Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 08 , August - 2026

Developing a Transparent Verification Mechanism for Digital Certificates Issued by OpenXPKI using a Merkle Tree

Somia Sroor
Department of Information
Technology Syrian Virtual University
(SVU) Damascus, Syria

Waseem Jneidi
Department of Information
Security Syrian Virtual University
(SVU) Damascus, Syria

***Abstract*—Public Key Infrastructure (PKI) has traditionally relied on centralized Certificate Authorities (CAs), and this architecture introduces a critical single point of failure; a compromised CA can issue fraudulent certificates without immediate detection. In this context, traditional validation protocols, such as CRLs and OCSP, operate reactively post-issuance and fail to architecturally prevent unauthorized generation. This research addresses this architectural gap by designing and implementing a Certificate Transparency (CT) mechanism that is managed locally within an OpenXPKI enterprise environment. This architecture enables independent cryptographic verification of certificate issuance operations while keeping sensitive internal network data isolated from public registries. The proposed system strictly enforces pre-issuance logging by modifying the OpenXPKI workflow to generate a preliminary certificate prior to the final issuance of an X.509 certificate. This preliminary certificate is registered on a standalone server (CT Log) based on a Merkle Tree structure. Following registration, the server issues a Signed Certificate Timestamp (SCT) that is directly embedded in the final certificate structure, while a client-side web application employs a tile-based storage model to support independent cryptographic verification. Experimental evaluations have demonstrated the system's ability to deterministically detect any retroactive tampering with records. The fail-closed architecture also succeeded in categorically preventing the issuance of any unverified certificates during server failure simulations. On the operational level, performance measurements confirmed that this integration adds a delay of less than one second to the issuance cycle, with a linear growth in hash tile storage of 32 bytes per added certificate. These results confirm that incorporating a Merkle Tree-based verification layer into enterprise infrastructure replaces trust assumptions with mathematical proofs, providing organizations with tamper-evident auditing capabilities without compromising operational efficiency.**

***Index Terms*—Public Key Infrastructure (PKI), OpenXPKI, Digital Certificates, Certificate Transparency (CT), Merkle Tree, Pre-certificates, Signed Certificate Timestamp (SCT), Pre-issuance Logging, TesseraCT Library, Independent Auditability, Tamper-Evidence, Information Security.**

I. Introduction

The hierarchical trust model underpinning Public Key Infrastructure (PKI) relies heavily on the absolute authority of Certificate Authorities (CAs), creating a critical single point of failure. High-profile breaches [5], [12] demonstrated how compromised keys, despite stringent key management guidelines [2], lead to the undetected generation of fraudulent certificates. Furthermore, traditional X.509 validation protocols [3], such as Certificate Revocation Lists (CRLs) and the Online Certificate Status Protocol (OCSP), are strictly reactive, post-issuance measures. Addressing malicious issuance requires a preventative approach that cryptographically logs the creation process itself.

To address these vulnerabilities, the Internet Engineering Task Force (IETF) standardized Certificate Transparency (CT) [11], mandating public, append-only registries. Utilizing Merkle tree structures, CT logs issue Signed Certificate Timestamps (SCTs) as cryptographic proofs. However, applying this standard directly within closed enterprise environments violates data sovereignty by exposing sensitive internal Subject Distinguished Names (Subject DNs) and network topologies.

Conversely, enterprise certificate management systems, notably OpenXPKI, rely on relational databases lacking independent cryptographic verification. Consequently, they are vulnerable to retroactive tampering, where compromised servers or insiders can alter records without leaving deterministic proofs, confining reliability to post-hoc audits.

This research bridges this architectural gap by proposing a 'Privately-Operated Transparency' model. The objective is to engineer a cryptographic verification layer integrated directly into the OpenXPKI workflow. This system enforces strict pre-issuance logging—requiring a Pre-certificate and an SCT from a standalone enterprise CT log before final X.509 issuance—using a lightweight, tile-based architecture that avoids Distributed Ledger Technology (DLT) overhead.

This study tests several technical hypotheses regarding verifiable transparency in enterprise PKI. It posits that a Merkle tree-based mechanism can integrate seamlessly via middleware, enabling relying parties to perform independent mathematical verification. The proposed model guarantees the deterministic detection of retroactive tampering while keeping operational latency and storage consumption within acceptable margins.

