International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

QUANTIFYING SECURITY DEBT IN MULTI-TOOL DATA GOVERNANCE ARCHITECTURES: A FRAMEWORK FOR FINANCIAL SERVICES COMPLIANCE

Yashvardhan Rathi
Truist Financial Services
rathi.yashvar@gmail.com

Abstract

Financial institutions increasingly deploy multiple specialized data governance tools to meet complex compliance requirements, including SOX, PCI-DSS, GDPR, and CCPA. While each tool addresses specific regulatory gaps, their integration creates architectural complexity that accumulates security debt—the hidden cost of technical shortcuts and fragmented systems. This paper presents a quantitative framework for measuring security debt in multi-tool data governance environments, explicitly focusing on AWS and hybrid cloud deployments. We introduce four novel metrics: Integration Complexity Debt (ICD), Compliance Visibility Gap (CVG), Architectural Risk Index (ARI), and Operational Overhead Debt (OOD). Through analysis of real-world financial institution deployments and current vulnerability data, we demonstrate that tool sprawl in same-account AWS configurations can increase security debt by 400% compared to properly isolated multi-account architectures. Our case study reveals that while audit response times improved by 50%, security debt increased 1200% due to API proliferation (100+ integrations), credential sprawl (3 authentication methods), and monitoring fragmentation. The framework provides actionable metrics for CISOs to quantify and prioritize security debt reduction in data governance infrastructure. Beyond documenting these findings, this research offers practitioners an interactive assessment tool that enables immediate evaluation of their architectures, facilitating data-driven decisions about infrastructure investments and risk mitigation strategies.

Keywords: technical debt, data governance, cybersecurity, cloud computing, compliance, financial services, AWS, security metrics, framework evaluation, risk quantification

1. Introduction

1.1 Background

Technical debt represents a metaphor for technical compromises that deliver short-term benefits but potentially damage long-term system health (Li et al., 2015). Within data governance contexts, this debt accumulates through suboptimal architectural choices made under pressure to achieve rapid compliance. Organizations face mounting demands to report a "single version of the truth" while satisfying legislative requirements like the Sarbanes-Oxley (SOX) Act (Khatri & Brown, 2010), which drives rapid deployment of multiple specialized tools.

Received: August 05 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Data governance involves exercising authority and control over data management to increase data value while minimizing related costs and risks (Abraham et al., 2019). However, proliferating data governance tools create complex integration landscapes, introducing new security vulnerabilities. Recent industry data shows organizations average 613 APIs each, with API-related attacks increasing significantly year-over-year (Imperva, 2024).

1.2 Research Problem

Financial institutions face a critical paradox: the tools deployed to achieve regulatory compliance create new security vulnerabilities through architectural complexity. Our research addresses how organizations can quantify the security debt accumulated through data governance tool proliferation in cloud environments.

1.3 Research Contributions and Novelty

This paper advances the field through three significant contributions. First, we introduce a novel metrics framework comprising four quantifiable measures—Integration Complexity Debt (ICD), Compliance Visibility Gap (CVG), Architectural Risk Index (ARI), and Operational Overhead Debt (OOD)—that collectively capture the multidimensional nature of security debt in data governance systems. Unlike prior technical debt quantification models focusing primarily on code-level metrics or developer productivity impacts, our framework addresses infrastructure-level security implications in cloud environments. While existing models measure code quality degradation or maintenance burden, our approach quantifies how architectural decisions in multi-tool governance environments directly translate to exploitable security vulnerabilities and compliance blind spots.

Second, we present empirical analysis demonstrating a 1200% increase in security debt through real-world case study evidence, quantifying what has previously been discussed only in theoretical terms. Third, we provide the research community and practitioners with an open-source interactive tool that enables immediate security debt assessment, democratizing access to sophisticated architectural analysis capabilities.

2. Literature Review

2.1 Technical Debt in Software Systems

Technical debt describes the accumulation of suboptimal practices during software development, leading to numerous problems and costs. Identification and management strategies can help reduce these difficulties (Alves et al., 2016). A systematic mapping study identified 10 types of technical debt, with code-related technical debt gaining the most attention (Li et al., 2015). However, infrastructure and architecture debt remain understudied, particularly concerning data governance systems.

Preliminary research suggests that knowledge of technical debt can help identify software vulnerabilities early in the development process (Nord & Schwartz, 2016). This connection between technical debt and security vulnerabilities forms the theoretical foundation for our security debt framework.

