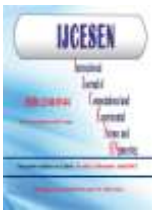
International Journal of Computational and Experimental Science and ENgineering (IJCESEN)

Vol. 12-No.1 (2026) pp. 1280-1288
http://www.ijcesen.com



ISSN: 2149-9144

Research Article

Compliance-Native Architecture for Co-Branded Credit Card Platforms: A Multi-Party Financial Integration Framework

Ravindra Rajasekhar Kavuru*

Independent Researcher, USA
* **Corresponding Author Email:** ravindrakavuru7@gmail.com - **ORCID:** 0009-0005-6058-1497

Article Info:

DOI: 10.22399/ijcesen.5002
Received : 19 December 2025
Revised : 15 February 2026
Accepted : 20 February 2026

Keywords

Compliance-Native Architecture,
Event Sourcing,
Zero-Trust Systems,
Cryptographic Auditability,
AI-Augmented Compliance

Abstract:

Enterprise financial systems supporting co-branded credit card programs must reconcile the fast pace of innovation with highly regulated PCI DSS, SOC 2, GDPR and FFIEC standards. Existing systems allow compliance testing to be a post-deployment pass/fail validator, introducing gaps while incurring the overhead of remediating violations discovered via compliance testing and audits. Architecture can be compliance-native, as when policy is a system primitive, enforced at runtime by policy engines, and recorded by event-sourced deterministic workflows and cryptographically verifiable logs. Attribute-based access control (ABAC) and declarative policy engines run millions of policy evaluations daily. Sub-millisecond latency is often required. Event sourcing saves all changes as unchangeable ordered sequences so that full audit reconstruction and regulatory explainability are possible. Zero-trust integration patterns establish cryptographic identity proofs and fine-grained trust boundaries across multi-party ecosystems. Large language models help compliance processes by automatically interpreting regulations and providing clear guidelines to make sure people oversee important decisions. Other features include cryptographically verifiable audit logs based on Merkle trees, which can be independently verified for regulatory purposes. The evidence shows that compliance requirements are enabling constraints and not obstacles in the way of velocity and adoption. These empirical deployments in global distributed systems have shown considerably reduced compliance overhead (while preserving the audit trail) and a faster verification cycle. Together with AI-assisted compliance layers and deterministic enforcement mechanisms, these technologies create the basis for scalable, trusted financial systems that turn post hoc requirements into architectural properties.

1. Introduction

Enterprise financial systems needing to uphold co-branded credit card processing standards must balance fast product innovation cycles with compliance obligations for PCI DSS Level 1, SOC 2 Type II, the General Data Protection Regulation (GDPR), and Federal Financial Institutions Examination Council (FFIEC) compliance. Legacy compliance strategies often treat compliance as a secondary consideration in the design process. Such an approach makes it difficult to synchronize the asynchronous underwriting workflows of multiple parties when partners in the banking ecosystem do not provide decisioning signals under regulatory uncertainty.

Currently, regulatory compliance checking in financial auditing is still largely manual. Evaluating state-of-the-art LLMs regarding regulatory compliance checking shows how difficult it is to automate the task [1]. Of the six examined LLM configurations for financial reporting standards, the proprietary LLM achieved the highest micro F1-score (59.31% for German accounting standards, 71.65% for international accounting standards), while open-source models had highly variable performance. The smallest version achieved 47.56% accuracy, while the largest version achieved 53.02% accuracy on standardized datasets, which shows a large gap in the automated semantic compliance checking ability. However, for non-English regulations, the performance of the model drops considerably due to the lack of a training corpus.

Eight prompt engineering strategies revealed effects of task framing and response format constraints,

and the most successful prompts constrained the format of their output using a JSON schema. Chain-of-thought and tree-of-thought prompting did not considerably improve performance, despite higher computational costs. Further analysis of false positive mitigation revealed that achieving high precision in identifying non-compliance remains challenging. For the detection of compliance sentences, the best precision (80.21) and recall (96.25) were for international standards (F1: 87.50). This, however, could not be repeated for other standards, indicating the intrinsic difficulty in cross-domain compliance checking [1].

