Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Tamper-Evident Incident Response and Digital Evidence Management for Connected Vehicles Using Permissioned Blockchain

Md Shahanur Islam Shagor

smshagor.ru@gmail.com

Voronezh State University of Forestry and Technologies https://orcid.org/0009-0003-7730-3202

Research Article

Keywords: blockchain, connected vehicles, cybersecurity incident response, digital evidence, Internet of Vehicles, tamper-evident logging, V2X, chain of custody

Posted Date: September 18th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11073242/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

Tamper-Evident Incident Response and Digital Evidence Management for Connected Vehicles Using Permissioned Blockchain

Md Shahanur Islam Shagor

Independent Researcher, Department of Computer Science and Microelectronics Engineering,
Voronezh State University of Forestry and Technologies, Voronezh, Russia

Corresponding author: smshagor.dev@gmail.com

Abstract—Connected vehicles require incident-response mechanisms that preserve forensic integrity without moving high-volume or privacy-sensitive telemetry directly into a distributed ledger. This paper presents a permissioned-blockchain-assisted incident-response architecture in which normalized evidence is maintained in a local tamper-evident journal while compact integrity checkpoints are periodically committed to a validator ledger. Each evidence record is canonicalized, sequence-linked with SHA3-256, and authenticated with HMAC-SHA-256 under a key domain separated from operator authorization. A latched incident state machine prevents automatic recovery after containment events: acknowledgement and recovery actions are independently authenticated, timestamp-bound, nonce-protected, and subject to a healthy-observation window, evidence-chain validation, and a safety-interlock confirmation for safety-critical incidents. The design therefore couples chain-of-custody preservation with operational recovery rather than treating blockchain as a passive archive. A controlled x86-64 software benchmark shows mean record-construction latency of 19.15 microseconds for the authenticated hash-chain path, full verification of 50,000 evidence records in 693.89 ms, and approximately 99.5% lower ledger data volume when one 192-byte checkpoint is stored per 64 records instead of placing a 620-byte normalized record on-chain. Adversarial validation detects field modification, deletion, reordering, and forged evidence authentication and rejects replayed operator authorization and unsafe recovery. The results support a practical separation between local evidentiary detail, distributed integrity assurance, and fail-closed incident recovery for connected-vehicle security operations.

Index Terms—blockchain, connected vehicles, cybersecurity incident response, digital evidence, Internet of Vehicles, tamper-evident logging, V2X, chain of custody.

I. INTRODUCTION

Connected vehicles continuously exchange safety, control, diagnostic, and cooperative-awareness information across in-vehicle networks, roadside infrastructure, backend services, and vehicle-to-everything (V2X) links. This connectivity improves situational awareness and mobility services, but it also creates an evidentiary problem when a cyber incident occurs. Security monitors may observe replay attempts, forged service responses, integrity failures, abnormal synchronization behavior, or contradictory control-plane events. The evidence needed to reconstruct such incidents is distributed across components, generated at high rate, and potentially mutable by an attacker who has already gained partial system access. A response mechanism must therefore answer two questions simultaneously: whether the evidence can still be trusted, and whether the system is safe to return from containment to normal operation.

Traditional centralized logging is operationally convenient but places strong trust in the host and storage administrator. Blockchain-based vehicle-forensics research has addressed parts of this problem by distributing evidence references, access decisions, or event records among multiple stakeholders. Block4Forensic introduced a lightweight blockchain model for connected-vehicle forensics [1]. Later work considered smart-contract-managed accident evidence and access control [2], multisource incident reporting with machine learning and blockchain [3], and traffic-event verification in smart vehicles [4]. More recent systems address post-quantum forensic privacy [5] and protected cross-platform vehicle forensic data [6]. These contributions establish the value of distributed integrity and accountability, but incident recovery itself is often modeled separately from evidence integrity. In practice, that separation can be dangerous: a system may produce an immutable audit record and still recover automatically while evidence is inconsistent, while a new containment event is pending, or after a stale operator command has been replayed.

This paper addresses the gap by treating evidence preservation and recovery authorization as one security protocol. The proposed architecture uses local, value-minimized, tamper-evident records for detailed incident chronology and a permissioned blockchain only for compact integrity checkpoints. A latched state machine connects the evidentiary status to operational recovery. An incident cannot become recoverable simply because time has passed; it must be acknowledged by an authorized operator, followed by a sequence of healthy observations, no new containment evidence, valid local and monitor evidence chains, a fresh signed recovery authorization, and, when the incident is safety-critical, explicit safety-interlock confirmation.