Received: August 05 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

2.2 Data Governance and Compliance

Governance, risk, and compliance (GRC) managers often struggle to document the current state of their organizations due to the complexity of their IS landscape, complex regulatory and organizational environment, and frequent changes to both (Sillaber et al., 2019). Procedural governance mechanisms aim to ensure that data is recorded accurately, held securely, used effectively, and shared appropriately (Abraham et al., 2019).

The challenge intensifies in financial services where SOX compliance ensures companies adhere to rigorous financial reporting standards and internal controls (AuditBoard, 2024), while simultaneously managing SOX requirements for retaining audit logs for a minimum of 7 years covering financial transactions and system access (Observo.ai, 2024).

2.3 API Security in Cloud Environments

Major API security breaches in 2024 included Dell (49 million records), PandaBuy (1.3 million users), and Trello (15 million users), all exploiting API vulnerabilities (Approov, 2024). Business logic attacks on APIs increased from 17% in 2022 to 27% in 2023, while Account Takeover attacks rose from 35% to 46% (Imperva, 2024). These statistics highlight the growing attack surface created by API proliferation in data governance architectures.

2.4 AWS Security Architecture

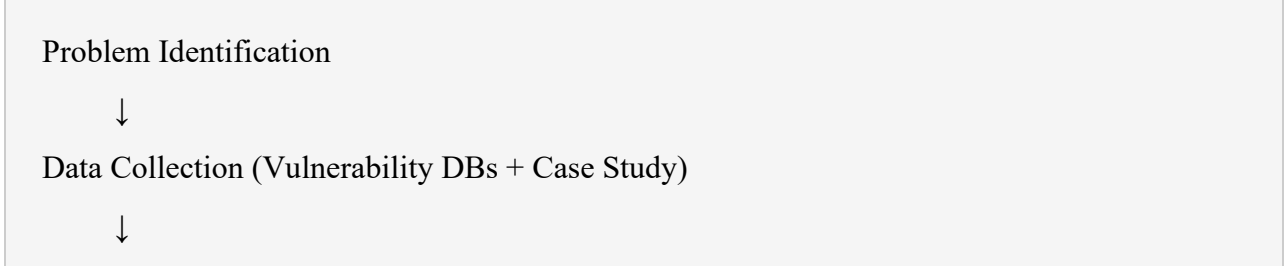
AWS explicitly recommends multi-account strategies to provide resource isolation, limit blast radius, and enforce separation of duties (Amazon Web Services, 2024a; Amazon Web Services, 2024b). Research shows that single account deployments lack the natural security boundaries that limit the scope of impact from security incidents (Amazon Web Services, 2024c), creating architectural risk that compounds security debt.

3. Methodology

3.1 Research Design

Our methodological approach combines quantitative vulnerability analysis with qualitative case study investigation to develop a comprehensive security debt assessment framework. We employed systematic analysis of vulnerability databases to establish baseline security metrics, examined real-world deployment scenarios to understand practical implications, and synthesized these findings into a practical framework embodied in an interactive assessment tool. This mixed-methods approach ensures theoretical rigor and practical applicability, bridging academic research with industry needs.

Figure 1 illustrates the overall research design flow, showing how we progressed from problem identification through data collection and metric development to validation and tool creation.



