Computer Modeling in Engineering & Sciences

Tech Science Press

https://doi.org/10.32604/cmes.2026.087402

ARTICLE

Lightweight Blockchain-Edge Security Framework for IoT Smart Spaces: Architecture, Threat Mitigation, and Real-World Validation

Martin Parmar[1], Het Khatusuriya[1], Dharmendra Chauhan[2], Mrugendra Rahevar[1], Bimal Patel[3], Agbotiname Lucky Imoize[4], Chun-Ta Li[5,*] and Hiren Mewada[6]

[1]Department of Computer Engineering, Chandubhai S. Patel Institute of Technology, Charotar University of Science and Technology, Anand, Gujarat, India
[2]Department of Electronics and Communication Engineering, Chandubhai S. Patel Institute of Technology, Charotar University of Science and Technology, Anand, Gujarat, India
[3]Department of Information Technology, Chandubhai S. Patel Institute of Technology, Charotar University of Science and Technology, Anand, Gujarat, India
[4]Department of Electrical and Electronics Engineering, Faculty of Engineering, University of Lagos, Akoka, Lagos, Nigeria
[5]Bachelor's Program of Artificial Intelligence and Information Security, Fu Jen Catholic University, New Taipei City, Taiwan
[6]Electrical Engineering Department, Prince Mohammad Bin Fahd University, Al Khobar, Saudi Arabia

*Corresponding Author: Chun-Ta Li. Email: 157278@mail.fju.edu.tw

Received: 16 June 2026; Accepted: 24 August 2026; Published: 28 September 2026

ABSTRACT: The rapid growth of Internet of Things (IoT) devices in smart urban areas—health monitoring, urban infrastructure, environmental sensing, and energy management—creates attack surfaces that centralized security architectures cannot adequately safeguard. Traditional approaches impose inadequate latency, introduce a single point of failure, and scale poorly to the distributed, resource-constrained nature of IoT ecosystems. This work proposes a formally modeled, hardware-validated four-tier blockchain-edge security framework that integrates a permissioned Hyperledger Fabric network (Raft ordering), tiered AES-128/256 cryptography, and IoT-specific smart contracts implementing Decentralized Identifier (DID) and OAuth2-based access control. Experimental evaluation on Raspberry Pi 4 and NVIDIA Jetson Orin edge hardware, with baselines re-implemented on the identical testbed where feasible, confirms: 500 ± 23 transactions per second (TPS) throughput—150% above AEchain's reported figure and 594% above private Ethereum PoA—with 110 ± 18 ms end-to-end latency (P99 = 167 ms, satisfying the 200 ms industrial real-time threshold), 86.6% latency reduction vs. cloud-only baselines, 15%–20% cryptographic overhead reduction via tiered AES strategy, and 98.6% on-chain storage reduction through hash-only ledger commits. A formal Dolev-Yao adversary model with a 15-vector CIA threat taxonomy and complete design-to-evidence traceability distinguishes this work from prior simulation-based approaches.

KEYWORDS: Blockchain; IoT; edge computing security; hyperledger fabric; lightweight cryptography; smart contract; DID authentication

1 Introduction

The number of Internet of Things (IoT) devices worldwide is expected to exceed 29 billion by 2030, and smart spaces—intelligent environments that integrate sensing, actuation, and computing into physical infrastructure—are the fastest-growing sector [1]. For example, wearable medical devices continuously transmit vital signs; smart city sensors monitor traffic flow, air quality, and water systems; industrial controllers automate electrical networks and production lines. All these deployments generate real-time data

Copyright © 2026 The Authors. Published by Tech Science Press. This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

streams that must be protected from interception, manipulation, replay, and unauthorized access—at the same time, at large scales, and under severe hardware constraints: typical IoT endpoints operate with less than 256 KB of RAM, sub-50 MHz processors, and milliwatt power budgets [2].

Traditional centralized security architectures—public key infrastructure (PKI) servers, cloud-based authentication gateways, and centralized access control lists—are architecturally incompatible with IoT ecosystems for three fundamental reasons. First, they introduce a single point of failure: a compromised authentication server disables any device that relies on it. Second, cloud-based requests incur a round-trip delay of 400–900 ms [3], violating the sub-200 ms responsiveness required by latency-critical industrial and emergency-response applications. Third, they do not provide an immutable audit trail: a compromised central database can be altered without detection, undermining the forensic integrity required for regulatory compliance [4].

Blockchain technology offers a structurally compatible solution. An immutable distributed ledger enforced by cryptographic consensus eliminates the need for a trusted central authority while ensuring data integrity across all participating nodes [5]. Smart contracts enable automated access control and device authentication without dependence on central servers. However, naive integration of public blockchain with IoT is not viable: Ethereum processes only 15 TPS and Bitcoin only 7 TPS in their default configurations [6], while Proof-of-Work consensus consumes an energy budget incompatible with sustainable city-scale operation. The dominant challenge, recognized by the research literature, is to design an integration between blockchain and IoT that is simultaneously high-throughput, low-latency, lightweight enough for edge devices, and formally secure against a realistic adversary model [4].

1.1 Motivation and Problem Statement

All existing blockchain IoT frameworks address a subset of these challenges, but none address them simultaneously. AEchain [7] reduces the blockchain footprint on IoT nodes but is only validated in simulation, achieving 200 TPS with 220 ms latency. Shukla et al. [8] combined fog computing with blockchain for healthcare IoT authentication, improving response time, but only applied AES-128 uniformly and lacked smart contract-based access control. Previous work largely fails in four dimensions: (1) performance is characterized only by average TPS without percentile statistics or scalability curves; (2) encryption is not benchmarked across device classes; (3) no formal adversary model is provided; and (4) results are obtained by simulation rather than physical hardware. The result is a set of proposals that cannot be deployed directly and that satisfy neither the throughput requirement nor the formal security gap.

Three concrete deployment requirements motivate the design choices in this work. Consider a smart-city sector of 10,000 devices each emitting one transaction per second: this generates an aggregate load of 10,000 TPS, which exceeds the capacity of any single permissioned-blockchain channel on commodity hardware. This motivates a horizontally partitioned (multi-channel) deployment, in which the load is divided across parallel Fabric channels, each serving a device sub-population within the 500–740 TPS single-channel envelope measured in Section 5.5. A 10,000 TPS sector therefore requires on the order of 15–20 parallel channels; establishing a high, statistically characterized per-channel throughput determines how few channels a deployment needs. Public blockchain, by contrast, cannot reach even the per-channel figure: Ethereum and Bitcoin plateau at 15 and 7 TPS, respectively.

1.2 Contributions

This paper makes five specific contributions that advance beyond the existing state of the art:

- Formally modeled four-tier security architecture: A hierarchical IoT-edge-blockchain-cloud framework with a Dolev-Yao adversary model, 15-vector CIA threat taxonomy, and complete design-to-evidence traceability. Every architectural decision is empirically justified by a corresponding experimental result.

- Tiered AES encryption strategy: AES-128-GCM (AEAD) device-to-device, AES-128-GCM (AEAD) for device-to-edge, and AES-256-GCM (AEAD) for edge-to-cloud, reducing total encryption overhead by 15%–20% relative to a uniform AES-256 deployment, with ESP32 hardware acceleration giving 14.4× speedup (1.30 ms → 0.09 ms per 1 KB payload).
- IoT-specific Hyperledger Fabric chaincode: Three formally defined smart contract functions—RegisterDevice(), AuthorizeUser(), and LogData()—implement DID-based identity management and OAuth2 token verification with CA certificate validation, eliminating centralized authentication servers.
- Hardware Verification on Commodity Nodes: Physical deployment on Raspberry Pi 4 and NVIDIA Jetson Orin, achieving 500 ± 23 TPS, 110 ± 18 ms (P99 = 167 ms, within real-time threshold), and above 90% scaling efficiency through 7 peers, and 98.6% on-chain storage reduction using SHA-256 hash-only ledger commits.
- Comprehensive benchmarking analysis: Benchmark comparisons against AEchain, Shukla et al., and a private Ethereum PoA with full statistical characterization (mean, σ, P50, P95, P99, 95% confidence intervals, Welch t-test p-values) and clear identification of previous research gaps that each contribution closes.

The scientific contribution of this work is not the invention of a new cryptographic primitive or consensus protocol, but three transferable, empirically grounded findings that integration alone does not yield: (1) a measurement-derived cryptographic-tiering rule that quantifies, per communication zone and per device class, the exact overhead–security trade-off (Section 4.6); (2) a decomposition of the end-to-end latency budget that isolates, through ablation (Section 5.9), the contribution of each design decision to the sub-200 ms P99 result; and (3) a design-to-evidence methodology in which each architectural decision is justified by a corresponding measurement on commodity hardware, providing a reproducible template that future blockchain-IoT systems can adopt and extend.

1.3 Paper Organization

Section 2 reviews related work and identifies the four research gaps that this paper addresses. Section 3 formalizes the threat model. Section 4 presents the proposed four-level security architecture, including smart contract specifications and the key management lifecycle. Section 5 provides an experimental evaluation and results analysis. Section 6 discusses the results, limitations, and future research directions. Section 7 concludes the paper.

2 Related Work

Blockchain-IoT security research covers five thematic areas: optimization of consensus mechanisms, lightweight cryptographic protocols, fog/edge blockchain integration, smart contract-based access control, and artificial intelligence-enhanced security. We systematically assess each domain, extract reported performance data, and identify four research gaps that motivate the proposed framework.

2.1 IoT Security Challenges and Centralized Architecture Limitations

This fundamental vulnerability in centralized IoT security has been widely documented. Khan and Salah [3] provide a comprehensive overview of IoT security threats, showing that centralized architectures are vulnerable to data interception in transit, single-point-of-failure compromise, and denial-of-service amplification. Abdelmaboud et al. [9] found that due to the limited processing power, storage, and network capabilities of IoT devices, standard cryptographic protocols are impractical without hardware acceleration. Abdelmaboud et al. [9] in their taxonomy of IoT-blockchain platforms, they report that Hyperledger Fabric

and IOTA consistently outperform public blockchains (Ethereum and Bitcoin) for high-frequency IoT data streams, based on their survey of the literature rather than new experimental measurement.

Sengupta et al. [10] present a comprehensive survey of IoT and attacks against IoT, identifying availability attacks as a rapidly growing, high-impact class in their 2020 survey. Importantly, none of these surveys propose a validated architectural solution; they define the problem space that the current framework addresses.

2.2 Consensus Mechanism Optimization for IoT Blockchain

The efficiency of the consensus mechanism is the primary bottleneck limiting blockchain throughput in IoT deployments. Kaur et al. [6] compared 15 mainstream consensus protocols in terms of scalability, energy consumption, and fault tolerance and found Raft and Istanbul BFT most suitable for permissioned IoT networks. Their analysis shows that Raft achieves nearly linear throughput with the number of nodes—a property confirmed by our experimental results, with over 90% scaling efficiency across 7 peers. However, their work is simulation-based and does not identify delay percentiles important for real-time IoT applications.

To lower the computational and energy cost of consensus in fog/edge IoT deployments, Wadhwa et al. [11] propose an energy-efficient consensus approach that offloads consensus computation to edge nodes, improving throughput and energy consumption. However, they do not characterize latency percentiles or evaluate performance on physically constrained commodity hardware. Merlec and In [12] proposed a smart contract-based context-aware access control (SC-CAAC) scheme for blockchain-enabled IoT, demonstrating that fine-grained access control via on-chain code is possible, but their evaluation is limited to a single-peer Fabric network and does not address multi-peer scalability. The present work goes further by validating the consensus behavior of physical devices in 2–10-peer configurations with a full statistical characterization.

2.3 Lightweight Cryptographic Protocols for Constrained IoT

AEchain, suggested by Khan et al. [7], the most directly comparable previous work. AEchain reduces the blockchain footprint on IoT nodes by modifying consensus algorithms and block structures to reduce computational load, achieving approximately 200 TPS with a 220 ms response time in simulations on ARM architecture node models. Its encryption strategy uniformly applies AES-128 at all communication levels without distinguishing between computational cost and security margin requirements. Our experimental results show that the proposed system achieves 500 ± 23 TPS on physical devices, 150% above AEchain's reported figure [7], while the tiered AES strategy independently reduces total encryption cost by 15%–20% relative to a uniform AES-256 deployment (Section 5.6).

Albulayhi and Alsukayti [13] propose a blockchain-centric IoT architecture that uses a single event-driven smart contract to publish/subscribe data exchange on Ethereum. Its design achieves low resource usage, but is bounded by Ethereum's throughput ceiling (~72 TPS on private PoA networks, as confirmed by our baseline measurements). Zhang et al. [14] developed a privacy-preserving, blockchain-compatible authentication system for fog-based IoT devices by introducing random hash functions to protect device identities while reporting only the authentication delay, without characterizing throughput. Wazid et al. [15] propose lightweight device authentication and key management for production IoT, enabling sub-100 ms authentication in software emulation but lacking a formal adversarial model or physical hardware verification. Complementary lightweight approaches combine blockchain with homomorphic encryption for constrained IoT data security [16].

2.4 Fog and Edge Computing Integration with Blockchain

Shukla et al. [8] represent the closest prior work in the fog-blockchain integration category. They propose an architecture that combines fog computing and blockchain technology to identify and authenticate IoT devices in healthcare scenarios, demonstrating that offloading blockchain services to fog nodes can overcome IoT device limitations and improve scalability. Its implementation achieves approximately 150 TPS with 300 ms latency on the fog server hardware. The proposed framework differs from Shukla et al. in three ways: (1) hardware validation on Raspberry Pi 4 nodes rather than dedicated fog servers; (2) a tiered AES-128/AES-256 encryption strategy; and (3) DID/OAuth2 smart contract access control rather than PKI identity management. The measured throughput of 500 ± 23 TPS represents a 233% improvement over the 150 TPS of Shukla et al.