The contribution is fourfold. First, a chain-of-custody construction is defined for normalized vehicle-security evidence using deterministic canonicalization, SHA3-256 sequence linking, and HMAC-based record authentication, while raw payloads and credentials are excluded from the journal. Second, compact ledger checkpoints bind a local journal tail to a permissioned validator history, reducing ledger growth without sacrificing externally auditable integrity. Third, evidence-authentication keys are separated from operator-authorization keys, and operator actions are bound to the incident identifier, action, timestamp, and nonce to resist forgery and replay. Fourth, a fail-closed recovery state machine is formalized and evaluated under evidence tampering, deletion, reordering, forged authentication tags, replayed operator commands, new evidence after acknowledgement, and missing safety-interlock confirmation.

II. RELATED WORK

A. Vehicular Digital Forensics and Blockchain

Cebe et al. proposed Block4Forensic as a lightweight blockchain framework for forensic applications of connected vehicles, emphasizing distributed evidence integrity and forensic data collection across vehicle entities [1]. Philip and Saravanaguru subsequently presented a smart-contract evidence-management framework for accident investigation, including dynamic stakeholder access control and a comparison of on-chain and off-chain storage [2]. Their later multisource framework combined event correlation, trust assessment, machine learning, and blockchain-supported evidence management to address conflicting or malicious incident reports [3]. These studies demonstrate that forensic usefulness depends not only on preserving data but also on deciding which event reports are sufficiently trustworthy to preserve.

Pujol et al. investigated blockchain-based traffic-event verification in smart vehicles and showed how distributed validation can support cooperative event credibility [4]. Zhang et al. extended IoV forensics toward post-quantum blockchain constructs, focusing on privacy, traceability, and resistance to cryptographic threats from quantum

computing [5]. Li et al. proposed PSVDF for vehicle digital forensics using protected cross-platform storage and blockchain views to improve evidence authenticity and availability [6]. These works primarily optimize forensic preservation, privacy, validation, or access. The present study instead focuses on the operational boundary between incident containment and recovery, while using blockchain as an external integrity anchor rather than as the primary store for every evidence item.

B. Immutable Logging and Incident Response

Blockchain-backed logging has also been studied outside the vehicular domain. Pourmajidi et al. proposed immutable log storage over private and public blockchains and used hierarchical organization to address scalability concerns [7]. Such work motivates anchoring strategies in which distributed consensus protects compact integrity commitments while bulk records remain in a more efficient local or off-chain tier. The same principle is especially relevant to connected vehicles because raw sensor and runtime data can be voluminous, privacy-sensitive, and subject to retention restrictions.

Incident-response guidance emphasizes preparation, detection, response, recovery, and the preservation of evidence needed for analysis. NIST SP 800-61 Rev. 3 treats incident response as an integrated cybersecurity-risk-management function [8], while NIST SP 800-86 provides practical guidance for integrating forensic techniques into incident handling [9]. ISO/SAE 21434 similarly establishes lifecycle-oriented cybersecurity engineering expectations for road vehicles [10]. These standards do not prescribe the specific cryptographic journal and ledger architecture developed here, but they motivate clear security boundaries, traceable actions, and controlled recovery decisions.

C. Research Gap

Three limitations remain visible across the literature. First, immutable storage alone does not determine when a compromised or contained vehicle function should be restored. Second, a single cryptographic trust domain for both evidence creation and operator control creates unnecessary common-mode risk. Third, storing every record on a ledger can inflate consensus traffic and replicated storage. The proposed design therefore combines a locally efficient authenticated chain, periodic permissioned-ledger checkpoints, independently authenticated human actions, and recovery invariants that fail closed whenever evidence integrity or system health cannot be established.

III. SYSTEM AND THREAT MODEL

A. System Entities

The system contains five logical entities. The runtime security monitor M observes security-relevant events from authenticated subsystems and emits normalized incident decisions. The incident manager I maintains the incident lifecycle. The evidence journal J stores normalized, value-minimized transition records. A permissioned validator set V = {v1, ..., vn} accepts compact checkpoint transactions. Finally, an operator or investigator O acknowledges incidents, authorizes recovery, and verifies evidence. The architecture is shown in Fig. 1. No direct vehicle actuation is performed by the evidence manager; containment and safe-mode outputs are requests consumed by separately governed safety mechanisms.

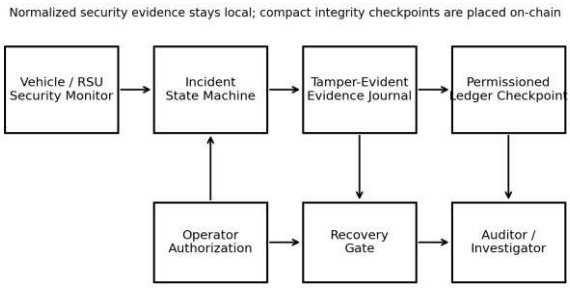


FIGURE 1. Architecture of the proposed tamper-evident incident-response and evidence-management framework.

