ISSN: 2754-6659

Journal of Artificial Intelligence & Cloud Computing

SCIENTIFIC Research and Community

Review Article

Open Access

Architecting Resilient Online Transaction Platforms with Java in a Cloud-Native World

Rajesh Kotha[1]* and Pavan Kumar Joshi[2]

[1]Software Development Engineering Advisor, Fiserv, USA

[2]Director Information Technology, Fiserv, USA

ABSTRACT

The exponential growth of digital payments and e-commerce has revolutionized the consumer purchasing experience, enabling seamless transactions across borders. In this landscape, electronic money (e-money) and digital payment platforms have emerged as key enablers, optimized transaction efficiency and enhancing communication between businesses and financial institutions. As these solutions become integral to global commerce, the resilience of payment platforms—particularly within cloud-native architectures—has become a critical concern. This paper examines the integration of cloud-native design principles into modern payment platforms, emphasizing the need for architectural approaches that prioritize resilience, scalability, and security. As businesses increasingly rely on digital payment solutions, the demand for platforms capable of handling fluctuating transaction volumes and potential security threats is more pressing than ever. The research focuses on best practices for building resilient payment systems using Java and Spring Boot, exploring how cloud-native techniques such as microservices architecture improve agility and responsiveness. Additionally, it highlights the importance of employing layered security strategies and automated resilience mechanisms—such as circuit breakers and bulkheads—to safeguard system reliability. This study offers a comprehensive guide to architecting secure, resilient, and adaptable payment platforms that meet the evolving demands of the digital economy.

***Corresponding author**

Rajesh Kotha, Software Development Engineering Advisor, Fiserv, USA.

Received: October 05, 2022; **Accepted:** October 09, 2022; **Published:** October 17, 2022

Keywords: Online Payment Platforms, Cloud-Native Architecture, Digital Economy, Microservices Architecture, Java Resilience Strategies, Application Security, Platform Scalability, Layered Security

Introduction

Payments and e-commerce are perhaps the fastest-growing industries. A customer can buy any desired product while he/she is at home because e-commerce makes web-based shopping easier and better thus eliminating geographical boundaries between buyers and sellers. Similarly, it is also possible for sellers to sell their products everywhere, which makes why online shopping and electronic money easy and fast. E-money or e-payment facilities are one of the clean and efficient methods of transacting and are important in the design of efficient payment systems. It integrates the transaction processes and communications of the seller/business and the finance community within the e-commerce structure. This system is becoming prevalent in the developing and developed world and, therefore, the world. There are several methods by which the clients can purchase goods with them, for instance credit cards, debit cards, electronic funds transfers (E.F.T.), etc. [1]. There is a huge need for resilience in online platforms, especially in the field of cloud-native technologies. The word "cloud-native" was used a lot when cloud computing was first introduced in 2006, which seems pretty clear now. That being said, the word almost disappeared. Suddenly, and in the last few years, the term has been used more and more, and it seems to be picking up speed [2].

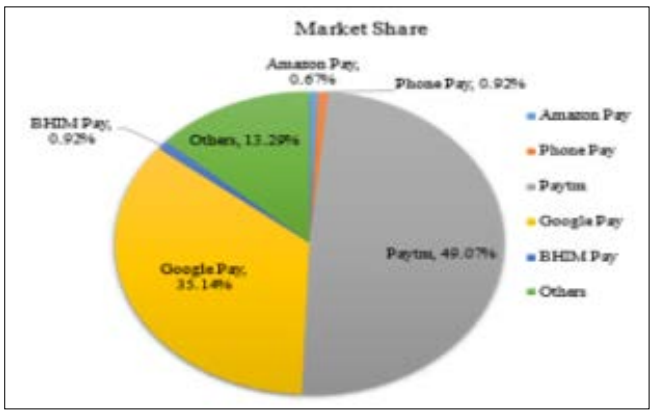

Figure 1: Percentage of Studies Conducted in each Category

Figure 1 depicts the percentage of studies conducted in each category of digital payment technologies. Digital transactions and payment systems have several issues, like scalability, security issues, and performance bottlenecks. These issues can cause delays, high costs, and a lack of trust from users. Cloud-native design patterns, which integrate microservice and cloud-focused design patterns to provide characteristics like massive scalability and resilience, can help tackle this issue [3,4]. However, putting these patterns to use is a difficult procedure that raises several security issues. In this work, we investigate these security issues and use cloud-native design techniques to reimagine and launch a single

SaaS application. We also consider appropriate countermeasures for multilayer security to be used at the application and cloud networking layers.

Redesigning payment platforms to leverage cloud-native technologies involves a careful migration from monolithic systems to microservices, accompanied by the implementation of layered security measures. By integrating security protocols at both the application and cloud networking layers, businesses can better protect themselves against potential threats. Continuous monitoring of these systems will also play a vital role in detecting and responding to security incidents promptly. Ultimately, architecting resilient payment platforms in a cloud-native environment is crucial for meeting the demands of an evolving e-commerce landscape and ensuring the trust of users in digital payment solutions.

Motivation for this study arises from the rapid evolution of digital payment systems and the increasing reliance on cloud-native architectures to support them. As businesses transition to these innovative solutions, there is a pressing need for resilient, secure, and scalable payment platforms capable of handling diverse transaction loads and potential security threats. The aim of this paper is to investigate effective architectural patterns and best practices for developing such platforms using Java and the Spring Boot framework, ultimately providing a comprehensive framework that addresses the unique challenges posed by cloud environments and enhances the agility of payment systems in the digital economy.

