Beyond the Pipeline: Embedding Generative AI Risk Intelligence into Cloud-Native Financial Transaction Architectures

Ganesh Pambala
Independent Researcher
ganeshpambal@gmail.com

Abstract

The increasing frequency, sophistication, and financial impact of cyberattacks against financial institutions demand a proactive instead of a reactive approach. One current direction consists of the adoption of Cloud-Native Architecture principles supported by DevOps practices driven by Automation and AIOps tools supplied as-a-Service by cloud providers. Taking into consideration the property of security for the deployment of Payment Systems is paramount, it is proposed a Security-as-a-Service solution using Generative Artificial Intelligence techniques to induce Risk Models. This approach applied within the Design-and-Build phases of the DevOps Automation might link the application code with the risk/security posture of the Payment System and support effective decision-making regarding the acceptance/rejection of the software deployment or update.

Generative Artificial Intelligence techniques have been used for textual completion since the introduction of the Transformer architecture. It can be used to induce Risk Models since usually a Risk Model can be represented in a tabular structure summarized by its three fundamental components: Asset, Threat, and Vulnerability. Recent advances in confidential computing and secure enclaves enable the processing of sensitive information without exposing the input data.

Keywords: DevOps, financial services, payments, security, cloud, microservices, infrastructure, continuous delivery, automation, generative AI.

1. Introduction

Digital payment enables an instantaneous transfer mechanism for payments, which is much faster than traditional banking channels. In recent years, the volume of digital payment transactions has increased significantly and is expected to continue growing at high rates; therefore, it is essential that such payment systems are secure, reliable, and available. However, the demand for continuous delivery of new features and services introduces new challenges to the operational aspects of payment systems, leading to a need for a more frequent and automated way of releasing and deploying code that requires the collaboration of Development and Operations teams.

The objective of this research work is to demonstrate a cloud-native payment architecture that applies security-by-design concepts and automated CI/CD pipelines for the end-to-end secure development and deployment of a digital payment system. Cloud-native DevOps Security sites (DevSecOps) practices, i.e., the combination of a cloud-native operational stack with security, privacy, compliance, reliability, and resilience perspectives, enhance the security of the digital payment stack. AI-based Generative Adversarial networks (GANs) are integrated to automate the identification of new attack patterns and provide the detected anomalies as extra risk factors for the new payments to be processed. Shaping cloud-native financial payment systems around these key directions makes them more secure and reliable while providing higher-quality input for subsequent business processes.

ISSN: 3050-9696 E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 03 Issue: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 09 ACCEPTED: FEBRUARY 24 PUBLISHED: MARCH 18

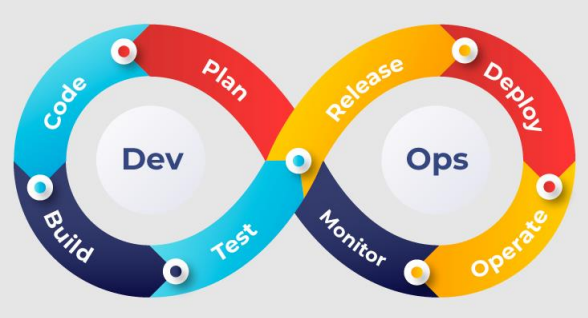


Fig 1: Generative AI in DevOps

1.1. Background and Significance

Recent reports reveal that criminal groups are increasingly targeting financial institutions through cyberattacks, money laundering, and illegal operations. Although payment processing services provided by companies such as PayPal, Adyen, and Stripe are becoming more popular, little attention is paid to the security of underlying payment systems. As payment service providers (PSPs) now have to process higher volumes of transactions and diverse transaction types across multiple channels such as web-based, mobile, QR code, NFC, and bank transfer, security, performance, and availability have become significant concerns. Cloud-native architecture, that is, the DevOps- and SRE-driven implementation of the supporting infrastructure, and integrated risk modeling and analysis based on Generative AI can complement enhancements to core payment processing subsystems, addressing these concerns.

Emerging concepts, products, and services exploiting the potential of Generative AI promise disruptive advances across multiple domains. Within risk management, using Generative AI for forward-looking risk analysis remains underexplored, particularly in generating and updating the underlying risk models. Large Language Models (LLMs), which are recent breakthroughs in Generative AI, can significantly improve the data management-related tasks of risk modeling and data preparation toward risk analysis.

Equation 1: SLA and Availability (ties to "SLA specifies % uptime" and Active-Active redundancy)

Step 1: Define variables

- Let T = total time window (e.g., 30 days)
- Let D = total downtime in that window

Step 2: Availability definition

$$A = \frac{T - D}{T} = 1 - \frac{D}{T}$$

Step 3: Downtime allowed for an SLA
If SLA requires $A \geq A^*$, then:

$$1 - \frac{D}{T} \geq A^* \Rightarrow \frac{D}{T} \leq 1 - A^* \Rightarrow D \leq (1 - A^*)T$$