B. Evidence and Privacy Model

Evidence records contain security metadata required to reconstruct state transitions: incident identifier, transition type, time, incident level, recommended containment action, safety-critical flag, monitor event count, monitor tail hash, bounded source/category labels, the previous record hash, and the cryptographic fields defined below. Raw packet bodies, passwords, session secrets, full vehicle identifiers, and raw sensor streams are intentionally excluded. This choice minimizes evidentiary exposure and makes the journal a chronology of security decisions rather than an unrestricted telemetry archive.

C. Adversary Model

The adversary may read or modify local journal files, delete or reorder records, replay previously valid operator authorizations, submit forged authorizations, or trigger new containment evidence after an earlier acknowledgement. A network adversary may delay checkpoint propagation or cause temporary loss of validator connectivity. A minority of authorized validators may be Byzantine. The design assumes the cryptographic keys used for evidence authentication and operator authorization are not simultaneously compromised and that the permissioned ledger preserves its configured quorum assumption. An attacker with both unrestricted storage-write access and the evidence-authentication key can recompute a local chain; this case is addressed only when a previously published ledger checkpoint exists. The architecture is tamper-evident, not physically tamper-proof, and does not claim automotive functional-safety certification.

D. Security Objectives

The design targets six invariants: (1) modification, deletion, insertion, or reordering of committed evidence must be detectable; (2) a valid operator action must not be reusable outside its freshness window or more than once; (3) acknowledgement must not itself imply recovery eligibility; (4) any new containment evidence must invalidate a prior recovery trajectory; (5) safety-critical recovery must require an additional interlock confirmation; and (6) loss of local or monitor evidence integrity must force a fail-closed decision rather than a silent return to normal operation.

IV. PROPOSED METHODOLOGY

A. Canonical Evidence Construction

Let ri denote the normalized evidence metadata for sequence i. A deterministic canonicalization function C(.) sorts keys, uses an unambiguous UTF-8 representation, rejects non-finite numeric values, and fixes field names and domains. Determinism is essential because two verifiers must derive identical bytes from the same logical record. The previous record hash is included in ri before hashing, so chronological position is cryptographically bound to the record.

$$h_i = \text{SHA3-256}(\, C(r_i)\,), \quad r_i.\text{previous} = h_\{i\text{-}1\} \quad (1)$$

The authenticated evidence tag uses a dedicated evidence key Ke and a protocol domain De. Domain separation prevents an HMAC produced for one protocol purpose from being accepted as an operator authorization or another message class.

$$\text{tau_i} = \text{HMAC-SHA-256}(\, K_e, D_e \,||\, C(r_i) \,||\, h_i\,) \quad (2)$$

A verifier starts from the known genesis hash h0 and checks sequence continuity, the previous-hash field, recomputed SHA3-256 digest, and HMAC tag for every record. Verification is performed against persisted bytes rather than only an in-memory state so that post-open file tampering is not hidden by cached data.

B. Permissioned Blockchain Checkpointing

Replicating every evidence record through consensus would maximize direct ledger visibility but needlessly duplicates local metadata. Instead, the proposed architecture creates a checkpoint after B records or immediately at important lifecycle transitions. For checkpoint j ending at local sequence i, the commitment is

$$c_j = \text{SHA3-256}(\, \text{journal_id} \,||\, i \,||\, h_i \,||\, s_i \,||\, t_j\,) \quad (3)$$

where si is the incident state and tj is the checkpoint time. The checkpoint transaction contains c_j, the ending sequence, a journal pseudonym, the state code, and validator-ledger metadata. A quorum-certified checkpoint makes later local rewriting detectable: even an attacker that recomputes subsequent local hashes cannot change the historical tail h_i without disagreeing with the ledger commitment. The checkpoint interval B trades ledger volume against the maximum number of newly generated records that are not yet externally anchored. The experiments use B = 64.

C. Incident Lifecycle and Latching

Figure 2 shows the lifecycle. CLEAR denotes no active incident. A containment-level decision opens a fresh incident and latches it in OPEN. Only a fresh authorized acknowledgement moves the incident to ACKNOWLEDGED. Healthy observations then move the incident into RECOVERY_PENDING and increment a health counter. New containment evidence at any point before recovery resets the state to OPEN and clears the earlier acknowledgement. This prevents an operator approval from remaining valid across a materially changed threat context.