Utilizing the Design Science Research Methodology

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 08 , August - 2026

(DSRM) [13], this study develops an operational prototype linking OpenXPKI with a TesseraCT-based log server. The system is evaluated for real-time inclusion verification, fail-closed determinism during outages, and the detection of retroactive tampering.

The scope is confined to standard X.509 issuance within closed networks using OpenXPKI and TesseraCT, excluding DLTs and broader PKI key management. Due to proof-of-concept resource constraints, auditor interfaces are co-located with the log server. Consequently, this model focuses on on-demand verification rather than continuous background auditing or mitigating split-view attacks.

The remainder of the paper is structured as follows: Section 2 reviews related literature, Section 3 Proposed Architecture and Implementation, Section 4 analyzes the experimental results, and Section 5 concludes the study.

II. Related Work

Previous research addressing the structural vulnerabilities of centralized CAs has explored various architectural paths. While standard Certificate Transparency (RFC 6962) [11] provides robust tamper detection, its reliance on public logs violates enterprise data sovereignty. Decentralized blockchain solutions, such as ProofChain [14] and CertLedger [9], utilize smart contracts and distributed consensus to unify certificate states and eliminate single points of failure. However, these approaches introduce prohibitive computational overhead, extended response latency, and severe storage bloat, rendering them incompatible with high-throughput enterprise environments.

Alternative paradigms sacrifice deterministic security or standards compliance. Probabilistic frameworks [6] rely on belief systems and continuous human feedback rather than cryptographic proof, while decentralized vault architectures [8] abandon the global X.509 standard entirely, resulting in a critical loss of external interoperability. Privacy-enhancing overlays like SVT-CT [10] demand persistent external monitors, adding unacceptable architectural complexity. Furthermore, efforts to optimize storage, including LegoLog [4] or relational database SubTrees [1], improve specific I/O metrics but remain constrained by legacy database architectures and fail to optimize cumulative storage natively.

The literature reveals a persistent trade-off between deterministic security, operational efficiency, and digital sovereignty. This study addresses this exact gap by integrating a lightweight, independent Merkle tree verification layer natively into the OpenXPKI workflow. This approach preserves full X.509 compatibility and ensures enterprise data sovereignty while actively avoiding the operational complexities and storage bloat associated with distributed ledgers.

III. Proposed Architecture and Implementation

This section details the structural design and practical execution of the ”Privately-Operated Transparency” model. The system shifts the issuance paradigm by enforcing a synchronous pre-issuance logging constraint directly within the OpenXPKI engine, utilizing a lightweight, tile-based Merkle tree storage.

A. Infrastructure and Node Separation

To prevent unilateral control over the issuance path, the Service-Oriented Architecture (SOA) is distributed across two functionally isolated Debian GNU/Linux 12 nodes. As illustrated in Fig. 1, this setup establishes a secure data flow between the primary issuance environment and the independent auditing components. The Issuance Node hosts the OpenXPKI workflow engine and MariaDB. It manages standard certificate lifecycles but is structurally restricted from final signing until all transparency requirements are met. Concurrently, the Audit Node hosts a Tessera-backed CT log server, POSIX tile storage, and the verification dashboard.

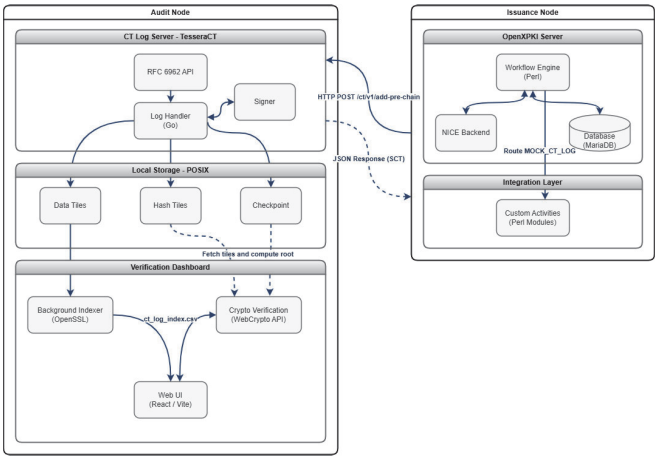

Fig. 1. System Architecture and Node Separation.

