Journal of Artificial Intelligence and Semiconductor Integrated Hardware (JAISIH)
Volume 1 | Issue 2 |April-June 2026 | Pages 07-13
DOI: https://doi.org/10.5281/zenodo.21412754
e-ISSN: 3139-5775
Review Article OPEN ACCESS https://jaisih.org/index.php/jaisih/issue/archive

Review of Observability Driven Security Engineering in Modern Cloud Native Applications

Dr Chintal Kumar Patel

Associate Professor, CSE Geetanjali Institute of Technical Studies

chintal.patel@gits.ac.in

Corresponding Author: *Chintal Kumar Patel, (chintal.patel@gits.ac.in)

Received: 24 April, 2026 | **Revised:** 12 May 2026 | **Accepted:** 14 June 2026 |

***Abstract*— One advantage of cloud-native apps is that they form the backbone of decentralised computer networks. The characteristics of cloud-native apps are scalability, flexibility, and resilience. However, the dynamics of microservices, containers, and Kubernetes architectures create many difficulties in system monitoring and identification of its vulnerabilities. The traditional approach to system monitoring based on the use of static pre-defined rules and threshold alerts does not provide enough information and is often unable to identify anomalies or potential dangers. The aim of this paper is to present an enhanced AI-enabled cloud-native observability framework for improving system monitoring, detection of system vulnerabilities, and decision-making processes. In addition, the existing technologies and techniques for cloud-native observability, intelligent monitoring, and security such as distributed tracing, eBPF, open Telemetry, artificial intelligence, and machine learning analyzed. Comparative review of the existing technologies demonstrates their advantages, disadvantages, and application possibilities in cloud-native architecture. It shows how the intelligent approach to monitoring can improve the fault detection and optimization of resource consumption, as well as enhance security monitoring and autonomous operation of cloud. Finally, the problems faced in the current state of research discussed and future research directions outlined.**

***Keywords*—Cloud-Native, Cloud Computing, Observability, Artificial Intelligence, Machine Learning, Kubernetes, Microservices, Open Telemetry, Anomaly Detection.**

I. INTRODUCTION

The use of cloud-native apps has completely altered how businesses track system and observability metrics [1]. Monitoring application performance, identifying bottlenecks, and maintaining system stability has gotten more challenging with the change from monolithic to distributed microservices [2][3]. In modern studies, it has been shown that the microservices architecture adds a lot of operational complexity, and distributed systems can show fault propagation patterns that can propagate through several services with milliseconds of delay across service boundaries. In this technical review, the focus on the key practices and tools for successful monitoring and observability in cloud-native environments.

In today's cloud-native environment, a change in mindset from traditional monitoring to a broader observability model is required to gain real-time visibility into complex distributed systems. Monitoring flaky containers, moving service meshes and auto-scaling infrastructure while also keeping visibility into the performance of the apps and user experience is a difficult task for the organizations [4][5]. Distributed systems can produce interconnected data streams across multiple abstraction layers that are generated upon deployment, often from a variety of cloud-native sources, generating significantly more telemetry data than monolithic applications, which in turn demand sophisticated mechanisms for correlation to ensure the coherence of the system.

The economic impact of lack of observability in cloud-native scenarios is significant [6]. The findings from distributed systems research underscore the critical role of robust observability solutions in improving the reliability and efficiency of systems, using advanced monitoring systems to gain deeper insights into complex system behaviours [7]. By taking a full-stack approach to observability, organizations have seen tangible benefits in their incident response times and system stability, especially with cascading failures often found in microservices environments.

Traditional monitoring tools, however, are unable to meet the operation monitoring objectives of containerised microservices deployed over dispersed IT infrastructures. The IIoT's advent and the merging of the edge and cloud for mission-critical IIoT applications have made operators' lives more complicated [8]. The management of users, application data, tenants, nodes, and networks in various configurations—which include integrating multi-site resource islands over heterogeneous networks—requires a variety of operation monitoring tools. Monitoring metrics for bare-metal servers, VM instances, or application containers isn't enough for varied operations (Ops) teams to diagnose performance issues; they also need to understand the topological linkages between these entities [9].

Despite monitoring's long history as a fundamental IT function, a number of factors, including agile development methodologies, cloud-native deployments, and new DevOps procedures, have started to highlight its limits [10]. The need to keep an eye on all parts of the IT ecosystem, including

DOI: https://doi.org/10.5281/zenodo.21412754

systems, applications, and infrastructures, in order to react quickly to crises has changed because of these considerations [11]. One example is Kubernetes, which coordinates most edge infrastructures and serves to conceal much of the underlying complexity [12]. On the other hand, the platform itself is made up of several separate elements that work together. There are several reasons why using traditional monitoring tools doesn't work right away: some parts that affect the containerised application are hosted outside of it; the underlying environment is also more dynamic; and applications are deployed more quickly.

A. *Structure of the Paper*

The paper structure is as follows: Section II discusses observability and security in modern distributed systems. Section III describes the proposed AI-enhanced cloud-native observability framework. Section IV reviews key techniques and technologies. The relevant literature is reviewed in Section V. Lastly, the work is concluded and future research goals are outlined in Section VI.