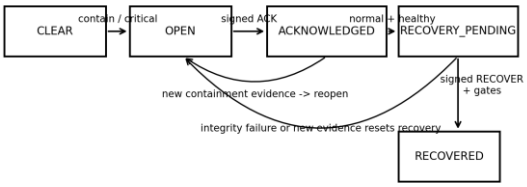


FIGURE 2. Latched incident lifecycle and fail-closed recovery transitions.

D. Replay-Resistant Operator Authorization

The operator-control trust domain uses a separate key Ko. For action a in {ACKNOWLEDGE, RECOVER}, incident identifier q, timestamp t, and random nonce n, the authorization value is

$$\text{sigma} = \text{HMAC-SHA-256}(\, K_o,\, D_o \,||\, a \,||\, q \,||\, t \,||\, n \,) \quad (4)$$

A receiver verifies the protocol version, expected action, incident binding, timestamp freshness, nonce format, authentication value, and nonce uniqueness. Accepted nonces are held in a bounded replay cache until their freshness interval expires. A saturated replay cache fails closed instead of evicting an active nonce in a way that could re-enable a recent replay.

E. Recovery Gate

Recovery is deliberately stricter than acknowledgement. Let Jv indicate that the local evidence chain verifies, Mv that the monitor evidence chain verifies, N that the current decision is NORMAL with no containment action, H that the required number of consecutive healthy observations has been reached, A that a fresh RECOVER authorization has verified, and S that the safety interlock is confirmed when required. Recovery is permitted only when

$$R = J_v \text{ AND } M_v \text{ AND } N \text{ AND } H \text{ AND } A \text{ AND } (\text{not critical OR } S). \quad (5)$$

The implementation used for evaluation requires three healthy observations. A WATCH or audit-only state is not counted as healthy. Automatic recovery is never permitted. If the monitor evidence chain itself becomes invalid, the decision is elevated to a safety-critical containment condition, which prevents the integrity subsystem from treating corrupted monitoring evidence as proof of health.

F. Evidence Verification Procedure

An investigator verifies a case in two layers. First, the complete local journal is checked from h0 through the last record using (1) and (2), ensuring sequence and authentication continuity. Second, each relevant ledger checkpoint is fetched from the permissioned history and recomputed from the corresponding local tail. A valid local chain with a mismatching historical checkpoint indicates retroactive rewriting after the checkpoint was issued. Conversely, a valid checkpoint with a missing local segment indicates loss or deletion of locally retained evidence. This two-tier construction therefore distinguishes local continuity from externally witnessed integrity.

V. SECURITY ANALYSIS

A. Record Modification, Deletion, and Reordering

Changing any protected field changes C(ri), which changes hi and invalidates taui. Deleting a record causes the next record to reference a previous hash not present at the expected sequence. Reordering causes both sequence and previous-hash mismatches. Insertion similarly breaks the expected sequence unless all later records are regenerated, in which case an earlier blockchain checkpoint exposes the rewritten tail. The construction therefore provides tamper evidence for ordinary local edits and ledger-backed rollback evidence across anchored prefixes.

B. Forged Evidence and Key Separation

A record whose hash is correctly recomputed but whose authentication tag is not produced with Ke fails (2). Separating Ke from Ko limits cross-protocol damage: disclosure of an operator authorization credential does not grant the ability to rewrite evidence tags, while disclosure of an evidence credential does not authorize incident recovery. This does not remove the need for secure key custody, rotation, and access control; it reduces the blast radius of a single credential failure.

C. Operator Replay and Staleness

A recorded ACKNOWLEDGE or RECOVER message includes a unique nonce and bounded timestamp. Reuse of the message is rejected if the nonce has already been accepted, while delayed delivery outside the configured time window fails freshness validation. Binding the incident identifier and action prevents an acknowledgement for one incident from being interpreted as recovery for another. This property is important because incident-control messages can remain cryptographically valid long after their operational meaning has expired unless freshness is explicitly enforced.

D. Unsafe Recovery and New Evidence

The recovery gate prevents a healthy snapshot from erasing an earlier incident. The system first requires explicit acknowledgement and then a healthy observation window. If a new containment event arrives, the acknowledgement is invalidated and the incident returns to OPEN. Safety-critical incidents additionally require a separate interlock confirmation. Thus the design resists three failure modes that immutable storage alone cannot prevent: premature recovery, stale authorization reuse, and recovery after the threat context has changed.

E. Validator and Endpoint Compromise

Permissioned checkpointing inherits the safety assumptions of the validator consensus. A malicious minority that cannot reach the configured quorum cannot rewrite already finalized checkpoints. If enough validators collude to violate the ledger assumption, ledger anchoring no longer provides independent witness integrity. Likewise, compromise of both the local evidence key and the vehicle storage can conceal tampering between the last finalized checkpoint and the current head. Shorter checkpoint intervals reduce this exposure at the cost of higher ledger traffic. Hardware-backed key storage and independent timestamp authorities are complementary controls but are outside the evaluated scope.