B. Pre-issuance Logging Workflow

The integration modifies the OpenXPKI core engine via custom middleware to execute a mandatory three-step transparency sequence before the final X.509 issuance. Fig. 2 models the temporal dependencies and synchronous blocking constraints of this re-engineered workflow, detailing the SCT injection within the certificate lifecycle:

- **Preparation and Freezing:** The PreparePrecertificate module halts the standard workflow. It freezes the serial number and validity period in the database to guarantee exact cryptographic equivalence, then injects a critical poison extension (OID 1.3.6.1.4.1.11129.2.4.3) to generate a Pre-certificate.
- **Log Submission:** The Pre-certificate is transmitted via HTTP POST to the Audit Node. The workflow enters a blocking state until a valid cryptographic response is received.
- **SCT Injection:** The log server returns a JSON payload containing the timestamp. The InjectSCTExtension module decodes this payload into a binary Signed Certificate Timestamp (SCT) and injects it as a non-critical extension (OID 1.3.6.1.4.1.11129.2.4.2) into the final certificate.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 08 , August - 2026

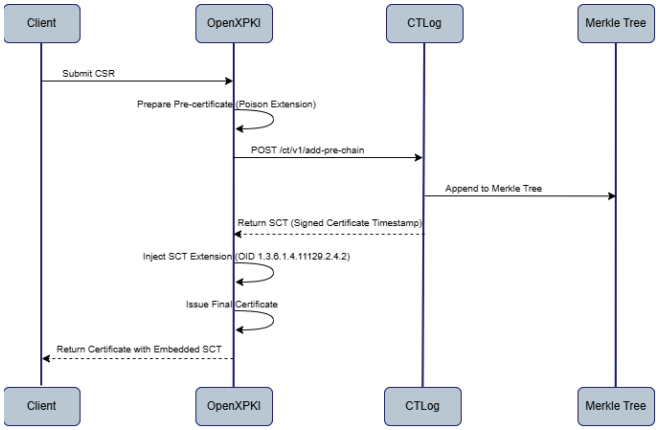

Fig. 2. Pre-issuance Logging Sequence Diagram.

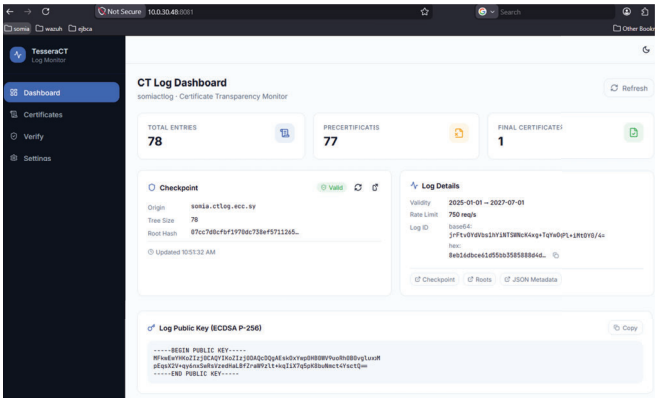

Fig. 3. Client-Side Monitoring Dashboard.

C. Tile-Based Storage (TesseraCT)

Bypassing legacy relational databases, the CT log operates on an append-only file system. Certificates are processed as leaf nodes in a Merkle tree and physically stored using a cumulative tile model. Data tiles retain the immutable binary encodings, while Hash tiles store the 32-byte SHA-256 node hashes. A background indexer processes these tiles using OpenSSL to expose metadata for client-side auditing.

D. Client-Side Cryptographic Verification

The architecture transfers the auditing burden to relying parties via a React/Vite dashboard. Using WebCrypto APIs, the dashboard fetches the static hash tiles and executes independent proofs. It verifies the ECDSA P-256 signature of the log's checkpoint (Signed Tree Head) and mathematically reconstructs the root hash to prove inclusion, utilizing standard RFC 6962 rules:

Leaf node hashing:

$$H_{leaf} = \text{SHA-256}(0x00 \parallel \text{entry}) \tag{1}$$

Internal node hashing:

$$H_{parent} = \text{SHA-256}(0x01 \parallel \text{left} \parallel \text{right}) \tag{2}$$

The application compares the calculated value against the documented root in the checkpoint to mathematically prove inclusion. Fig. 3 displays the client-side dashboard used to execute these proofs and monitor log statistics.

