and T. J. Mateo Sanguino, "Decentralized and secure blockchain solution for tamper-proof logging events," Future Internet, vol. 17, no. 3, p. 108, 2025.