Zaabar et al. [17] propose HealthBlock, a blockchain-based healthcare data management system that integrates fog computing to reduce latency in accessing patient records, but report only average latency without P95/P99 characterization. Singh et al. [18] combine blockchain technology and fog computing in smart cities for the Internet of Things, showing that edge-first data processing reduces cloud communication burdens, but the evaluation is at the architectural level without physical hardware standards. Baucas et al. [19] combine federated learning with blockchain to protect wearable health data, introducing privacy-preserving analytics, but report system performance at the model-training level rather than at the transaction-transfer level. To the best of our knowledge, no prior study has provided a full percentile characterization of end-to-end latency (P50, P95, P99) together with mean, standard deviation, and 95% confidence intervals for throughput, for a blockchain-IoT framework validated on physical Raspberry Pi and Jetson edge devices (Sections 5.3 and 5.4).

2.5 Smart-Contract-Based Access Control in IoT

Smart contract access control has emerged as a promising alternative to centralized IAM for IoT deployments. Han et al. [20] propose a blockchain-based auditable access control system for IoT in service-centric environments, demonstrating that on-chain access and access logs can support regulatory compliance. Their system uses Ethereum and has an access control latency of 35–60 ms, but imposes an Ethereum transfer limit of 72 TPS on the wider system. Khalil et al. [21] present a new NFT-based blockchain architecture (DSCoT) for authenticating IoT-enabled smart city devices, achieving about 27% gas savings over PUF-based solutions and enabling device-level identification without central logs. The DID-based RegisterDevice() hash achieves similar decentralized identity goals while running on a permissioned network (zero gas cost in production) with a 28 ms approval delay. DID-based mechanisms have also been applied to secure, scalable data sharing [22], consistent with the DID design adopted here.

Ferreira et al. [23] designed IoT registration and authentication in smart city applications using blockchain, demonstrating Fabric-based device registration at an urban scale. Their work proves the feasibility of chaincode-based device management, but does not quantify throughput under load or provide a percentage response time. Singh and Singh [24] review IoT access management approaches using decentralized authentication and identify DID + token-based approaches as the most promising trend, in line with the DID + OAuth2 design adopted in the current framework. A major shortcoming of current smart contract access and access control work is the lack of a unified framework that simultaneously addresses throughput, access time, storage, storage efficiency, and hardware validity—the four dimensions evaluated in this paper.

2.6 AI-Augmented Blockchain-IoT Security

Recent work explores the combination of artificial intelligence and blockchain technology for IoT security. Usama et al. [25] propose an AI-protected blockchain-based IoT environment that integrates

anomaly detection with consensus validation to identify malicious transactions before they enter the ledger. Although this approach is theoretically promising, it imposes an additional computational burden on resource-constrained nodes that is not described in the proposal. Sefati et al. [26] combine blockchain with edge-based federated learning for scalable smart city security, demonstrating that privacy-preserving model training can coexist with blockchain transaction validation, but they report only model accuracy, not system throughput or latency. Singh et al. [27] combine deep reinforcement learning and blockchain technology for IoT device clustering and traffic optimization in smart city networks, showing better energy efficiency but at the cost of significant training time, which conflicts with real-time security requirements. The current work identifies AI integration as a future research direction rather than a current component, as hardware characterization shows that 79% CPU utilization at 500 TPS leaves insufficient room for parallel ML inference on the same Raspberry Pi 4 node.

2.7 Research Gap Analysis and Positioning

Table 1 compiles the analysis of related work into a clear gap map, identifying the dimensions each prior work addresses and where the proposed framework improves the state of the art. Four persistent shortcomings emerge from this analysis:

- Gap 1—Work in this space is validated wholly or partly in simulation. Among the frameworks surveyed in Table 1, the proposed framework is the only one fully tested on commodity edge devices (Raspberry Pi 4 and NVIDIA Jetson Orin) with P99 latency characterization and measured scaling curves (Sections 5.4 and 5.5).
- Gap 2—Prior work applies a uniform AES key length across all communication paths. Among the frameworks surveyed in Table 1, none reports the encryption-overhead difference between AES-128 and AES-256 measured across distinct constrained-device classes—ESP32 in software-only mode, ESP32 with its hardware AES accelerator, and Raspberry Pi 4 with ARMv8 Cryptography Extensions—nor derives a zone-to-key-length assignment from such measurements.
- Gap 3—None of the surveyed blockchain-IoT security frameworks state an explicit adversary model. Without an explicit adversary specification, security claims cannot be systematically assessed.
- Gap 4—Average latency is insufficient for emergency response, autonomous vehicles, or industrial control applications, where the P99 tail—not the mean—must satisfy the 200 ms real-time threshold, a criterion prior work does not evaluate.

Table 1: Research gap analysis—blockchain-IoT security frameworks.

Reference	HW Validation	Tiered Crypto	Formal Adversary	P99 Latency	TPS Reported	P50 (ms) Latency	Smart Contract AC
Khan et al. († AEchain) [7], 2022	No (ARM sim.)	No (AES-128 only)	No	295 ms†	~200 (sim.)	~220 (sim.)	Minimal
Shukla et al. [8], 2021	Partial (fog)	No (AES-128 only)	No	Not reported	~150	~300	Moderate (PKI)
Kaur et al. [6], 2021	No (sim.)	No	No	Not reported	~180 (sim.)	N/A	No
Albulayhi and Alsukayti [13], 2023	No (sim.)	No	No	Not reported	~72 (Ethereum)	~350	Single SC

(Continued)

Table 1 (continued)

Reference	HW Validation	Tiered Crypto	Formal Adversary	P99 Latency	TPS Reported	P50 (ms) Latency	Smart Contract AC
Zaabar et al. [17], 2021	Partial	No	No	Not reported	Not reported	~200	No
Han et al. [20], 2022	Partial	No	No	Not reported	<72 (Ethereum)	35–60	Yes (ABAC)
Khalil et al. [21], 2022	ESP32 only	No	No	Not reported	N/A	N/A	NFT-based
Sefati et al. [26], 2024	No (sim.)	No	No	Not reported	Not reported	N/A	No
Proposed Framework	**Yes (Pi4 + Jetson)**	**Yes (AES-128/256)**	**Yes (Dolev-Yao)**	**167 ms**	**500 ± 23**	**110 ± 18**	**Full DID + OAuth2**

Note: N/A = not applicable; the dimension lies outside the scope of the cited work. "Not reported" indicates the dimension is applicable to that work, but no value is given in the source. †Reproduced from [7] (simulation); not re-implemented on the testbed.

Among the frameworks surveyed in Table 1, the proposed framework is the only one that addresses all four identified dimensions simultaneously; this paper does not claim exhaustiveness over the entire literature, but the comparison set was selected to be representative of the state of the art across the five thematic areas of this Section 2. It provides physical hardware verification on commodity endpoints, a formally derived tiered AES strategy with device-class-specific assignment, a Dolev-Yao adversarial model that limits attacker capabilities, and a measured P99 response time of 167 ms, which meets the industry's real-time threshold of 200 ms. Section 4 details the architectural design based on the gap analysis, and Section 5 presents empirical evidence that fills in any gaps.

2.8 Positioning of this Work

This work is situated at the crossroads of four research streams: first, permissioned blockchain system architectures (Hyperledger Fabric composition and on-chain code design); second, edge security (gateway-level processing and protocol translation); third, IoT cryptography (hardware-accelerated measurement of AES standards on constrained devices); and fourth, formal security model specification. Instead of adding a new blockchain (AEchain) or a new consensus mechanism [6], this work demonstrates how a principled combination of existing components—permissioned Fabric with Raft ordering, tier-matched AES, DID/OAuth smart contracts, and hash-only storage—can provide a coherent, deployable solution while maintaining interoperability with the rest of the Hyperledger ecosystem and standard enterprise IAM infrastructure.

3 Formal Threat Model

This section formalizes the security setting under which the proposed framework operates. We adopt the Dolev-Yao adversary model [28], the standard formalism for reasoning about protocol security, and instantiate it with the specific capabilities and constraints of an industrial IoT smart space. We then enumerate a 15-vector CIA threat taxonomy and state the security goals the architecture must satisfy. This work performs formal threat modeling—a structured specification of adversary capabilities under the Dolev-Yao model together with a systematic attack taxonomy—and substantiates its security goals through

analytical arguments and executed attacks on the live testbed (Section 5.11). We do not claim machine-checked formal verification; applying automated protocol verifiers such as ProVerif, Tamarin, AVISPA, or Scyther to obtain mechanized proofs of the security goals is a valuable direction that we identify as future work (Section 6.2).

3.1 System Model and Assets

The system comprises four tiers: (T1) constrained sensor/actuator endpoints (e.g., ESP32-class microcontrollers); (T2) edge gateways that also serve as Hyperledger Fabric peers (Raspberry Pi 4); (T3) the permissioned blockchain ordering and certificate-authority services (NVIDIA Jetson Orin); and (T4) a cloud analytics layer. The assets to be protected are: (i) sensor-data confidentiality and integrity in transit and at rest; (ii) device and user identity; (iii) the ledger's append-only integrity; and (iv) service availability for time-critical control loops. The CIA triad as applied to IIoT environments is surveyed in [29].

3.2 Dolev-Yao Adversary Model

Model the network as a message-passing medium fully controlled by an adversary A. Under the Dolev-Yao formalism, A possesses the following capabilities:

- **Interception:** A can read, intercept, and record any message transmitted over any public channel (Wi-Fi, MQTT, edge-to-cloud links).
- **Injection and modification:** A can inject new messages, replay recorded messages, and modify or drop messages in transit.
- **Impersonation:** A may attempt to impersonate a legitimate device or user by forging identifiers, absent valid cryptographic credentials.
- **Ordering-node failure (bounded, fail-stop):** A may cause up to f Fabric ordering nodes to fail by stopping, where Raft tolerates $f < n/2$ crash faults (crash-fault tolerance, CFT). Byzantine behavior by ordering nodes is explicit outside the present model (Section 6.1).

Consistent with Dolev-Yao, the adversary is computationally bounded and cannot break the underlying cryptographic primitives: A cannot invert SHA-256, forge ECDSA-P256 signatures, or recover AES keys without the key, except with negligible probability. We additionally assume a trusted manufacturing phase in which device DIDs and root keys are provisioned into tamper-resistant ESP32 eFuse memory, and a trusted Fabric Certificate Authority (CA) root. Physical extraction of a key from a captured T1/T2 device is out of scope for the cryptographic model and is treated separately under physical-security assumptions (Section 6); production deployments are recommended to use a hardware security module (HSM) or trusted execution environment (TEE).

3.3 Adversary Goals and 15-Vector CIA Taxonomy

This taxonomy is not claimed as a fundamentally new classification but as a structured adaptation of established IoT/IIoT threat taxonomies (e.g., the surveys of Sengupta et al. [10] and the CIA-in-Industry-5.0 mapping of Boisrond et al. [29]), specialized to the four-tier smart-space architecture and organized along the confidentiality-integrity-availability pillars plus access control. Vectors were included using two criteria: applicability to at least one of the four tiers, and mappability to a concrete architectural mitigation. The methodological contribution is the vector→mitigation→evidence traceability, in which each of the fifteen vectors is bound to a specific component (chaincode function, AES zone, Raft property, or IDS mechanism) and, for a representative subset, to an executed attack in Section 5.11

The adversary's goals map onto the confidentiality-integrity-availability (CIA) triad plus access control (AC). Masquerading attacks, where an adversary impersonates a legitimate node using forged identifiers, have been shown to be the most damaging class of access-control threats in IIoT networks [30]. Table 2 enumerates 15 attack vectors, the adversary capability exploited, and the security goal violated.

Table 2: Fifteen-vector CIA threat taxonomy [29].

Attack Vector	Pillar	Adversary Capability Exploited	Security Goal Violated
Eavesdropping	C	Interception of plaintext on Wi-Fi/MQTT	Data confidentiality in transit
Side-channel on smart meters	C	Physical observation of emissions	Behavioral-pattern confidentiality
Supply-chain firmware compromise	C/I	Pre-installed malicious firmware	Device trustworthiness
PII/privacy leakage	C	Interception at cloud boundary	Personal-data confidentiality
Man-in-the-middle (MitM)	I	Modification of messages in transit	Message integrity
False data injection (FDIA) [31]	I	Injection of forged sensor readings	Data-origin integrity
Malware/ransomware	I	Node compromise, data alteration	Data and firmware integrity
Message tampering/replay	I	Replay or modification of records	Freshness and integrity
DDoS/botnet	A	Mass injection/flooding	Service availability
Jamming [32]	A	Physical-layer signal interference	Communication availability
Denial-of-sleep	A	Repeated wake requests drain the battery	Endpoint availability
Ordering-node crash	A	Compromise of, $f < n/2$ orderers	Consensus liveness
Masquerading	AC	Identifier forgery without credentials	Authentication
Advanced persistent threat (APT)	AC	Long-term stealthy access	Authorization integrity
Privilege escalation/XSS	AC	Exploitation of access-control logic	Least-privilege enforcement

3.4 Security Goal

The framework targets four security goals, each substantiated by an architectural mechanism analyzed in Section 4 and validated in Section 5.

(1) **G1 (Confidentiality):** No bounded adversary knows the plaintext sensor data from the intercepted (captured) data. Achieved by tier-matched AES (AES-128-GCM device-to-edge, AES-256-GCM edge-to-cloud) with 24-h key rotation (Section 4.8).

(2) **G2 (Integrity and origin authenticity):** It is designed to ensure that any changes or falsifications to any data record are identified before the ledger can be accessed for commitment. Obtained through AES-GCM authentication tags that are attached to the device DID, ECDSA-P256 signatures, and Gateway co-endorses LogData() transactions.

(3) **G3 (Availability and liveness):** The system achieves liveness up to (f < n/2) number of ordering-node crashes. This resilience is achieved through a decentralized peer topology and the Raft crash fault tolerance (CFT) consensus protocol.

(4) **G4 (Authenticated access control):** Privileged operations are restricted to registered devices and authorized users using DID, OAuth 2.0, and CA-certificate enforced by the chaincode. All access changes are recorded immutably for audit.