E. Security Considerations and Standards Compliance

This localized transparency model is particularly optimal for infrastructure operating under a National Root CA, where certificates are trusted nationally rather than globally, ensuring that localized trust boundaries are cryptographically maintained. The implemented architecture ensures stringent tamper resistance through forced serial number consistency, time-bound SCT issuance, and mathematically verifiable ECDSA signatures. Furthermore, the system is engineered to maintain strict alignment with the standard object identifiers mandated by Certificate Transparency protocols. Table I summarizes the system's compliance mechanisms with the RFC 6962 standard parameters.

TABLE I
SYSTEM COMPLIANCE WITH RFC 6962 STANDARD PARAMETERS

Standard Requirement	Implementation Mechanism
Poison Extension	Implemented using OID 1.3.6.1.4.1.11129.2.4.3 as a critical flag to prevent operational misuse.
SCT List Extension	Injected into the final certificate using OID 1.3.6.1.4.1.11129.2.4.2 as a non-critical extension.
Pre-certificate Submission	Executed synchronously via the /ct/v1/add-pre-chain REST endpoint.
SCT Structure	Encoded as a binary structure containing version, log ID, timestamp, extensions, and signature.

IV. Evaluation and Results

The evaluation methodology assessed the functional determinism and operational efficiency of the proposed architecture, specifically validating the four core research hypotheses. To evaluate architectural integration (H1), baseline testing confirmed the viability of modifying OpenXPKI workflows to mandate a synchronous Signed Certificate Timestamp (SCT) prior to final issuance. Simulated log outages validated the fail-closed nature of this integration. Upon intentionally stopping the log service, the OpenXPKI engine immediately aborted the signing phase. Server logs recorded an HTTP 500 'Connection refused' error (see Fig. 4a), while the user interface reflected policy violations and suspended the workflow with a 'CT_SUBMIT_FAILED' exception (see Fig. 4b and 4d), preventing the generation of unverified certificates. Once the log service was manually restored via the server terminal (see Fig. 4c), the preserved request state allowed for manual resumption, culminating in successful certificate issuance and publication (see Fig. 4e).

Regarding real-time transparency and independent verification (H2), baseline testing evaluated the system under standard operational conditions. Server logs recorded the pre-issuance sequence across tracked sessions (see Fig. 5a and

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 08 , August - 2026

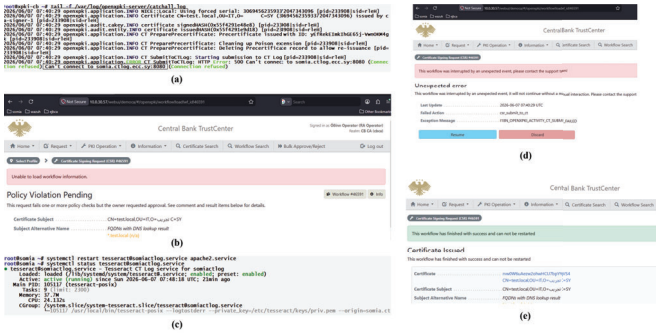

Fig. 4. Fail-closed determinism during a simulated log outage: (a) Server logs capturing the HTTP 500 connection error; (b) OpenXPKI WebUI displaying policy violation status; (c) Terminal execution restoring the log service; (d) OpenXPKI WebUI suspending the workflow; (e) Successful manual resumption and issuance upon log service recovery.

5b)—generating the pre-certificate, securing the SCT, and finalizing issuance—executing within a one-second window, while strictly maintaining cryptographic consistency through forced serial numbers. The standalone client dashboard verified this temporal synchronization by reflecting the exact 'Not Before' timestamp (see Fig. 5c). Most importantly, the client-side application executed a successful mathematical inclusion proof ('Inclusion VERIFIED') by matching the locally computed root hash against the log's expected root (see Fig. 5d). This confirms that relying parties can verify log integrity and certificate inclusion without implicit trust in the central CA database, effectively decoupling the auditing process from the issuing authority.