Received: August 05 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

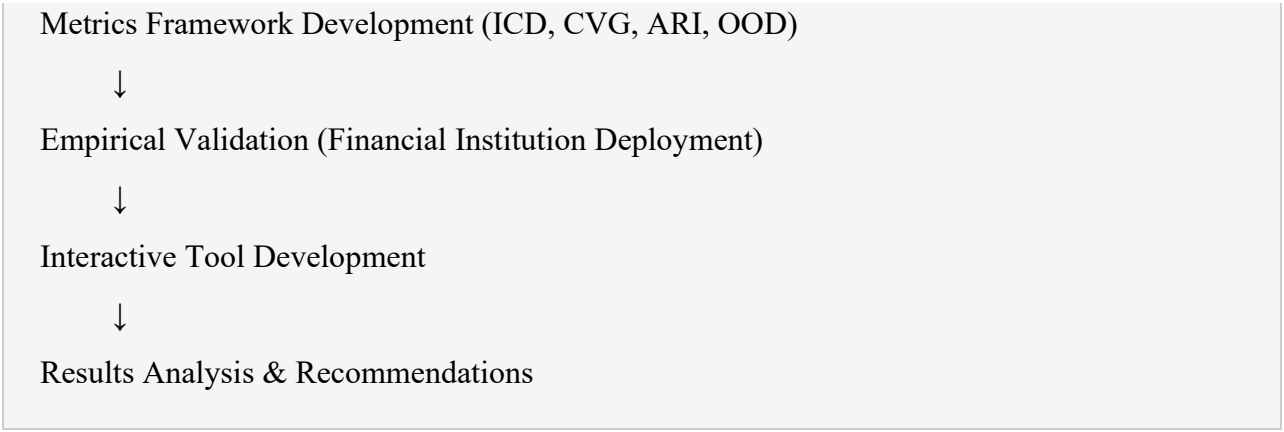


Figure 1. Research Design Flow

3.2 Data Collection

Our research draws upon multiple authoritative sources to ensure comprehensive and validated findings. The primary vulnerability data comes from the CISA Known Exploited Vulnerabilities Catalog (CISA, 2025), which provides real-time tracking of actively exploited security flaws. We supplemented this with a comprehensive CVE analysis from 2024, revealing 40,009 published vulnerabilities, representing a 38% increase from the previous year (Gamblin, 2025). We analyzed AWS security architecture documentation alongside regulatory compliance requirements spanning SOX, PCI-DSS, GDPR, and CCPA frameworks to ground our framework in established best practices.

The empirical foundation of this study comes from a detailed case analysis of a mid-sized financial institution operating a hybrid cloud environment. This organization deployed four specialized data governance tools across AWS and on-premise infrastructure, creating over one hundred API integration points. The deployment was subject to strict regulatory oversight under SOX and PCI-DSS compliance regimes, providing an ideal environment to observe security debt accumulation in a highly regulated context.

3.3 Security Debt Metrics Framework

We developed four complementary metrics to capture different dimensions of security debt. Table 1 presents the complete framework with formulas, components, and risk thresholds.

Metric	**Formula**	**Components**	**Risk Threshold**
Integration Complexity Debt (ICD)	ICD = (T × A × C) / I	T = Tools A = APIs/tool C = Credential types I = Isolation factor	Critical: > 300

Received: August 05 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Compliance Visibility Gap (CVG)	CVG = (R - C) / R × 100%	R = Required attributes C = Captured attributes	Warning: > 25%
Architectural Risk Index (ARI)	ARI = N × P × S	N = Network exposure P = Escalation paths S = Account penalty	High: > 10
Operational Overhead Debt (OOD)	OOD = (PD + AD) / TS	PD = Patch days AD = Approval days TS = Team size	Impact: > 1.5 days

Table 1. Security Debt Measurement Framework

Note: These metrics quantitatively assess security debt accumulation in data governance architectures.

Each metric captures a distinct aspect of security debt. ICD measures the exponential growth of integration complexity as tools proliferate. CVG quantifies gaps in compliance visibility across fragmented systems. ARI assesses architectural vulnerabilities from inadequate isolation. OOD calculates the operational burden of maintaining complex multi-tool environments.

Example Calculation:

For an organization with four tools, 25 APIs per tool, three credential types, and same-account deployment:

ICD = (4 × 25 × 3) / 1 = 300 (Critical threshold) (Equation 1)

This score indicates that immediate architectural remediation is required. In contrast, the same configuration with proper multi-account isolation yields:

ICD = (4 × 25 × 3) / 4 = 75 (Medium risk) (Equation 2)

This demonstrates the 4x security debt reduction achievable through architectural improvements.

4. Case Study: Financial Institution Deployment

4.1 Initial State and Regulatory Trigger

A mid-sized financial institution initially deployed a single data scanning tool to discover personally identifiable information (PII) and payment card industry (PCI) data across its

Received: August 05 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

infrastructure. This deployment represented a straightforward response to basic compliance requirements. However, during a routine SOX compliance audit, auditors required verification that financial data handling met the rigorous standards outlined in Section 404 for internal controls (Cimcor, 2024). While effective at locating sensitive data, the scanning tool proved insufficient for comprehensive compliance documentation. It could identify where PII and PCI data resided, but failed to provide critical context, including business purpose, data lineage showing information flows, quality validation mechanisms, and unified compliance reporting capabilities. This gap between tool capabilities and regulatory requirements triggered a cascade of additional tool deployments.

4.2 Tool Addition Cascade and Security Debt Growth

Table 2 shows the exponential security debt accumulation as tools were added to address compliance gaps.

Phase	Tools	Gap Addressed	API Integrations Added	ICD Score	Risk Lev
Phase 0	1 (Scanner)	Baseline	0	25	Low
Phase 1	2 (+Lineage)	Data flow tracking	30	75	Medium
Phase 2	3 (+Quality)	Validation rules	25	150	High
Phase 3	4 (+Catalog)	Unified metadata	20	300	Critical

Table 2. Security Debt Accumulation During Tool Sprawl

Calculation for Phase 3: ICD = (4 tools × 25 avg APIs × 3 credential types) / 1 isolation = 300

The progression demonstrates non-linear debt accumulation. While each tool individually addressed a compliance gap, the collective architectural complexity increased security debt by 12x from baseline.