Step 4: Active-Active (2 independent sites)
If each site has availability A, then each site's downtime probability is $(1 - A)$.
Service is down only if **both** sites are down:

$$A_{AA} = 1 - (1 - A)^2 = 2A - A^2$$

Step 5: N-replica redundancy

$$A_N = 1 - (1 - A)^N$$

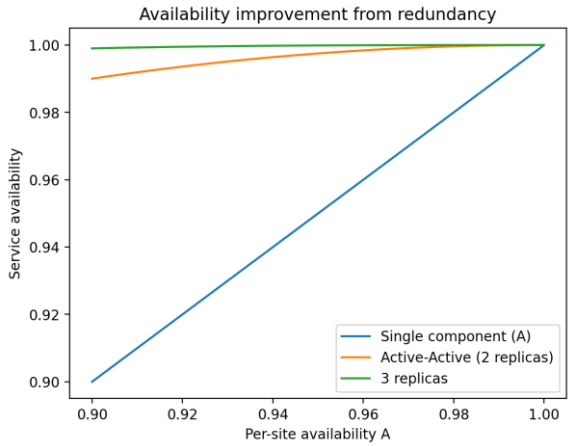
→ The PDF shows a **graph of A_N vs A** for single site, 2 replicas, and 3 replicas.

2. Foundational Principles and Requirements

Design principles and requirements form the broader foundation for the secure cloud-native financial payment system architecture. As financial payment systems typically deal with sensitive customer data, compromises of such systems would cause severe security, privacy, and compliance (SPC) violations. Therefore, SPC is the primary requirement. Apart from SPC, reliability and availability are very important. Such systems should provide service consistently and be available to end customers even during a network or data-center failure via cloud-native design.

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 03 Issue: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 09 ACCEPTED: FEBRUARY 24 PUBLISHED: MARCH 18

Information security comprises measures intended to safeguard a computer's digital information from modification, disruption, destruction, or disclosure. In simple language, it is the protection of information systems from compromise. For cloud-native financial payment systems, SPC requirements include securing sensitive customer data stored in or passing through the system; achieving privacy compliance in exposing data to generative AI tools and cloud-based analytics; securing the AI/ML and prediction models (which also contain sensitive data such as bank authorization keys) from theft and/or malicious prediction; defining and implementing strong data access and sharing policies to leak minimal or no data; and achieving legal compliance regarding financial data handling, as mandated by abroad-based regulations (for example, GDPR, PCI-DSS) by efficiently managing data location in distributed multi-cloud deployments.



2.1. Security, Privacy, and Compliance

Architectural and component-level security is paramount for any payment processing system handling financial transactions of end customers. Payment systems must operate in accordance with standards imposed by the Payment Card Industry Data Security Standard (PCI DSS). PCI DSS defines a set of minimum-security requirements that cover various components of the system and its surroundings.

The PCI requirements mandate a comprehensive definition, design, establishment, and continuous management of the Payment Card Account Data Environment (PA-DSS). This management process includes the determination of applicable level 1, 2, 3, and 4 functions to be implemented and operated by acquirers, merchants, and service providers. The specific requirements include protecting cardholder data during transmission and storage, creating and maintaining secure systems and applications, implementing strong access authentication, restricting access based on role, tracking and monitoring network resources, a policy and regular testing of security measures, and frequently supporting system monitoring access. Alongside these requirements, the PCI DSS requires various security practices such as risk analysis, development of a security policy, regular security awareness training for staff, and continuity of operations testing.

Data privacy is a core requirement and challenge for cloud-native payments. To be serious about creating a business-driven privacy-preserving architecture, payments must primarily operate around real customer data that is sensitive and private (real PCI personal sensitive information). In essence, all payment transactions can concern sensitive real data. PCI represents some of the sensitive data that can be used for data privacy-preserving risk analysis when the original data cannot be used or applied. Thus, whenever sensitive data are involved in the card transaction processes, generating privacy-preserving synthetic data will always be a key. Synthetic data for payments play an important role in many business cases, enabling payments to deeply understand customer experience and optimize their products and services. The effort to create an effective solution to properly generate synthetic payment data will nicely balance the business class with the privacy-preserving class.

Security measures must also fulfil the security assessment requirements. These must include proper definition of the risk detection and risk analysis procedure for the success of the risk model to be created and applied to reinforce the overall security of the business and enable quick and targeted incident responses and security controls (that can be considered as remediation) during incidents.

ISSN: 3050-9696 E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 03 Issue: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 09 ACCEPTED: FEBRUARY 24 PUBLISHED: MARCH 18

2.2. Reliability and Availability

Service availability should be maximized to minimize business disruption. Market stability may allow components to be brought offline for maintenance during off-peak hours and supported by Disaster Recovery capabilities for extensive failures. The Service Level Agreement (SLA) specifies the percentage of uptime that must be guaranteed. Service Reliability Engineering (SRE) can be adopted and complemented with chaos-engineering approaches to improve failure detection, testing, and simulation.

If designed without maintenance windows, the platform is designed with redundancy for reliability, availability, and scalability. An Active-Active Data Center (DC) failover is an example of achieving high availability; on each side, the capacity is doubled. Two redundant DCs, each covering the other side, along with a Data Loss Prevention (DLP) system to ensure an Information Security Management System (ISMS), thus generating a Highly Available DR (HA-DR) approach. Active-passive architecture allows for a greater divide between DCs and greater separation of risks. Heat maps can ensure investment and focus are allocated where losses can be reduced the most.