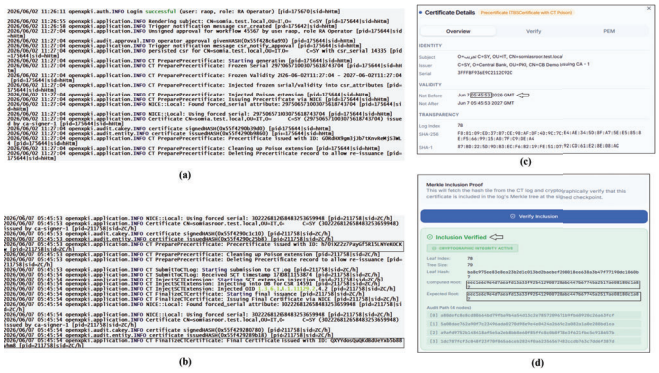

Fig. 5. Baseline real-time transparency and independent verification: (a) Initial server logs recording the operational request; (b) Server logs demonstrating a sub-second issuance cycle with forced serial numbers; (c) Dashboard reflecting immediate temporal synchronization; (d) Successful client-side mathematical inclusion proof validating cryptographic integrity.

To assess security determinism (H3), fault injection scripts evaluated the Merkle tree's sensitivity to retroactive tampering across two distinct vectors. First, a local tamper test mutated the raw leaf data of a specific certificate via an injection script (see Fig. 6a). The client dashboard immediately detected this as a data integrity failure, invalidating the inclusion proof exclusively for that targeted certificate (see Fig. 6b). Adjacent records (Indices 74 and 76) verified successfully, proving that the architecture effectively isolates local data corruption without compromising the overall root hash (see Fig. 6c).

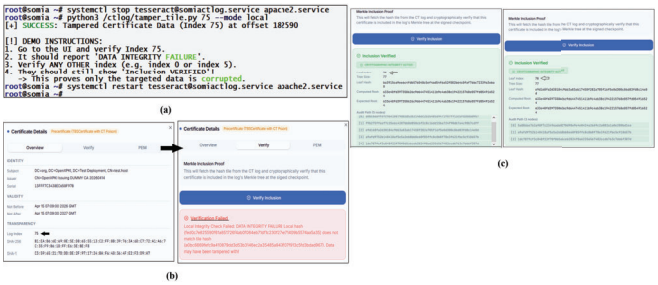

Fig. 6. Tamper-evidence evaluation for local data corruption: (a) Fault injection script execution targeting a specific leaf node; (b) Localized data integrity failure detected by the client dashboard; (c) Fault isolation showing successful verification of adjacent records.

Second, a global tamper test targeted an internal hash node (Index 73) using a fault injection script (see Fig. 7a). This structural mutation cascaded upward, altering the computed global root hash. Consequently, the dashboard reported a verification failure for the targeted entry (see Fig. 7b), and more importantly, caused inclusion proofs for all other certificates across the log—including unaffected adjacent records—to fail simultaneously due to a root mismatch (see Fig. 7c). This differential response confirms the system's capacity to deterministically expose both isolated data manipulation and systemic structural tampering.

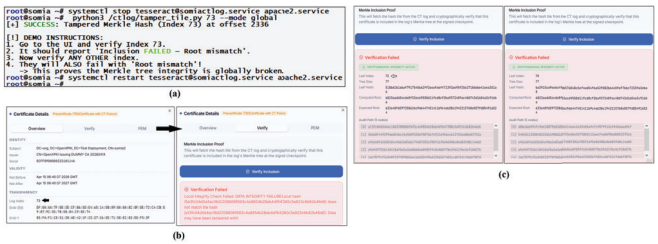

Fig. 7. Tamper-evidence evaluation for global structural corruption: (a) Fault injection script execution targeting an internal hash node; (b) Verification failure detected by the client dashboard for the targeted index; (c) Cascading verification failure across unaffected adjacent log entries due to compromised root integrity.

Finally, measuring operational efficiency and scalability (H4) revealed that the logarithmic complexity of the Merkle tree and the lightweight REST API integration maintained issuance latency at a negligible overhead of approximately one second. Observation of the file system demonstrated distinct scaling behaviours for both storage components. As illustrated in Fig. 8, hash tiles exhibited a strict, deterministic linear growth of exactly 32 bytes per certificate (corresponding to the SHA-256 output). Concurrently, data tiles grew proportionally based on the raw binary size of the certificates. This consolidated visualisation confirms that the tile-based model scales efficiently, completely avoiding the cumulative storage bloat characteristic of Distributed Ledger Technologies (DLT).

