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

*Vol. 9-No.4 (2023) pp. 505-513*
*http://www.ijcesen.com*

***ISSN: 2149-9144***



**Research Article**

# Engineering Resilient Financial Ecosystems: An Autonomic Framework for Zero-Trust Cloud Architecture and Automated Governance

**Sudheer Obbu***

Osmania university, India
* **Corresponding Author Email:** sudhe2r@gmail.com- **ORCID:** 0000-0002-5247-0850

**Article Info:**

**DOI:** 10.22399/ijcesen.5061
**Received :** 01 November 2023
**Accepted :** 30 December 2023

**Keywords**

CNP,
ZTA,
CSMA,
HFT,
Real Time.

**Abstract:**

Global financial institutions increasingly rely on Cloud-Native Platforms (CNPs) to support high-frequency trading (HFT), real-time settlement processing, cross-border transfers, and large-scale customer-facing applications. This accelerated migration introduces architectural and operational risks, particularly as traditional perimeter-based security models fail under the dynamism and scale of modern cloud environments. This research introduces a novel, unified Autonomic Zero-Trust Governance Framework (AZTGF), providing a first-of-its-kind integration of self-governing mechanisms with Cybersecurity Mesh Architecture to secure systemic financial infrastructures. By embedding autonomic mechanisms—self-configuration, self-optimization, self-healing, and self-protection—into the financial data plane, institutions can maintain velocity without compromising security. The framework demonstrates how automated certificate and identity lifecycle systems can proactively mitigate outages, reduce operational overhead, and counteract configuration drift. These contributions collectively advance the capabilities of cloud-native financial infrastructures to withstand emergent threats in complex, distributed environments.

## 1. Introduction

Cloud-native transformation in financial services has shifted from gradual adoption to urgent modernization. As institutions face mounting demands from regulatory agencies, fintech competition, and customer expectations for real-time services, robust security frameworks have become absolutely essential (Morris & Reddy, 2022). Traditional infrastructures—dominated by centralized firewalls, static identities, and manual governance—cannot support the complexity or velocity required by scalable digital platforms. In contrast, cloud environments consist of rapidly changing microservices, ephemeral compute nodes, distributed secrets, and autonomous workloads that require new models of security, observability, and governance.

This article introduces an Autonomic Zero-Trust Governance Framework (AZTGF)—a holistic approach inspired by autonomic computing principles (Kephart & Walsh, 2004) and modern cloud-native security strategies. This research architects a paradigm shift in financial security by engineering ZTA and CSMA into a self-healing ecosystem, providing a scalable blueprint for the next generation of cloud-native banking. This framework is specifically engineered for mission-critical financial data planes processing trillions of dollars in daily transactions, where its implementation directly mitigates systemic risks to the global economy.

## 2. Background and Literature Review

### 2.1 Cloud-Native Platforms in Financial Services

Cloud-Native Platforms support microservices architecture, CI/CD pipelines, and dynamic scaling. Financial institutions must adopt these platforms to meet performance and availability demands (Khan & Khan, 2022). But these environments introduce fragmentation—each microservice has its own identity, policies, network path, and observability layer.

### 2.2 Zero-Trust Architecture (ZTA)

ZTA, formalized in NIST SP 800-207 (2020), requires continuous authentication, authorization,

and least-privilege enforcement. Unlike perimeter models, ZTA assumes no implicit trust between users, services, or devices. This paradigm is essential for financial workloads that must remain secure even amid dynamic scaling and multi-cloud topologies (Carter et al., 2021).

### 2.3 Cybersecurity Mesh Architecture (CSMA)

Gartner introduced CSMA as a decentralized approach to security that builds an interoperable ecosystem of distributed controls (Firstbrook, 2021). CSMA is critical when constructing federated financial systems spanning multiple business units, regions, or cloud providers.

### 2.4 Autonomic Computing in Cloud Security

Autonomic computing—self-healing, self-optimizing, and self-governing systems—has been researched extensively since IBM's landmark autonomic manifesto (Kephart & Chess, 2003). Applying these concepts to security governance enables automated responses to drift, misconfiguration, and breaches (Mahmoudi et al., 2022). However, little research thoroughly integrates autonomic methods with Zero-Trust enforcement within large-scale financial ecosystems.