Equation 2: Risk Modeling and Risk Scoring (ties to "risk scoring system" + risk model)

Step 1: Consider discrete scenarios s (e.g., account takeover, data leak, DoS).
Let:

- $P(s)$ = probability of scenario s
- $L(s)$ = loss if scenario s occurs

Step 2: Expected loss risk:

$$\text{Risk} = \mathbb{E}[L] = \sum_{s} P(s)\, L(s)$$

For each tuple i (asset, threat, vulnerability):

- likelihood $p_i \in [0,1]$
- impact I_i (e.g., dollars)

$$\text{Risk}_i = p_i \cdot I_i, \quad \text{Total risk} = \sum_i \text{Risk}_i$$

→ The PDF includes a **table** with sample Asset–Threat–Vulnerability rows and a **bar chart** of expected loss by asset.

Let transaction features be $x = [x_1, \ldots, x_d]$.
Linear score:

$$z = w_0 + \sum_{j=1}^{d} w_j\, x_j$$

Convert to probability via sigmoid:

$$p = \sigma(z) = \frac{1}{1 + e^{-z}}$$

Decision rule (example):

$$\text{Approve if } p < \tau, \quad \text{else hold/review/deny}$$

→ The PDF includes the **logistic curve graph** p vs z.

3. Architectural Overview

Initial payment transaction processing stages, including ingestion, validation, authorization, encryption, and key management, are discussed. A secure stack leveraging function already available in public cloud providers helps reduce development time and costs. Integrated operations, security, and privacy risk analysis tools, based on generative AI, provide more resilient security controls. Sourcing specialist cloud functions from these providers allows payment transactions to be processed and supported by Pays-as-You-Use business models with tools from Cloud Service Providers.

Foundational principles related to cloud-native architecture are first examined, laying the basis for the design of cloud-native payment transactions, based on services offered by cloud providers. Payments from their inception through ingestion and validation are highlighted, capturing how they fit into the financial risk environment. Many payment

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 03 Issue: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 09 ACCEPTED: FEBRUARY 24 PUBLISHED: MARCH 18

transactions are generated from orders placed on retail websites; a model that captures their risk architecture is briefly outlined, and a high-level perspective of data flows during integrated development and deployment is introduced, followed by more detail on general automated deployment.

3.1. Cloud-Native Paradigms for Payments

Delivering high-performance, scalable, and fault-tolerant online payment processing systems is a significant engineering challenge. To address these challenges, the payment stack can be fully rebuilt by utilizing the most up-to-date cloud-native and DevOps paradigms. Payment request ingestion and validation as well as payment transaction authorization and settlement can run as fully managed services in the cloud. Generative AI-based risk analysis can provide risk probability scores and supporting evidence with fast response times and high throughput. DevSecOps pipelines can provide an automated framework for monitoring, testing, and securing all aspects of the system.

All payment transactions are passed through a transaction-router service that performs ingestion, transaction validation, risk scoring, log forwarding, and alert generation. Ingestion implements message flow control, load balancing, and fault detection, while validation checks for missing, invalid, duplicate, and corrupted messages. The transaction-router-proxy service acts as an ingress controller for the transaction-router service and provides domain name system services. An automated risk-scoring system uses generative AI techniques and a risk-model server to build and deploy risk classification models that respond in near real time with predictions and supporting evidence. All alerts and log records are forwarded to the security-information-and-event-management platform. Payment transaction requests are ingested, validated, authorized, and sent for settlement.

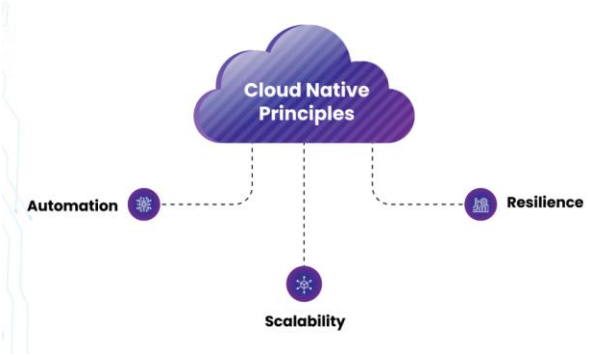


Fig 2: Cloud-Native DevOps

3.2. DevOps and SRE Practices

Security postures are verified and validated early in the software lifecycle, preferably on every commit, by employing integrated security toolchains in continuous integration/continuous delivery (CI/CD). The DevOps philosophy includes an absolute commitment to supporting change in production environments that may include more than 50% of servers every day, requiring cyber-resilient security implementations that expect, detect, and recover from compromise rather than attempting to prevent it. Specifically, reliance on perimeter security should diminish as systems move to the cloud and operating environments become shared. Failure of any control should be anticipated, enabling rapid recovery without significant brand reputation loss. National standards recommend reducing recovery time to less than 24 hours for major financial systems, and the trend is toward much faster recovery times of less than 8 hours. Indeed, fast recovery is becoming a cornerstone of cyber-resilient security for financial services, particularly in retail payment systems.