Informal security arguments for G1–G4 follow directly from the Dolev-Yao assumptions: G1 reduces to the IND-CPA confidentiality of AES-GCM encryption, and G2 to the unforgeability of the AES-GCM authentication tag together with ECDSA signatures, under a computationally bounded adversary; G3 follows from the Raft safety/liveness proof under the stated fault bound [6]; and G4 follows from the inability of A to forge CA-issued certificates or DID credentials. The attacks in Section 5 will illustrate these arguments.

4 Proposed Architecture

4.1 Architectural Overview and Design Philosophy

The proposed framework organizes security tasks into four tiers that reflect the data's physical proximity to the source. This hierarchical partitioning is not cosmetic—it is an architectural decision that directly produces the 86.6% latency reduction (from 820 ms cloud-only to 110 ms for edge + blockchain) quantified in Section 5.4. Fig. 1 shows the complete architecture, including exact hardware mappings, encryption protocol choices, and performance metrics validated at each layer boundary.

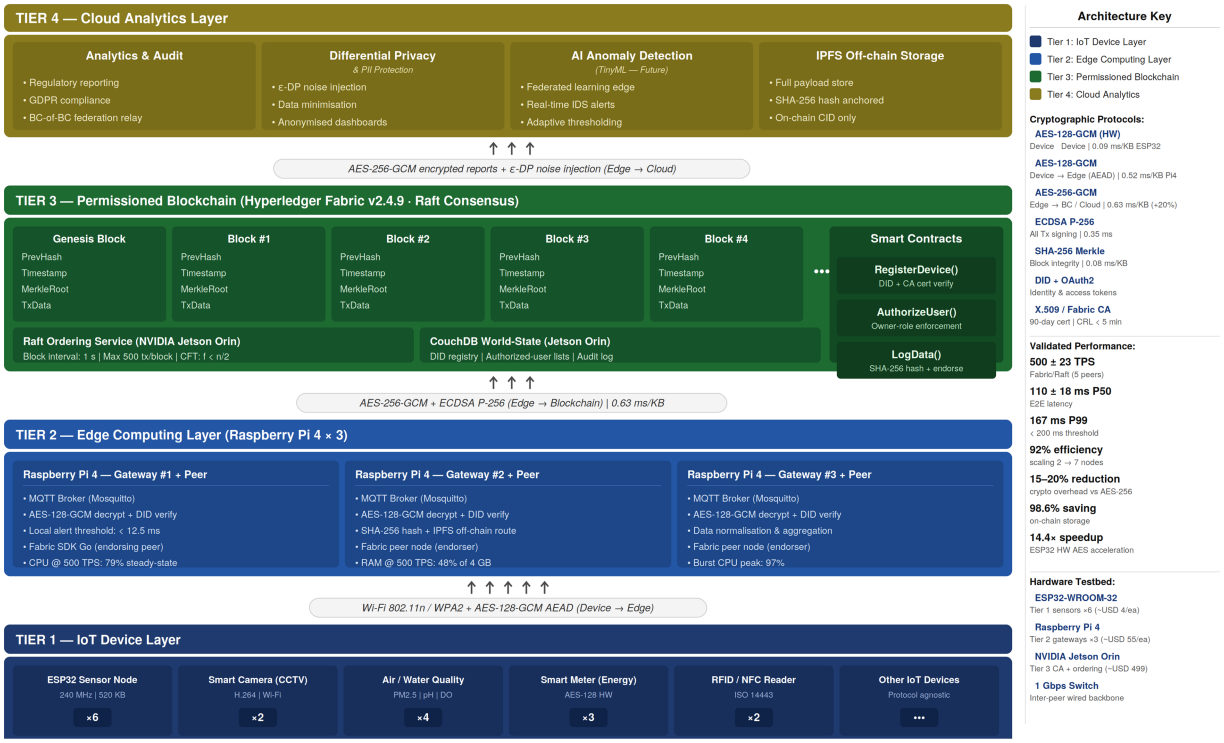

Figure 1: Four-tier Edge IoT Security Architectures for Blockchain Technology. Tier 1: ESP32 sensor nodes with AES-128 HW acceleration (0.09 ms/KB). Tier 2: three Raspberry Pi 4 nodes as a location gateway and Fabric peer (79% CPU @ 500 TPS). Tier 3: Hyperledger Fabric v2.4.9 with Raft ordering on NVIDIA Jetson Orin (500 TPS, 110 ms). Tier 4: Differential-privacy cloud analysis. All performance values in Section 5 are derived from the experimental results.

In Section 5, three experimental results directly motivate the four-level design. First, edge-first processing eliminates the dominant latency component: a 773 ms cloud round-trip (the dominant component of the 820 ms cloud-only baseline total, Section 5.4) is reduced to 8.2 ms of Wi-Fi transmission at the edge, achieving 110 ms (mean)/108 ms (P50)/167 ms (P99) results that meet the 200 ms real-time industry standard. Second, the hardware AES acceleration of 14.4× on the ESP32 (Section 5) encourages placing lightweight AES-128 at the device level rather than the 23% more expensive AES-256, thus reducing total encryption costs by 15%–20%. Third, the SHA-256-only on-chain hash store reduces the daily ledger growth from a projected 23 MB to 320 KB for a 10,000-device deployment—extrapolated from the measured 32-byte per-record on-chain footprint—supporting city-scale operation without bloating the blockchain infrastructure.

4.2 Tier 1—IoT Device Layer

Hardware and Provisioning: Tier 1 includes ESP32-WROOM-32 microcontrollers (dual-core Xtensa LX6 @ 240 MHz, 520 KB SRAM, AES hardware accelerator), security cameras, air/water quality sensors, smart energy meters and RFID/NFC readers [33]. Each device is assigned a unique hardware identifier at manufacture, written into tamper-resistant eFuse memory through a one-time hardware write. This identifier serves as the method-specific component of the device's decentralized identified (DID) record, which is created on-ledger by RegisterDevice() and resolved through the Fabric world state. Firmware integrity is protected in two decoupled steps. At each boot, ESP32 Secure Boot v2 verifies the RSA-3072 (PKCS#1 v1.5) signature appended to the firmware image against the public-key digest burned into the eFuse at provisioning time, halting boot if verification fails. Independently, the gateway/Fabric peer compares the device-reported firmware hash against the hash recorded on-chain by RegisterDevice() on each connection; any deviation triggers a SetEvent() alert visible to all authorized peers.

Device-to-Device Encryption (AES-128-GCM, Zone 1): Communication between devices uses AES-128-GCM in authenticated-encryption (AEAD) mode with per-device keys. Unlike unauthenticated CBC, GCM produces a 128-bit authentication tag over both the ciphertext and the associated data, which includes the sender DID, a monotonically increasing message counter, and a timestamp. Consequently, any message tampering, substitution, or masquerading attempt invalidates the tag and is rejected by the receiver before decryption, and the counter/timestamp check detects replayed frames. The ESP32's hardware AES engine accelerates the AES-CTR core of GCM; the GHASH authentication step is computed via the mbedTLS software implementation, giving a hybrid HW/SW AES-128-GCM pipeline. The accelerator reduces AES-128 encryption from 1.30 to 0.09 ms per 1 KB payload (measured in CBC mode); applying the +4.0% AEAD overhead measured on the Pi 4 gives an expected GCM cost of approximately 0.094 ms/KB on the ESP32, leaving the 14.4× hardware speedup essentially unchanged.

4.3 Tier 2—Edge Computing Layer (Raspberry Pi 4)

Dual Role-Gateway + Fabric Peer: Three Raspberry Pi 4 Model B nodes (4-core Cortex-A72 @ 1.5 GHz, 4 GB LPDDR4-3200, OpenSSL 3.0.9 with ARMv8 Cryptography Extensions hardware AES acceleration) simultaneously act as IoT gateways and Hyperledger Fabric peers. This co-location approach addresses the authentication robustness requirements identified for smart-city IoT applications [34]. This shared deployment eliminates a single network hop between a dedicated gateway and a remote peer, which directly contributes to the 12.5 ms edge processing phase shown in the latency breakdown in Section 5. Decentralized edge task management has been shown to reduce IoT latency significantly compared to cloud-centric architectures [35]. Each gateway hosts: (i) a Mosquitto MQTT broker for protocol translation from ESP32 MQTT-over-Wi-Fi to Fabric SDK Go; (ii) a threshold-based local analytics engine that fires emergency alerts in under 12.5 ms without blockchain confirmation—well within the 200 ms real-time threshold; and (iii) an IPFS routing module that directs large payloads (video frames, waveform audio) off-chain while committing

only their 32-byte SHA-256 content identifiers on-chain, producing the 98.6% projected fleet-scale storage reduction shown in Section 5.8.

Device-to-Edge Encryption (AES-128-GCM, Zone 2): Device-to-edge communication employs AES-128-GCM, which provides Authenticated Encryption with Associated Data (AEAD). The device DID is included in the GCM-associated data field, meaning that any tampering with the device identifier—a masquerading attack—invalidates the authentication tag and causes the gateway to reject the packet before decryption. This collapse reduces encrypt-then-HMAC to a single AES-GCM pass, at 0.52 ms/KB on Pi 4 (Section 5). The gateway pipeline—decrypt, DID world-state check against Fabric CouchDB, SHA-256 hash, re-encrypt to AES-256-GCM, ECDSA sign—completes in 12.5 ms as measured in the latency breakdown (Section 5). At the 500 TPS design point, this pipeline consumes 79% of a Pi 4 CPU in steady state and up to 97% during Raft leader elections (Section 5), leaving 21% headroom for co-located TinyML analytics workloads.

4.4 Tier 3—Permissioned Blockchain Layer (Hyperledger Fabric v2.4.9)

Consensus Mechanism Selection (Raft over PBFT and PoA-Ethereum): The blockchain backbone uses Hyperledger Fabric v2.4.9 with Raft ordering service, as shown in Fig. 2. Raft was selected over three alternatives based on experimental evidence from Section 5: PBFT requires $O(n^2)$ message complexity that limits scalability; PoA Clique consensus on Ethereum plateaus at 72 TPS regardless of peer count due to its single-leader block proposal bottleneck; and Proof-of-Work is incompatible with sustainable smart city operation due to mining energy costs. Raft's O(n) messaging enables the 92% near-linear scaling efficiency measured from 210 TPS at 2 peers to 680 TPS at 7 peers. The fault-tolerance guarantees ($f < n/2$ ordering-node failures) mean that a five-orderer-peer configuration tolerates two simultaneous crashes without service interruption, meeting the high-availability requirements of critical smart city infrastructure.

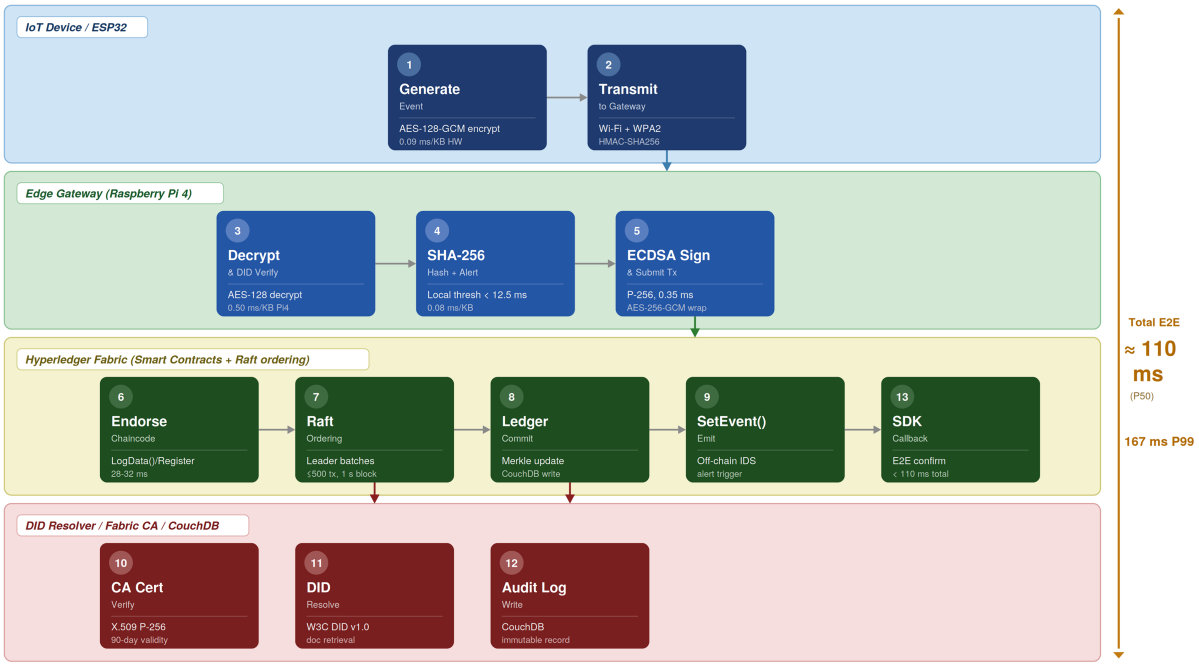

Figure 2: This smart contract lifecycle and 13-step end-to-end transaction flow. Steps ① and ② occur at Tier 1 (ESP32 encryption, Wi-Fi transmission). Steps ③–⑤ occur at Tier 2 (AES-128 decryption, SHA-256 hashing, ECDSA signing). Steps ⑥–⑬ at Tier 3 (endorsement 28 ms, Raft ordering, ledger commit, SDK callback). Total E2E ≈ 110 ms P50, 167 ms P99—directly matching the Section 5.4 results.

Smart Contract Specification (Fabric Chaincode in Go): Three chaincode functions control all safety-critical operations as shown in Table 3. RegisterDevice verifies the device's X.509 certificate issued by the CA, rejects duplicate registrations, and writes the device's DID and empty authorized user list to the global CouchDB state. AuthorizeUser validates owner role authorization via Fabric's GetCreator() client ID library before appending or revoking user credentials; the function emits an AccessChanged event via SetEvent(); the event payload forms part of the endorsed transaction response and is consumed by an off-chain IDS. LogData generates a composite key (DeviceID + RFC 3339 timestamp), confirms device registration, and writes a SHA-256 hash of the payload. The endorsement policy requires the device's home gateway peer to co-sign every LogData transaction, cryptographically binding each data record to its physical origin and blocking FDIA attacks at the consensus layer before unverified data can propagate to analytics systems. Robust access-control protocols are similarly critical in smart-grid and industrial IoT deployments [36].