VI. EXPERIMENTAL METHODOLOGY

A. Test Environment and Baselines

Experiments were executed in a controlled Linux x86-64 environment using five virtual CPU cores of an AMD EPYC 9V74 processor and Python 3.13.5. Timing values are software measurements rather than road-test measurements. Each normalized record contained incident state, action, monitor integrity metadata, source/category labels, a previous hash, SHA3-256 digest, and HMAC-SHA-256 authentication tag. Construction microbenchmarks used 30,000 records per run and five runs. Scaling tests used 100 to 50,000 records. Storage analysis compared the proposed checkpoint design with a hypothetical

design that places every 620-byte normalized evidence record on the permissioned ledger. The checkpoint record was modeled as 192 bytes and emitted every 64 local records.

Three integrity baselines were considered conceptually: a mutable centralized event log with no cryptographic continuity, a SHA3-only hash chain without keyed authentication or recovery authorization, and a full-record ledger model that writes each normalized record through consensus. The proposed method combines keyed local evidence, periodic ledger checkpointing, replay-resistant operator control, and lifecycle recovery gates. The full-record ledger baseline is used for storage comparison, not for claiming measured consensus latency, because consensus performance depends strongly on validator implementation and network topology.

Parameter	Value
CPU	AMD EPYC 9V74, 5 virtual cores
Runtime	Python 3.13.5, Linux x86-64
Hash	SHA3-256
Record authentication	HMAC-SHA-256
Operator freshness window	120 s
Healthy observations	3
Checkpoint interval B	64 records
Construction repetitions	5 x 30,000 records

TABLE I. Controlled evaluation configuration.

B. Metrics and Adversarial Scenarios

Measured metrics were mean cryptographic construction time per record, journal construction/write time, complete-chain verification time, serialized record size, operator authorization verification time, and attack detection outcome. Adversarial tests modified a protected field, deleted a record, swapped adjacent records, replaced an authentication tag, replayed an already accepted operator command, supplied a forged operator tag, introduced new containment evidence after acknowledgement, and attempted safety-critical recovery without the interlock confirmation. Each test was expected to fail closed at the first violated invariant.

VII. RESULTS AND DISCUSSION

A. Evidence Construction and Verification Cost

Table II reports cryptographic construction cost. Canonical serialization alone required 7.47 microseconds per record. Adding SHA3-256 sequence linking increased the mean to 16.34 microseconds, while adding the HMAC evidence tag increased it to 19.15 microseconds. Thus keyed evidence authentication added approximately 2.81 microseconds per record over an unkeyed hash chain in this environment. The result indicates that cryptographic record construction is small relative to typical V2X network and application latencies, although embedded target measurements are still required before deployment claims can be made.

Path	Mean (us/record)	Std. dev. (us)
Canonical only	7.469	0.109
SHA3 chain	16.343	0.190
SHA3 + HMAC	19.152	0.220

TABLE II. Mean evidence-construction latency over five runs.

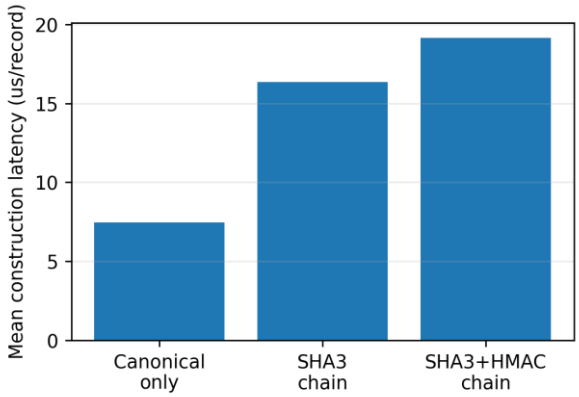


FIGURE 3. Mean record-construction latency for progressively stronger integrity paths.

Verification scaled approximately linearly with evidence volume. A 10,000-record journal verified in 139.82 ms, while 50,000 records verified in 693.89 ms, equivalent to approximately 13.88 microseconds per record at the largest tested scale. Journal construction and sequential write time for the same 50,000 records was 995.48 ms. Figure 4 shows the scaling trend. The benchmark performed one durability synchronization at the end of each generated journal; systems that synchronize every record can incur higher storage-device-dependent latency.

Records	Build/write (ms)	Verify (ms)	Avg. bytes/record
100	2.25	2.38	721.6
1,000	20.96	14.73	723.6
5,000	102.20	70.55	724.9
10,000	203.18	139.82	725.6
50,000	995.48	693.89	726.9

TABLE III. Evidence-journal scaling results.