Cloud-native SRE principles and practices automate operational workloads, increase operational robustness, and minimize operational toil. A blameless culture normalizes the postmortem process, enables the emergence of training and knowledge documents that are not the sole responsibility of the affected individuals, and facilitates passage into corrective engineering. Redundant and diverse

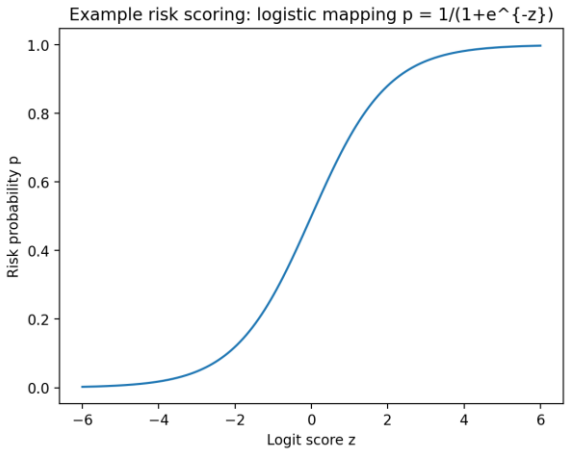
ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 03 Issue: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 09 ACCEPTED: FEBRUARY 24 PUBLISHED: MARCH 18

implementation of devops.gov Sachdeva's principles assures continuous security assessment during production, development, and design phases. Detection of anomalies signals the emergence of novel attack patterns as soon as possible, and a minimalist alerting strategy increases the signal-to-noise ratio. Penetration tests, exercises, continual testing of backup and restore procedures exercise and validate the success of these treatments.

4. Secure Payment Processing Stack

The secure payment stack described is designed for processing transactions Trusted Execution Environment (TEE) and Smart Contract architectures avoid reliance on privileged code performing critical processing. Transactions are generated by clients and travel through a sequence of functional blocks: ingestion and validation, who is accessing the transition, assets involved, policies governing transition from state to state of stakeholders, and the proper regenerated data to shape the transaction.

Ingestion readies the transaction for approval by systems of record (SoR) that ensure desired properties are met when providing consent. Approval or denial decisions capture how available resources and privileges relate to enclosed policies, while logs record pending and awaited responses. During analysis activities, detections and risk-result correlations are fed back for completeness, risk sensitivity optimization, and communication retraining. Reactions to approvable transactions are responsive automatic transactions or semaphore-type notifications.



4.1. Transaction Ingestion and Validation

Although ingestion pipelines are typically considered part of data processing rather than application runtime, the points of integration of inbound payment messages of a financial system have close relationships to some of the explanations and analyses of security and stability. In the cases of security and compliance, the relationship involves the initiation of transactions and their interaction with detection mechanisms, such as anti-money-laundering services. In the case of reliability and availability, the distinction between application runtime and data processing can be blurred by considerations relating to both the applicability of the processing to payments still in transit and the desire to investigate holistically other payment-system activities that are more frequently proxied over the messaging layers than executed in line with the traditional batch-processing paradigm.

Transaction ingestion consists of admitting external messages from a payment enabler, such as a merchant service provider, a payment network, or a payment gateway. Each payment message must be validated to ensure its completeness and conformity with accepted structure prior to further action. Validation must be in accordance with the acceptance criteria of the adjustment service performing the validation, with the criteria addressing both general issues and any business activity checks deemed necessary.

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 03 Issue: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 09 ACCEPTED: FEBRUARY 24 PUBLISHED: MARCH 18

Depending on the especially relevant focus for the detection mechanism, alerts about messages failing to validate might be flagged to either risk or operational business analysts for decision-making.

Equation 3: GAN / Adversarial Training Objective (ties to GAN mention + adversarial training)

Let:

- Generator $G(z;\theta_G)$ maps noise $z \sim p(z)$ to synthetic sample $G(z)$
- Discriminator $D(x;\theta_D)$ outputs probability sample x is real

Step 1: Discriminator wants

- $D(x)$ high for real data $x \sim p_{\text{data}}$
- $D\big(G(z)\big)$ low for generated data

So maximize:

$$V(D,G) = \mathbb{E}_{x\sim p_{data}}[\log D(x)] + \mathbb{E}_{z\sim p(z)}\left[\log\left(1 - D\big(G(z)\big)\right)\right]$$

Step 2: Min–max game

$$\max_D V(D,G) \quad \text{and} \quad \min_G V(D,G)$$

Step 3: Common generator alternative (better gradients)
Instead of minimizing $\log\left(1 - D\big(G(z)\big)\right)$, maximize $\log\left(D\big(G(z)\big)\right)$.

4.2. Authorization, Authentication, and Access Control
Authorization, authentication, and access control are critical components of any secure system. Different services and users require different access privileges. Authorization mechanisms automatically grant or restrict transaction approval authority based on user role and transaction type. Payment services should implement multifactor authentication.

Everyone risks losing or exposing sensitive information, such as passwords, security tokens, and cryptographic keys. They can share legitimate information due to phishing, collusion, or other attacks. Access control limits the exposure of secret information in case it falls into the wrong hands. It also mitigates security risks caused by shared information. Payment services need to monitor critical services, personnel, and resources constantly.