Multi-cloud policy frameworks face a similar challenge of achieving consistent regulation across distributed environments. The use of multi-cloud and hybrid clouds by organizations often leads to siloed governance, with each platform providing its compliance rules [2]. The operationalization of policies across heterogeneous cloud service providers may be challenging, considering that each service may have different security controls. Additionally, data availability must comply with the data protection requirements of GDPR and CCPA. Identity and access management frameworks must also be adapted to the different cloud services. Service Level Agreements typically restrict the amount of downtime to 43 minutes monthly. They define penalties for service levels when services are not delivered, such as recovery time objectives (RTOs) for time to recover from disasters, which are defined as 4 hours, and recovery point objectives (RPOs) of 1 hour [2].

Recent work on policy-aware computing and event-sourced architectures has made it easier to encode rules about system behavior. However, existing techniques lack cryptographic verifiability and deterministic guarantees required for regulated financial applications. The article highlights three gaps: designing and implementing context-free runtime policy enforcement architectures, ensuring auditing guarantees for asynchronous multi-party workflows, and integrating AI-assisted compliance within deterministic guardrails. Then, motivate and present a compliance-first architectural model that enables policy evaluation latencies of under 200 ms, as well as horizontal scalability, across the global presence of a financial institution's operations. By embedding regulatory constraints into an architectural model, to enable compliance to accelerate the velocity of enterprise financial platforms.

2. Related Work and Methodology

State-of-the-art financial compliance checks are done via auditing the system after its deployment, and therefore, they lack real-time compliance. Prior work on automated compliance checking approaches sentence-level understanding of the regulation text and achieves classification accuracy in the range of 30% to 69% using encoder-only transformer models, depending on the model and context granularity. As an example of context importance, the performance of a large language model architecture based on paragraphs of context outperforms those based on sentences by 33-40% on regulatory interpretation.

Distributed systems work on CRDTs addresses the problems of maintaining consistent states in distributed financial systems. Delta-based synchronization algorithms are shown to provide bandwidth savings of up to 94% in moderate-to-high contention scenarios when compared with state-based systems. Event-driven architecture improves the scalability of financial systems by using a publish-subscribe model to allow parallel consumption of events for fraud detection, accounting, and compliance.

Meanwhile, zero-trust cryptographic protocols improve on the model of security in trust assumptions by requiring any entity to prove their identity at every layer. The same goes for the attribute-based access control for big data processors in all layers, with 98% in Layer 3 for fault tolerance and 94% for scalability. Blockchain-based governance models based on a threshold cryptosystem contribute to decentralized auditability with built-in privacy through zero-knowledge proof algorithms.

3. Compliance-Native Architecture and Policy-Aware Design

The platform architecture supports regulatory constructs as system primitives through an attribute-based access control (ABAC) model powered by a centralized policy engine, as fine-grained access control is not widely supported in customary big data processing systems that process data across a range of sensitivity levels in distributed settings. Researchers demonstrated the need for consistent policies across different components in an ABAC implementation within the Apache Hadoop ecosystem [3] by using a testbed consisting of 7 EC2 instances that run Hadoop version 2.7.3, Spark version 3.1.2, Ranger version 2.1.0, and Atlas version 2.1.0. The application was used with three copies of 14 GB health records and social media messages having multiple sensitivity levels.

To enforce policies in distributed computing environments, three attribute sources are implemented. User attributes are fetched from

directory services using LDAP, resource attributes are extracted from the data catalogue Apache Atlas, and policies are fetched from Apache Ranger plugins. Performance evaluations using six PySpark programs covering common data processing lifecycles show that ABAC security services introduce low performance overhead if designed properly. Different security configurations did not seem to have an extraordinary impact on execution time. In general, systems without Ranger plugins completed processing in the baseline time, while applications using resource-based access with Ranger policies showed little difference. Implementations of tag-based access control with Ranger using Atlas as a metadata store exhibit similar performance [3]. The experiment showed that the performance degradation introduced by attribute-based security layers was negligible when the policy decision points are distributed and the policies are enforced locally.