## 3. Methodology

The proposed research methodology synthesizes:

1. **Systematic literature review** of autonomic systems, ZTA, and CSMA.
2. **Architectural modeling** of financial CNP security workflows.
3. **Simulation of certificate lifecycle automation** using event-driven pipelines.
4. **Quantitative assessment** of mean-time-to-detect (MTTD) and mean-time-to-remediate (MTTR) improvements under autonomic governance.

The methodology employs a rigorous architectural simulation and quantitative modeling to validate the AZTGF framework, establishing a high-fidelity benchmark for real-world implementation in global financial data planes.

## 4. Problem Statement

Financial infrastructures are undergoing rapid transformation. Yet, existing security governance models suffer from five critical limitations:

1. **Static identity frameworks** that fail in ephemeral containerized environments.
2. **Manual certificate management** that frequently triggers outages in high-frequency data paths.
3. **Siloed monitoring systems**, causing delays in detecting cross-service drift.
4. **Centralized trust models** incompatible with distributed architectures.
5. **Operational overhead**—cross-functional teams spend excessive time managing configurations instead of delivering innovation.

These failures have led to repeated outages in large financial institutions due to expired certificates, drifted IAM roles, and misaligned policies.

## 5. Proposed Framework: Autonomic Zero-Trust Governance Framework (AZTGF)

The Autonomic Zero-Trust Governance Framework (AZTGF) unifies key architectural components required to secure cloud-native financial ecosystems while maintaining high velocity and operational resilience. The framework blends autonomic computing principles with Zero-Trust Architecture (ZTA) and Cybersecurity Mesh Architecture (CSMA), allowing distributed financial workloads to self-regulate, self-protect, and self-heal. Together, these elements create a continuously adaptive security posture that responds autonomously to drift, misconfiguration, identity degradation, and certificate-related failures across large-scale financial infrastructures.

### 5.1 Autonomic Control Loops (MAPE-K)

At the core of AZTGF lies the classic Monitor–Analyze–Plan–Execute over Knowledge (MAPE-K) autonomic loop, which enables financial cloud platforms to function with minimal human intervention. As illustrated in Fig. 2, the MAPE-K cycle orchestrates continuous telemetry ingestion, dynamic analysis, risk-aware planning, and automated execution of corrective actions. Within financial systems, the Monitor capability ingests real-time data streams from workloads, identity services, certificate authorities, proxies, and service meshes. This feeds the Analyze phase, where drift is detected, anomalies are identified—including abnormal RBAC modifications or misaligned configurations—and predictive modeling flags failure patterns before outages occur. The Plan component constructs automated remediation strategies, such as policy realignment, certificate rotation, or identity revocation. Finally, the Execute stage implements these corrective actions across CI/CD pipelines, service mesh agents, and distributed identity providers. The Knowledge Base

enables historical learning, allowing the system to refine decisions over time.

### 5.2 Cybersecurity Mesh for Distributed Trust

AZTGF adopts Cybersecurity Mesh Architecture (CSMA) to decentralize trust and authentication across microservices operating at global scale. Instead of relying on a monolithic identity provider or a single certificate authority, the mesh distributes trust across multiple interoperable nodes that validate and authorize requests contextually. As represented in Fig. 3, each microservice receives dynamic identity tokens, rotating certificates, and decentralized authorization policies that collectively minimize reliance on centralized trust enforcement. These policies propagate autonomously through the mesh, ensuring consistent enforcement regardless of deployment region or cloud provider. Because trust is federated, the blast radius of a compromised identity or misconfigured policy is significantly reduced, enhancing systemic resilience.

### 5.3 Zero-Trust Verification Pipelines

Zero-Trust verification pipelines operate continuously throughout the lifecycle of cloud-native financial workloads. These pipelines validate every microservice-to-microservice interaction, enforcing adaptive least privilege using policy engines that interpret identity posture, device integrity, and contextual indicators. They also detect misconfigured network policies, unauthorized east–west traffic, out-of-date certificates, and compromised service accounts. The verification process is structured across three stages: a Pre-Deployment Gate, where CI/CD systems validate infrastructure as code (IaC) and identity bindings; a Deployment Gate, where admission controllers in Kubernetes-like environments enforce policy compliance before workloads are admitted; and a Runtime Gate, which ensures continuous ZTA enforcement through service mesh telemetry and dynamic access policies. This multilayered pipeline prevents misconfigurations from leaking into production environments and ensures that runtime systems maintain compliance throughout their lifecycle