Access control may make services less usable. Payments require keys for credit checks, fraud detection, and other monitoring and control tasks. Security mechanisms should minimize risk without increasing usability issues beyond reasonable limits. Data Minima can help by collecting and processing only the data necessary for approval and execution or by using privacy-preserving methods.

For payments relying on public-key cryptography, private keys must be protected. Protecting keys while allowing them to decrypt secret information is a classic problem, and several solutions have been proposed.

5. Generative AI Risk Analysis Integration

Risk analysis is essential for all payment processing businesses, and integrating increasingly accurate Generative AI-based Risk Analysis modules in an easy-to-use way can really boost transaction security and data relevance for payment companies. Payment Systems for Risk Analysis offers a framework to model financial transaction risk with Generative AI. Generative techniques can help streamline the process of identifying risks in various Transactions like Payments – Debit/Credit etc., Investment Transactions, Domain-based Transactions, Device-level Transactions, Corporate Transactions and Financial Transactions etc., through advanced AI modelling tricks. Risk in any domain is almost inevitable, and with growing Cybersecurity Threats and Intrusion Attacks, it is exceedingly likely that Generative AI are going to play a pivotal role in risk identification across multiple domains.

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 03 Issue: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 09 ACCEPTED: FEBRUARY 24 PUBLISHED: MARCH 18

The Data Generation Flow components generate training data and provide a scrutinizing Data Advancement process which enhances risk model performance. Generative AI-based Risk Analysis should collect data such that Privacy for Payment Transaction is very important. Solution Privacy Approach enables extracting the facility to be used for finding data at higher accuracy without the data leakage but without gaining solution converting keys. Security measures while dealing with generative AI technologies are critical in Data Merging Process involved along with Generative AI. The risks in Generative AI are novel and varied due to the very nature of the underlying AI models and their application in a multitude of tasks, which hence become very domain specific. The risks differ based on the path followed by the generative AI solution: data preparation, model development, model deployment, or application.

Fig 3: Generative AI Risk Analysis Integration

5.1. Risk Modeling with Generative Techniques

Risk models are based on uncertainty in the values of the variables involved. In a cartoonish view of financial transactions, risk can depend on, for example, the identity of the transaction initiator, whether it is a buy or a sale, the relationship of the buyer and the seller, how much money flows and who is receiving it. In a more realistic setting where hundreds to thousands of attributes come into play, whether explicitly or implicitly, the direct use of generative AI methods requires a more delicate handling and cannot simply consist of representing in a symbolic way the different classes and their ritual actions. However, the intuitive capabilities of generative AI methods can be exploited by using them for the modeling of major risk exposure categories inside the transactions, which is the key aspect concerning the complicated and specific nature of the specific market where the transactions belong to.

To get a sense about how much money is needed to influence the market for a given transaction (spatial risk), the set of all transactions of a particular market category forming the set of transactions that happened in that market category at a time will be used to provide the context of the transaction. The identity of the user will not be used in this first-generation model. Generative AI will be used to model the amount of funds required to significantly influence the supply demand balance in the market. In time, the Stable diffusion or Imagen method could then provide this spatial risk index to all the transactions in its domain.

5.2. Data Handling, Privacy Preservation, and Model Security\

Modern payment systems collect tremendous amounts of sensitive personal information. Such data are valuable targets for unauthorized access, where success is likely given that most organizations have weak shields like user passwords. The GDPR imposes strict conditions on the processing and storage of personal data, mandating that it be minimized and, if truly required for the purposes pursued, employed only in an irreversible form. Thus, using personal information in neural networks must be governed by a combination of the GDPR principles of data minimization, proper anonymization, and appropriate defense of the learning models.

The principle of data minimization indicates that during training, the generative techniques should work over as little real data as possible. User data from other systems can be inserted into the training pipeline in such a way that only a small subset of samples of actual users of the payment system can be employed with GANs to generate new synthetic users that differ statistically from real users in a way that does not compromise privacy. Proper anonymization is particularly important in the payment context since the bank transactions could reveal information about private and sensitive behaviors of the users, once these transaction flows can be reconstructed. As a general

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 03 Issue: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 09 ACCEPTED: FEBRUARY 24 PUBLISHED: MARCH 18

principle, data cannot be restored from the generated samples by using a GAN, however, if information about the user payment behavior is included during the training, it could be very difficult to ensure that user privacy is preserved and does not compromise the legal requirements. Data anonymization techniques have been proposed and proven successful in ensuring the protection of the personal data used in GAN training without compromising the quality of the generated data and images.

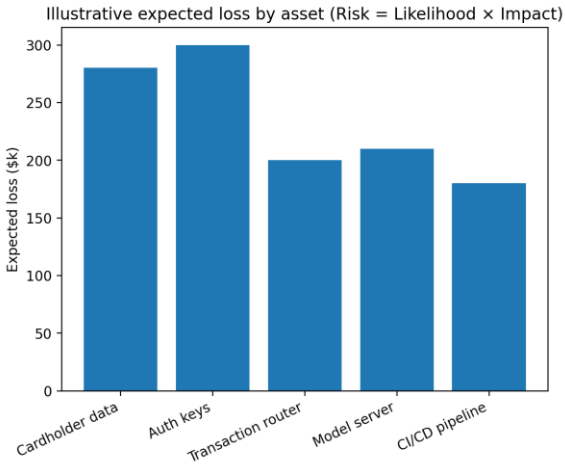
A product of the Generative AI Risk Analysis feature is a generative model able to simulate the damage caused by potential cyberattacks. The generated model is not allowed to be used in real scenarios. Organizations investing their resources in composing and training such a model must be protected against the scenario where the model is stolen, and its use becomes free. The GAN model must hide every detail and specificity of the real data, thus rendering the generated model a poor emulator of the original data. Adversarial training could be used as an appropriate strategy to generate such a defense buffering the above risk.