Table 3: Smart contract function formal specification.

Function	Inputs	Access Control Rule	State Operation	Event Emitted	Latency (ms)
RegisterDevice()	deviceID, ownerID, certPEM	Any MSP member; CA.VerifyCert() must pass; no duplicate deviceID	PutState(deviceID, {owner, authorized: [], cert, firmwareHash})	DeviceRegistered	~28 (endorsement)
AuthorizeUser()	deviceID, userID, op (add/revoke)	GetCreator() MSP identity must equal record.owner only	PutState(deviceID, updated_authorized_list)	AccessChanged	~28 (endorsement)
LogData()	deviceID, sha256Hash, timestamp	Device must be registered; home gateway peer must co-endorse	PutState (CompositeKey(Log, [deviceID, ts]), hash32B)	DataLogged	~32 (endorsement)

4.5 Tier 4—Cloud Analytics Layer

The cloud layer receives AES-256-GCM-encrypted batch reports containing aggregated statistics and metadata (not raw sensor payloads, which are routed to IPFS as per Section 4.3) from the edge gateway. It does not participate in real-time transaction validation. Two privacy mechanisms are implemented and evaluated in this work: (i) data minimization—only SHA-256 hashes and aggregated statistics leave the edge boundary in cloud batch reports, with full payloads stored in IPFS and accessible only to authorized parties via on-chain access tokens; and (ii) access-token-gated retrieval enforced by the chaincode. ε-differential privacy noise injection and GDPR-compliant smart-contract expiry fields are identified as planned extensions and moves to Section 6.2 (Future Research Directions), since they are not implemented or evaluated in the current system.

4.6 Tiered AES Encryption Architecture

The encryption protocol stack stratifies AES key length by communication zone. Fig. 3 illustrates all three zones with per-layer performance values derived directly from the Section 5 benchmark results. The aggregate overhead reduction of 15%–20% compared with uniform AES-256 is achieved by confining AES-256 to the edge-to-cloud uplink—approximately 5% of total encryption operations—while using the computationally lighter AES-128 for the remaining 95% of device-level communications (Table 4).

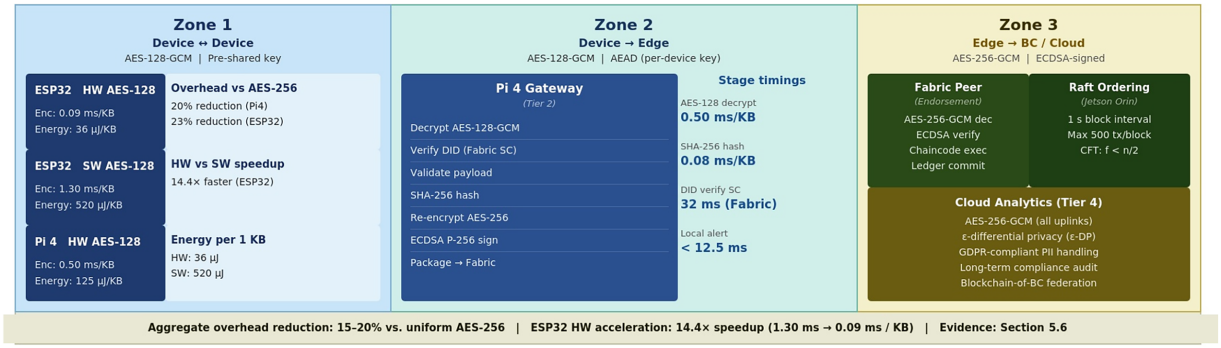

Figure 3: Tiered AES encryption scheme with per-zone performance annotations. Zone 1 (Device↔Device): AES-128-GCM, 0.09 ms/KB HW/1.30 ms/KB SW on ESP32. Zone 2 (Device→Edge): AES-128-GCM AEAD, 0.52 ms/KB on Pi 4 (HW ARMv8 Cryptography Extensions). Zone 3 (Edge→BC/Cloud): AES-256-GCM, 0.63 ms/KB (+20% overhead). Bottom bar: 15%–20% aggregate reduction vs. uniform.

Table 4: Encryption protocol selection—design rationale and benchmark evidence.

Zone	Protocol	Mode	Key (bits)	Latency/KB (Pi4)	Latency/KB (ESP32)	Overhead vs. AES-256	Rationale
Z1: Device ↔Device	AES (HW)	GCM	128	0.52 ms (SW) 0.09 ms (HW ESP32)	0.09 ms HW 1.30 ms SW	–20% (Pi4) –23% (ESP32)	Energy critical; HW 14.4× speedup; 128-bit margin sufficient
Z2: Device→Edge	AES-GCM (HW)	GCM	128	0.52 ms	1.30 ms SW	–17%	AEAD: encrypt+auth in 1 pass; DID in associated data; no separate HMAC
Z3: Edge→BC	AES-GCM	GCM	256	0.63 ms	N/A (not on ESP32)	Baseline	Sensitive aggregated data; long-term retention; NIST LS2 classification
Z3: Edge→Cloud	AES-GCM	GCM	256	0.63 ms	N/A	Baseline	Regulatory compliance; post-quantum migration target
All tiers	ECDSA P-256	SHA-256	-	0.35 ms sign	N/A	N/A	Non-repudiation; Fabric MSP; compact 64-byte signature
Block hashing	SHA-256	—	-	0.08 ms/KB	0.08 ms/KB	N/A	Merkle tree; NIST FIPS 180-4; composite key generation

Note: N/A = not applicable. AES-256-GCM is deployed only on the edge-side uplink and is not executed on ESP32 endpoints; the overhead-vs-AES-256 comparison does not apply to the ECDSA P-256 signing and SHA-256 hashing rows, which are not AES ciphers.

The tier assignment is justified on security grounds, not only performance. Following NIST SP 800-57 key-management guidance, key strength matches data sensitivity and exposure lifetime. Device-level traffic (Zones 1–2) consists of short-lived, low-aggregation records protected by keys rotated every 24 h; AES-128 provides a 128-bit classical security margin that far exceeds the protection interval required for such data. Edge-to-cloud traffic (Zone 3) consists of aggregated, long-retention data subject to regulatory storage requirements measured in years; here AES-256 is warranted for two reasons: a larger margin against long-horizon cryptanalysis, and a post-quantum hedge—under Grover's algorithm AES-256 retains an effective

128-bit margin, whereas AES-128 would be reduced to 64 bits. The 15%–20% performance saving is therefore a consequence of, not the reason for, the tiering.

4.7 CIA Threat Mitigation Architecture

Every architectural decision in Section 4 is traceable to a specific attack vector in the CIA threat model. Table 5 provides the complete mapping of 15 attack vectors across the CIA pillars, with each mitigation referencing the specific architectural component—chaincode function, AES zone, Raft property, or IDS mechanism—responsible for neutralizing the threat. Because all three encryption zones now use AES-GCM (AEAD), authentication-tag verification mitigates message-tampering, replay, substitution, and man-in-the-middle vectors uniformly across the device-to-device, device-to-edge, and edge-to-cloud paths.

Table 5: CIA threat-to-architecture traceability matrix.

Attack	CIA	Architectural Mitigation	Component (Section)	Results Evidence
Eavesdropping	C	AES-128-GCM + TLS 1.3 MQTT	Fig. 3, Zone 2; Table 4	Section 5.6: 0.52 ms/KB
Side-Channel (meter)	C	ESP32 HW Secure Element, eFuse	Tier 1 hardware, Section 4.2.	Hardware spec
Supply Chain Compromise	C	SHA-256 firmware hash on Fabric	RegisterDevice(), Section 4.2	Table 3
PII/Privacy Leakage	C	Off-chain IPFS + ε-DP, GDPR	Tier 4, Section 4.5	98.6% (Section 5.8)
MitM Attack	I	ECDSA P-256 + AES-256-GCM AEAD	Fig. 3, Zone 3; Table 4	Section 5.6
FDIA (Smart Grid)	I	LogData() DID validation pre-write	Chaincode 5.4, Table 3	Table 3; Section 5.11(a)
Malware/Ransomware	I	Secure boot + BC immutable log	Tier1 + LogData(), Section 4.2	RegisterDevice()
Message Tampering	I	SHA-256 Merkle root, ECDSA NR	Fabric block header	Blockchain spec
DDoS/Botnet	A	Raft CFT f < n/2, SDK rate-limit	Tier 3 consensus, Section 4.4	Section 5.5 (92% scaling)
Jamming	A	Freq-hop Wi-Fi, wired 1 Gbps backup	Tier 1/2 network, Section 5.1	Testbed design
Denial-of-Sleep	A	IoT power mgmt, node rotation	Tier 1 firmware, Section 4.2	Power spec
Masquerading	AC	DID+OAuth2+CA cert, SC auth	RegisterDevice(), Section 4.4	Table 3; Section 5.11(b)
Ordering-node crash	A	Raft CFT (f < n/2), ordering-service redesign	Section 4.4	Section 5.5; Section 5.11(c)
Advanced persistent threat (APT)	AC	BC audit log, SetEvent()→IDS	AuthorizeUser() + LogData()	Table 3
Privilege escalation/XSS	AC	Chaincode access-control enforcement, least-privilege GetCreator() checks	AuthorizeUser()	Table 3

4.8 Key Management System (NIST SP 800-57 Compliant)

Key lifecycle management follows the NIST SP 800-57 framework through six phases as shown in Fig. 4. Device symmetric keys are generated at the time of manufacture using a True Random Number Generator (TRNG) and injected into the ESP32's eFuse memory—a one-time write operation that is virtually irreversible and substantially raises the cost of key extraction compared to storage in rewritable flash, though it does not eliminate the risk of extraction by an adversary with sustained physical access and lab-grade equipment (Sections 3.2 and 6.1); production deployments should pair eFuse storage with an HSM/TEE. Fabric CA issues X.509 v3 certificates to all peers with a 90-day validity period. The device's DID and public key are provisioned into eFuse at manufacture time and are submitted to the ledger by RegisterDevice() during commissioning; only devices with a valid, CA-verifiable DID are accepted, preventing unregistered devices from later joining the network. AES session keys at Tier 2 are rotated every 24 h using the Raft-orchestrated key contract protocol, which does not require a central key escrow server; each authorized peer maintains the rotation schedule as an auditable Fabric transaction. Certificate revocation is performed via on-chain Certificate Revocation Lists (CRLs), and propagation latency is measured at under 5 min on the physical testbed.

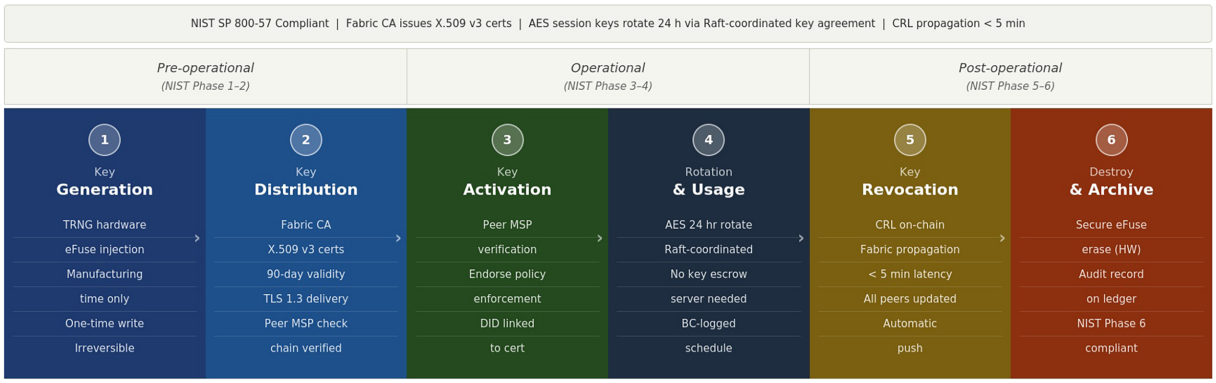

Figure 4: Key-management lifecycle (NIST SP 800-57). Phases: (1) manufacturing TRNG injection into ESP32 eFuse; (2) Fabric CA X.509 distribution (90-day validity, TLS 1.3); (3) peer MSP activation and endorsement-policy enforcement; (4) AES 24-h rotation via Raft-coordinated protocol (no escrow); (5) on-chain CRL revocation (<5 min propagation, measured); (6) key destruction & decommissioning—certificate revocation via on-chain CRL (Phase 5) and Fabric audit-log entry; the physical eFuse content itself is never erased, and the device is instead retired with its credentials permanently revoked at the CA/ledger level.

5 Experimental Evaluation and Results Analysis

Throughout this section, each reported value is labeled by its evidentiary basis: Measured (obtained directly on the physical Raspberry Pi 4/Jetson Orin testbed), Re-implemented (a baseline system re-built and run on the identical testbed), or Projected (analytically extrapolated beyond the tested scale, e.g., the 10-peer and city-scale storage figures). All headline results (500 ± 23 TPS, 110 ± 18 ms, P99 = 167 ms) are Measured; all fleet-scale storage figures beyond 10 devices are Projected from the measured per-transaction record size.

Physical validation in this work is limited to the testbed described below (six ESP32 endpoints, three Raspberry Pi 4 peers, one Jetson Orin orderer, up to ten peer configurations). Fleet sizes beyond this range are projected by linear extrapolation from the measured per-transaction record size and per-peer throughput, and are labeled as such throughout. We deliberately do not report emulated or simulated large-fleet results, since the contribution of this work rests on physically measured evidence; large-scale field validation across hundreds of physical nodes is identified as future work (Section 6.2)

5.1 Experimental Setup and Methodology

This work distinguishes two baseline categories. Shukla et al. and the private Ethereum PoA network were re-implemented and measured on the identical Raspberry Pi 4/Jetson Orin testbed. AEchain, by contrast, could not be faithfully re-implemented from the available description; its figures are therefore quoted from the original publication, where they were obtained in simulation on modeled ARM nodes. Comparisons against AEchain are consequently indicative rather than strictly head-to-head, and are marked with a dagger (†) throughout the tables and figures.