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 08 , August - 2026

These results confirm that integrating a Merkle tree-based verification layer enforces mandatory transparency, enables independent auditing, and guarantees tamper detection without compromising enterprise scalability.

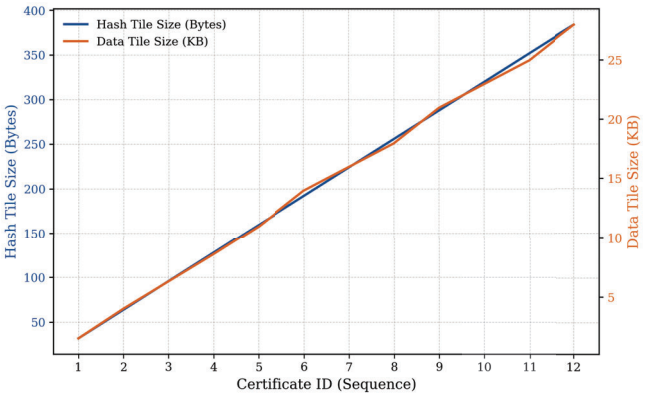

Fig. 8. Storage Scalability: Deterministic Hash Growth vs. Proportional Data Growth.

A. Scope and Limitations

While the empirical evaluation proves the viability of Privately-Operated Transparency, this study acknowledges specific architectural, operational, and technical boundaries.

Objective Scope: The research is strictly confined to enhancing the transparency of standard X.509 certificate issuance via Merkle tree inclusion proofs. It deliberately excludes broader PKI security policies (e.g., private key management), deep cryptanalysis of RSA/ECDSA algorithms, and decentralized paradigms such as Self-Sovereign Identity (SSI).

Technical Boundaries: The integration middleware and auditor-side proof-of-concept are tailored exclusively to OpenXPKI and a standalone CT Log Server. Consequently, the model excludes integration with alternative CA platforms, such as Microsoft ADCS or EJBCA. Furthermore, blockchain and Distributed Ledger Technologies (DLTs) were explicitly excluded in favor of the lightweight efficiency of Merkle trees.

Operational Setting and Concurrent Throughput: The development and evaluation phases were executed within a simulated virtual environment designed to mirror the stringent security conditions of closed enterprise networks. The architecture was evaluated without high-speed Hardware Security Module (HSM) offloading, limiting the assessment to functional determinism rather than maximum throughput stress testing. Theoretically, under extreme concurrent loads (e.g., 1,000 requests per minute), the tile-based storage would handle I/O efficiently due to its atomic file appends. However, because pre-issuance logging is a synchronous constraint, the primary bottleneck would shift to the OpenXPKI MariaDB transaction locks and software-based signing. Scaling this architecture for peak enterprise loads would mandate HSM offloading and asynchronous queueing for log submissions.

Security Limitations: Due to prototype resource constraints, the auditor interface is currently co-located with the log server. While this restricts the system to on-demand verification and creates a theoretical single point of failure, a true production deployment would strictly decouple these components. In a live enterprise environment, the auditing application must reside on entirely separate infrastructure (e.g., client-side workstations or dedicated external monitoring subnets) to guarantee cryptographic independence. Furthermore, the current model does not yet deploy continuous background auditing daemons or gossip protocols, which are necessary to fully mitigate split-view attacks.

V. Conclusion and Recommendations

A. Architectural Impact

This research successfully mitigates the structural vulnerabilities of centralized PKI trust models by engineering a ”Privately-Operated Transparency” architecture natively integrated into the OpenXPKI environment. By enforcing a mandatory pre-issuance logging constraint via a Merkle tree-based registry, the system actively shifts the enterprise paradigm from implicit trust to cryptographically verifiable security. The implementation achieves deterministic tamper-evidence without exposing internal network topologies to public registries, successfully balancing absolute data sovereignty with independent auditability.

B. Engineering Contributions and Recommendations

The engineering contributions of this study are threefold: establishing fail-closed determinism that structurally halts undocumented certificate generation; proving the operational viability of tile-based storage with minimal latency and linear capacity growth; and successfully decoupling verification workloads to enable on-demand, auditor-side mathematical proofs. For practical enterprise deployment, cybersecurity administrators must formally integrate pre-issuance logging constraints into their Certificate Policy and Certification Practice Statement (CP/CPS) governance frameworks. Furthermore, maintaining cryptographic integrity mandates a strict physical and logical separation of duties between the central CA operations and the CT log administration to prevent unilateral system control.

