Blockchain-Assisted Data Integrity Verification for Electronic Supply Chain Management

Benjamin Clarke

United Kingdom

Abstract

This study proposes a blockchain-assisted data integrity verification framework for electronic supply chain management, focusing on improving information authenticity, traceability, and transaction security across distributed supply networks. The proposed architecture integrates permissioned blockchain technology, cryptographic hashing, and smart contracts to establish tamper-evident records of electronic component procurement, manufacturing, transportation, and distribution activities. A decentralized verification mechanism enables authorized stakeholders to validate transaction histories and detect unauthorized modifications without relying on a centralized database. Smart contracts automate integrity checks, supplier authentication, and compliance verification based on predefined operational requirements. The framework incorporates off-chain data storage and on-chain hash verification to minimize computational overhead while maintaining data confidentiality and auditability. Performance evaluation considers transaction throughput, verification latency, storage efficiency, scalability, and resistance to data manipulation. The proposed approach supports transparent supply chain operations, counterfeit component traceability, secure information exchange, and reliable decision-making in electronic manufacturing and distribution networks.

Keywords: Blockchain Technology; Data Integrity Verification; Electronic Supply Chain Management; Smart Contracts; Supply Chain Traceability.

1. Introduction

Enterprise applications depend on reliable diagnostics to detect runtime errors, failed workflows, API issues, slow transactions, and integration faults [1]. Logs are the main evidence source for understanding what happened before, during, and after an incident. Effective application diagnostics require structured logging, traceability, contextual metadata, error classification, and operational visibility [2]. However, fixed logging strategies often create two problems. Low-detail logs may miss important causes, while high-detail logs may generate excessive storage cost and alert noise [3].

The main challenge is that diagnostic needs change during runtime. A normal transaction may require only basic logs, but a failing workflow, repeated exception, or security-sensitive event may need deeper traces [4]. Adaptive logging improves this process by changing log depth according to system condition, transaction risk, and error behavior [5]. AI-assisted monitoring can identify when additional diagnostic detail is needed and when logging can return to normal levels. This article proposes an adaptive logging framework for enterprise application diagnostics.

2. Methodology

The proposed framework captures log metadata from application servers, API gateways, databases, background jobs, authentication modules, and integration services [6]. It records event type, severity, timestamp, user role, transaction ID, request path, error frequency, response time, affected component, and workflow state. These features are converted into diagnostic context profiles. Adaptive logging requires context because the same event may be low-risk in one workflow but critical in another [7].

The logging controller classifies runtime states as normal, warning, error-prone, performance-risk, security-sensitive, workflow-critical, or incident-active [8]. Based on this classification, it adjusts log level, trace depth, parameter capture, correlation ID tracking, and diagnostic enrichment. Lightweight scoring and policy rules are used so logging decisions remain fast and explainable. Sensitive fields are masked before storage, and transaction-critical events are linked with workflow and database identifiers for easier diagnosis.

The framework is evaluated using simulated enterprise applications involving APIs, database workflows, authentication events, scheduled jobs, and service integrations. Static logging is compared with adaptive logging under repeated exceptions, slow requests, failed jobs, API timeout, suspicious login, and workflow interruption scenarios. Evaluation metrics include diagnosis time, log volume reduction, root-cause evidence quality, missed-event rate, storage overhead, alert usefulness, and administrator acceptance rate. Feedback from resolved incidents, ignored logs, and administrator notes is used to refine logging policies over time.

3. Results and Discussion

The results show that adaptive logging improves diagnostics by capturing richer evidence only when runtime risk increases. In the baseline condition, static logs either lack enough detail or produce large volumes of low-value records. With the proposed framework, detailed traces are activated for high-risk events and reduced during normal execution. The strongest improvement appears in intermittent failures, where deeper logs are needed only around the failure window.

The discussion shows that adaptive logging must balance diagnostic depth with privacy and storage cost. Excessive logging can expose sensitive data or overload monitoring systems, while weak logging may hide failure causes. The main limitation is that log-level decisions depend on accurate runtime classification. Regular policy review and masking controls are necessary for safe long-term use.

4. Conclusion

This article presented an adaptive logging framework for enterprise application diagnostics. The framework adjusts log detail using runtime context, risk scoring, workflow state, and error behavior. It improves troubleshooting by increasing diagnostic evidence during abnormal conditions while reducing unnecessary log noise during normal operation. The study shows that adaptive logging can strengthen enterprise diagnostics when combined with structured metadata, privacy controls, and continuous incident feedback.

References

1. rao Gondi, S. (2020). Intelligent Fault Detection and Classification in Semiconductor Manufacturing Using Statistical Learning Techniques. Transactions on Smart Electrical and Computational Systems, 7, 26-31.
2. Xu, X., Rahman, F., Shakya, B., Vassilev, A., Forte, D., & Tehranipoor, M. (2019). Electronics supply chain integrity enabled by blockchain. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, *24*(3), 1-25.
3. Islam, M. N., & Kundu, S. (2019). Enabling ic traceability via blockchain pegged to embedded puf. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, *24*(3), 1-23.
4. Hastig, G. M., & Sodhi, M. S. (2020). Blockchain for supply chain traceability: Business requirements and critical success factors. *Production and Operations Management*, *29*(4), 935-954.
5. Jabbar, S., Lloyd, H., Hammoudeh, M., Adebisi, B., & Raza, U. (2021). Blockchain-enabled supply chain: analysis, challenges, and future directions: S. Jabbar et al.. *Multimedia systems*, *27*(4), 787-806.
6. Malik, S., Dedeoglu, V., Kanhere, S. S., & Jurdak, R. (2019, July). Trustchain: Trust management in blockchain and iot supported supply chains. In *2019 IEEE international conference on blockchain (Blockchain)* (pp. 184-193). IEEE.
7. rao Gondi, S. (2021). Big Data Analytics Framework for Process Optimization and Yield Improvement in Semiconductor Production. Journal of Privacy-Aware Computing Systems, 8, 52-57.

8. Sahai, S., Singh, N., & Dayama, P. (2020, November). Enabling privacy and traceability in supply chains using blockchain and zero knowledge proofs. In *2020 IEEE International conference on blockchain (Blockchain)* (pp. 134-143). IEEE.