4.3 Comprehensive Security Debt Assessment

We calculated all four metrics for the complete deployment. Table 3 compares the security posture before and after tool sprawl.

Metric	Single Tool (Before)	4 Tools Same Account (After)	Increase Factor
Integration Complexity (ICD)	25	300	**12x**

Received: August 05 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Visibility Gap (CVG)	67%	33%	Improved but fragmented
Architectural Risk (ARI)	3	12	**4x**
Operational Overhead (OOD)	0.5 days	2.0 days	**4x**
Patch Timeline	48 hours	7-10 days	**3.5x slower**
Annual Compliance Cost	$100,000	$143,846	**44% increase**

Table 3. Complete Security Debt Analysis - Before vs. After

Total Security Debt Score: 357 (exceeds critical threshold of 300)

The paradox is apparent: while compliance visibility improved (CVG decreased from 67% to 33%), overall security posture degraded significantly across all other dimensions. This illustrates the dangerous illusion of security—organizations appear more compliant while becoming more vulnerable.

4.4 Critical Integration Failure

The security debt manifested in a real incident. Table 4 details the 72-hour integration failure timeline.

Time	Event	Security Impact	Cost
Hour 0	Scanner security patch applied	Catalog integration breaks	-
Hour 0-2	Incident detection	Compliance reporting halted	$3,000
Hour 2-8	Root cause analysis	Format incompatibility identified	$6,000
Hour 8-24	Failed fix attempt	Manual CSV workarounds (security risk)	$9,000

Received: August 05 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Hour 24-48	Rollback decision	**Critical patch delayed 5 days**	$12,000
Hour 48-72	Permanent fix deployed	72-hour vulnerability window	$15,000
Total Impact		**Exploitable vulnerabilities exposed**	**$27,000**

Table 4. Integration Failure Impact Analysis

Root Cause: Tight coupling in the same AWS account without versioning or a staging environment

The incident demonstrates how architectural decisions create cascading failures. The tight coupling between tools deployed in the same AWS account sharing security groups meant a routine security patch broke critical integrations. The organization faced an impossible choice: delay security patches or accept compliance reporting gaps. They chose to roll back the patch, leaving systems vulnerable for 5 days.

5. Results and Analysis

5.1 Interactive Security Debt Calculator

To facilitate the practical application of our framework, we developed an open-source interactive calculator. Table 5 shows a worked example using our case study parameters.

Table 5. Interactive Calculator - Case Study Example

Input Parameters:

- Tools: 4
- API Connections per Tool: 25
- Credential Types: 3 (API keys, OAuth, IAM)
- Isolation Factor: 1 (same account)
- Network Exposure: 3/5
- Team Size: 4 people

Calculated Results:

ICD = (4 × 25 × 3) / 1 = 300 (CRITICAL) (Equation 3)

CVG = (6 - 4) / 6 × 100% = 33% (Equation 4)

ARI = 3 × 2 × 2 = 12 (HIGH RISK) (Equation 5)

OOD = 5 / 4 = 1.25 days per update (Equation 6)

Received: August 05 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Recommendations Generated:

- Critical: ICD score 300 - immediate architectural review required
- Warning: 33% visibility gap - implement unified logging
- Warning: High architectural risk - migrate to multi-account

Multi-Account Scenario:

ICD = (4 × 25 × 3) / 4 = 75 (MEDIUM) (Equation 7)

ARI = 3 × 2 × 1 = 6 (MEDIUM) (Equation 8)

Projected savings: $53,846 annually

The calculator validates our framework by allowing real-time assessment of different architectural scenarios. The 4x reduction in ICD score when moving from same-account to multi-account deployment demonstrates the quantifiable benefit of proper isolation.

5.2 Vulnerability Exposure Analysis

We calculated the organization's total vulnerability exposure based on industry benchmarks showing an average of 15 CVE vulnerabilities per software category (Gamblin, 2025). With four distinct data governance tools deployed, the mathematical exposure reaches 60 potential vulnerabilities before considering integration complexities. The extensive API integration layer introduces additional risk vectors, with over one hundred integration points creating an estimated five high-risk attack pathways based on typical exploitation rates of five percent. Perhaps most critically, the same-account deployment architecture eliminates natural isolation boundaries between tools, meaning a compromise of any single component could cascade across the entire data governance infrastructure.