Structure of the Paper

The following paper is structured as; Section II provide the cloud-native architecture for payment platforms, Section III discussed the systematic mapping process for cloud-native services, Section IV give the overview of digital payment application, Section V discussed the resilience of payment platforms, Section VI provide the java in building resilient payment platforms, Section VII discussed some challenges for security in cloud-native services, Section VIII provide the literature review , at last section provide the conclusion and future work.

Cloud-Native Architecture for Payment Platforms

A software program that is cloud-native is a distributed, adaptable, and horizontally scalable system. It is composed of as few stateful components as possible that maintain state separation using (micro) services. The program is created utilizing cloud-focused design patterns in each of its self-contained deployment blocks, and it is operated on a self-service elastic platform [5]. Publicly available and versioned APIs are the primary means by which services communicate with one another in a cloud-native application design. API-based cooperation is the term for this. These APIs can utilize alternative protocols and serialization formats, but they typically employ HTTP REST and JSON serialization. The single deployment components of the architecture are created and connected using a set of cloud-focused patterns, such as the circuit breaker pattern, the twelve-factor app collection, or cloud computing patterns. Finally, these microservices are deployed and operated via self-contained deployment units, or containers, using self-service agile infrastructure tools. These systems provide additional operating functions, like load balancing and dynamic routing, log and metric collection, and application health management, on top of IaaS infrastructures [6]. Among these functionalities are on-demand and automated application instance scaling. Figure 2 depicts the overall cloud-native design.

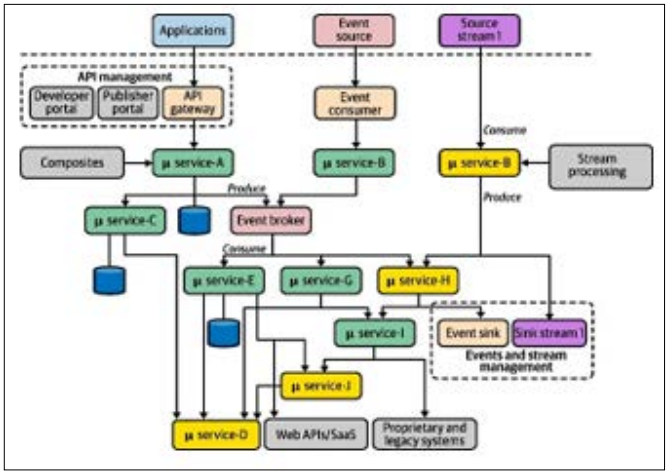

Figure 2: Cloud Native Applications Architecture

Infrastructure-as-a-Service, or IaaS, is the name of the service model [7]. "Compute" computers are the IT resources that are used over a network for networking, storage, and/or processing power. The operating systems and applications that are installed in the virtual machines (VMs) that the IaaS customer rents are their own responsibility. The hardware is handled by the cloud provider. The uses of IaaS services (such as virtual servers, storage, and networking) may be restricted if a user has an agreement with an IaaS provider. AWS's EC2, S3, Route 53, and other services are an example of an IaaS offering. As a member of Google Cloud Platform, the company's IaaS and PaaS group, Compute Engine is likewise an IaaS service. The business in question is equipped with a private cloud. When hosting is done on-premise, it takes place at the client's location and may be managed by the client themselves [8]. You can leverage Microsoft's Azure, IBM's SoftLayer/Bluemix, or VMware's vSphere to put up an internal cloud PaaS or IaaS. While their investment banking divisions would exclusively utilize Office 365 on-premise, some banks' retail banking operations might use it in the cloud. In a similar vein, Splunk is a software that many EU banks use for data gathering, monitoring, and analysis. It can be utilized through SaaS or installed on-premise [9].

Benefits of Cloud-Native Systems in Payment Services

The following Cloud-Native systems in Payment services Benefits are:

- **Digital Identification and Verification and KYC:** The main benefit of cloud services is the ability to process photographs instantly utilizing cloud-stored legal documents and images to verify and identify the individual as well as learn more about them.
- **Digital Signature:** Digital signatures are an additional option that enable users to access documents from any location in the world. This reduces the expense of operating a business and facilitates doing business. Investing in AI and the Internet of Things is a wise strategy to keep ahead of the competition in the market, even though it is a one-time event.
- **Digital Payments:** Digital payments have increased significantly over time, and the global acceptance of these payments has been facilitated by the current epidemic. The compliance agreement enables users to buy and sell goods globally. In this instance, cloud-based digital payments have an impact.
- **Banking Through Smart Speakers:** Banks and IT businesses are heavily investing in speech biometrics,

which is implemented through Internet of Things devices such as smart speakers. Once again, banking through smart speakers will revolutionize the industry and spur massive cloud infrastructure development initiatives [10].

Systematic Mapping Process for Cloud Native Services

Cloud-native services alter software application design, development, deployment, and management in cloud computing. This cloud-based solution improves application performance, availability, scalability, and efficiency. Through principles and techniques, cloud-native services deliver durable, scalable, and portable applications. Containerization isolates application components and dependencies into lightweight containers, supporting these services [11-13]. Containerization optimizes resource utilization, streamlines deployment, and ensures environment consistency.