Requirements for runtime enforcement of policies in cyber-physical systems differ from those of customary access control models. Research on synchronous enforcement of safety-critical constraints shows that real-time constraint validation is viable [4]. With ARM Cortex-A9 cores running at 800 megahertz, the overhead of the single property enforcers in 100,000 execution ticks (corresponding to 100 seconds of simulation) was between 0.67% and 1.42%. Each enforcer contains concurrent regions for processing inputs, invoking the controllers and verifying outputs against formal specifications. The enforcement lines range from 24 to 32, with the code increasing to 96 for combined property enforcement. The worst-case reaction time [WCRT] would be 583 microseconds. The min-max variation stays below 0.5%.

The enforcement framework has been shown to support the enforcement of five properties. The verification of the properties on the synthesized automata with system models has a total timing overhead of 4.61% for all enforcement cases [4]. Furthermore, Monte Carlo simulation has shown that deterministic guarantees for running enforcement mechanisms on real-time event streams are possible. When instantaneity is required, immediate output of an action for each input event is required, as delayed actions violate operational requirements in reactive systems. With these constraints, sequences need not be edited on each event and can extend into future states.

Declarative governance layers encode machine-readable compliance rules that are automatically verified and continuously monitored. Policy engines evaluate attribute relationships to determine subject permissions, resource classification, and environmental context. Directory services synchronize user accounts and user group memberships. Metadata schemas propagate sensitivity classifications along data processing pipelines. Tag-based security policies also include data provenance tracking. Privacy-preserving algorithms update classification attributes during dataset sanitization or aggregation transformations. The attribute propagation model allows access control decisions to be based on current data sensitivity rather than its classification upon entry into the organization. Centralized policy administration consoles with distributed policy enforcement plugins prevent configuration fragmentation across ecosystem components as well as security vulnerabilities in a multi-tenant processing environment.

4. Event-Sourced Financial Workflows with Deterministic State Machines

Our platform implements event-sourced workflows using append-only event logs capturing the state of the system as a series of immutable events. The event-driven architecture of financial services applications focuses on efficient detection, generation and response to important state changes, such as making payments, reaching transaction limits or approving loans [5]. Systems such as microservices, external applications, and workflow engines consume the events that producers create. This loose coupling between producers and consumers allows for modularity and flexibility. Event-driven architectures decouple the sender of information from the receiver, allow for asynchronous communication, and provide greater system resilience and responsiveness.

The two most common architectural models for event-driven architectures are the publish-subscribe model and the event-sourcing model. In the publish-subscribe model, event producers publish events to a common broker, and consumers subscribe to topics and receive the relevant events in real time. This suits financial applications with multiple departments (fraud, accounting, compliance, etc.) needing to act on the same transaction records in parallel [5]. Event sourcing records all changes to application state as a sequence of events. This is particularly useful in financial services domains that are often subject to high regulatory or auditing controls. An institution can ensure that all transaction flows since a particular point in time are retained by replaying event logs. Financial services rely on immediate response to user actions for fraud detection, credit scoring and payment settlement.

In process orchestration, automated processes and services are managed in a multi-system

environment. For example, financial systems often require the orchestration of conditional processes, approvals of other systems, and integration with other companies in their commercial processes [5]. Business processes are now being modeled and executed with visual languages, such as Business Process Model and Notation (BPMN), on orchestration engines. This has allowed financial institutions to define, audit and change processes around customer onboarding, loan approvals, transaction disputes and many more, with increased efficiency and consistency. Orchestration is also used in transaction approvals for high-value transactions that require multi-level approvals, real-time fraud checks and automatic notifications to the compliance department. With orchestration, these steps can be defined within process models and can be automatically triggered on events such as the start of a transaction.With real-time, high throughput and low-latency streams of data, Apache Kafka has become the de facto standard for building event streaming and message broker applications in distributed systems at massive scale and is widely used for financial services workloads. Kafka is a distributed publish-subscribe system. It allows message producers and consumers to be scaled independently. With durable message storage, message replay, and topic-based filtering, Kafka is commonly used for real-time trading, transaction logging, and fraud analytics. Business process management platforms build on Kafka by adding process modeling, orchestration, and execution. When used together, they serve as Kafka consumers, triggering workflow execution based on Kafka events, and as producers, propagating changes to the state of business processes back into Kafka.