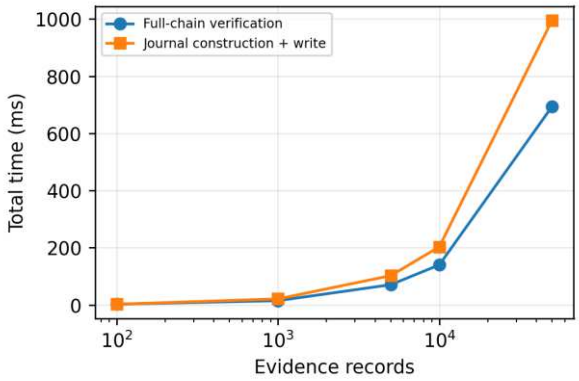


FIGURE 4. Journal construction/write and full-chain verification time versus evidence volume.

B. Ledger Storage Reduction

Checkpointing substantially reduces replicated ledger volume. At 50,000 local records, storing every 620-byte normalized record would require approximately 31.0 MB of ledger payload before protocol overhead, whereas 782 checkpoints at 192 bytes each require approximately 150.1 kB. Across the evaluated scales, the modeled payload reduction remains approximately 99.5%. This result does not imply that local storage is eliminated; the complete authenticated journal remains locally retained according to policy. The reduction concerns only data that must be replicated and agreed through the permissioned ledger.

Local records	Checkpoint bytes	Full-record ledger bytes	Reduction
1,000	3,072	620,000	99.505%
10,000	30,144	6,200,000	99.514%
50,000	150,144	31,000,000	99.516%
100,000	300,096	62,000,000	99.516%

TABLE IV. Modeled replicated-ledger payload with B = 64.

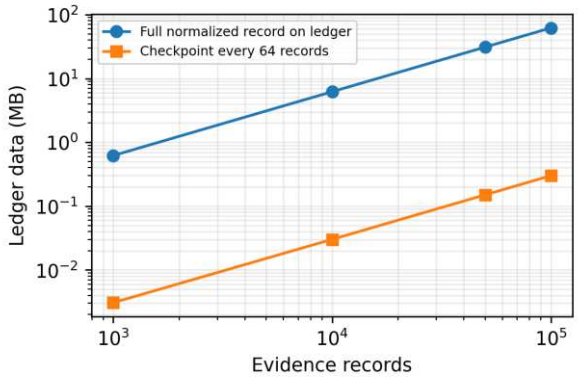


FIGURE 5. Ledger payload growth for full-record replication and periodic checkpointing.

C. Adversarial Evidence Tests

All four direct journal-tampering scenarios were detected. Field modification, deletion, reordering, and forged authentication were rejected at or before the altered sequence. In the 200-record tamper experiment, attacks introduced at sequence 81 were detected after 80 valid records, demonstrating that failure location is bounded by the first violated link rather than the end of the file. Replayed operator authorization was rejected by the nonce cache, forged operator tags failed authentication, and a recovery request lacking safety-interlock confirmation was rejected for a safety-critical incident. New containment evidence after acknowledgement reopened the same incident and forced re-acknowledgement.

Attack / unsafe action	Mutable log	Hash chain	Proposed
Field modification	Not assured	Detected	Detected
Record deletion	Not assured	Detected	Detected
Record reordering	Not assured	Detected	Detected
Forged evidence tag	N/A	Not keyed	Rejected
Operator replay	No control	No control	Rejected
Recovery without interlock	No control	No control	Rejected

TABLE V. Security-property comparison under evaluated adversarial actions.

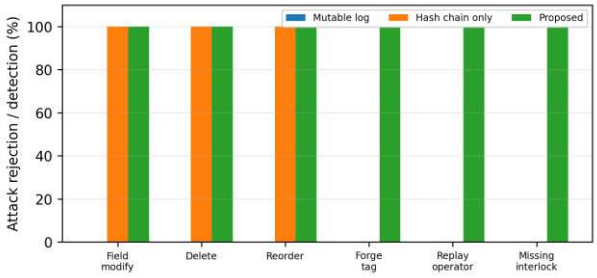


FIGURE 6. Detection/rejection coverage of the evaluated attack classes.

D. Operator Authorization Cost

The measured mean cost of producing and immediately verifying a keyed operator-authorization message was 3.34 microseconds per action over 50,000 iterations. The cryptographic cost is therefore negligible relative to human response time. More importantly, security comes from binding the authorization to an incident identifier, an expected action, a freshness window, and a single-use nonce. A low cryptographic latency without those semantic bindings would still permit operational replay.