The architectural complexity manifests most clearly in patch management timelines. A single-tool environment typically enables security patch deployment within 48 hours, allowing rapid response to emerging threats. However, the multi-tool same-account configuration extends this timeline dramatically to seven to ten days due to coordination overhead, integration testing requirements, and the risk of cascading failures. Organizations adopting proper multi-account isolation can reduce this timeline to three to five days through parallel testing capabilities, demonstrating how architectural choices directly impact security response capabilities.

The 72-hour integration failure documented in our case study created an acute vulnerability window during which 60 potential vulnerabilities remained unpatched while five high-risk API attack vectors remained actively exploitable. This incident transforms abstract vulnerability counts into concrete security exposure, illustrating how architectural debt translates to measurable risk.

5.3 Architecture Comparison and ROI Analysis

Table 6 demonstrates the security debt reduction achievable through architectural improvements.

Received: August 05 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Architecture	ICD	CVG	ARI	Annual Cost	ROI Year 3
Current State (Same Account)	300	33%	12	$143,846	Baseline
Hybrid Isolation (Critical tools separated)	150	25%	8	$110,000	+$23,846/year
Full Multi-Account (AWS best practice)	75	15%	6	$90,000	+$53,846/year

Table 6. Architectural Approaches - Security Debt Comparison

Multi-Account Migration Costs:

- One-time setup: $50,000
- Migration effort: $75,000
- **Total investment: $125,000**
- **Payback period: 2.3 years**
- **5-year ROI: $144,230**

Calculation: Annual savings ($53,846) × 5 years - Initial investment ($125,000) = $144,230

The analysis proves that despite significant upfront investment, multi-account architecture delivers substantial returns through reduced operational overhead, faster patch cycles, and elimination of integration failure incidents.

6. Discussion

6.1 Key Findings

The empirical analysis reveals four fundamental insights about security debt in data governance architectures. First, security debt grows exponentially rather than linearly as organizations add tools to their environment. Our ICD calculation framework demonstrates this multiplicative effect clearly: the transition from one to two tools creates a threefold increase (from 25 to 75), followed by successive doublings as each additional tool compounds the complexity. This exponential pattern contradicts the assumption that adding one more tool represents marginal additional complexity.

Second, deploying multiple tools within a single AWS account amplifies security debt by factors of two to four compared to properly isolated multi-account architectures. This amplification effect violates AWS multi-account best practices for resource isolation and blast radius limitation (Amazon Web Services, 2024a). The isolation factor in our ICD formula

Received: August 05 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

quantifies this architectural risk precisely: maintaining tools in a single account yields an ICD score of 300. In contrast, proper account separation reduces this to 75, a fourfold improvement achieved purely through architectural reorganization without changing the underlying tools.

Third, integration brittleness creates critical vulnerability windows extending beyond typical patch cycles. The catalog failure case demonstrates how tight coupling transforms routine maintenance into security crises. The 72-hour exposure window allowed known API vulnerabilities to remain exploitable, creating potential pathways for account takeover attacks (Approov, 2024). Our Architectural Risk Index captures this vulnerability through quantifying privilege escalation paths and network exposure, which quadrupled from 3 to 12 as tools became increasingly interdependent.

Fourth, and perhaps most paradoxically, organizations can improve compliance visibility while degrading overall security posture. The compliance visibility gap improved by 50 percent as measured by our CVG metric, decreasing from 67 percent to 33 percent. However, this improvement came at the cost of a twelvefold increase in integration complexity, a fourfold increase in architectural risk, and a fourfold increase in operational overhead. Organizations achieve regulatory compliance checkboxes while becoming more vulnerable—the "compliance-security paradox" or the dangerous illusion of security.

6.2 Theoretical Implications

This research extends the existing technical debt taxonomy by introducing infrastructure-level security debt specific to data governance (Li et al., 2015). While previous studies focused on code-level debt, our framework addresses architectural complexity in cloud environments. We bridge data governance literature focused on policies and procedures (Abraham et al., 2019) with practical security metrics that CISOs can use for risk assessment and budget justification.

6.3 Practical Implications

The framework delivers three critical capabilities for different stakeholder groups within financial institutions. The quantitative risk assessment transforms subjective architectural discussions into objective, measurable evaluations for Chief Information Security Officers and security leadership teams. Rather than debating whether tool sprawl "feels" risky, teams can calculate precise security debt scores and track their evolution over time. The framework enables architecture decision support by allowing organizations to compare security debt across deployment options before committing resources, preventing costly mistakes that require extensive remediation later. Perhaps most valuably for resource-constrained security teams, the framework provides compelling ROI justification, demonstrating that annual savings of $53,846 justify an initial investment of $125,000 in multi-account migration—arguments that resonate with budget-conscious executives.