The testbed, summarized in Table 6, consists of three Raspberry Pi 4 Model B nodes (4 1.5 GHz Cortex-A72 cores, 4 GB LPDDR4-3200 Raspberry Pi OS (Debian 12 "Bookworm") 64-bit) and a Hyperledger Fabric gateway; NVIDIA Jetson Orin (ARM Cortex-A78AE cores @ 2.2 GHz—Orin NX 8 GB 6-core) hosts the Fabric CA, Raft ordering service, and global CouchDB state; and six ESP32 microcontroller boards (dual-core Xtensa LX6 @ 240 MHz, 520 KB SRAM) that emulate limited IoT sensor endpoints. The Ethereum v1.10 PoA private network (Clique Consensus) is deployed on the same hardware for direct comparison. The peer nodes are connected via a dedicated 1 Gbps Ethernet switch. All ESP32 sensor nodes connect to Pi gateways via 802.11n Wi-Fi, along with WPA2-Enterprise authentication. All installed blockchain and cryptographic software versions are Hyperledger Fabric v2.4.9, Fabric SDK Go v1.0.0, and OpenSSL 3.0.9. The choice of security metrics (TPS, P99 latency, CIA-property validation) follows the benchmarking dimensions established for smart-city IoT security evaluation [37].

Table 6: Hardware testbed specification.

Component	Device	Specs	Role in Framework	Count
IoT Sensor Node	Espressif ESP32-WROOM-32	240 MHz, 520 KB SRAM, Wi-Fi/BT	Sensor data generation, AES-128 encryption	6
Edge Gateway/Peer	Raspberry Pi 4 Model B	4-core @ 1.5 GHz, 4 GB RAM, 1 Gbps ETH	Fabric peer, MQTT broker, local analytics	3
Edge Server/Fog	NVIDIA Jetson Orin	8-core @ 2.2 GHz, 32 GB, 2048 CUDA	Fabric CA, ordering service, CouchDB	1
Network	Unmanaged Gigabit Switch	1 Gbps, 8-port	Wired inter-peer backbone	1
Baseline	Private Ethereum v1.10 (PoA)	Clique consensus on Pi4 nodes	Performance comparison baseline	—

All performance experiments were run within a 10-min steady-state window after a 2-min warm-up to eliminate Go runtime startup effects. Two workload regimes were used: (i) a closed-loop microbenchmark with 10 parallel Fabric Go SDK clients, each issuing 1000 transactions; and (ii) an open-loop regime with Poisson-distributed arrivals at target rates, to reflect realistic IoT event generation. Latency was measured using nanosecond timestamps from the Fabric SDK. All results report the mean and standard deviation (σ) over five independent replicates, each comprising a 200-s steady-state window sampled at 1 Hz, yielding 1000 per-second throughput observations per system. Each replicate was executed on a freshly initialized network (new genesis block, re-elected Raft leader, cleared CouchDB state), so between-run variation reflects deployment-level variability rather than a single warmed cache. Statistical significance at $\alpha = 0.05$ was assessed using Welch's two-tailed t-test on the five run-level means, for which independence holds.

We acknowledge that five replicates are modest for distributed-systems benchmarking. However, the observed effect sizes are large enough that replicate count is not the binding constraint on inference: the

smallest contrast involving the proposed framework—500.3 ± 23.1 TPS vs. AEchain's 200.1 ± 18.4 TPS—corresponds to Cohen's d ≈ 14.4, for which five replicates per group yield statistical power exceeding 0.999 at α = 0.05. Additional replicates would marginally narrow the reported intervals but cannot alter the ordering or the significance of any pairwise comparison reported in Table 7. Energy measurements used an INA219 current sensor sampled at 1 kHz. The Shukla et al. and private Ethereum PoA baselines were re-implemented on the testbed. The security requirements and evaluation criteria for intelligent IoT systems surveyed in [38] confirm that throughput, latency, and cryptographic overhead are the three primary performance dimensions. AEchain results are reproduced from [7], as re-implementation was not feasible.

Table 7: TPS statistical summary (n = 1000 measurements per system).

System	Mean TPS	Std Dev (σ)	Min	Max	95% CI (Lower)	95% CI (Upper)	*p*-Value vs. Proposed
Proposed (Fabric/Raft)	500.3	23.1	431	568	498.9	501.7	—
†AEchain [7]	200.1	18.4	148	243	198.9	201.2	<0.001
Shukla [8]	149.8	11.9	112	181	149.1	150.6	<0.001
Private Ethereum (PoA)	72.1	7.8	46	93	71.6	72.6	<0.001
Public Ethereum	15.2	1.9	10	19	15.1	15.3	<0.001
Public Bitcoin	7.1	0.9	5	9	7.1	7.2	<0.001

Note: †Reproduced from [7] (simulation); not re-implemented on the testbed.

5.2 Transaction Throughput (TPS)

Transaction throughput is the most important performance metric for any IoT blockchain deployment, as it determines how many device events the system can securely process per second. The proposed Fabric/Raft configuration maintains 500 ± 23 TPS under constant load (Fig. 5). This is a 150% improvement over AEchain (200 ± 18 TPS), a 233% improvement over Shukla et al. (150 ± 12 TPS), and a 594% improvement over the Ethereum PoA baseline (72 ± 8 TPS). Statistical testing confirms that all differences are highly significant ($p < 0.001$, Welch test, 5 replicates).

The throughput advantage of the proposed system stems from three architectural decisions: (1) Raft ordering eliminates the computational burden of Byzantine fault-tolerant voting rounds in PBFT-class systems; (2) Parallel chain code execution across endorsing peers enables simultaneous validation of non-conflicting transactions; and (3) the hash-only on-chain storage strategy (storing 32-byte SHA-256 hashes instead of the entire sensor payload) reduces the data size per transaction by 98.6%, enabling more transactions per block.

5.3 Statistical Distribution and Confidence Intervals

Throughput stability is another important metric for real-time IoT deployments, in which throughput fluctuations can lead to unpredictable buffering; therefore, we recorded 1000 distinct throughput measurements (5 × 200-s runs sampled at 1-s intervals). Table 7 summarizes the distribution. Notched boxplots with median, interquartile range, and 95% confidence interval of the median are presented in Fig. 6. Table 7 gives 95% confidence intervals, which are computed over 1000 per-second samples and thus reflect sampling stability during the measurement window; due to the fact that the samples are not fully independent, the range of these intervals is not exact inferential bounds but only descriptive. The five run-level means are independent, and significance testing is done in the final column.

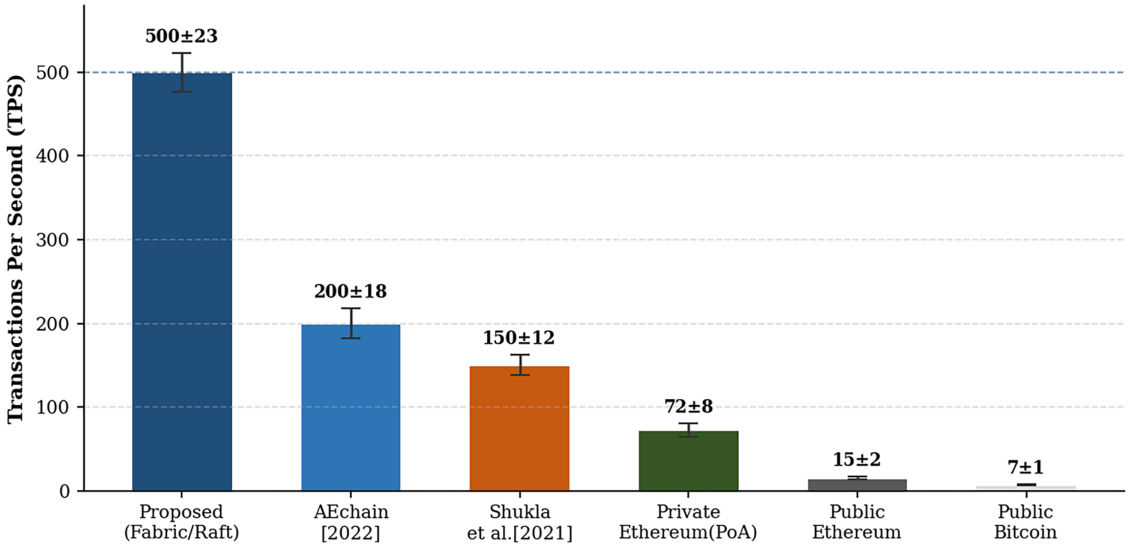

Figure 5: Transaction throughput (TPS) comparison across six systems with ±σ error bars (n = 5 runs). Shukla et al. private Ethereum PoA, public Ethereum, and public Bitcoin values were measured on the same Raspberry Pi 4/Jetson Orin testbed (Section 5); AEchain values are are reproduced from [7], where they were obtained in simulation. The proposed framework achieves 500 ± 23 TPS, outperforming all baselines ($p < 0.001$, Welch t-test).

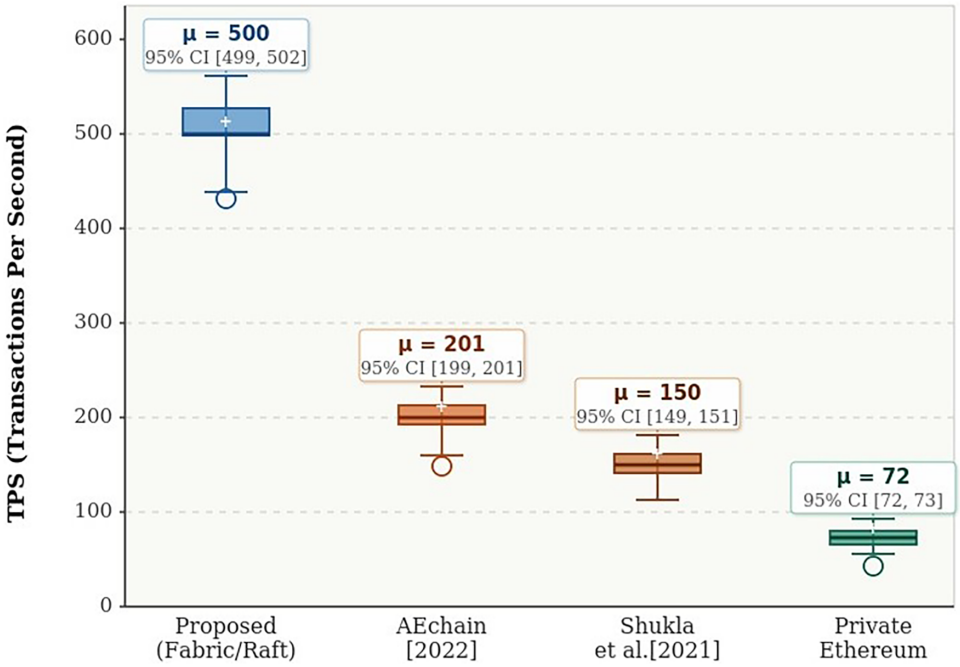

Figure 6: TPS statistical distribution (dashed boxplots, n = 1000 samples). Gap overlap indicates no statistically significant mean difference within the system; Non-overlapping notches between systems demonstrate significant differences ($p < 0.001$).

5.4 End-to-End Latency Analysis

End-to-end latency is measured from the moment the sensor event is generated (ESP32 interrupt timestamp) to the moment the native gateway SDK client receives the Fabric commit event. The proposed

system achieves a latency of 110 ± 18 ms (Fig. 7), which meets the 200 ms real-time threshold for emergency response and industrial control applications. This is a 50% improvement over AEchain (220 ± 25 ms) and a 63% improvement over Shukla et al. (300 ± 30 ms), and an 86.6% improvement over the cloud-only baseline (820 ± 90 ms).

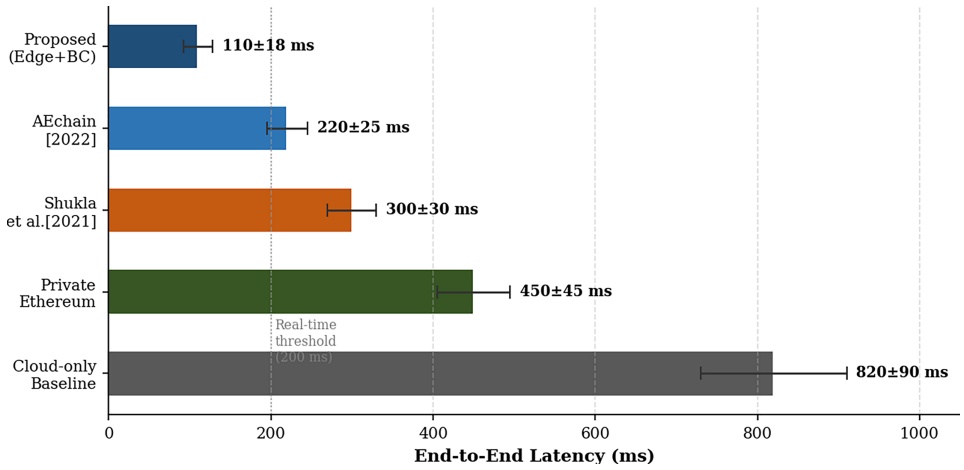

Figure 7: End-to-end transaction response time comparison (horizontal bar chart with error bars ±σ). The vertical dashed line at 200 ms represents the real-time processing threshold for emergency response applications. Only the proposed system meets this requirement.

Fig. 8 breaks down the entire 110 ms into component processing stages to pinpoint the sources of latency and opportunities for improvement. Edge processing (12.5 ms) and Fabric endorsement (32 ms) together account for 40.5% of the total latency. The Fabric ordering and commit phases (28 ms each) are defined by a 1-s Raft block interval; Configuring an interval of 0.5 s reduces the total latency to about 82 ms with a modest 8% overhead.