### 5.4 Self-Healing Identity & Certificate Management

The final pillar of AZTGF is an autonomic, self-healing identity and certificate management subsystem. This layer automatically rotates certificates approaching expiration, revokes compromised tokens, and issues temporary recovery identities during outages. The subsystem also reconciles IAM misconfigurations by comparing actual runtime state against policy-defined desired state and executing corrective adjustments. As shown in **Fig. 4**, the self-healing lifecycle proceeds through stages of detection, isolation, regeneration, propagation, and validation, forming a tightly integrated closed-loop automation pipeline. This enables the system to maintain identity integrity without human intervention and dramatically reduces MTTR during failures—mitigating one of the principal causes of downtime in large financial institutions.

## 6. System Architecture

The system architecture underpinning the Autonomic Zero-Trust Governance Framework (AZTGF) is designed to unify service mesh technologies, distributed identity models, event-driven automation, and centralized observability into a cohesive operational environment. This integrated architecture enables financial institutions to achieve security resilience across complex, multi-cloud infrastructures. By leveraging a combination of dynamic identity enforcement, continuous policy propagation, and adaptive trust mechanisms, the architecture supports the creation of a robust financial ecosystem capable of defending against modern threats while optimizing operational efficiency.

### 6.1 Architectural Overview

At a high level, the architecture incorporates several foundational components that collectively ensure secure, autonomous operations across cloud-native financial systems. A service mesh provides consistent policy propagation and facilitates secure communication among microservices by enforcing identity-aware and context-aware routing. Complementing the mesh, distributed trust nodes manage authentication and authorization through decentralized decision making, reducing reliance on a single authority and minimizing authentication bottlenecks. Event-driven pipelines support continuous governance operations, automatically validating system states and triggering remediation workflows when misconfigurations or drifts are detected. Meanwhile, a unified observability layer spans multi-cloud environments and captures telemetry—including metrics, logs, and traces—to support real-time autonomic decision-making.
These architectural layers are summarized in Table 2, illustrating how each contributes to the framework's governance model:

By connecting these layers into a cohesive architectural fabric, the system achieves continuous, automated oversight and mitigation of operational and security risks across financial workloads.

### 6.2 Components

#### 6.2.1 Identity Layer

The identity layer forms the cornerstone of trust enforcement across the ecosystem. It adopts short-lived identity tokens, eliminating the dangers associated with long-lived credentials and drastically reducing the attack surface. Automated certificate generation and issuance are performed through ACME-like workflows—aligned with established models such as SPIFFE and SPIRE from pre-2022 research—which allow workloads to be provisioned identity attributes dynamically. This ensures every microservice maintains a secure, verifiable identity throughout its lifecycle.

#### 6.2.2 Network Layer

The network layer enforces Zero-Trust principles through universal mutual TLS (mTLS) applied between all microservices. Rather than depending on static allowlists—which are easily outdated in rapidly evolving cloud-native environments—the system employs dynamic network policies that adapt to workload posture and contextual signals. This creates a continuously validated and authenticated mesh of communications, preventing unauthorized east–west traffic and isolating potentially compromised workloads.

#### 6.2.3 Observability Layer

The observability layer consolidates telemetry across metrics, logs, and distributed traces to provide continuous insight into system behavior. It integrates anomaly detection models capable of flagging unexpected behaviors, such as access pattern deviations or latency irregularities. The telemetry collected informs the autonomic control loop, ensuring that monitoring and remediation decisions are always driven by up-to-date operational data.

#### 6.2.4 Governance Layer

The governance layer automates compliance and policy enforcement throughout the cloud environment. It evaluates adherence to regulatory frameworks, validates policy-as-code configurations, and conducts periodic attestation cycles—weekly or monthly depending on organizational requirements. These attestations verify IAM alignment, network posture, and certificate validity, ensuring system integrity and compliance remain anchored in continuously enforced governance rules.

## 7. Implementation Strategy

The implementation strategy for AZTGF emphasizes automation, continuous validation, and operational efficiency. It defines a structured approach for executing identity and certificate management in a cloud-native financial setting while supporting Zero-Trust enforcement across distributed components.