6. DevOps Automation and Cloud-Native Tools

Empowering an automated, secure, and reliable architecture for a modern payment system through the ecosystem of CI/CD pipelines built using cloud-native tools and DevOps principles.

Cloud-native and DevOps practices enable fast and secure delivery of modern business applications. They lay down principles that can help deploy complex architectures in a reliable manner and with shorter time to market. Security of payment systems is of utmost importance and needs to be used as part of the CI/CD pipeline. Security testing, including dynamic and static application security testing, runtime application self-protection (RASP), infrastructure as code scanning, API testing, and compliance checks must be integrated into the CI/CD pipelines using a set of tools.

The principles of Infrastructure as Code (Isac) must be followed, using tools like Terraform, CDK, AWS Cloud Formation, and Deployments in Kubernetes. The environments must be easily reproducible, enabling support for production-like environments for staging, testing, UAT, and development. Immutable infrastructure principles supported by tools like AWS AMI Builder, Packer and Jenkins/Kubernetes Master are essential. The deployment pipeline on Kubernetes helps build, scan, test, sign, and deploy container images in the registry on successful testing. Security scanning must also include image scanning in the container registry and mandatory checking of vulnerability assessment reports before deploying the application in production.



6.1. CI/CD Pipelines for Secure Payments

The application and data pipelines must be validated and updated regularly to achieve safe incremental changes to the digital payment service. Continuous Integration and Continuous Deployment (CI/CD) methods can be successfully applied using cloud and DevOps practices. Security measures must also be integrated into the CI/CD process so that code is reviewed not only for functionality, but also for security vulnerabilities. One approach, a DevSecOps pipeline, integrates Security into the continuous testing phase of traditional DevOps CI/CD use.

ISSN: 3050-9696 E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 03 Issue: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 09 ACCEPTED: FEBRUARY 24 PUBLISHED: MARCH 18

Continuous Integration for Cloud-Native Microservices follows the pattern of CI/CD developed for cloud-based microservices application development. Virtual Container technology enables Developer and Frequent Short Feedback Cycles, and common tools for Container Build, Quality Test and Deployment are using or re-implementing. However, a cloud-based approach focuses on cooperation between different development teams rather than reducing personal wait time and resource use.

Any CI/CD Build should also include automatic Security Testing. It should include checks to ensure vulnerability scans of applications are a normal part of the CI/CD Cycle, and that these are not skipped. An internal standard on connection to external resources should be developed and applied through a CI/CD checkpoint. Penetration Testing of the Container Image prior to going to Production should also be enforced.

Securing continuous testing of the application regularly with Fuzz Testing is suggested to support secure deployment of the cloud-native financial payments service. Regular Transient Environment Debugging, Scanning and Penetration Testing and Automated Secrets Protection ensure Security Checks occur regularly.

Equation 4: CI/CD Security Gate Math (ties to "security testing must be integrated into CI/CD")

Step 1: Escape probability across all gates
If independent:

$$P(\text{escape}) = \prod_i (1 - q_i)$$

Step 2: Detection probability

$$P(\text{detect}) = 1 - \prod_i (1 - q_i)$$

→ The PDF includes a **bar chart** showing the pipeline security checkpoints (illustrative).

6.2. Infrastructure as Code and Immutable Infrastructure

To enhance management simplicity with multi-cloud deployment strategies that reduce recovery times, cloud-native services are designed as first-class components without compromise, enabling immutable infrastructure management models with infrastructure as code configured in provision workflows. Deployment-management patterns supporting DevOps and SRE teams are simplified through automated enablement of preconfigured continuous integration and delivery processes as part of provisioning workflows. Ad-hoc integration of multiple developer teams and dev packages through dev cloud environments enhance development speed. Validation capabilities, such as dynamic QA environments using regression and performance testing, are already integrated with the infrastructure as code through pipeline stages.