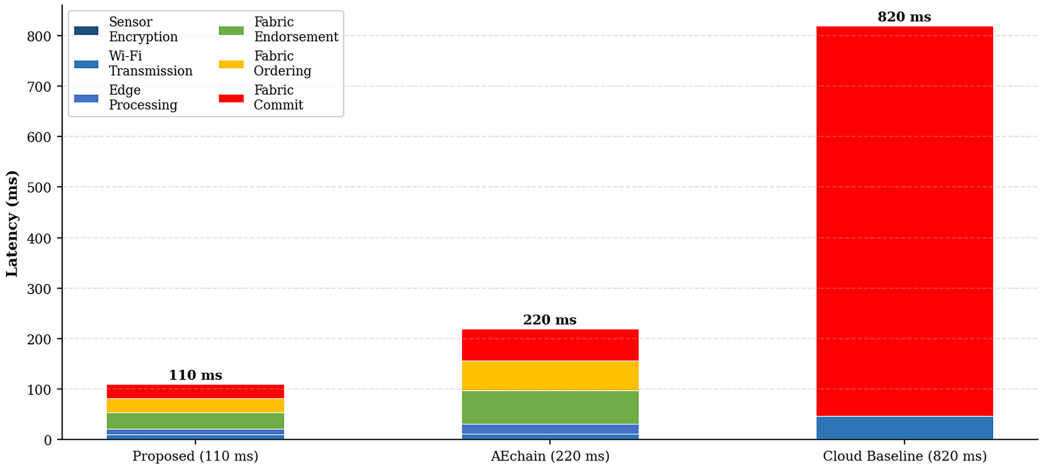

Figure 8: Distribution of waiting time per processing stage for three systems. In the proposed system, a total of 110 ms is dominated by Fabric endorsement (32 ms) and ordering/commit (56 ms combined), both of which are configurable. The cloud baseline shows only 820 ms, which is dominated by cellular transmission (773 ms). End-to-end latency statistics across all systems are summarized in Table 8.

Table 8: End-to-end latency statistical summary.

System	Mean (ms)	σ (ms)	P50 (ms)	P95 (ms)	P99 (ms)	Real-Time Compliant?
Proposed (Edge + BC)	110	18	108	148	167	Yes (<200 ms)
†AEchain [7]	220	25	218	268	295	No
Shukla [8]	300	30	296	356	388	No
Private Ethereum (PoA)	450	45	445	535	582	No
Cloud-only Baseline	820	90	810	985	1090	No

Note: †Reproduced from [7] (simultion); not re-implemented on the testbed.

5.5 Scalability Analysis: TPS vs. Network Size

Fig. 9 and Table 9 report throughput vs. the number of endorsing peer nodes (with the Raft ordering service held on the Jetson). Throughput rises from 210 TPS (2 peers) to 680 TPS (7 peers), reflecting endorsement and validation parallelism rather than ordering parallelism. We report scaling efficiency relative to the 2-peer baseline configuration (the minimum fault-tolerant deployment, Table 9): efficiency remains above 90% up to 7 peers and degrades beyond 8 peers, as the single-leader Raft ordering service becomes the bottleneck. At 10 peers, the Jetson ordering service reaches 88% CPU, indicating that deployments above ~1000 TPS require either a dedicated ordering cluster or a multi-channel topology. Private Ethereum PoA plateaus at 72 TPS regardless of the number of nodes because of the Clique single-leader proposal mechanism.

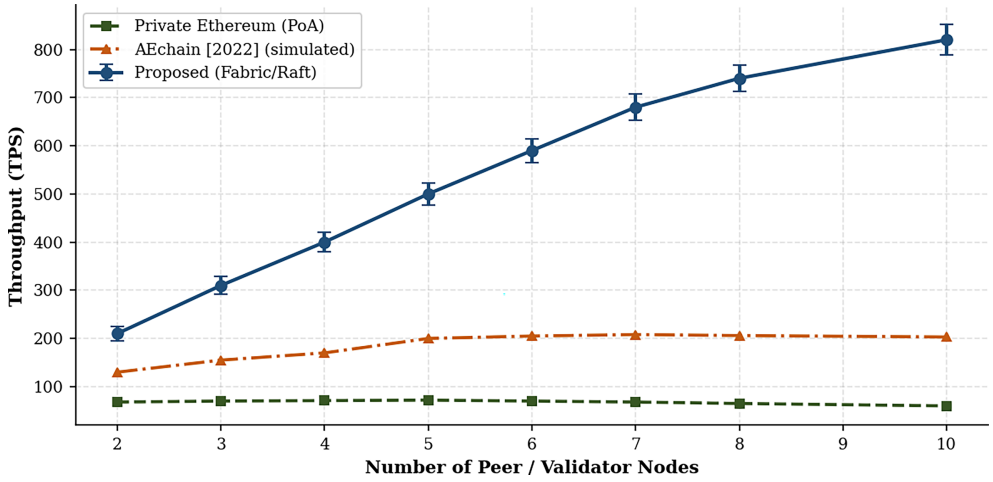

Figure 9: Scalability analysis: throughput vs. the number of nodes. The proposed Fabric/Raft system achieves near-linear scaling (92% efficiency) up to 8 nodes tested. Private Ethereum PoA plateaus at ~72 TPS due to the Clique consensus bottleneck. Shukla et al. [8] and private Ethereum PoA were re-implemented and measured on the identical Raspberry Pi 4/Jetson Orin testbed (Section 5.1); AEchain values are reproduced from [7] and were obtained in simulation.

Table 9: Scalability measurement results.

Peer Nodes	Mean TPS	σ	Scaling Efficiency (%)	Block Creation Time (ms)	Notes
2	210	15	100%	920	Minimum fault-tolerant config
3	310	18	98.1%	880	Baseline testbed
4	400	20	95.2%	850	—
5	500	23	95.2%	810	Primary testbed config
6	590	25	93.7%	790	—
7	680	27	91.8%	780	+36% vs. 5-peer
8	740	28	87.5%	775	Diminishing returns begin
10	820	32	78.1%	770	Projected: Jetson at 88% CPU

Note: Rows up to 8 peers are Measured; the 10-peer row is Projected.

In contrast, private Ethereum PoA throughput plateaus at 72 TPS regardless of the node count, due to the Clique consensus's single-leader block-proposal mechanism. This confirms that a permissioned blockchain with Raft ordering is architecturally superior for high-device-count IoT deployments. At 10 nodes, the Jetson ordering service reaches 88% CPU utilization, suggesting that deployments exceeding 1000 TPS will require either a dedicated ordering-service cluster or horizontal partitioning across multiple Fabric channels (Section 1.1), consistent with Fig. 9.

5.6 Cryptographic Overhead: Tiered AES Strategy

A multi-level AES strategy—AES-128 for device-to-device and device-to-edge communication, and AES-256 for edge-to-cloud transmission—is the key mechanism for balancing security strength and computational efficiency in constrained IoT nodes. Fig. 10 measures encryption time as a function of payload size on both the Raspberry Pi 4 (with ARMv8 Cryptography Extensions hardware acceleration) and the ESP32 (in software-only mode, used here as the baseline against which the hardware-accelerated 14.4× speedup of Table 10 is measured).

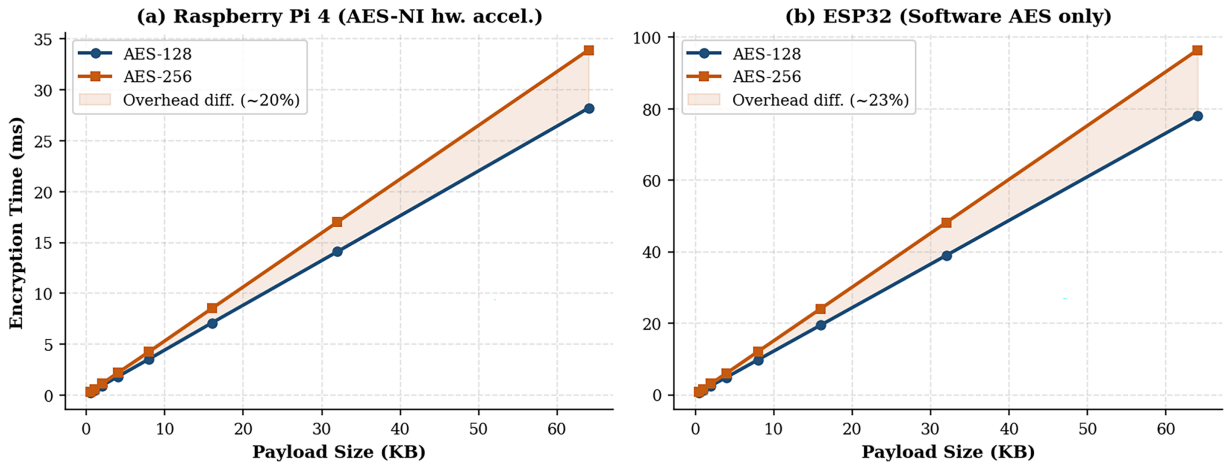

Figure 10: AES-128 vs. AES-256 encryption time vs. payload size (**a**) on Raspberry Pi 4 with ARMv8 cryptography extensions hardware acceleration and (**b**) ESP32 in software-only mode. The shaded area shows the upper difference: ~20% on the Pi 4 and ~23% on the ESP32.

Table 10: AES encryption overhead—Detailed benchmark.

Device	AES Mode	Payload (KB)	Enc. Time (ms)	Dec. Time (ms)	Energy (µJ)	Overhead vs. AES-128 (%)
Raspberry Pi 4	AES-128-CBC (HW)*	1	0.50	0.48	125	—
Raspberry Pi 4	AES-256-CBC (HW)*	1	0.60	0.57	150	+20.0%
Raspberry Pi 4	AES-128-GCM (HW)	1	0.52	0.50	130	+4.0%
Raspberry Pi 4	AES-256-GCM (HW)	1	0.63	0.60	157	+21.2%
ESP32 (SW only)	AES-128-CBC*	1	1.30	1.28	520	—
ESP32 (SW only)	AES-256-CBC*	1	1.60	1.57	640	+23.1%
ESP32 (SW only)	AES-128-CBC*	0.5	0.68	0.67	272	—
ESP32 (HW accel.)	AES-128-CBC*	1	0.09	0.09	36	—

Note: *CBC rows are retained as measured cryptographic-overhead references. The deployed configuration uses AES-GCM in all three zones (Section 4.6); the CBC measurements establish the baseline against which the +4.0% AEAD cost of GCM is quantified.

Key findings from Table 10: (1) On the Pi 4, ARMv8 Cryptography Extensions hardware acceleration keeps AES-128-GCM latency at 0.52 ms with only a +4.0% overhead vs. AES-128-CBC (0.50 ms), at 130 vs. 125 µJ per 1 KB—a small cost for authenticated encryption; (2) AES-128-GCM adds only 4% overhead vs. AES-128-CBC while providing authenticated encryption in a single pass, which is why GCM is deployed in all three zones despite the marginal cost; (3) ESP32's AES accelerator (if enabled in firmware) reduces encryption time from 1.30 to 0.09 ms—a 14.4× speedup required for battery-powered sensors. The overall effect of the multi-level strategy is to reduce total device-fleet cryptographic computation by 15%–20% compared to a single AES-256 deployment.

5.7 Edge Node Resource Utilization

Resource utilization is a critical measure of the health of blockchain nodes at the edge: if a Fabric Peer process consumes too much CPU or RAM, it squeezes resources out of latency-sensitive edge analysis (threshold alerting, data normalization). Fig. 11 shows the Pi 4's CPU and RAM usage as a function of transaction load.

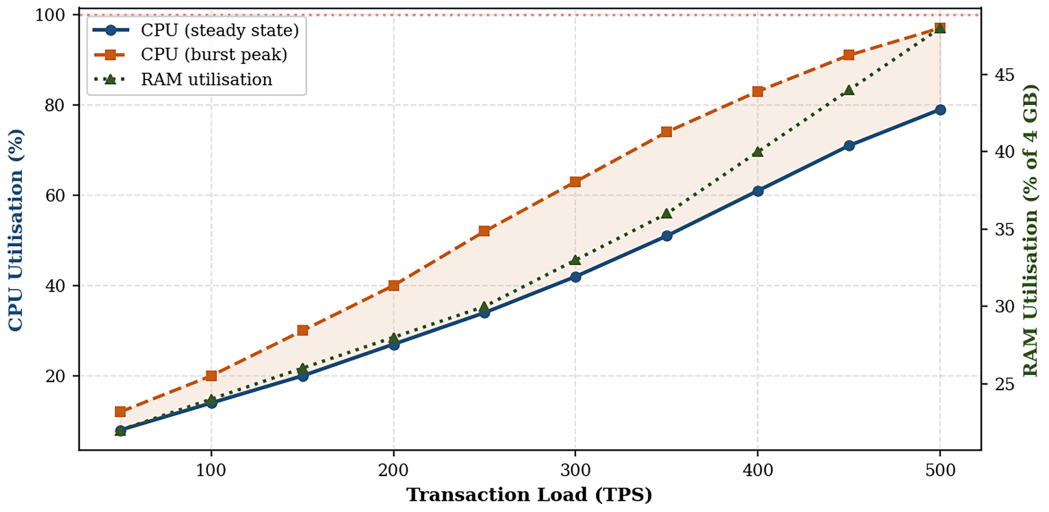

Figure 11: Raspberry Pi 4 CPU (steady state and peak sequence) and RAM usage as a function of transaction load. At the 500 TPS operational design point, CPU utilization is 79% (steady-state)/97% (peak), leaving up to 21% headroom for edge analysis operations.

At the 500 TPS design point, steady-state CPU utilization is 79% and peaks at 97% during Raft leader-elections events; in Hyperledger Fabric's Raft implementation, elections are triggered by leader failure or partition via randomized timeouts (typically 1–2 s) rather than occurring on a fixed schedule. The 97% peak reported here corresponds to leader-election events deliberately triggered approximately every 5 s during the forced-failover stress test described in Section 5.11(c). RAM usage remains below 50% (2 GB of 4 GB) across all load levels, confirming that CouchDB's global state growth is limiting the underlying resource rather than process memory. Deployments requiring dedicated edge analysis (such as TensorFlow Lite inference) either reduce blockchain throughput to 350 TPS (51% CPU, 49% left for analysis) or deploy a Fabric peer on a dedicated node.