DevOps, a scientific approach that promotes collaboration and task automation between development and operations teams, is the foundation for software development and delivery. Code integration, testing, and deployment are all automated and go by the name of Continuous Integration and Continuous Deployment (CI/CD). This improves software and lowers the risk of launching new features. DevOps uses Infrastructure as Code (IaC) to design and maintain infrastructure based on code, ensuring consistency and repeatability. This methodical integration of automation, version control, and infrastructure management makes developing and implementing cloud-native software more dependable, efficient, and user-friendly.

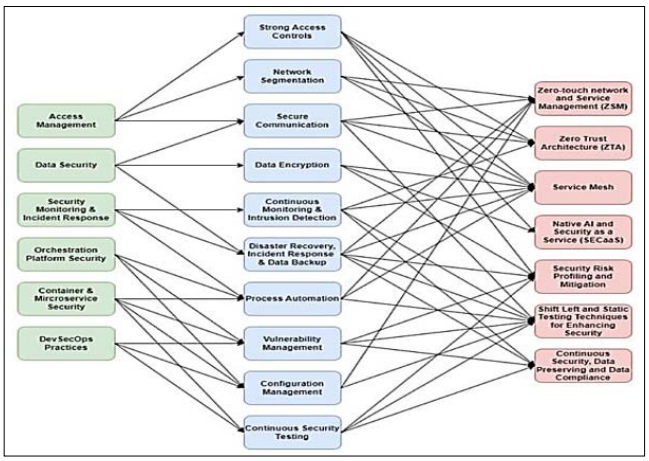

Figure 3: A Systematic Mapping Process

A thorough design process that incorporates several crucial elements, functionality, and security choices for cloud-native applications is depicted in Figure 3. Important sections are highlighted in green features are highlighted in blue, and responses are highlighted in red for aesthetic purposes.

Cloud-native microservices divide applications into loosely linked, deployable services. Each service can be designed, deployed, scaled, and updated separately for a specific business purpose [14]. This flexible, agile, and scalable architecture allows organizations to swiftly add new goods and meet changing demands. Microservices use containers to separate applications and dependencies. With their lightweight and portable runtime environment, containers maintain computer consistency. They streamline application scalability, deployment, and management, saving resources and speeding up launch. Kubernetes orchestrates cloud-native container deployment, scalability, and administration [15,16]. They ensure application reliability using service discovery, load balancing, and self-healing. Orchestration systems allow developers focus on app design and deployment, not infrastructure.

Overview of Digital Payment Application

Online applications, are software programs that allow users to make browse, purchase products or services from their mobile devices or computers. They can be used to make peer-to-peer payments, pay bills, or pay businesses for products [17]. Using a computerized payment method in place of cash or cheques is known as digital payment [18]. The utilization of digital funds is growing in popularity since they are more user-friendly and straightforward. Customers can also make digital payments at any time and from any location. This expedites the transaction process and makes digital payments a beneficial substitute for conventional payment methods [19]. The image shows the current digital payment options in addition to their soon-to-be-released replacements. The Figure 4 depicts the digital payment technologies while Figure 5 shows the no. of studies conducted in these fields.

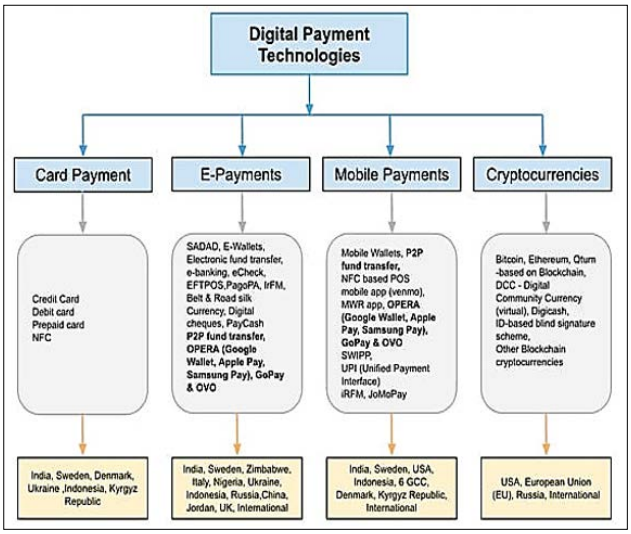

Figure 4: Digital Payment Technologies

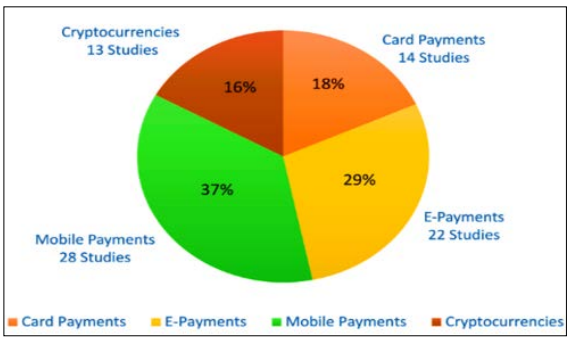

Figure 5: The Number and Percentage of Research Studies Conducted on Each Kind of Digital Payment Technology

Resilience in Online Application Platforms

Resilience in microservices is about designing systems that can handle failures gracefully. It involves anticipating potential points of failure, implementing mechanisms to recover from those failures, and ensuring that the system continues to operate, even if some components fail. In a microservices architecture, where services are loosely coupled and often rely on network communication, resilience becomes a crucial aspect of system design.