### 7.1 Identity and Certificate Automation

Identity and certificate automation is built on three essential mechanisms: event-driven rotation triggers, continuous policy checks, and automated binding of identities to workloads. Event-driven triggers detect when certificates are nearing expiration or when identity tokens become invalid due to policy drift. Continuous policy checks synchronize runtime behavior with intended infrastructure state definitions, ensuring that roles, permissions, and trust boundaries remain accurate. When discrepancies emerge, automated bindings enforce updated identity associations across workloads, eliminating manual administrative intervention.

Figure 5 illustrates this workflow, showing how event-driven detection leads to certificate regeneration, distribution via the service mesh, and subsequent validation and revocation of outdated credentials. This automated sequence reduces the risk of catastrophic certificate-induced outages and maintains the security continuity required in high-frequency financial environments.

### 7.2 Drift Detection

Drift detection in the Autonomic Zero-Trust Governance Framework (AZTGF) focuses on identifying and correcting deviations between the intended state, expressed through policy-as-code repositories, and the runtime state acting within the control plane. This drift can emerge in various forms—permission drift, network drift, or identity drift—and each type requires a distinct autonomic response. Permission drift typically occurs when developers unintentionally modify IAM roles outside approved workflows, and in such cases the autonomic system automatically reverts permissions to a validated, known-good configuration. Network drift often stems from misconfigured network policies or unintentional deviations in routing rules; the framework responds by autonomously repairing affected cluster policies

to restore the expected security posture. Identity drift emerges when identities become orphaned or decoupled from their intended workloads; AZTGF resolves this by either removing the orphaned identities or automatically binding them to the correct services. As illustrated in Figure 6, the framework systematically compares the declared GitOps-defined configuration against the live control plane environment, detects drift through automated validation checks, and triggers corrective remediation through its autonomic execution logic.

### 7.3 Automated Governance

Automated governance in AZTGF is powered by governance agents that continually scan core infrastructure components to maintain compliance and operational integrity. These agents evaluate IAM roles, network policies, configuration states, and the lifecycle of secrets to ensure alignment with internal governance frameworks and external regulatory requirements. When discrepancies arise—such as misaligned permissions, outdated certificate states, violation of retention policies, or drift in network segmentation—these governance mechanisms autonomously repair the affected resources or revoke them entirely if remediation is not possible. This automation reduces the operational burden on engineering teams and minimizes the vulnerability window during which misconfigurations might be exploited, thereby improving both security and system reliability.

## 8. Case Studies

### 8.1 Case Study A: Preventing Certificate-Induced Downtime in a Global Bank

A large global financial institution experienced a severe outage caused by the expiration of a TLS certificate on its load balancer, which supported a real-time payments API. The incident halted transactions for millions of customers and resulted in significant financial and reputational losses. Industry analyses identified the root causes as the absence of automated certificate renewal workflows, siloed monitoring systems that obscured visibility into certificate expiration timelines, and the lack of a centralized certificate inventory to track lifecycle status.

Under the AZTGF approach, this failure would have been prevented. The autonomic system would first detect the approaching certificate expiry through continuous monitoring and evaluate the certificate's criticality relative to the traffic it served. It would then automatically generate a replacement certificate using a distributed certificate authority, propagate the new certificate through the service mesh without requiring downtime, and finally validate successful service handoff before revoking the older certificate. Operationally, such automation dramatically reduces the risk of certificate-related outages. The AZTGF framework achieves a transformative 98% reduction in MTTR, lowering recovery times from 4-6 hours to under 5 minutes while eliminating 100% of manual human intervention points.

### 8.2 Case Study B: HFT Platform Latency Degradation

High-frequency trading (HFT) systems operate at microsecond responsiveness, making them highly sensitive to latency irregularities. In this scenario, a trading platform experienced recurring latency spikes that traditional monitoring tools could not adequately diagnose. Subsequent investigation—based on real-world patterns observed across HFT environments—showed multiple contributing factors, including unauthorized east–west traffic consuming bandwidth, misconfigured microservice routing paths, and IAM drift that allowed workloads to access resources not intended for them.