Communication in the form of chatbot systems is a natural integration for understanding management of payments that integrate with specialized network management and payment fraud systems while providing access control, risk scoring, and management through natural language queries and responses. Risk analysis, including multi-agent gaming strategies at different levels of abstraction, via engineering with generative modeling techniques is a natural integration at the payment fraud detection at different abstraction levels, enabling consideration for modeling and visualization with a generative approach and response prediction from scene analysis and strategic

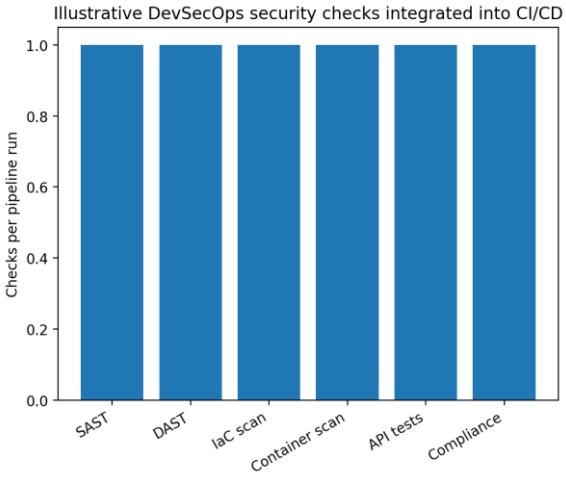
Compliance support through proactive analysis of emerging laws and regulations described in generative models is also possible. Integration of risk analysis through generative modeling directly enhances the security of fraud systems that utilize schema vulnerability scans, tomography pattern detection, malicious activity detection, and service node generation.

7. Conclusion

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 03 Issue: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 09 ACCEPTED: FEBRUARY 24 PUBLISHED: MARCH 18

The proposed architecture clearly expresses integrated, cloud-native principles and is intended to unlock the DevOps automation potential for highly sensitive financial payments systems. Generative AI techniques are utilized to enhance real-world risk analysis capabilities within natural constraints, while continuous integration, continuous deployment, and Site Reliability Engineering best practices spearhead continuous automation of end-to-end solution development, testing, validation, and deployment.

Emerging trends may redefine aspects nowadays considered key. Central banks are venturing into central bank digital currencies, offering the promise of free-money payments for all parties and the various implications of significant currency substitution within the financial economy. Market-based crypto-assets—beyond Bitcoin, still the poster child—offer novel forms of pseudo-private money and their own set of opportunities and hazards. Risk analysis using generative models, presently still an experimental endeavor, may become commonplace, with further incentives for organizations delivering real-time risk analysis models to gain a business advantage. Such models hold the potential for a new round of open data and privacy discussions, especially if cryptographic mechanisms such as differential privacy continue to prosper.



7.1. Emerging Trends

DevOps-oriented cloud-native financial payment architectures employing generative artificial intelligence (AI) risk assessment systems are truly novel. While separate elements exist in industry practice and academic publications, deploying them collectively to protect payment systems from risk is uncharted territory, although early indications suggest potential commercial or academic value.

Efficient risk analysis methods, notably those based on natural-language processing models—have potential applicability across verticals, including finance and payments. Yet existing literature has overlooked the integration of these capabilities into payment processing infrastructure. Furthermore, with security at the forefront of payment-system design, models, data, and methods used in generative risk analysis must preserve data confidentiality while delivering real-world utility, covering all payment-related operations across geographic or corporate boundaries. These requirements are often not fulfilled in existing models, which expose sensitive data, lack sound statistical foundations, or both. A solution capable of realizing the model without exposing the data would thus provide a technological leap.

8. References

[1] Almoras, M., Grundy, J., & Ibrahim, A. S. (2016). Collaboration-based cloud computing security management framework. Journal of Network and Computer Applications, 62, 31–44.

[2] Arora, S., Barak, B., & Brunner Meier, M. (2023). Measuring the macroeconomic impact of AI. Brookings Papers on Economic Activity, 2023(2), 1–49.

[3] Beyer, B., Jones, C., Petoff, J., & Murphy, N. R. (2016). Site reliability engineering:

ISSN: 3050-9696 E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 03 Issue: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 09 ACCEPTED: FEBRUARY 24 PUBLISHED: MARCH 18

How Google runs production systems. O'Reilly Media.

[4] Burns, B., Grant, B., Oppenheimer, D., Brewer, E., & Wilkes, J. (2016). Borg, Omega, and Kubernetes. Communications of the ACM, 59(5), 50–57.

[5] Carillo, F., Dal Pozzolo, A., Le Borgne, Y.-A., Caelen, O., Mazdzer, Y., & Bontempi, G. (2019). Scarff: A scalable framework for streaming credit card fraud detection with Spark. Information Fusion, 41, 182–194.

[6] CNCF (Cloud Native Computing Foundation). (2025). Cloud Native 2024: Approaching a decade of code, cloud, and change (Annual survey report). Linux Foundation Research.

[7] Dal Pozole, A., Borachio, G., Caelen, O., Lippi, C., & Bontempi, G. (2018). Credit card fraud detection: A realistic modeling and a novel learning strategy. IEEE Transactions on Neural Networks and Learning Systems, 29(8), 3784–3797.

[8] Dragoni, N., Lanese, I., Larsen, S. T., Mazzara, M., Mustafin, R., & Safina, L. (2017). Microservices: How to make your application scale. In E. Di Nitto & M. Harman (Eds.), Lecture Notes in Computer Science (Vol. 10311). Springer.

[9] Forsgren, N., Humble, J., & Kim, G. (2018). Accelerate: The science of lean software and DevOps. IT Revolution Press.

[10] Geng, T., Xu, Z., Li, H., Liu, X., Zhang, N., & Xiao, C. (2025). Prompt injection attacks on large language models: A survey of attack methods, root causes, and defense strategies. Computers, Materials & Continua, 2025, Article 074081.