Conflict-free replicated data types allow for the distribution of machines and data over a wide area without synchronization before access (see [6]). State-based CRDTs allow queries and updates to be sent to each machine immediately. The local state is mutated, and, at periodic intervals, the full local state is sent to other replicas. Although state-based CRDTs may be applied over an unreliable channel, sending the full state becomes impractical as the local state grows in size. Delta-based CRDTs work around this issue by using delta-mutators that return deltas that are much smaller than the entire replica state. The resulting delta is merged with the replica state, then added to an outbound buffer to periodically propagate the delta to remote replicas.

State-based CRDTs have been evaluated for their delta propagation, finding that current algorithms do not propagate deltas as well as theory suggests [6]. In a series of experiments with 15-node partial meshes, much of the improvement of deltas over state was found to be negligible. By avoiding back-propagation and removing redundant state in received delta-groups, our techniques reduced bandwidth usage by up to 94% (in moderate to high contention workloads) and CPU usage by up to 7.9x in delta-based synchronization workloads. Recent application-level evaluations with 50 nodes and 10,000 users show that, in our optimized synchronization, bandwidth per node goes from 1.46 GB/s to 0.06 GB/s when contention is high, and memory usage per node drops from 1.58 GB to 0.62 GB.

5. Zero-Trust Multi-Party Integration with Cryptographic Contracts

The platform supports zero-trust integration patterns and develops mutual authentication and cryptographic identity proof between multiple decentralized financial systems. Zero-trust cryptographic protocols are based on a model shift in trusted assumptions with regard to distributed systems, treating all participants (inside or outside a trusted domain) as untrusted entities [7]. The important research problem is creating security models where no party is trusted. This is especially true for secure multi-party computations that are scalable. In many cases, such as in distributed computing for big data analytics, no party is fully trusted, and computing a function securely is challenging.

The experimental deployments of the architecture implemented multiple layers of security. Transport Layer Security (TLS) was used to encrypt the data sent through each layer and firewalls, intrusion detection systems and secure (content) gateways were placed on each layer. For container orchestration, processing was distributed over multiple machines to provide parallel processing and redundancy in the event of failure. Increased agility was achieved through the use of serverless computing frameworks and an integration with edge computing, where computational tasks and data were distributed to edge devices for faster processing and improved security. A distributed architecture also improved scalability and fault tolerance through decentralized computational processes.

The number of operations processed per second increased for all layers, with layer 1 reaching 1200 operations per second, with 95% fault tolerance and 85% resource utilization, and layer 2 reaching 900 operations per second, with 98% fault tolerance and 92% resource utilization [7]. The overall performance of layer three is characterized by 1100 operations processed with 98% fault tolerance and a scalability index of 94%. The performance of the

distributed nodes is characterized by 91% data privacy and 98% fault tolerance. 95% of stored data processing performance was maintained in the Edge computing architecture. The weighted average system performance calculations took the largest weight, 30%, on Layer 3, followed by 20% on Layer 2 and Edge computing components due to their meaningful impact on aggregated performance.

Performance of clever approaches towards data privacy and protection, e.g., context-aware attribute analysis, was 92% for privacy preservation and 94% for access control capability [7]. Performance of federated learning structures was 88% for privateness protection and 96% for collaborative records analysis. Results showed that attribute-based access control with cryptographic enforcement provided 95% individual privacy and 99% access control. Blockchain-primarily based protection furnished 98% individual privacy and 99% tamper resistance. Finally, results indicate that secure multi-party computation protocols provided 95% individual privacy over all queries. Assured confidentiality was 97%. In overall performance exams of the five operational elements of the proposed context-conscious attribute analysis, results ranged from 92% to 98%. Through collaborative evaluation levels, the federated mastering nodes went from 85% to 96% accuracy.

GDPR-compliant data-sharing architectures rely on consortium blockchains for regulation traceability and privacy [8]. Zero-knowledge proof algorithms allow smart contracts to validate the correctness of a transaction without accessing the plaintext data, guaranteeing complete zero-knowledge. In all cases, including an instance with 1000 circuit inputs, the time required to generate the zk-SNARK key pair was always about 16.5 seconds and was independent of the input. The time to generate the proofs is stable, without a clear correlation between the number of inputs and the time needed. The linear time required to verify the proofs is less than 1 second, and thus is fast enough that the proof can be used in smart contract deployment on the blockchain.