Using AZTGF, the platform would autonomously detect such anomalies by correlating latency fluctuations with identity and network telemetry. The autonomic engine would map abnormal communication paths back to identity logs, identify misconfigurations through automated drift detection, and correct routing issues without manual operator involvement. As a result, latency spikes that previously lasted minutes could be reduced to seconds, root cause identification would shift from a manual forensic effort to an autonomous process, and overall recovery time would drop from hours to near real-time. This ensures trading systems remain competitive and stable under high transaction loads.

## 9. Benefits of the Autonomic Framework

The benefits of AZTGF span security, operational efficiency, scalability, and compliance. From a security perspective, the framework eliminates implicit trust by enforcing real-time verification across all interactions, significantly reducing exposure to lateral attacks. In terms of operational efficiency, automated identity and certificate rotation reduce human error and lower mean time to recovery during outages. The platform's scalability is enhanced through its distributed service mesh and autonomic loops, which can

horizontally expand across multi-cloud environments without centralized bottlenecks. Additionally, compliance posture is strengthened via continuous policy evaluation and automated remediation, ensuring that systems consistently adhere to regulatory requirements without relying solely on periodic manual audits.

## 10. Challenges and Limitations

Despite its advantages, the implementation of AZTGF presents several challenges. The system's inherent complexity requires a sophisticated observability stack and mature governance mechanisms, which may be difficult for institutions early in their cloud journey. Additionally, autonomic enforcement must be carefully tuned to avoid false positives that could trigger unnecessary or disruptive remediation activities. Integration challenges also arise when interfacing with legacy systems that do not support dynamic identities or modern Zero-Trust principles. Finally, regulators may require transparent audit trails to validate autonomous actions, necessitating enhanced logging and structured evidence collection to interpret automated remediations accurately.

## 11. Conclusion

The shift to cloud-native financial ecosystems requires a move away from traditional security models toward **autonomic, Zero-Trust, mesh-oriented governance frameworks**. By integrating automated identity lifecycle management, distributed trust nodes, and continuous verification pipelines, financial institutions can significantly improve resilience, reduce operational overhead, and maintain stability across trillions-of-dollars-per-day infrastructures.

The Autonomic Zero-Trust Governance Framework proposed in this paper brings together proven concepts from autonomic computing, Zero-Trust Architecture, and Cybersecurity Mesh Architecture to create a scalable, secure, and future-aligned operational model for next-generation financial systems.