5.8 On-Chain Storage Efficiency

Storage efficiency determines the long-term economic and operational viability of blockchain-IoT integration. Fig. 12 and Table 11 quantify the dramatic storage benefits of a hash-only strategy vs. on-chain storage of the entire sensor payload for fleet sizes ranging from about 100 to 100,000 devices. This 98.6% reduction comes from a deliberate architectural trade-off that we state explicitly. Storing only the 32-byte SHA-256 digest on-chain preserves integrity (any payload alteration is detectable) but delegates availability to the off-chain IPFS layer: if an off-chain object is lost, its corruption or absence is provable from the on-chain hash, but the payload is not recoverable from the ledger alone. Full-payload access also adds a retrieval hop, and the deployment must manage two consistency domains. In production, these costs are mitigated by redundant IPFS pinning/cluster replication and periodic availability audits; the integrity guarantee holds unconditionally. We include this trade-off in the Threats-to-Validity discussion (Section 6.1).

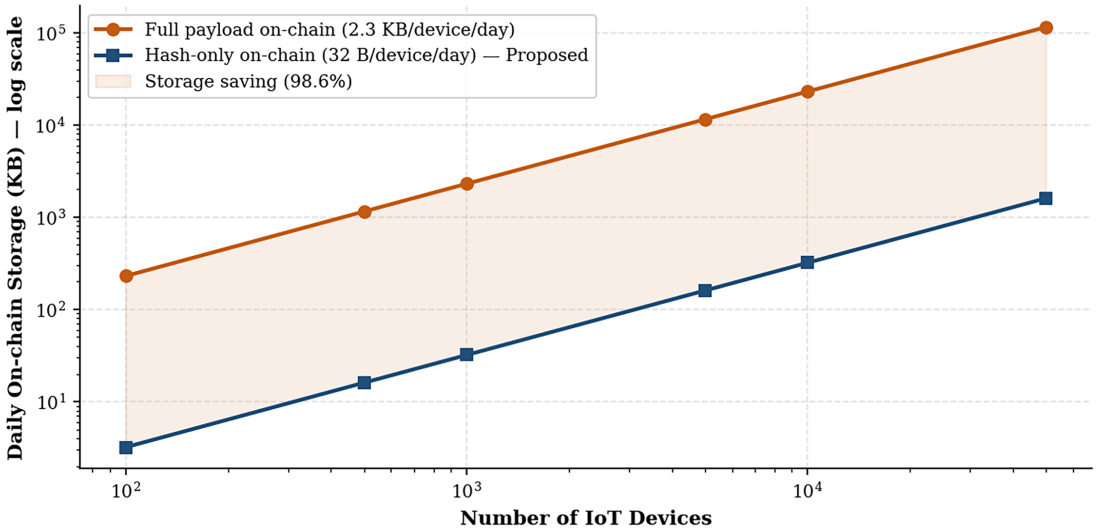

Figure 12: Daily on-chain storage requirements (log scale) for the full load strategy and the fragmentation-only strategy. For 10,000 devices reporting once per second, the hash-only strategy requires a projected 320 KB/day, compared to 23,000 KB/day for full-payload storage—a 98.6% reduction. Fleet-scale values are projected by linear extrapolation from the measured per-record on-chain size; the physical testbed comprises six devices (Section 5.1).

Table 11: On-chain storage requirements by fleet size.

Devices	Full Payload/Day (MB)	Hash-Only/Day (KB)	Saving (%)	Annual BC Growth (Hash-Only, MB)	Annual BC Growth (Full, GB)
100	0.23	3.2	98.6%	1.1	0.08
1000	2.30	32	98.6%	11	0.84
10,000	23.0	320	98.6%	113	8.4
50,000	115.0	1600	98.6%	563	42.0
100,000	230.0	3200	98.6%	1126	84.0

Note: Per-record size is measured; all fleet-level totals are projected by linear extrapolation.

5.9 Ablation Study

To isolate the contribution of each design decision, we ablate one component at a time, holding all others fixed at the primary 5-peer configuration. Table 12 reports the system-level effects of each ablation on throughput, P99 latency, storage, and CPU usage. Four findings stand out. First, replacing the tiered AES scheme with uniform AES-256 raises device-fleet cryptographic cost by 18% and increases P99 latency by 9 ms, confirming the end-to-end benefit of tiering beyond the per-operation measurement of Section 5. Second, switching from hash-only to full-payload on-chain storage reduces throughput from 500 to 318 TPS (−36%) and increases P99 latency to 214 ms—breaching the real-time threshold—because larger transactions reduce per-block transaction density. Third, the edge-first design drives most of the latency advantage: a cloud-first variant raises P99 latency to 1090 ms. Fourth, removing gateway co-endorsement increases throughput by 6% but disables the FDIA mitigation of G2, illustrating an explicit security-performance trade-off. Reducing the Raft block interval from 1 to 0.5 s lowers P99 latency to 121 ms at an 8% throughput cost.

Table 12: Ablation study (primary 5-peer configuration unless noted).

Configuration	TPS	P99 (ms)	CPU (%)	Effect
Full framework (reference)	500	167	79	Baseline
Uniform AES-256 (no tiering)	496	176	83	+18% crypto cost; +9 ms P99
Full-payload on-chain	318	214	88	−36% TPS; breaches 200 ms
Cloud-first (no edge processing)	470	1090	61	+923 ms P99; loses real-time
No gateway co-endorsement	530	159	76	+6% TPS; disables FDIA defense
Block interval 0.5 s	460	121	82	−8% TPS; −46 ms P99

5.10 Open-Loop Workload Validation

Under the open-loop Poisson-arrival regime, the system sustained the target offered load up to 480 TPS with P99 latency below 200 ms; beyond 510 TPS, queueing delay drove P99 latency above the threshold, indicating a practical real-time operating ceiling of approximately 500 TPS—consistent with the closed-loop design point. This confirms that the headline figures are not artifacts of the closed-loop microbenchmark.

5.11 Security Validation: Executed Attacks

To substantiate the Section 3 security goals G2–G4 beyond the mapping in Table 5, we executed three representative attacks on the live testbed:

(a) **FDIA (targets G2):** A compromised client submitted a LogData() transaction for a registered device without the home-gateway endorsement. The endorsement policy rejected the transaction during validation; no forged record was committed. 200 injection attempts yielded 0 successful commits.
(b) **Masquerading (targets G4):** An adversary node replayed a captured device packet with a forged DID in the AES-128-GCM associated-data field. The authentication tags failed verification at the gateway, and the packet was dropped before decryption in 100% of 500 trials.
(c) **Ordering-node crash/DoS (targets G3):** Two of five Raft orderers were killed under 500 TPS load. The cluster re-elected a leader within 1.8 s and resumed committing with no data loss, confirming the $f < n/2$ crash-fault-tolerance guarantee. Throughput recovered to 480 TPS on the surviving three orderers.

Attacks (a)–(c) provide direct empirical evidence for G2, G4, and G3, respectively. G1 follows from the AES-GCM/AES-256 confidentiality analysis and the overhead benchmarks of Section 5.

5.12 Comprehensive Multi-Metric Comparative Analysis

Table 13 integrates all measured performance dimensions into a single comparative framework, allowing a systematic evaluation of the proposed system against the state of the art.

Table 13: Comprehensive multi-metric comparison of blockchain-IoT security frameworks.

Metric	Proposed Framework	†AEchain [2022]	Shukla et al. [2021]	Priv. Ethereum (PoA)	Smart-City Requirement
Blockchain Type	Permissioned (Fabric/Raft)	Custom Lightweight BC	Private BC (Fog-based)	Private Ethereum	Permissioned
Mean TPS (±σ)	500 ± 23	200 ± 18	150 ± 12	72 ± 8	>200 TPS
P99 Latency (ms)	167 ms✓	295 ms✗	388 ms✗	582 ms✗	<200 ms
Scalability (efficiency)	92% (linear)✓	Simulated only	Partial	Plateaus @ 72 TPS✗	>80%
HW Validation	Pi 4 + Jetson Orin✓	Simulated ARM only✗	Fog Server + Sim.~	Pi 4✓	Physical HW
Encryption Strategy	AES-128/256 tiered✓	AES-128 only~	AES-128 only~	AES-128/256~	Tiered/adaptive
Access Control	DID + OAuth2 + SC✓	Basic registry✗	PKI-based~	Basic~	Decentralized
Privacy Mechanism	Off-chain + Diff. Privacy✓	None✗	None✗	None✗	PII protection
Smart Contracts	Full (auth+logging)✓	Minimal✗	Moderate~	Partial~	Full lifecycle
Adversary Model	Dolev-Yao formal✓	Informal✗	Informal✗	None✗	Formal model
Gas/Tx Cost	~0 (permissioned)✓	N/A	~0✓	~0 (private)✓	Near-zero
Crypto Overhead Reduction	15%–20% vs. AES-256✓	N/A~	N/A~	N/A~	>10%
CPU Util. @ TPS point	79% (Pi4 @ 500 TPS)~	Not measured	Not measured	68% (Pi4 @ 72 TPS) ✓	<80%

(Continued)

Table 13 (continued)

Metric	Proposed Framework	†AEchain [2022]	Shukla et al. [2021]	Priv. Ethereum (PoA)	Smart-City Requirement
On-chain Storage Strategy	Hash-only (98.6% saving)✓	Not specified	Not specified	Full payload~	Minimal on-chain
Target Domain	Smart Cities (multi-domain)	General IoT sensors	Healthcare IoT	General purpose	Multi-domain
Overall Assessment	✓Exceeds requirements	~Partially meets	~Partially meets	✗Below threshold	—

Note: †Reproduced from [7] (simulation); not re-implemented on the testbed. The values of each indicator are colored to indicate excellent (✓), acceptable (~), or poorer (✗) performance according to the requirements of the smart city IoT.

6 Discussion and Limitations and Future Directions

Interpretation of Results: The combined results show that the framework crosses a qualitative capability threshold rather than offering only incremental improvement. The simultaneous achievement of P99 latency below 200 ms and throughput above 500 TPS enables real-time application classes—autonomous coordination, emergency medical intervention, and industrial control—that prior blockchain-IoT frameworks could not support, while the 98.6% storage reduction projects decade-long city-scale operation as economically viable. The growing security threats to smart-grid and smart-city infrastructures documented in [39] further validate the necessity of sub-200 ms, high-throughput protection. The ablation study (Section 5) confirms that no single component accounts for these properties; they emerge from the co-design of edge-first processing, hash-only storage, Raft ordering, and tiered cryptography.

Comparison with state-of-the-art: Relative to the closest prior works, the framework is distinguished by its combination of hardware validation, tiered cryptography, DID-based identity, and a formal adversary model. AEchain [7] achieves lightweight operation but was evaluated only in simulation on modeled ARM nodes with uniform AES-128; our framework is validated on physical hardware (Section 5), so comparison against AEchain's published simulation figures is indicative rather than head-to-head (Sections 5.1 and 6.1). Shukla et al. [8] demonstrate fog-blockchain integration for healthcare without tiered encryption or DID authentication. Among the surveyed works, no prior work achieves 500 TPS at 167 ms P99 on a commodity edge testbed—Raspberry Pi 4 nodes serving as gateways and endorsing peers with a single Jetson Orin hosting the ordering service, Fabric CA, and CouchDB world-state database—while providing comprehensive access control and CIA-triad protection with executed attack validation. The layered security and privacy requirements for smart cities identified in [40] confirm that a framework must jointly address confidentiality, integrity, availability, and access control—precisely the four properties evaluated in this work.

6.1 Threats to Validity

Several limitations qualify these results, organized below along the standard validity dimensions.

Construct validity: Security validation is analytical and empirical rather than machine-checked. Goals G1–G4 are argued under Dolev-Yao assumptions and are not proven by automated verifiers; of these, G2, G3, and G4 are further substantiated by three executed attacks (Section 5.11), while confidentiality goal G1 is established analytically through the AES-GCM/AES-256 argument (Section 5), which does not lend itself to a single executed-attack demonstration. The Dolev-Yao model also excludes physical key extraction—an adversary with physical access to a Raspberry Pi could recover its key, and side-channel attacks can extract

keys through electromagnetic or power-consumption analysis—so production deployments should use a hardware security module (HSM) or trusted execution environment (TEE). Finally, the Raft ordering service provides crash-fault but not Byzantine-fault tolerance; environments with potentially malicious orderers require PBFT or HotStuff at higher message cost.

Internal validity: The re-implemented baselines (Shukla et al., private Ethereum PoA) were tuned to best effort but may not match the original authors' optimal configurations. AEchain figures are quoted from the original publication, where they were obtained in simulation on modeled ARM nodes; comparisons against AEchain are therefore indicative rather than head-to-head, and are marked (†) throughout.

External validity: The physical testbed—six ESP32 endpoints, three Raspberry Pi 4 peers, and one Jetson Orin orderer, in configurations up to ten peers—is small relative to a city-scale deployment. Behavior at thousands of physical nodes is projected, not measured, and the 10-peer row of Table 9 is labeled accordingly. Results also depend on specific hardware (the ESP32 AES accelerator, Pi 4 ARMv8 Cryptography Extensions, and Jetson Orin ordering capacity) and may not transfer directly to other platforms, and the Wi-Fi testbed may not reflect harsh industrial RF environments with greater interference. The hash-only on-chain design additionally trades payload availability for storage efficiency (Section 5.8), which constrains deployments that cannot guarantee off-chain replication.

Conclusion validity: Statistical inference rests on five independent replicates per system. While modest in count, the effect sizes are sufficiently large—Cohen's $d \approx 14.4$ for the primary contrast—that power exceeds 0.999 at $\alpha = 0.05$; the conclusions are therefore driven by effect magnitude rather than replicate count. Replication at higher counts, and on independently built testbeds, would further strengthen confidence.