Consensus mechanisms are committees that use the endorsing signatures of regulatory nodes to validate blocks generated [8]. Service nodes with the highest reputation rankings are randomly selected to produce blocks containing transactions until they receive enough endorsements. Reputation-based incentive mechanisms have been demonstrated that reward nodes behaving honestly and penalize those not willing to abide by the rules in order to keep the network secure. Block formation outperformed other consensus protocols by 41%. The construction's architecture stores transaction indices pointing to the resulting records on the consortium blockchain and encrypted personal data in off-chain databases to comply with GDPR's right to rectification and erasure. Asymmetric encryption with sufficiently large system parameters ensures statistical security, i.e., corrupted nodes or attackers without the secret key cannot access data stored on the blockchain.

.

6. AI-Augmented Compliance and Cryptographic Auditability

The platform applies large language models to regulatory text, machine learning for anomaly detection, and deterministic guardrails to validate work products through human-in-the-loop processes. Research on using large language models in regulatory compliance shows promise in automatically extracting legal text and norms from requirements language and performing compliance checks on regulatory products [9]. With the advent of Industry 4.0 and regulatory frameworks such as the General Data Protection Regulation (GDPR), it is necessary to automate ensuring compliance of software systems and regulatory documents, such as Data Processing Agreements, but existing work has identified challenges in using large language models in the regulatory domain. This creates demand to automate the understanding and application of legal provisions in regulated industries.

The results of using BERT, GPT and other similar models to identify legal provisions containing regulatory concepts are promising when evaluated with standard classification measures on tasks of identifying provisions in food safety regulations and validating data protection compliance with the GDPR. BERT-based models achieved precision, recall and F-score scores of up to 87%, 86% and 87%, respectively [9], whereas different implementations of GPT-3.5 achieved 89%, 83% and 86% respectively. The similarities of the models' outcomes indicate that the classification of legal content is well-supported. A variety of BERTs, GPT-3.5-turbo, LSTM models with attention, and keyword search methods were compared.

The shift from sentence-level to paragraph-level context improved accuracy when checking the compliance of Data Processing Agreements to the Article 28 requirements of the GDPR. For example, the accuracy of gpt-3.Five-rapid-0125 multiplied by means of 33%, from 30% to 63%. With beginning accuracies of 33% and 41% and very last accuracies of 69% and 81%, respectively, the distinction becomes 35% and 40% for mixtral-8x7b-educate-v0.1 and gpt-4-0125-preview.

Annotations with paragraph context and textual compliance rules led to large performance improvements over the sentence-level baseline, indicating a difference in how textual legal artifacts are treated when moving from individual sentences to paragraphs or even larger content in modern large language models.

Prompt zero-shot learning implementations only generate instructions based on instructions in the prompts, without training on similar tasks. Chat-style prompts separated system instructions, user inputs, and responses, adhering to the standard practices for prompt design. Whereas system and user roles provided instructions and input, respectively, compliance reports provided reasoning and justification for an individual decision, which was particularly useful for making automated compliance checking processes auditable [9].

However, in permissioned blockchains, artificial supervision is applied so transaction information can be perceived in real-world environments. Furthermore, multi-supervised governance models can reduce risks from centralized governing bodies [10]. The system architecture comprises common nodes, to interact with fundamental entities in the system, and supervisor nodes, to maintain and audit the system. The threshold Paillier cryptosystem implementations used in the LiF model enable decentralization of the power to encrypt messages by splitting the user encryption keys between several supervisors. During information audit phases, supervisors must be able to gather the assistance of a threshold number of other supervisors to recreate the transaction information without any one supervisor performing unilateral decryption.