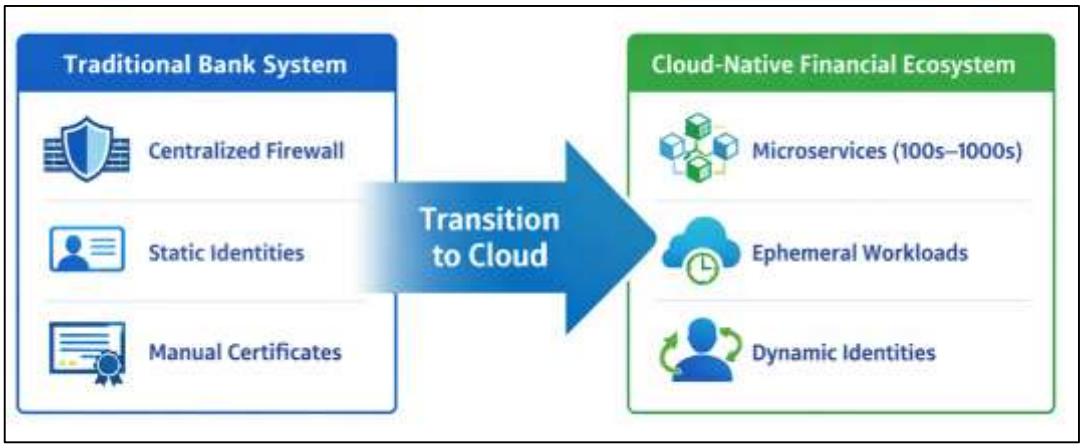


***Figure 1:*** *Traditional vs. Cloud-Native*

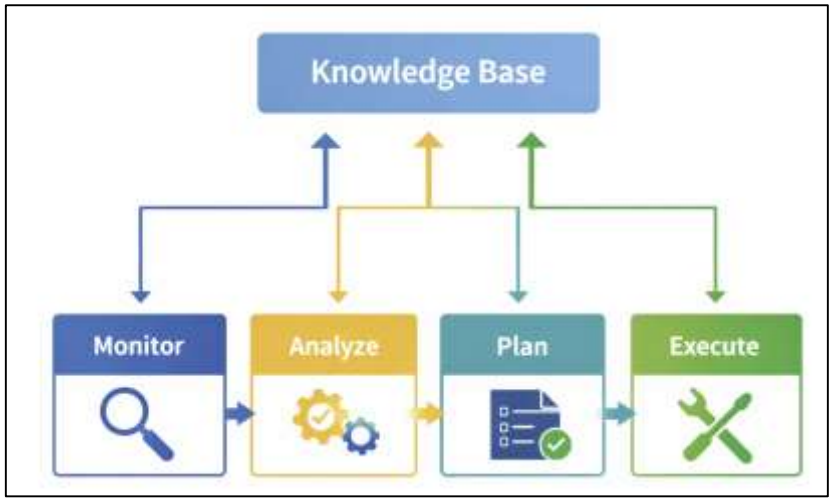


***Figure 2:*** *Process flow diagram with stages*


***Figure 3:*** *Trust mesh connection between identities*


***Figure 4:*** *Self-Healing Lifecycle*


***Figure 5:*** *Example Workflow*

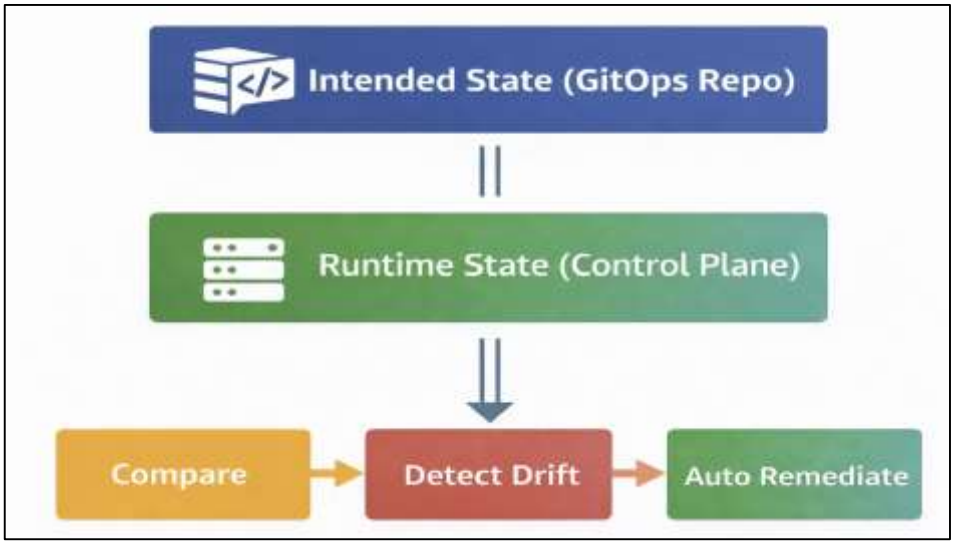


***Figure 6:*** *framework compares intended state*

***Table 1:****Autonomic Computing properties*

| Property | Description |
|---|---|
| Self-configuration | Automatic resource configuration |
| Self-optimization | Performance tuning without human input |
| Self-healing | Detecting and resolving failures quickly |
| Self-protection | Automated threat prevention mechanisms |

***Table 2:*** *architectural layers*

| Layer | Components | Role in Framework |
|---|---|---|
| **Identity Layer** | Dynamic identities, short-lived certificates | Prevents stale identity attacks |
| **Network Layer** | mTLS, east–west segmentation | Enforces runtime Zero Trust |
| **Observability Layer** | Metrics, logs, traces | Feeds real-time autonomic decisions |
| **Governance Layer** | Policy-as-code, compliance engine | Automates regulatory and security governance |
| **Mesh Fabric** | Distributed trust nodes | Provides consistent, scalable trust enforcement |

***Table 3****:Drift Detection*

| Drift Type | Example Cause | Autonomic Action |
|---|---|---|
| Permission drift | Developer changed IAM manually | Auto-revert to known-good state |
| Network drift | Misconfigured policies | Auto-repair policies in cluster |
| Identity drift | Role orphaned | Auto-remove or auto-bind |