[11] Gong, N. Z., & Liu, B. (2024). Trustworthy AI in practice: Attack surfaces, evaluation, and governance. ACM Computing Surveys, 56(10), 1–38.

[12] Google. (2020). Beyond Prod: A new model for production change management in large-scale infrastructure. Google Research.

[13] He, L., Zhang, Y., Wang, H., & Chen, J. (2025). LPRAG: Locally private retrieval-augmented generation with formal privacy guarantees. Information Sciences, 677, 120–139.

[14] Humble, J., & Farley, D. (2010). Continuous delivery: Reliable software releases through build, test, and deployment automation. Addison-Wesley.

[15] International Organization for Standardization. (2022). ISO 20022: Financial services—Universal financial industry message scheme. ISO.

[16] International Organization for Standardization. (2022). ISO/IEC 27001:2022 Information security, cybersecurity and privacy protection—Information security management systems—Requirements. ISO.

[17] Jaffal, N. O., Alkanofer, M., & Muheisen, D. (2025). Large language models in cybersecurity: A survey of applications, vulnerabilities, and defense techniques. AI, 6(9), 216.

ISSN: 3050-9696 E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 03 Issue: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 09 ACCEPTED: FEBRUARY 24 PUBLISHED: MARCH 18

[18] Kim, G., Humble, J., Debois, P., & Willis, J. (2016). The DevOps handbook. IT Revolution Press.

[19] Kreuz Berger, D., Kühl, N., & Hirschl, S. (2023). Machine learning operations (Mops): Overview, definition, and architecture. IEEE Access, 11, 31866–31879.

[20] Li, M. Q., Zhang, H., & Wu, J. (2025). Security concerns for large language models: A survey. arXiv.

[21] MITRE. (2024). SAFE-AI: A framework for securing AI-enabled systems (Technical report). MITRE.

[22] MITRE. (2025). MITRE ATLAS: Adversarial Threat Landscape for Artificial-Intelligence Systems (Knowledge Base). MITRE.

[23] National Institute of Standards and Technology. (2020). Zero trust architecture (NIST Special Publication 800-207). U.S. Department of Commerce.

[24] National Institute of Standards and Technology. (2020). Security and privacy controls for information systems and organizations (NIST Special Publication 800-53, Rev. 5). U.S. Department of Commerce.

[25] National Institute of Standards and Technology. (2022). Secure software development framework (SSDF) version 1.1: Recommendations for mitigating the risk of software vulnerabilities (NIST Special Publication 800-218). U.S. Department of Commerce.

[26] National Institute of Standards and Technology. (2023). Artificial intelligence risk management framework (AI RMF 1.0) (NIST AI 100-1). U.S. Department of Commerce.

[27] Newman, S. (2021). Building microservices (2nd ed.). O'Reilly Media.

[28] OWASP. (2023). OWASP Top 10 for Large Language Model Applications (v1.0). OWASP Foundation.

[29] Paleyes, A., Urma, R.-G., & Lawrence, N. D. (2020). Challenges in deploying machine learning: A survey of case studies. ACM Computing Surveys, 54(6), 1–36.

[30] Payment Card Industry Security Standards Council. (2024). Payment Card Industry Data Security Standard: Requirements and security assessment procedures (PCI DSS v4.0.1). PCI SSC.

[31] Rahman, M. A., & colleagues. (2024). Fine-tuned large language models (LLMs) for prompt injection vulnerability analysis. arXiv.

[32] Şaşal, S. (2025). Prompt injection attacks on large language models. In Proceedings of the International Conference on Security and Cryptography (SECRYPT). SCITEPRESS.

[33] Tabassi, E. (2023). Trustworthy and responsible AI: Risk management and measurement challenges. Journal of Information Policy, 13, 1–28.

[34] Xu, W., Liu, Y., & Chen, Z. (2025). A survey of attacks on large language models. arXiv.

ISSN: 3050-9696
E-ISSN: 3050-970X
EUROPEAN ADVANCED JOURNAL FOR SCIENCE & ENGINEERING
Volume: 03 Issue: 01
RECEIVED: JANUARY 26 REVISED: FEBRUARY 09 ACCEPTED: FEBRUARY 24 PUBLISHED: MARCH 18

[35] Yao, Y., Duan, J., Xu, K., Cai, Y., Sun, Z., & Zhang, Y. (2024). A survey on large language model (LLM) security and privacy: The good, the bad, and the ugly. High-Confidence Computing, 4, 100211.

[36] Zeng, S., Zhang, J., He, P., Xing, Y., Liu, Y., Xu, H., Ren, J., Wang, S., Yin, D., Chang, Y., & Tang, J. (2024). The good and the bad: Exploring privacy issues in retrieval-augmented generation (RAG). In Findings of the Association for Computational Linguistics: ACL 2024.

[37] Zhao, K., & colleagues. (2025). A survey on model extraction attacks and defenses for large language models. arXiv.

[38] Zhou, Y., Zhang, Z., & Chen, X. (2024). Cloud-native security for microservices: A systematic literature review. Journal of Systems and Software, 207, 111836.

[39] Zou, D., Wang, X., & Jin, H. (2023). Service mesh in production: Reliability, observability, and security for microservices. IEEE Software, 40(4), 52–60.