C. Future Work

While the current proof-of-concept validates on-demand verification, future development within closed enterprise networks will focus on transitioning to a continuous monitoring architecture. This entails deploying independent background auditing daemons to autonomously detect split-view attacks, designing local log truncation algorithms to archive expired certificates in high-volume environments, and establishing distributed, high-availability log replicas to ensure fault tolerance and uninterrupted cryptographic auditing.

References

[1] P. Ayyalasomayajula and M. Ramkumar, ”Optimization of Merkle Tree Structures: A Focus on Subtree Implementation,” in *Proc. 2023 Int. Conf. on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, 2023, pp. 59–67. [Online]. Available: https://doi.org/10.1109/CyberC58899.2023.00021

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 08 , August - 2026

[2] E. Barker, "Recommendation for Key Management," NIST Special Publication 800-57 Part 1 Rev. 5, National Institute of Standards and Technology, Gaithersburg, MD, 2020. [Online]. Available: https://doi.org/10.6028/NIST.SP.800-57pt1r5

[3] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," RFC 5280, May 2008. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc5280

[4] V. Fang et al., "LegoLog: A configurable transparency log," in *Proc. IEEE 10th Eur. Symp. on Security and Privacy (EuroS&P)*, 2025, pp. 1082–1103. [Online]. Available: https://doi.org/10.1109/EuroSP63326.2025.00066

[5] Fox-IT, "DigiNotar Certificate Authority breach 'Operation Black Tulip': Interim Report," 2011. [Online]. Available: https://www.teletrust.de/uploads/media/FOX-IT_Interim_Report_2011-09-05_pdf.pdf

[6] A. S. Wazan, R. Laborde, F. Barrère, and A. Benzekri, "The X.509 certificate quality," in *Proc. Third Int. Conf. on Digital Information Management (ICDIM)*, 2008, pp. 928–930. [Online]. Available: https://doi.org/10.1109/ICDIM.2008.4746813

[7] ITU-T, "Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks," ITU-T Recommendation X.509. [Online]. Available: http://handle.itu.int/11.1002/1000/11830-en

[8] C. Jadhav et al., "A Decentralized Vault-Based Trust Model for Secure Communication: Overcoming the Limitations of Certificate Authorities," in *Proc. 2026 Int. Conf. on Communication, Computing and Emerging Technologies (IC3ET)*, 2026, pp. 928–934. [Online]. Available: https://doi.org/10.1109/IC3ET64989.2026.11467316

[9] M. Y. Kubilay, M. S. Kiraz, and H. A. Mantar, "CertLedger: A New PKI Model with Certificate Transparency Based on Blockchain," *arXiv preprint arXiv:1806.03914*, 2018. [Online]. Available: http://arxiv.org/abs/1806.03914

[10] H. Kwon et al., "Certificate Transparency With Enhanced Privacy," *IEEE Trans. Dependable and Secure Comput.*, vol. 20, no. 5, pp. 3860–3872, 2023. [Online]. Available: https://doi.org/10.1109/TDSC.2022.3214235

[11] B. Laurie, A. Langley, and E. Kasper, "Certificate Transparency," RFC 6962, Jun. 2013. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc6962

[12] J. Nightingale, "Comodo Certificate Issue – Follow Up," Mozilla Security Blog, Mar. 2011. [Online]. Available: https://blog.mozilla.org/security/2011/03/25/comodo-certificate-issue-follow-up/

[13] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *J. Manage. Inf. Syst.*, vol. 24, no. 3, pp. 45–77, 2007. [Online]. Available: https://doi.org/10.2753/MIS0742-1222240302

[14] T. Saleem et al., "ProofChain: An X.509-compatible blockchain-based PKI framework with decentralized trust," *Comput. Netw.*, vol. 213, p. 109069, 2022. [Online]. Available: https://doi.org/10.1016/j.comnet.2022.109069

[15] Community Cryptography Specifications Project (C2SP), "Tile-based Transparency Logs," 2024. [Online]. Available: https://github.com/C2SP/C2SP/blob/main/tlog-tiles.md