Cloud architects gain quantitative validation of architectural principles they may have long advocated. Our isolation factor calculation proves empirically that AWS multi-account architecture transcends best practice status to become a security necessity. The fourfold reduction in security debt achieved through proper account isolation provides concrete

Received: August 05 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

evidence supporting architectural investments that might otherwise face scrutiny as "gold-plating" or unnecessary complexity.

Compliance teams discover through the CVG metric that meeting individual tool requirements does not guarantee comprehensive audit coverage. The framework reveals that fragmented compliance across multiple tools creates 33 percent visibility gaps even when each tool individually satisfies its designated requirements. This insight drives more strategic compliance planning that considers what tools can do and how their collective architecture enables or inhibits comprehensive regulatory visibility.

6.4 Limitations and Future Research

While robust within their context, this study's findings warrant careful consideration of scope limitations. The empirical foundation rests upon a detailed analysis of a single financial institution, which, while thoroughly documented, may not fully capture the diversity of organizational sizes, regulatory environments, and technical architectures encountered across the broader financial services sector. Organizations operating at different scales or facing different regulatory requirements may experience varying degrees of security debt accumulation, suggesting the need for multi-organization validation studies.

The framework's current implementation focuses on AWS cloud infrastructure, reflecting the platform's market dominance and the availability of detailed architectural guidance. However, organizations leveraging Azure, Google Cloud Platform, or multi-cloud strategies may encounter different security debt patterns due to platform-specific architectural paradigms and tooling ecosystems. Cross-platform validation would strengthen the framework's universal applicability and potentially reveal platform-specific debt mitigation strategies.

While grounded in established technical debt theory and validated through case study analysis, the metrics themselves require broader empirical validation across diverse industry sectors beyond financial services. Healthcare, government, and other highly regulated sectors may exhibit different security debt characteristics due to varying compliance requirements, risk tolerances, and operational constraints. While the framework demonstrates strong theoretical foundations and practical utility within financial services, generalizing these findings to other industries requires additional validation studies that account for sector-specific regulatory requirements, risk profiles, and operational constraints.

Future research should pursue several promising directions to extend and validate this framework. Automated security debt detection tools that continuously monitor ICD scores could provide real-time visibility into architectural degradation, enabling proactive intervention before critical thresholds are breached. Predictive models leveraging machine learning could forecast security debt accumulation based on tool addition patterns, allowing organizations to anticipate and prevent dangerous architectural states. Cross-platform framework validation spanning Azure and GCP would establish universal metrics that are applicable regardless of cloud provider choice. Finally, longitudinal studies tracking security debt impact on actual breach probability would validate the framework's predictive power, establishing definitive links between calculated security debt scores and real-world security incidents.

Received: August 05 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

6.5 Future Roadmap

Looking ahead, we envision three key extensions to this work. First, developing automated monitoring capabilities that continuously assess security debt in production environments would enable proactive risk management. Second, expanding the framework to multi-cloud environments would increase its applicability across diverse organizational contexts. Third, conducting longitudinal studies across multiple industries would validate the framework's predictive power and refine threshold values for different organizational contexts.

7. Recommendations

7.1 Immediate Actions (0-30 Days)

Organizations should begin their security debt reduction journey with a comprehensive assessment and documentation. The interactive calculator available through the open-source repository enables immediate security debt scoring without requiring complex analysis. Teams should access the tool, input their current configuration parameters, including tool counts, API integrations, and credential management approaches, and review the generated ICD, CVG, ARI, and OOD scores alongside specific recommendations. This initial assessment establishes baseline metrics essential for tracking improvement over time.

Critical threshold identification naturally follows the assessment results. ICD scores exceeding 300 demand immediate architectural review and executive escalation, signaling that integration complexity has reached dangerous levels requiring urgent intervention. CVG percentages above 25 indicate significant visibility gaps necessitating unified logging implementation to close compliance blind spots. ARI values above 10 suggest high architectural risk, warranting multi-account migration planning to establish proper isolation boundaries. OOD measurements exceeding 1.5 days per update reveal operational inefficiencies where automation investments could yield substantial returns.

Comprehensive architectural documentation provides the foundation for all subsequent improvements. Teams must create detailed inventories cataloging all data governance tools with version information, mapping every API integration point and authentication mechanism, documenting AWS account boundaries and security group configurations, and recording all credential management processes. This documentation transforms abstract architectural concerns into concrete, actionable information that guides remediation efforts.