Key Resilience Strategies in Microservices

The following key Resilience Strategies in Microservices are discussed below:

- **Circuit Breaker Pattern:** The circuit breaker pattern is a fundamental strategy for resilience in microservices. It prevents a service from repeatedly attempting to call a failing service, which can cause cascading failures. When the circuit breaker is "open," the service immediately returns an error or a fallback response, giving the failing service time to recover. This strategy prevents the system from being overwhelmed by repeated failure attempts and helps in containing the failure within a specific service.
- **Bulkhead Pattern:** The bulkhead pattern is inspired by the compartmentalized design of a ship, where failure in one compartment does not lead to the sinking of the entire vessel. In microservices, this pattern involves isolating different parts of the system to prevent a failure in one service from affecting others. By allocating separate resources (e.g., threads, memory) to different services, the bulkhead pattern ensures that even if one service fails, others can continue to operate normally.
- **Failover Mechanisms:** Failover mechanisms automatically switch to a standby system or service when the primary one fails. In microservices, this can be implemented at various levels, including the service, database, and network layers. By ensuring that a backup is always available, failover mechanisms help in maintaining service continuity during unexpected failures.
- One way to pay for items you purchase online is with an electronic payment. Doing business is made easier for both buyers and sellers by e-payment. The usage of electronic payments benefits consumers, vendors, online retailers, and the government in general. With electronic payments, transaction data can be saved for subsequent use in additional studies. Finding consumer trends and a product's market share in a certain region might be aided by this. Other advantages of e-payments include speedy payments, less time spent, decreased expenses, and increased buyer-user trust. Financial transactions occur as new technology is developed and applied to electronic payment systems. Good electronic payment technologies tend to ingrain themselves into users' minds and alter their expectations of how things should operate [20]. Figure 6 shows the Payment information registration system.

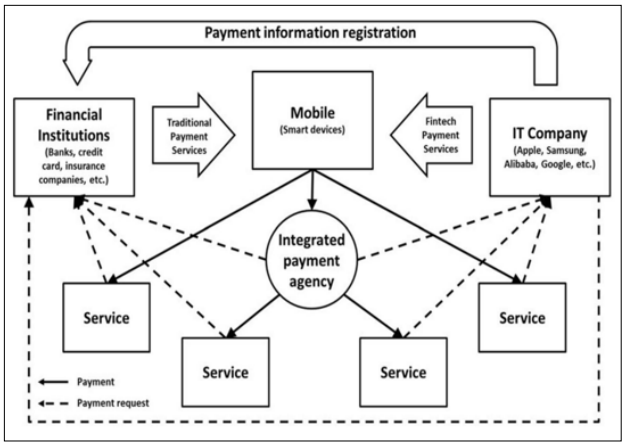

Figure 6: Payment Information Registration

Online payment processing can be categorized into two primary types: account-based and electronic currency techniques. Therefore, you can use an existing personal account, such as a bank account, to make payments utilizing account-based techniques. On the other side, digital currency techniques enable users to make payments using their digital money. Common account-based methods include credit and debit cards, online banking, and payment portals like PayPal. However, computer currency systems include things like smart cards and online payment systems like Bitcoin [21]. The novel concept of cloud-based mobile payments has the potential to address these problems related to faith. Parts of payment apps or entire apps are stored in cloud storage. The phone has unique features that make using it as a repeater safe. Relay technologies are occasionally viewed as potential hazards in the scientific literature. However, we believe that monitoring and recording financial activities in the cloud increases security and provides us with valuable information in the context of big data. The two main components of our cloud payment paradigm are mobile host card emulation (HCE) and servers hosting secure elements. As previously said, the most crucial component of the cloud model payment process is trust in mobile terminals. The TLS protocol makes reciprocal verification possible. TLS processing is used by security components or Trusted Execution Environment (TEE) architecture to ensure security. The majority of safe components can function with a Java Virtual Machine (JVM). A Java Card TLS program runs in one to ten seconds in the full mode of TLS and in two to five seconds in the reduced mode [22].

Java in Building Resilient Payment Platforms

In 1995, Sun Microsystems developed the computer language Java. Oracle assumed control of the Java project after purchasing Sun Microsystems in 2010. Its class-based and object-oriented design makes it popular among engineers. As per the source, it is among the top three most widely used computer languages. Thus far, Oracle has released several Java versions that are easier to use and more suited for programmers [23,24]. Java 8 is one of the most recent versions that is suitable for commercial use. This method accumulates to numerous things, such as Lambda Expressions, a new feature, allows programmers to more succinctly specify instances of single-method interfaces. Code can be viewed as data, and functionality as a method parameter. A new class named Streams offers a Stream API that enables you to perform functional operations on streams of elements. Its ability to work with the Collections API allows you to perform bulk actions on collections.

Java Spring Boot Framework

The base of the new Spring Boot framework is the Spring framework. Programmers may create web-based apps more quickly and easily with the help of this technology. Spring Boot requires less configuration than the Spring framework. It's time to provide you with a ton of XML descriptions. All that's required is careful selection of variables. The Spring software is built around Multi-tier Architecture. It appears as follows:

- **The Web Layer** – It takes care of requests from users and errors that other layers throw.
- **The Service Layer** – It implements the business rules of the application.
- **The Repository Layer** – Its responsibility is the database connection.