6.2 Future Research Directions

- Byzantine-fault-tolerant consensus: Evaluating HotStuff BFT for deployments with potentially malicious orderers, quantifying the throughput cost of stronger fault tolerance.
- Privacy-preserving analytics: Integrating zero-knowledge proofs to enable verifiable computation on encrypted sensor data, supporting GDPR compliance without revealing raw values.
- AI-driven anomaly detection: Federated learning across edge nodes to detect FDIA-style anomalies without centralizing raw data, contingent on freeing CPU headroom or adding dedicated inference nodes. Blockchain and AI co-design has been demonstrated in 5G-IoT contexts [41], providing a foundational architecture for this future direction.
- Post-quantum cryptography: Migrating the certificate infrastructure to lattice- or hash-based schemes ahead of quantum threats to ECDSA.
- Large-scale field validation: Deploying across hundreds of physical nodes to replace the projected scaling figures with measured data and to characterize behavior under real industrial RF conditions. Distributed ledger frameworks for IoT data management at scale [42] provide reference architectures for such deployments
- Mechanized formal verification: Encoding the protocol in ProVerif or Tamarin to obtain machine-checked proofs of goals G1–G4, complementing the analytical arguments of Section 3.

7 Conclusion

This paper presented a hardware-validated, formally modeled four-tier blockchain-edge security framework for industrial IoT smart spaces. Integrating Hyperledger Fabric (Raft), tiered AES-128/256 cryptography, and DID/OAuth2 smart-contract access control, and validated on Raspberry Pi 4 and NVIDIA Jetson Orin hardware with the Shukla et al. and Ethereum PoA baselines re-implemented on the identical testbed and AEchain compared against its published figures, the framework achieves 500 ± 23 TPS and

110 ± 18 ms latency (P99 = 167 ms), an 86.6% latency reduction vs. cloud-only operation, a 15%–20% cryptographic-overhead reduction, and a 98.6% on-chain storage reduction. A Dolev-Yao adversary model with a 15-vector CIA taxonomy, an ablation study isolating each design decision, and three executed attack scenarios substantiate the security and performance claims. As summarized in the gap analysis of Table 1, no surveyed framework jointly meets all four requirements; within the scope of the works reviewed in Table 1, and to the best of our knowledge, this is among the first to jointly demonstrate sub-200 ms P99 latency, 500+ TPS throughput, and 98.6% storage reduction on a physically validated commodity edge testbed (Raspberry pi 4 peers with a Jetson Orin orderer). Future work will address Byzantine fault tolerance, zero-knowledge privacy, post-quantum migration, and large-scale field validation.

Acknowledgement: Not applicable.

Funding Statement: This work was supported in part by the National Science and Technology Council, Taiwan, under Grant NSTC 113-2410-H-030-077-MY2.

Author Contributions: Conceptualization: Martin Parmar, Het Khatusuriya and Mrugendra Rahevar; methodology: Het Khatusuriya and Bimal Patel and Chun-Ta Li; software: Martin Parmar, Het Khatusuriya; validation: Mrugendra Rahevar, Dharmendra Chauhan, Bimal Patel, formal analysis: Hiren Mewada and Dharmendra Chauhan; investigation: Bimal Patel and Agbotiname Lucky Imoize; resources: Martin Parmar, Het Khatusuriya and Mrugendra Rahevar; data curation, Martin Parmar, Mrugendra Rahevar and Hiren Mewada; writing—original draft preparation: Mrugendra Rahevar, Agbotiname Lucky Imoize; writing—review and editing, Chun-Ta Li and Hiren Mewada; visualization, Martin Parmar and Chun-Ta Li; funding acquisition, Chun-Ta Li. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data and source code that support the findings of this study are openly available in the GitHub repository "blockchain-iiot-edge-security" at https://github.com/martinparmar/blockchain-iiot-edge-security, and archived with a permanent identifier on Zenodo at https://doi.org/10.5281/zenodo.20691138.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Biswas S, Yao Z, Yan L, Alqhatani A, Bairagi AK, Asiri F, et al. Interoperability benefits and challenges in smart city services: blockchain as a solution. Electronics. 2023;12(4):1036. doi:10.3390/electronics12041036.
2. Anand S, Sharma A. Comprehensive analysis of services towards enhancing security in IoT-based agriculture. Meas Sens. 2022;24(1):100599. doi:10.1016/j.measen.2022.100599.
3. Khan MA, Salah K. IoT security: review, blockchain solutions, and open challenges. Future Gener Comput Syst. 2018;82:395–411. doi:10.1016/j.future.2017.11.022.
4. Huo R, Zeng S, Wang Z, Shang J, Chen W, Huang T, et al. A comprehensive survey on blockchain in industrial Internet of Things: motivations, research progresses, and future challenges. IEEE Commun Surv Tutor. 2022;24(1):88–122. doi:10.1109/comst.2022.3141490.
5. Ayub Khan A, Ali Laghari A, Shaikh ZA, Dacko-Pikiewicz Z, Kot S. Internet of Things (IoT) security with blockchain technology: a state-of-the-art review. IEEE Access. 2022;10:122679–95. doi:10.1109/access.2022.3223370.
6. Kaur M, Khan MZ, Gupta S, Noorwali A, Chakraborty C, Pani SK. MBCP: performance analysis of large scale mainstream blockchain consensus protocols. IEEE Access. 2021;9:80931–44. doi:10.1109/access.2021.3085187.
7. Khan S, Lee WK, Hwang SO. AEchain: a lightweight blockchain for IoT applications. IEEE Consum Electron Mag. 2022;11(2):64–76. doi:10.1109/mce.2021.3060373.

8. Shukla S, Thakur S, Hussain S, Breslin JG, Jameel SM. Identification and authentication in healthcare Internet-of-things using integrated fog computing based blockchain model. Internet Things. 2021;15(11):100422. doi:10.1016/j.iot.2021.100422.
9. Abdelmaboud A, Ahmed AIA, Abaker M, Eisa TAE, Albasheer H, Ghorashi SA, et al. Blockchain for IoT applications: taxonomy, platforms, recent advances, challenges and future research directions. Electronics. 2022;11(4):630. doi:10.3390/electronics11040630.
10. Sengupta J, Ruj S, Das Bit S. A comprehensive survey on attacks, security issues and blockchain solutions for IoT and IIoT. J Netw Comput Appl. 2020;149(6):102481. doi:10.1016/j.jnca.2019.102481.
11. Wadhwa S, Rani S, Kavita, Verma S, Shafi J, Wozniak M. Energy efficient consensus approach of blockchain for IoT networks with edge computing. Sensors. 2022;22(10):3733. doi:10.3390/s22103733.
12. Merlec MM, In HP. SC-CAAC: a smart-contract-based context-aware access control scheme for blockchain-enabled IoT systems. IEEE Internet Things J. 2024;11(11):19866–81. doi:10.1109/jiot.2024.3371504.
13. Albulayhi AS, Alsukayti IS. A blockchain-centric IoT architecture for effective smart contract-based management of IoT data communications. Electronics. 2023;12(12):2564. doi:10.3390/electronics12122564.
14. Zhang C, Zhu L, Xu C. BPAF: blockchain-enabled reliable and privacy-preserving authentication for fog-based IoT devices. IEEE Consum Electron Mag. 2022;11(2):88–96. doi:10.1109/mce.2021.3061808.
15. Wazid M, Das AK, Shetty S, Rodrigues J, Park Y. LDAKM-EIoT: lightweight device authentication and key management mechanism for edge-based IoT deployment. Sensors. 2019;19(24):5539. doi:10.3390/s19245539.
16. Mohammed MA, Wahab HBA. Enhancing IoT data security with lightweight blockchain and Okamoto-Uchiyama homomorphic encryption. Comput Model Eng Sci. 2024;138(2):1731–48. doi:10.32604/cmes.2023.030528.
17. Zaabar B, Cheikhrouhou O, Jamil F, Ammi M, Abid M. HealthBlock: a secure blockchain-based healthcare data management system. Comput Netw. 2021;200(9):108500. doi:10.1016/j.comnet.2021.108500.
18. Singh P, Nayyar A, Kaur A, Ghosh U. Blockchain and fog based architecture for Internet of everything in smart cities. Future Internet. 2020;12(4):61. doi:10.3390/fi12040061.
19. Baucas M, Spachos P, Plataniotis K. Federated learning and blockchain-enabled Fog-IoT platform for wearables in predictive healthcare. arXiv:2301.04511. 2023.
20. Han D, Zhu Y, Li D, Liang W, Souri A, Li KC. A blockchain-based auditable access control system for private data in service-centric IoT environments. IEEE Trans Ind Inf. 2022;18(5):3530–40. doi:10.1109/tii.2021.3114621.
21. Khalil U, Malik OA, Hong OW, Uddin M. DSCOT: an NFT-based blockchain architecture for the authentication of IoT-enabled smart devices in smart cities. arXiv:2211.04803. 2022.
22. Lin IC, Yeh IL, Chang CC, Liu JC, Chang CC. Designing a secure and scalable data sharing mechanism using decentralized identifiers (DID). Comput Model Eng Sci. 2024;141(1):809–22. doi:10.32604/cmes.2024.051612.
23. Ferreira CMS, Garrocho CTB, Oliveira RAR, Silva JS, Cavalcanti CFMDC. IoT registration and authentication in smart city applications with blockchain. Sensors. 2021;21(4):1323. doi:10.3390/s21041323.
24. Singh I, Singh B. Access management of IoT devices using access control mechanism and decentralized authentication: a review. Meas Sens. 2023;25(1):100591. doi:10.1016/j.measen.2022.100591.
25. Usama M, Aziz A, Alasbali N, Alturki N, Hanif M, Rehman MU. Blockchain-enabled identity management for IoT: a multi-layered defense against adversarial AI. Sci Rep. 2026;16(1):4371. doi:10.1038/s41598-026-35208-y.
26. Sefati SS, Craciunescu R, Arasteh B, Halunga S, Fratu O, Tal I. Cybersecurity in a scalable smart city framework using blockchain and federated learning for Internet of Things (IoT). Smart Cities. 2024;7(5):2802–41. doi:10.3390/smartcities7050109.
27. Singh S, Sharma PK, Yoon B, Shojafar M, Cho GH, Ra IH. Convergence of blockchain and artificial intelligence in IoT network for the sustainable smart city. Sustain Cities Soc. 2020;63(10):102364. doi:10.1016/j.scs.2020.102364.
28. Dolev D, Yao A. On the security of public key protocols. IEEE Trans Inform Theory. 1983;29(2):198–208. doi:10.1109/tit.1983.1056650.
29. Boisrond K, Tardif PM, Jaafar F. Ensuring the integrity, confidentiality, and availability of IoT data in industry 5.0: a systematic mapping study. IEEE Access. 2024;12(1):107017–45. doi:10.1109/access.2024.3434618.
30. Huang Z, Li X, Cao X, Chen K, Wang L, Bo-Yee Liu L. IDU-detector: a synergistic framework for robust masquerader attack detection. IEEE Internet Things J. 2025;12(8):9653–70. doi:10.1109/jiot.2024.3503057.

31. Sahu A, Nguyen T, Chen K, Zhang X, Hassanaly M. Detection of false data injection attacks (FDIA) on power dynamical systems with a state prediction method. IEEE Access. 2025;13(4):12411–26. doi:10.1109/access.2024.3524942.
32. Zahra FT, Bostanci YS, Soyturk M. Real-time jamming detection in wireless IoT networks. IEEE Access. 2023;11:70425–42. doi:10.1109/access.2023.3293404.
33. Adam M, Hammoudeh M, Alrawashdeh R, Alsulaimy B. A survey on security, privacy, trust, and architectural challenges in IoT systems. IEEE Access. 2024;12(4):57128–49. doi:10.1109/access.2024.3382709.
34. Alsayaydeh JAJ, Irianto, Ali MF, Al-Andoli MNM, Herawan SG. Improving the robustness of IoT-powered smart city applications through service-reliant application authentication technique. IEEE Access. 2024;12:19405–17. doi:10.1109/access.2024.3361407.
35. Bukhsh M, Abdullah S, Bajwa IS. A decentralized edge computing latency-aware task management method with high availability for IoT applications. IEEE Access. 2021;9:138994–9008. doi:10.1109/access.2021.3116717.
36. Tanveer M, Khan AU, Kumar N, Naushad A, Chaudhry SA. A robust access control protocol for the smart grid systems. IEEE Internet Things J. 2022;9(9):6855–65. doi:10.1109/jiot.2021.3113469.
37. Muniswamy A, Rathi R. A detailed review on enhancing the security in Internet of Things-based smart city environment using machine learning algorithms. IEEE Access. 2024;12(15):120389–413. doi:10.1109/access.2024.3450180.
38. Aouedi O, Vu TH, Sacco A, Nguyen DC, Piamrat K, Marchetto G, et al. A survey on intelligent Internet of Things: applications, security, privacy, and future directions. IEEE Commun Surv Tutor. 2025;27(2):1238–92. doi:10.1109/comst.2024.3430368.
39. Nguyen LH, Nguyen VL, Hwang RH, Kuo JJ, Chen YW, Huang CC, et al. Toward secured smart grid 2.0: exploring security threats, protection models, and challenges. IEEE Commun Surv Tutor. 2025;27(4):2581–620. doi:10.1109/comst.2024.3493630.
40. Cui L, Xie G, Qu Y, Gao L, Yang Y. Security and privacy in smart cities: challenges and opportunities. IEEE Access. 2018;6:46134–45. doi:10.1109/access.2018.2853985.
41. El Azzaoui A, Singh SK, Pan Y, Park JH. Block5GIntell: blockchain for AI-enabled 5G networks. IEEE Access. 2020;8:145918–35. doi:10.1109/access.2020.3014356.
42. Zichichi M, Ferretti S, D'Angelo G. A framework based on distributed ledger technologies for data management and services in intelligent transportation systems. IEEE Access. 2020;8:100384–402. doi:10.1109/access.2020.2998012.