7.2 Short-term Actions (30-90 Days)

Multi-account strategy implementation represents the most impactful architectural improvement organizations can pursue. Following AWS Organizations best practices (Amazon Web Services, 2024b), institutions should establish separate accounts for each major tool category, creating natural isolation boundaries that prevent cross-tool compromise. A centralized logging account consolidates audit data while maintaining separation from operational systems. Dedicated security tooling accounts for house monitoring and protection capabilities without exposing them to potential compromises in workload environments. This workload isolation architecture fundamentally transforms the security posture by eliminating the same-account amplification effect documented in our analysis.

Received: August 05 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Testing infrastructure establishment prevents the integration failures that create critical vulnerability windows. Organizations should deploy staging environments that precisely mirror production configurations, enabling validation of changes before they impact critical systems. Integration testing automation removes manual error sources while accelerating validation cycles. Version compatibility validation procedures ensure tool updates maintain backward compatibility or provide clear migration paths. Robust rollback procedures allow rapid recovery from failed deployments without extended exposure windows like the 72-hour incident documented in our case study.

Unified logging deployment addresses the compliance visibility gaps revealed by CVG analysis. Centralized SIEM integration aggregates security event data from all governance tools into a single analytical platform. Cross-tool correlation rules detect attack patterns that span multiple systems, identifying threats invisible to individual tool monitoring. Compliance dashboard consolidation provides auditors with a comprehensive view of regulatory controls without manual data gathering across disparate systems. Real-time alerting capabilities enable rapid response to security events before they escalate into major incidents.

7.3 Long-term Actions (90+ Days)

Strategic security debt reduction requires systematic roadmap execution prioritized by calculated risk levels. Organizations should address critical tools exhibiting ICD scores exceeding 300 in the initial phase, as these represent imminent architectural failures requiring urgent remediation. The second phase targets high-risk integrations where ARI values surpass 10, indicating dangerous privilege escalation paths or network exposure requiring isolation improvements. The final phase optimizes operational efficiency by addressing OOD measurements above 1.5 days, implementing automation to reduce the manual overhead that delays security responses.

API gateway architecture implementation provides a comprehensive solution to credential sprawl challenges. Rather than managing three distinct authentication methods across multiple tools, organizations can establish centralized authentication through a unified gateway. This architectural pattern enables consistent rate limiting and throttling policies that prevent API abuse, implements sophisticated version management to eliminate compatibility failures like those documented in our case study, and enforces uniform security policies across all integrations. The gateway approach transforms the complex mesh of point-to-point integrations into a managed, monitorable architecture.

Compliance reporting automation eliminates the manual integration overhead that drives operational debt. Scheduled compliance exports automatically generate audit evidence without requiring manual data gathering across multiple systems. Automated evidence collection continuously validates that controls remain effective rather than discovering gaps during annual audits. Policy-as-code implementation enables version control and automated deployment of compliance requirements, ensuring consistency across environments. Continuous control monitoring provides real-time validation of regulatory compliance rather than periodic assessments that may miss transient violations.

7.4 Success Metrics

Received: August 05 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Organizations should establish quarterly tracking mechanisms to monitor security debt reduction progress and validate improvement initiatives. Integration complexity targets should aim for 25 percent ICD reduction each quarter through systematic architectural improvements and tool consolidation where appropriate. Compliance visibility improvements should progress toward achieving CVG measurements below 15 percent by the fourth quarter, indicating near-complete audit coverage across the governance landscape. Patch timeline objectives should reduce deployment cycles to under three days by the second quarter, demonstrating that architectural improvements have eliminated the coordination overhead that previously delayed security responses. Integration failure tracking should achieve zero-incident status, proving that proper isolation and testing procedures have eliminated the cascading failures that create critical vulnerability windows. These metrics transform abstract security debt reduction into measurable, achievable targets demonstrating tangible progress toward architectural excellence.

8. Conclusion

This research demonstrates that data governance tool proliferation creates measurable security debt exceeding 12x baseline complexity. Our four-metric framework—featuring Integration Complexity Debt (ICD), Compliance Visibility Gap (CVG), Architectural Risk Index (ARI), and Operational Overhead Debt (OOD)—provides quantitative tools for tracking and prioritizing remediation efforts.

The case study illustrates real-world consequences: a routine security patch was delayed 5 days due to integration brittleness, creating exploitable vulnerability windows. The 72-hour incident cost $27,000 and exposed 60 potential vulnerabilities. Our analysis proves that while organizations achieve compliance visibility improvements (67% → 33% gap), overall security posture degrades significantly across integration complexity (12x increase), architectural risk (4x increase), and operational overhead (4x increase).