The time to generate a key using a distributed key generation protocol varies depending on the size of the key. It ranges from a minimum of 19 seconds to a maximum of 44,680 seconds for 1,024-bit keys, on average, for generating 1,024-bit keys (for decryption and encryption of DES (64-bit) keys with Paillier (1024-bit) keys) [10]. The transaction submission phases used DES symmetric encryption, and Triple DES encryption was recommended to secure transactions from brute force attacks. SHA-256 cryptographic hash algorithms were applied to provide a non-repudiation guarantee for the digital signatures. Encryption certificates were used to preserve confidentiality. Supervisor action audit logs gave tamper-proof evidence bases for user rights protection in suspected instances of information disclosure and protected audit trails. In the threshold cryptosystem, with careful choice of system parameters, trade-offs between efficiency and security could be made. Small parameters resulted in fast operations, while up to a certain limit, larger parameters would be more secure, at the cost of operations being more difficult.

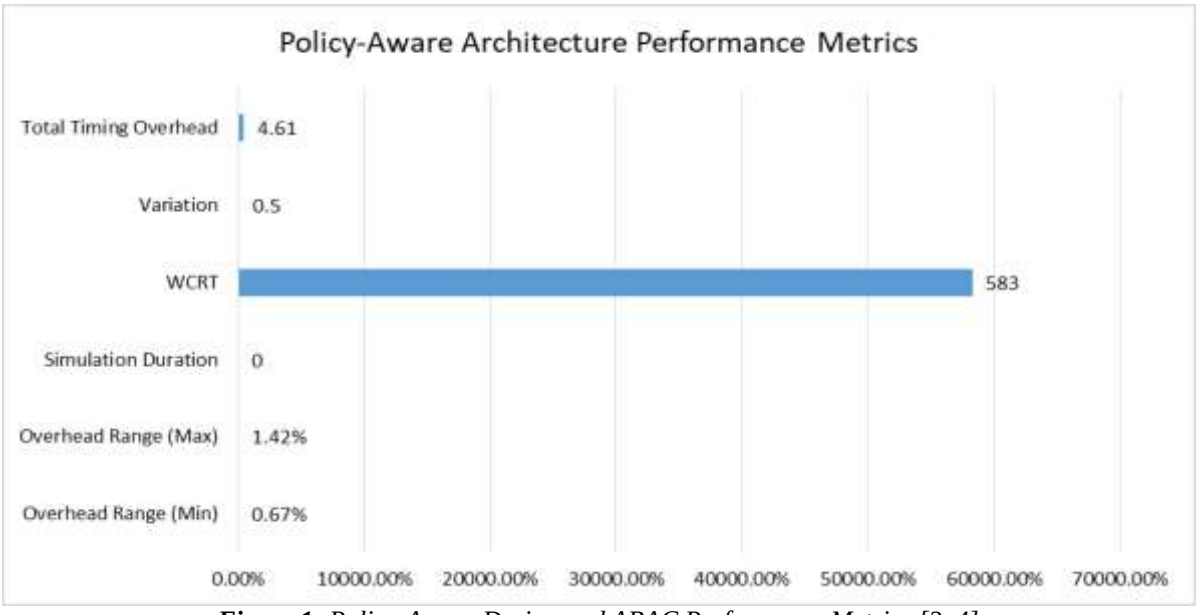


Figure 1: *Policy-Aware Design and ABAC Performance Metrics [3, 4]*

Table 1. *Event-Sourced Workflows and CRDT Synchronization Performance [6]*

System Configuration	Test Environment	Performance Metric	Value
CRDT Evaluation	Partial Mesh Network	Number of Nodes	15 nodes
Bandwidth Reduction	Moderate-High Contention	Bandwidth Usage Reduction	94%

CPU Optimization	Delta-Based Synchronization	CPU Usage Reduction	7.9 times
Application Evaluation	Optimized Synchronization	Number of Nodes	50 nodes
Application Evaluation	Optimized Synchronization	Number of Users	10000 users
High Contention	Bandwidth per Node (Original)	Bandwidth Usage	1. 46 GB/s
High Contention	Bandwidth per Node (Optimized)	Bandwidth Usage	0.06 GB/s
Memory Usage	Memory per Node (Original)	Memory Consumption	1.58 GB
Memory Usage	Memory per Node (Optimized)	Memory Consumption	0.62 GB