E. Interpretation

The results support the design choice of keeping detailed evidence local while anchoring compact commitments. Full-chain verification of tens of thousands of records remains sub-second in the evaluated environment, and periodic checkpointing avoids replicated-ledger growth proportional to every event. The architecture also demonstrates that evidence integrity and recovery logic can reinforce one another: journal or monitor corruption is not merely reported to an auditor later; it directly blocks a recovery transition. This is the principal distinction from passive forensic storage.

VIII. COMPARATIVE ANALYSIS

Table VI summarizes the architectural differences among representative vehicular-forensics directions and the proposed framework. Existing blockchain-forensics systems provide valuable mechanisms for distributed evidence preservation, access control, privacy, or event validation [1]-[6]. The proposed architecture is complementary rather than a replacement: it narrows the problem to runtime incident lifecycle integrity and makes recovery conditional on the same evidence guarantees used for post-incident investigation.

Approach	Primary focus	Distributed integrity	Recovery gate	Replay-safe operator control
Block4Forensic [1]	Vehicle forensics	Yes	Not central	Not central
Smart-contract evidence [2]	Accident evidence/access	Yes	Not central	Not central
Multisource framework [3]	Event trust + evidence	Yes	Not central	Not central
Post-quantum IoV forensics [5]	Privacy/PQ security	Yes	Not central	Not central
PSVDF [6]	Protected forensic data	Yes	Not central	Not central
Proposed	Incident lifecycle + evidence	Yes, checkpointed	Yes	Yes

TABLE VI. Qualitative comparison with representative vehicular digital-forensics directions.

A full-record ledger offers strong distributed visibility but scales replicated storage with event volume and can expose more metadata to validators. A pure local hash chain reduces ledger dependence but lacks a distributed historical witness and, without keyed authentication, cannot distinguish a legitimately generated record from a chain recomputed by a local attacker. The proposed hybrid design uses the local journal for efficient chronology, keyed authentication for source integrity, and the ledger for externally witnessed checkpoints. The checkpoint interval can be tuned to the threat model: critical state transitions can be anchored immediately while routine evidence can be batched.

The framework is also different from a conventional incident ticketing system. An incident ticket can document that an operator approved recovery, but documentation alone does not cryptographically bind the approval to the incident state observed at that time. Here, acknowledgement and recovery are distinct authenticated operations. Evidence arriving after acknowledgement invalidates the earlier approval, and safety-critical recovery requires an independent interlock condition. These rules convert operational policy into verifiable state-transition invariants.

IX. LIMITATIONS AND FUTURE WORK

The evaluation is a controlled software benchmark and does not constitute validation on production electronic control units, roadside units, cellular V2X stacks, or certified automotive safety platforms. Storage and cryptographic timings depend on processor, filesystem, and durability policy. The blockchain storage comparison models payload volume but does not claim a universal consensus latency because finality depends on validator count, protocol, network delay, and implementation. The threat model also assumes that evidence and operator keys are not jointly compromised and that the validator quorum remains within its configured Byzantine tolerance.

Future work should evaluate the architecture on embedded automotive hardware with hardware-backed key custody, real permissioned-validator deployments under delay and packet loss, and long-duration evidence retention. Additional work is needed on privacy-preserving auditor queries, cross-domain checkpoint federation, secure time sources, key rotation across open incidents, and formal verification of the state machine. Integration with vehicle cybersecurity engineering processes should also be studied without conflating forensic integrity with functional-safety certification.

X. CONCLUSION

This paper presented a tamper-evident incident-response and digital-evidence architecture for connected vehicles that couples chain-of-custody integrity with fail-closed recovery. Normalized evidence is canonicalized, sequence-linked by SHA3-256, and authenticated with a dedicated HMAC key, while periodic permissioned-ledger checkpoints provide an externally witnessed integrity boundary without replicating every local record. Separate replay-resistant operator authorization protects acknowledgement and recovery. A latched state machine requires a healthy observation window, valid evidence chains, fresh authorization, and safety-interlock confirmation when necessary. Controlled measurements showed 19.15 microseconds mean authenticated record-construction cost, 693.89 ms verification time for 50,000 records, and approximately 99.5% lower modeled ledger payload than full-record replication at a checkpoint interval of 64. Adversarial tests detected journal modification, deletion, reordering, and forged authentication and rejected replay and unsafe recovery. The central result is that distributed evidence integrity can be used not only for retrospective investigation but also as an active precondition for safe cyber-incident recovery.

ACKNOWLEDGMENT AND DISCLOSURE

The author declares no external funding for this study. A generative artificial-intelligence tool was used to assist language drafting and document formatting during manuscript preparation. The author reviewed and verified the technical content, references, calculations, and conclusions and takes responsibility for the final manuscript.

DATA AVAILABILITY

No external dataset was used. The numerical results reported in this article were produced from the controlled benchmark configuration and adversarial scenarios described in Section VI.