The interactive calculator enables immediate security debt assessment, demonstrating that AWS multi-account architecture reduces debt by 75% (ICD: 300 → 75) and delivers $53,846 annual savings with a 2.3-year payback period. Organizations must balance compliance requirements with architectural security, recognizing that rapid tool deployment creates technical debt requiring strategic management.

As GRC managers continue struggling with complex IS landscapes and regulatory environments (Sillaber et al., 2019), this framework provides essential metrics for navigating the compliance-security paradox in modern financial services. The average organization's 613 APIs represent a significant attack surface (Imperva, 2024) that requires careful architectural planning rather than reactive tool addition.

Key Takeaway: Security debt is not inevitable—it is quantifiable, preventable, and reversible through proper cloud architecture. Organizations that proactively measure and manage security debt will achieve compliance and security, rather than the dangerous illusion of one at the expense of the other.

References

Received: August 05 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

1. Abraham, R., vom Brocke, J., & Schneider, J. (2019). Data governance: A conceptual framework, structured review, and research agenda. *International Journal of Information Management*, *49*, 424–438. https://doi.org/10.1016/j.ijinfomgt.2019.07.008

2. Alves, N. S., Mendes, T. S., de Mendonça, M. G., Spínola, R. O., Shull, F., & Seaman, C. (2016). Identification and management of technical debt: A systematic mapping study. *Information and Software Technology*, *70*, 100-121. https://doi.org/10.1016/j.infsof.2015.10.008

3. Amazon Web Services. (2024a). *Benefits of Using Multiple AWS Accounts*. https://docs.aws.amazon.com/whitepapers/latest/organizing-your-aws-environment/benefits-of-using-multiple-aws-accounts.html

4. Amazon Web Services. (2024b). *Best Practices for a Multi-Account Environment*. https://docs.aws.amazon.com/organizations/latest/userguide/orgs_best-practices.html

5. Amazon Web Services. (2024c). *AWS Multi-Account Strategy for Your AWS Control Tower Landing Zone*. https://docs.aws.amazon.com/controltower/latest/userguide/aws-multi-account-landing-zone.html

6. Approov. (2024). *Major API Security Breaches of 2024: Causes and Prevention Strategies*. https://approov.io/blog/how-poor-api-security-led-to-major-breaches-in-2024

7. AuditBoard. (2024). *What Is SOX Compliance? 2025 Complete Guide*. https://auditboard.com/blog/sox-compliance

8. Cimcor. (2024). *What to Expect During a SOX Compliance Audit*. https://www.cimcor.com/blog/what-to-expect-during-a-sox-compliance-audit

9. CISA. (2025). *Known Exploited Vulnerabilities Catalog*. Cybersecurity and Infrastructure Security Agency. https://www.cisa.gov/known-exploited-vulnerabilities-catalog

10. Gamblin, J. (2025, January 5). *2024 CVE Data Review*. https://jerrygamblin.com/2025/01/05/2024-cve-data-review/

11. Imperva. (2024). *The State of API Security in 2024*. https://www.imperva.com/blog/state-of-api-security-in-2024/

12. Khatri, V., & Brown, C. V. (2010). Designing data governance. *Communications of the ACM*, *53*(1), 148-152. https://doi.org/10.1145/1629175.1629210

13. Li, Z., Avgeriou, P., & Liang, P. (2015). A systematic mapping study on technical debt and its management. *Journal of Systems and Software*, *101*, 193-220. https://doi.org/10.1016/j.jss.2014.12.027

14. Nord, R., & Schwartz, E. (2016, August 8). *Early Software Vulnerability Detection with Technical Debt*. Carnegie Mellon University Software Engineering Institute. https://insights.sei.cmu.edu/blog/early-software-vulnerability-detection-with-technical-debt/

Received: August 05 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

15. Observo.ai. (2024). *Observability 101: Log Retention Requirements for Regulatory Compliance*. https://www.observo.ai/post/log-retention-requirements-for-regulatory-compliance

16. Pentest-Tools. (2024). *Benchmark of Top Network Vulnerability Scanners in 2024*. https://pentest-tools.com/benchmarks/network-vulnerability-scanners

17. Sillaber, C., Mussmann, A., & Breu, R. (2019). Experience: Data and information quality challenges in governance, risk, and compliance management. *Journal of Data and Information Quality*, *11*(2), Article 6. https://doi.org/10.1145/3297721

Author Note

Interactive Security Debt Calculator available at: Security Debt Calculator - Data Governance Framework

Received: August 05 2025