Although it's a strong tool, Spring Boot is not enough to develop microservices; other tools are required to handle basic architectural elements like communication. To assist you with this, Spring Cloud offers standard patterns for distributed systems [25].

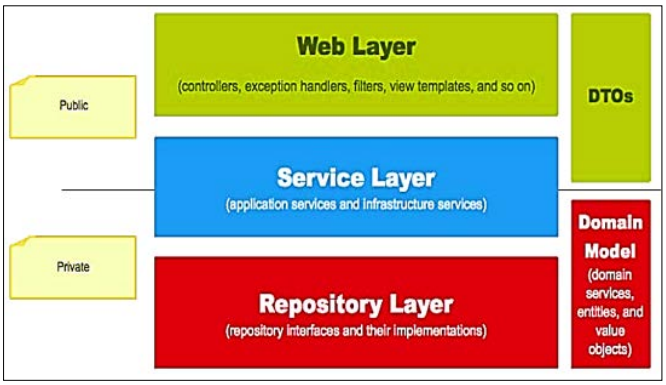

Figure 7: Shows the Typical Three-Tier Architecture Found in a Lot OF Online Apps

Figure 7 depicts the illustrates a typical three-tier architecture commonly used in web applications, including a Web Layer for handling user interactions, a Service Layer for business logic and transaction management, and a Repository Layer for data access. It also highlights the use of DTOs for data transfer and a Domain Model for encapsulating business logic.

Autonomic and Cognitive Security Management

Many technologies will underpin cloud-native service cybersecurity frameworks. Paradigm-based frameworks like enabling technologies are examined in this section. Microservices are being used to make modern programs more portable, scalable, and reliable in cloud-native contexts. This approach creates new cloud-native application growth and complexity security issues. Intelligent and automated solutions are needed to reduce human workload in cloud-native application security management. Figure 8 shows a paradigm based on ETSI ZSM design principles for autonomous safe resource management in varied areas. This framework introduces node, end-to-end, and inter-slice AI-powered closed loops. Thus, it can swiftly and efficiently detect and mitigate security problems near the source, limiting network proliferation.

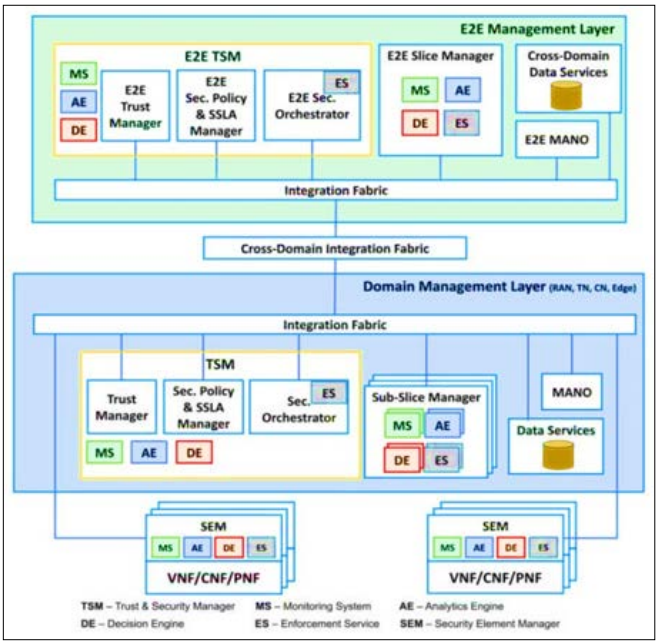

Figure 8: The Architecture of a Cognitive and Autonomous Security Management System

Figure 8 shows the architecture of an autonomous and cognitive security management system. By establishing layers, the technology enhances E2E security over the entire region. From network nodes to end-to-end links and levels between slices, AI-driven closed loops are created. By identifying and resolving security issues at their source, this fine-grained approach prevents them from propagating to other areas of the network. The ETSI ZSM design standards serve as the foundation for this framework. This is made possible by service-based design and independent security control. This design permits users who are authorized to access security control services to do so via an integration fabric. Integration fabric locates, files, and makes calls to security management services. Between these management services and other management services, there is more communication. Historical security management service data and expertise are stored by data services in the same or a different location and provided to the framework.

Challenges for Security in Cloud-Native Services

Cloud-native services offer numerous advantages, such as cost-effectiveness, flexibility, and the capacity to expand as needed. On the other hand, developing secure cloud-native services is challenging due to the numerous issues that surround them. This section is devoted to a cursory examination of a few of them. Because of the way cloud-native services manage and store sensitive data, it might be more difficult to maintain data security. Maintaining confidence, abiding by privacy regulations, and averting potential legal, financial, and societal issues all depend on the security of user data and private information. Furthermore, maintaining a consistent security posture is more difficult with cloud-native services due to their dynamic and scalable nature. Scalability is a big plus, but it also implies that security concerns must be properly considered. The concept that cloud-native services are harder to secure due to their distributed nature is supported by an examination of the many risks that they are likely to encounter [26]:

• **Malware Attacks:** Malware can target cloud-native services due to their dispersed deployment over multiple servers. Attackers have the ability to penetrate one area of the infrastructure before moving on to other areas that are connected to it. Data breaches, illegal access, or service disruptions could result from this. Additionally, vulnerabilities in cloud-native designs might be exploited by malware, compromising the security and integrity of the entire system [27].