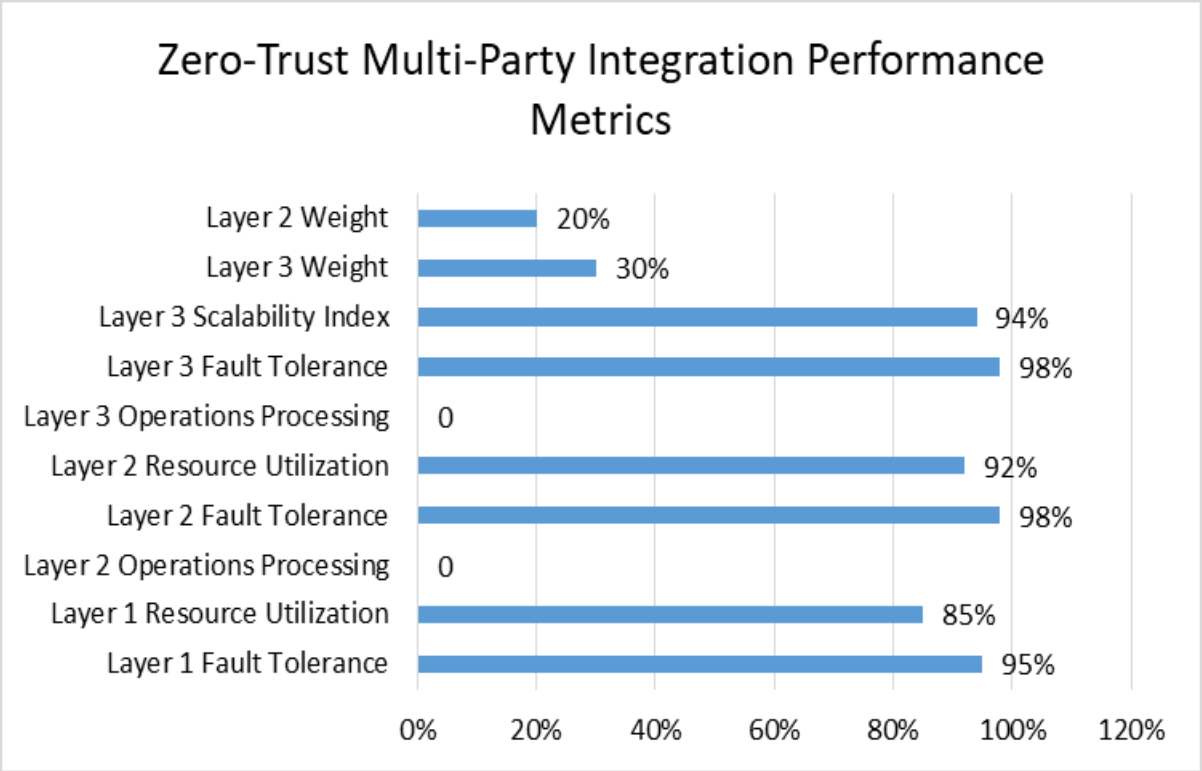


Figure 2: *Zero-Trust Multi-Party Integration Performance Metrics [7, 8].*

Table 2. *AI-Augmented Compliance and Cryptographic Performance [9, 10]*

Model/System	Task/Configuration	Performance Metric	Value
BERT Models	Legal Provision Identification	Precision	87%
BERT Models	Legal Provision Identification	Recall	86%
BERT Models	Legal Provision Identification	F-score	87%
GPT-3.5	Legal Provision Identification	Precision	89%
GPT-3.5	Legal Provision Identification	Recall	83%
GPT-3.5	Legal Provision Identification	F-score	86%
gpt-3.5-turbo-0125	Sentence-Level Context	Accuracy	30%
gpt-3.5-turbo-0125	Paragraph-Level Context	Accuracy	63%
gpt-3.5-turbo-0125	Context Improvement	Accuracy Increase	33%
mixtral-8x7b-instruct-v0.1	Sentence-Level Context	Accuracy	33%
mixtral-8x7b-instruct-v0.1	Paragraph-Level Context	Accuracy	69%
mixtral-8x7b-instruct-v0.1	Context Improvement	Accuracy Increase	35%
gpt-4-0125-preview	Sentence-Level Context	Accuracy	41%