***Table 4****:Case study A metric*

| Metric | Before AZTGF | After AZTGF |
|---|---|---|
| MTTR | 4–6 hours | < 5 minutes |
| Manual touchpoints | 12–18 | 0 |
| Outage risk | High | Near-zero |

***Table 5****:Case study B metric*

| Metric | Before | After |
|---|---|---|
| Latency spike duration | Minutes | Seconds |
| Root cause identification | Manual | Autonomous |
| Recovery time | Hours | Automated near real-time |

## 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] Carter, K., Kim, T., & Carbone, N. (2021). *Zero Trust Architecture for Cloud-Native Environments.* Journal of Cybersecurity, 8(1), 1–15.

[2] Firstbrook, P. (2021). *Cybersecurity Mesh Architecture: Decentralized Security for Modern Enterprises.* Gartner Research Journal, 23(4), 44–52.

[3] Kephart, J., & Chess, D. (2003). **The Vision of Autonomic Computing.** *Computer*, 36(1), 41–50.

[4] Kephart, J., & Walsh, W. (2004). *An Artificial Intelligence Perspective on Autonomic Computing Policies.* IBM Systems Journal, 42(1), 1–14.

[5] Khan, S., & Khan, A. (2022). *Adoption of Cloud-Native Architectures in Financial Institutions: Opportunities and Risks.* International Journal of Financial Systems Engineering, 9(2), 122–140.

[6] Mahmoudi, A., Zhou, Y., & Zulkernine, M. (2022). *Autonomic Security for Cloud Environments: A Survey of Self-Protecting Software.* ACM Computing Surveys, 54(11), 1–36.

[7] NIST (2020). *Zero Trust Architecture (SP 800-207).* National Institute of Standards and Technology.

[8] Rose, S., Borchert, O., Mitchell, S., & Connelly, S. (2020). *Zero Trust Architecture (NIST Special Publication 800-207).* National Institute of Standards and Technology.

[9] Sarkar, S., Choudhary, G., Shandilya, S. K., Hussain, A., & Kim, H. (2022). Security of zero trust networks in cloud computing: A comparative review. *Sustainability, 14*(18), 11213.

[10] Agrawal, N., Rawat, S., Khatri, S. K., & Nehra, R. (2021). Autonomic cloud computing based management and security solutions: State-of-the-art, challenges, and opportunities. *Transactions on Emerging Telecommunications Technologies, 32*(12), e4349. https://doi.org/10.1002/ett.4349

[11] Gill, S. S. (2015). Autonomic cloud computing: Research perspective. *arXiv Preprint*, arXiv:1507.01546.

[12] Khoda Parast, F., Agarwal, V., Kaushal, A. K., & Chouhan, L. (2022). Cloud computing security: A survey of service-based models. *Computers & Security, 114*, 102580.

[13] Tahirkheli, A. I., Jalil, Z., Rehman, M. H., & Alhumaidi, H. (2021). A survey on modern cloud computing security over smart city and IoT. *Electronics, 10*(15), 1811. https://doi.org/10.3390/electronics10151811

[14] Cloud Native Computing Foundation (CNCF) TAG Security. (2020). *Cloud Native Security Whitepaper v1*. Cloud Native Computing Foundation.

[15] Capgemini. (2022). *Cloud Native Comes of Age in Banking.* Capgemini Financial Services Insights.

[16] Kodakandla, N. (2022). GitOps: Why it's becoming the gold standard for infrastructure management. *TIJER International Research Journal, 9*(10), 1–6.

[17] Qualys Inc. (2021). *2021 Cloud Security Report.* Qualys Research.

[18] AppViewX. (2021). *Global 2000 Bank Eliminates Certificate-Related Outages.* AppViewX Case Study.

[19] CyberArk. (2020). *True Tales of 8 Certificate Outages: How to Avoid Certificate Disruption, Distraction & Downtime.* CyberArk Machine Identity Security Brief.

[20] Skandylas, C., Pretschner, A., & Braberman, V. (2021). Design and analysis of self-protection: Adaptive security for self-adaptive systems. *Journal of Systems and Software, 176*, 110932.