• **Man-in-the-Middle (MITM) Attacks:** Cloud-native services are also more vulnerable to man-in-the-middle assaults due to their dispersed nature. These kinds of attacks involve someone listening in on a discussion between two individuals and altering it without the victims' knowledge. This can result in the loss, unauthorized access to, or alteration of private data. MITM attacks can exploit compromised certificates, lax encryption techniques, or gaps in communication links[28].

• **DDoS Attacks:** Every type of attack on Industrial IoT (IioT) and strategies for defending against it: A network or service may experience service outages if a large number of compromised devices overwhelm it with data. We refer to this as a Denial of Service (DDoS) assault. These attacks, also referred to as "Cloud Zombie" attacks, can be extremely detrimental to cloud-native applications that rely on network connectivity and access [29]. The ability of cloud settings to grow or decrease automatically has also been leveraged by a few variations of this attack.

Literature Review

This section includes the associated work on cloud-native Java payments. This paper aims to provide an in-depth analysis of the ways in which cloud-native technologies provide scalable and dependable payments.

Wang, Hahn and Sutrave, outline the various types of mobile payment systems and discuss a model for making mobile payments. We compile a list of security elements that consumers desire and those that are already available in mobile payment systems. We discuss three main security threats: data leaks, SSL/TLS flaws, and malware. We discuss four main security issues: detecting malware, utilizing multi-factor authentication, preventing data breaches, and identifying and thwarting fraud in mobile payment systems [30].

First, Zhang et al., discuss the adversary model, specifications, and system architecture of BCPay. They then discuss its design in considerable detail. Our security study scores BCPay with robust fairness as well as soundness. This implies that assaults such as recording discussions or altering the rules have no effect on the fairness. Our analysis of BCPay's performance reveals that it operates with exceptional efficiency when it comes to both transaction volume and calculation costs. We also develop a blockchain-based protocol for fog computing task outsourcing and a blockchain-based system for cloud computing data ownership verification to demonstrate the potential applications of BCPay [31].

As demonstrated by Zouina and Outtai, (2019), a distributed payment system built on Blockchain technology and payment tokens is suggested. By preventing unauthorized use of the customer's payment card details (PAN, CVV) as a result of third parties managing the payment system poorly, these payment tokens shield them against identity theft. A private Blockchain consortium contributes to the safety and privacy of the proposed payment system. Three private blockchains comprise this consortium: the acquisition and issuer banks' blockchains, the third private blockchain, which is the issuer bank; and Bank Authority, the interoperability domain. Particular nodes verify and check transactions to identify and thwart fraud attempts, thereby strengthening the security of the payment system [32].

Chun, (2019) investigate who bears responsibility for breaches of security or privacy in electronic transactions. Many devices and architectures, such as PCs, smartphones, tablets, smart meters, sensors, automobiles, and more, are used to send data. Along with the cloud and the Internet of Things, electronic trade likewise has well defined network structures. Security and privacy are compromised, though. This essay demonstrates that if a consumer loses a significant amount of money and there is a high likelihood of security or privacy breaches, the corporation may be held accountable. However, the extent of the customer's culpability depends on their attitude towards risk, the amount of money they lose, and the effectiveness of their security investment [33].

A R Sri Nandhini, discuss cloud-native apps as a replacement for on-premise software in their 2020 study. We use Platform as a Service (PaaS) for Application Streaming on-premise instead of installing on-premise application software on each server as is customary. Furthermore, the study attempts to elucidate the superiority of native-cloud applications over on-premise ones [34].

The survey by, Prokhorenko and Ali Babar examines the most recent architectural techniques proposed to increase the resilience of Cloud-, Fog-, and Edge-based systems. This paper offers a flexible taxonomy to examine several approaches to developing distributed systems that are robust. Additionally, this work presents a capability-based cyber-foraging technique that considers a physical node's capabilities in order to increase the overall resilience of the system. This survey also discusses trust issues and solutions within the framework of system resilience and reliability [35].

This Table 1 provides a concise overview of each paper, highlighting the main focus, technologies involved, and the security or resilience mechanisms they describe.

Table 1: Summary of Related Work for Payments with Java in a Cloud-Native World

Reference	Year	Topic	System/ Technology	Key Focus	Security/Privacy Mechanisms
Wang et al. [30]	2016	Mobile Payment Processing	Mobile Payment Systems	Malware, SSL/TLS vulnerabilities, and data breaches are examples of security threats. Stopping data leaks, employing multifactor authentication (MFA), identifying malware, and identifying and preventing fraud are some of the challenges.	Desired security services, existing mechanisms
Y. Zhang et al. [31]	2018	Blockchain-Based Payment (BCPay)	Blockchain	BCPay architecture, Soundness, Robust Fairness; Performance analysis and blockchain-based applications	Resilient to eavesdropping and malleability attacks
M. Zouina et al. [32]	2019	Distributed Payment System	Blockchain Consortium	Payment tokens to protect against identity theft; Security/privacy through private blockchains and special nodes for fraud detection	Blockchain to secure payment tokens, interoperability via "Bank Authority" blockchain

Chun et al. [33]	2019	Liability in Electronic Transactions	Electronic Commerce	Analysis of liability in security/privacy breaches; Factors affecting liability decisions	Liability depends on customer risk attitude, firm-side investment in security.
A R Sri Nandhini et al. [34]	2020	Cloud-Native Application Implementation	Cloud PaaS	Advantages of using PaaS for application streaming over on-premise software	-
Prokhorenko and Ali Babar [35]	2020	System Resilience in Distributed Systems	Cloud, Fog, Edge	Taxonomy of resilience approaches; Capability-based cyber-foraging framework for resilience improvement	Focus on system resilience, trust issues, and capability analysis.