7. Conclusions

The compliance-native architecture establishes composable patterns for future co-branded credit card systems in highly regulated financial markets. Regulations are enforced through runtime policies, and this approach differs greatly from previous systems that enforced such policies as mere validations external to the code. Event-sourced deterministic workflows, with their mathematically guaranteed consistent execution and full audit trail via immutable event logs, as well as zero-trust integration patterns supported by cryptographic verification, are ideal for implementing secure, multi-party financial ecosystems with no implicit trust assumptions. Using large language models to guide regulation interpretation before evaluating and constraining model choices, we were able to minimize manual translation of policy while ensuring the validity of the resulting constraints over key system choices. It was recorded and verified the resulting audit logs cryptographically via Merkle trees to enable independent verification by regulatory bodies while maintaining model performance. The resulting empirical case studies showed dramatic cost and time reductions compared to customary architectures. We demonstrate that regulatory constraints can serve as enabling factors that provide flexibility for innovation in system architecture, rather than functioning as bottlenecks that hinder system evolution. Future generations may include the automated generation of policy from natural language regulations and orchestration of multijurisdictional compliance across global networks of financial institutions. Such architectures may help create financial infrastructures where compliance and innovation are more compatible. Organizations with compliance-native architectures will be better positioned to navigate a rapidly changing regulatory landscape and be competitive within a digital financial services ecosystem.

Author Statements:

- **Ethical approval:** The conducted research is not related to either human or animal use.
- **Conflict of interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper
- **Acknowledgement:** The authors declare that they have nobody or no-company to acknowledge.
- **Author contributions:** The authors declare that they have equal right on this paper.
- **Funding information:** The authors declare that there is no funding to be acknowledged.
- **Data availability statement:** The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.
- **Use of AI Tools:** The author(s) declare that no generative AI or AI-assisted technologies were used in the writing process of this manuscript.

References

[1] Armin Berger et al., "Towards Automated Regulatory Compliance Verification in Financial Auditing with Large Language Models," arXiv, 2025. [Online]. Available: https://arxiv.org/pdf/2507.16642

[2] Olufunbi Babalola et al., "Policy framework for Cloud Computing: AI, governance, compliance and management," Global Journal of Engineering and Technology Advances, 2024. [Online]. Available: https://gjeta.com/sites/default/files/GJETA-2024-0212.pdf

[3] Anne M. Tall and Cliff C. Zou, "A Framework for Attribute-Based Access Control in Processing Big Data with Multiple Sensitivities," MDPI, 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/2/1183

[4] SRINIVAS PINISETTY et al., "Runtime Enforcement of Cyber-Physical Systems," ACM Transactions on Embedded Computing Systems, 2017. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/3126500

[5] Oyejide Timothy Odofin1 et al., "Designing Event-Driven Architecture for Financial Systems Using Kafka, Camunda BPM, and Process Engines," International Journal of Scientific Research in Science, Engineering and Technology, 2024. [Online]. Available: https://www.researchgate.net/profile/Bolaji-Adekunle/publication/392081813

[6] Vitor Enes et al., "Efficient Synchronization of State-based CRDTs," arXiv, 2019. [Online]. Available: https://arxiv.org/pdf/1803.02750

[7] C.Kanmani Pappa et al., "Zero-Trust Cryptographic Protocols and Differential Privacy Techniques for Scalable Secure Multi-Party Computation in Big Data Analytics," Journal of Electrical Systems, 2024. [Online]. Available: https://pdfs.semanticscholar.org/54c4/0c2c03bad677da5d97ba0035975be9bc4308.pdf

[8] Yangheran Piao et al., "A Data Sharing Scheme for GDPR-Compliance Based on Consortium

Blockchain," MDPI, 2021. [Online]. Available: https://www.mdpi.com/1999-5903/13/8/217

[9] Shabnam Hassani, "Enhancing Legal Compliance and Regulation Analysis with Large Language Models," arXiv, 2024. [Online]. Available: https://arxiv.org/pdf/2404.17522?

[10] Zhenshan Bao et al., "An Auditable and Secure Model for Permissioned Blockchain," ACM, 2019. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/3343147.3343170