REFERENCES

[1] M. Cebe, E. Erdin, K. Akkaya, H. Aksu, and A. S. Uluagac, "Block4Forensic: An integrated lightweight blockchain framework for forensics applications of connected vehicles," IEEE Commun. Mag., vol. 56, no. 10, pp. 50–57, 2018, doi: 10.1109/MCOM.2018.1800137.

[2] A. O. Philip and R. A. K. Saravanaguru, "Smart contract based digital evidence management framework over blockchain for vehicle accident investigation in IoV era," J. King Saud Univ. Comput. Inf. Sci., vol. 34, no. 7, pp. 4031–4046, 2022, doi: 10.1016/j.jksuci.2022.06.001.

[3] A. O. Philip and R. A. K. Saravanaguru, "Multisource traffic incident reporting and evidence management in Internet of Vehicles using machine learning and blockchain," Eng. Appl. Artif. Intell., vol. 117, Art. no. 105630, 2023, doi: 10.1016/j.engappai.2022.105630.

[4] F. A. Pujol, H. Mora, T. Ramírez, C. Rocamora, and A. Bedón, "Blockchain-based framework for traffic event verification in smart vehicles," IEEE Access, vol. 12, pp. 9251–9266, 2024, doi: 10.1109/ACCESS.2024.3352738.

[5] Z. Zhang, Z. Cao, and Y. Wang, "Forensics system for Internet of Vehicles based on post-quantum blockchain," Sensors, vol. 25, no. 19, Art. no. 6038, 2025, doi: 10.3390/s25196038.

[6] M. Li, W. Liu, Q. Xia, Y. Li, and Z. Xu, "PSVDF: A data protection framework for vehicle digital forensics utilizing blockchain view and cross-platform secure space," Blockchain: Res. Appl., vol. 7, no. 3, Art. no. 100393, 2026, doi: 10.1016/j.bcra.2025.100393.

[7] W. Pourmajidi, L. Zhang, J. Steinbacher, T. Erwin, and A. V. Miranskyy, "Immutable log storage as a service on private and public blockchains," IEEE Trans. Serv. Comput., vol. 16, no. 1, pp. 356–369, 2023, doi: 10.1109/TSC.2021.3120690.

[8] A. Nelson, S. Rekhi, M. Souppaya, and K. Scarfone, Incident Response Recommendations and Considerations for Cybersecurity Risk Management: A CSF 2.0 Community Profile, NIST SP 800-61 Rev. 3, Apr. 2025, doi: 10.6028/NIST.SP.800-61r3.

[9] K. Kent, S. Chevalier, T. Grance, and H. Dang, Guide to Integrating Forensic Techniques into Incident Response, NIST SP 800-86, Aug. 2006, doi: 10.6028/NIST.SP.800-86.

[10] ISO/SAE 21434:2021, Road Vehicles—Cybersecurity Engineering, International Organization for Standardization and SAE International, 2021.

[11] National Institute of Standards and Technology, SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions, FIPS PUB 202, Aug. 2015, doi: 10.6028/NIST.FIPS.202.

[12] H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-hashing for message authentication," RFC 2104, Feb. 1997, doi: 10.17487/RFC2104.

[13] M. A. Ferrag, M. Derdour, M. Mukherjee, A. Derhab, L. Maglaras, and H. Janicke, "Blockchain technologies for the Internet of Things: Research issues and challenges," IEEE Internet Things J., vol. 6, no. 2, pp. 2188–2204, 2019, doi: 10.1109/JIOT.2018.2882794.

[14] C. Lin, D. He, N. Kumar, K.-K. R. Choo, A. Vinel, and X. Huang, "Security and privacy for the Internet of Drones: Challenges and solutions," IEEE Commun. Mag., vol. 56, no. 1, pp. 64–69, 2018, doi: 10.1109/MCOM.2017.1700390.

[15] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[16] M. Conti, E. Sandeep Kumar, C. Lal, and S. Ruj, "A survey on security and privacy issues of Bitcoin," IEEE Commun. Surveys Tuts., vol. 20, no. 4, pp. 3416–3452, 2018, doi: 10.1109/COMST.2018.2842460.

AUTHOR BIOGRAPHY

SHAHANUR ISLAM SHAGOR is an independent researcher and software engineer working across trustworthy artificial intelligence, autonomous and connected systems, cybersecurity, distributed systems, and edge computing. His research interests include secure V2X communication, blockchain-assisted cyber-physical security, privacy-preserving machine learning, autonomous-system resilience, and reproducible software validation. He focuses on translating security mechanisms into experimentally testable system designs with explicit threat boundaries and operational failure modes.