Conclusion and Future Work

In the present world which is advancing radically in its digital space, it is important that there is a growing need for payment systems that are strong, safe, and efficient. In conclusion, architecting resilient payment platforms in a cloud-native environment is crucial for businesses seeking to thrive in the digital economy. The integration of Java and the Spring Boot framework enables developers to build scalable and efficient microservices that enhance the overall performance of payment systems. However, as these platforms evolve, they face significant security challenges that must be addressed through a proactive approach to security management. Implementing layered security strategies and resilience patterns is essential to safeguard sensitive transaction data and maintain user trust. Future work should focus on refining these architectural practices and exploring emerging technologies that can further enhance the resilience and security of payment platforms, ensuring they can effectively respond to the complexities of the modern digital landscape.

Future work in architecting resilient payment platforms should focus on integrating AI and machine learning for enhanced fraud detection and risk management. Exploring blockchain technology can improve transaction transparency and security. Developing standardised frameworks for microservices and APIs will facilitate interoperability, while automated testing and continuous deployment strategies will ensure ongoing reliability. Additionally, researching the impact of emerging cloud technologies, like serverless and edge computing, will provide insights for future architectural designs.

References

1. RF Olanrewaju, B Ul Islam Khan, MM Ul Islam Mattoo, F Anwar, AN Anis, et al. (2017) "Securing electronic transactions via payment gateways – a systematic review," Int. J. Internet Technol. Secur. Trans doi: 10.1504/IJITST.2017.089781.
2. N Kratzke, PC Quint (2017) "Understanding cloud-native applications after 10 years of cloud computing - A systematic mapping study," J. Syst. Softw doi: 10.1016/j.jss.2017.01.001.
3. MT VISHWA VIJAY Kumar, MUKUL Tripathi, SATISH KUMAR Tyagi, SK Shukla (2007) "An integrated real time optimisation approach (IRTO) for physical programming based redundancy allocation problem," Proc. 3rd Int. Conf. Reliab. Saf. Eng. Udaypur, Rajasthan, India 692-704.
4. VV Kumar (2014) "An interactive product development model in remanufacturing environment : a chaos-based artificial bee colony approach," https://scholarsmine.mst.edu/masters_theses/7244/.
5. VV Kumar, FTS Chan, N Mishra, V Kumar (2010) "Environmental integrated closed loop logistics model: An artificial bee colony approach," in SCMIS 2010 - Proceedings of 2010 8th International Conference on Supply Chain Management and Information Systems: Logistics Systems and Engineering.
6. N Kratzke, R Peinl (2016) "ClouNS-a Cloud-Native Application Reference Model for Enterprise Architects," in Proceedings - IEEE International Enterprise Distributed Object Computing Workshop, EDOCW doi: 10.1109/EDOCW.2016.7584353.
7. VV Kumar, SR Yadav, FW Liou, SN Balakrishnan (2013) "A digital interface for the part designers and the fixture designers for a reconfigurable assembly system," Math. Probl. Eng doi: 10.1155/2013/943702.
8. V Kumar, VV Kumar, N Mishra, FTS Chan, B Gnanasekar (2010) "Warranty failure analysis in service supply Chain a multi-agent framework," in SCMIS 2010 - Proceedings of 2010 8th International Conference on Supply Chain Management and Information Systems: Logistics Systems and Engineering.
9. WK Hon, C Millard (2018) "Banking in the cloud: Part 1 – banks' use of cloud services," Comput. Law Secur. Rev doi: 10.1016/j.clsr.2017.11.005.
10. D Vivek, S Rakesh, RS Walimbe, A Mohanty (2020) "The Role of CLOUD in FinTech and RegTech," Ann. Dunarea Jos Univ. Galati. Fascicle I. Econ. Appl. Informatics doi: 10.35219/eai15840409130.
11. D Gannon, Barga R, Sundaresan N, Goasguen S, Gustaffson N, et al. (2017) "An Asynchronous Panel Discussion: What Are Cloud-Native Applications?," IEEE Cloud Computing doi: 10.1109/MCC.2017.4250941.
12. VV Kumar, M Tripathi, MK Pandey, MK Tiwari (2009) "Physical programming and conjoint analysis-based redundancy allocation in multistate systems: A Taguchi embedded algorithm selection and control (TAS&C) approach," Proc. Inst. Mech. Eng. Part O J. Risk Reliab 3: 215-232.
13. SKR Anumandla, VK Yarlagadda, SCR Vennapusa, KR V Kothapalli (2020) "Unveiling the Influence of Artificial Intelligence on Resource Management and Sustainable Development: A Comprehensive Investigation," Technol. \& Manag. Rev vol. 5: 45-65.
14. N Alshuqayran, N Ali, R Evans (2016) "A systematic mapping study in microservice architecture," in Proceedings - 2016 IEEE 9th International Conference on Service-Oriented Computing and Applications, SOCA doi: 10.1109/SOCA.2016.15.
15. N Poulton (2019) "The Kubernetes Book," Angew. Chemie Int. Ed 6: 951-952.
16. VV Kumar, MK. Pandey, MK Tiwari, D Ben-Arieh (2010) "Simultaneous optimization of parts and operations sequences in SSMS: A chaos embedded Taguchi particle swarm

optimization approach," J. Intell. Manuf doi: 10.1007/s10845-008-0175-4.

17. SBR Kumar, SA Rabara, JR Martin (2009) "A system model and protocol for mobile payment consortia system," in Proceedings of the International Symposium on Test and Measurement doi: 10.1109/ICTM.2009.5413011.
18. BP Vamsi Krishna Yarlagadda, Sai Sirisha Maddula, Dipakkumar Kanubhai Sachani, Kishore Mullangi, Sunil Kumar Reddy Anumandla (2020) "Unlocking Business Insights with XBRL: Leveraging Digital Tools for Financial Transparency and Efficiency," Asian Account. Audit. Adv 11: 101-116.
19. M Massoth, T Bingel (2009) "Performance of different mobile payment service concepts compared with a NFC-based solution," in Proceedings of the 2009 4th International Conference on Internet and Web Applications and Services, ICIW doi: 10.1109/ICIW.2009.112.
20. DS Islamiati, D Agata, AR Anom Besari (2019) "Design and Implementation of Various Payment System for Product Transaction in Mobile Application," in IES 2019 - International Electronics Symposium: The Role of Techno-Intelligence in Creating an Open Energy System Towards Energy Democracy, Proceedings doi: 10.1109/ELECSYM.2019.8901643.
21. FH Al-Hawari, MS Habahbeh (2020) "Secure and robust web services for e-payment of tuition fees," Int. J. Eng. Res. Technol doi: 10.37624/ijert/13.7.2020.1795-1801.
22. P Urien, X Aghina (2016) "Secure Mobile Payments Based on Cloud Services: Concepts and Experiments," in Proceedings - 2nd IEEE International Conference on Big Data Security on Cloud, IEEE BigDataSecurity 2016, 2nd IEEE International Conference on High Performance and Smart Computing, IEEE HPSC 2016 and IEEE International Conference on Intelligent Data and S doi: 10.1109/BigDataSecurity-HPSC-IDS.2016.48.
23. RP Vamsi Krishna Yarlagadda (2018) "Secure Programming with SAS: Mitigating Risks and Protecting Data Integrity," Eng. Int 6: 211-222.
24. VKY Nicholas Richardson, Rajani Pydipalli, Sai Sirisha Maddula, Sunil Kumar Reddy Anumandla (2019) "Role-Based Access Control in SAS Programming: Enhancing Security and Authorization," Int. J. Reciprocal Symmetry Theor. Phys 6: 31-42.
25. K Gos, W Zabierowski (2020) "The Comparison of Microservice and Monolithic Architecture," in International Conference on Perspective Technologies and Methods in MEMS Design doi: 10.1109/MEMSTECH49584.2020.9109514.
26. Y Shah, S. Sengupta (2020) "A survey on Classification of Cyber-attacks on IoT and IIoT devices," in 2020 11th IEEE Annual Ubiquitous Computing, Electronics and Mobile Communication Conference, UEMCON doi: 10.1109/UEMCON51285.2020.9298138.
27. MN Alenezi, H Alabdulrazzaq, AA Alshaher, MM Alkharang (2020) "Evolution of Malware Threats and Techniques: A Review," Int. J. Commun. Networks Inf. Secur doi: 10.17762/ijcnis.v12i3.4723.
28. M Conti, N Dragoni, V Lesyk (2016) "A Survey of Man in the Middle Attacks," IEEE Communications Surveys and Tutorials doi: 10.1109/COMST.2016.2548426.
29. AC Panchal, VM Khadse, PN Mahalle (2018) "Security Issues in IIoT: A Comprehensive Survey of Attacks on IIoT and Its Countermeasures," in Proceedings - 2018 IEEE Global Conference on Wireless Computing and Networking, GCWCN doi: 10.1109/GCWCN.2018.8668630.
30. Y Wang, C Hahn, K Sutrave (2016) "Mobile payment security, threats, and challenges," in Proceedings of the 2016 2nd Conference on Mobile and Secure Services, MOBISECSERV 2016 doi: 10.1109/MOBISECSERV.2016.7440226.
31. . Zhang, RH Deng, X Liu, D Zheng (2018) "Blockchain based efficient and robust fair payment for outsourcing services in cloud computing," Inf. Sci. (Ny) doi: 10.1016/j.ins.2018.06.018.
32. M Zouina, B Outtai (2019) "Towards a distributed token based payment system using blockchain technology," in Proceedings - 2019 International Conference on Advanced Communication Technologies and Networking, CommNet 2019 doi: 10.1109/COMMNET.2019.8742380.
33. SH Chun (2019) "E-commerce liability and security breaches in mobile payment for e-business sustainability," Sustain doi: 10.3390/su11030715.
34. A R Sri Nandhini (2020) "Impact of Implementing Cloud Native Applications in Replacement to on-Premise Applications," Int. J. Eng. Res doi: 10.17577/ijertv9is061021.
35. V Prokhorenko, M Ali Babar (2020) "Architectural resilience in cloud, fog and edge systems: A survey," IEEE Access doi: 10.1109/ACCESS.2020.2971007.