II. Observability And Security Engineering Data In Modern Distributed Systems

The cloud-native applications that have been deployed today use microservices, containers, and orchestration frameworks in order to ensure that highly available and highly resilient services are delivered. The distributed nature of the cloud environment presents numerous difficulties when it comes to monitoring, failure detection, and providing security. With cloud-native observability, the performance of the system is monitored through metrics, logs, and distributed traces. At the same time, it is important to ensure the security of the dynamically changing workload. Thus, observability and security are key aspects of today's distributed environment [13][14].

A. *Cloud-Native Systems*

The concept of computing as an architecture using containers, microservices, Kubernetes, and orchestration for designing scalable and resilient applications [15]. Discuss that such applications tend to be very distributed and in a state of constant evolution, which makes it difficult to use existing methods for monitoring. State that cloud-native architectures need sophisticated monitoring capabilities.

B. *Cloud-Native Observability*

The capacity to gather and analyse telemetry data from distributed systems in order to analyse the internal state of cloud-native apps is known as observability in cloud-native applications [16][17]. While conventional monitoring is based on metric and alert definitions, observability offers a comprehensive understanding of application behavior. It enables organizations to quickly identify performance problems, identify errors, identify the causes of errors, and maintain reliability of systems in cloud-based environments. Observability is especially critical for microservices architecture, where many interconnected components communicate with each other through distributed systems.

1) *Metrics*

These are data points collected periodically that reveal useful insight into the well-being of an application or infrastructure. They can take many forms, including but not limited to: CPU utilisation, memory usage, request delay, error rates, network throughput, etc. These metrics allow the operator to evaluate the performance of a system, detect certain trends, and set alarms upon crossing certain thresholds.

2) *Logs*

These are records created by applications, containers, operating systems, and cloud services containing information about events that happen within the course of executing a system. Logs contain all kinds of details such as actions of users, errors and warnings, and general behavior of an application. They can be used to investigate problems, security incidents, and other things.

3) *Distributed Traces*

It involves following the entire process of request processing from one microservice to another [18]. The process is traced through different components and records information about the execution of requests and any timing-related data. Distributed tracing is especially useful in investigating cloud-native applications' interaction.

4) *Telemetry Data Collection*

The cloud native observability framework leverages telemetry data gathered from applications, containers, virtual machines [19], Kubernetes clusters, and network components. These include metrics, logs, traces, and events that get aggregated in centralized observability solutions.

5) *Cloud-Native Observability Solutions*

A number of open-source and proprietary solutions can be used for cloud-native observability. For example, metrics can be gathered using Prometheus, logs managed with Elasticsearch, Fluent, and Kibana (EFK), traces collected using Jaeger and Zipkin, and visualized with Grafana. Also, there is an open standard Open Telemetry that standardizes telemetry data collection.

6) *Advantages of Cloud-Native Observability*

Cloud-native observability ensures continuous insight into the state of health and performance of cloud-native applications. It allows detecting anomalies proactively, analyzing the root causes, reducing downtime, optimizing resources usage, improving customer experience, and making decisions more efficiently [20][21].

C. *Security in Cloud-Native Environments*

Security can be described as the processes, tools, and techniques that help secure cloud applications, containers, microservices, and the cloud environment itself from any security risks at all times. Cloud-native environments are very dynamic, distributed, and always changing; hence, they are more vulnerable to certain kinds of security risks such as configuration errors, breaches [22], container risks, and runtime risks. A successful approach to cloud-native security is incorporating security throughout the development and deployment process.

1) *Container Security*

These are lightweight and portable. But if the container images have any vulnerability or if the configurations are insecure, then the applications are exposed to cybersecurity risks. Container security includes image scanning, use of trusted images, implementation of the principle of least privilege, and continuous monitoring of container runtime to prevent unauthorized access.

DOI: https://doi.org/10.5281/zenodo.21412754

2) Kubernetes Security

This is commonly used for container orchestration purposes. Thus, Kubernetes is an essential part of cloud-native architecture. Some security practices include role-based access controls, securing the API server, use of network policies [23], namespace segregation, pod security standards, and updates to the Kubernetes cluster to minimize the attack surface .

3) Identity and Access Management (IAM)

The security feature ensures that the cloud resources may only be accessed by authorised persons, services, and applications. Methods for managing identities and access include authentication, authorisation, role-based access control, and MFA.

4) Runtime Security and Threat Detection

The main purpose of runtime security is to ensure that applications and containers are secure during their runtime. With the help of continuous monitoring of suspicious actions like privilege escalation, running malicious processes, malware, abnormal network communications, and container escapes, it becomes possible to detect and respond to threats faster and mitigate any damage.

5) Vulnerability Management

It helps to protect cloud-native applications since they use various software libraries, container images, and other third-party components which are potentially vulnerable to different types of threats. The processes involved in vulnerability management include scanning, risk assessment, patching, dependencies monitoring, and compliance verification [24][25].

6) Network Security and Policy Enforcement

In order to protect microservices and cloud resources from being accessed by unauthorized users and to avoid data breaches, network security techniques should be used. Some of them are encryption, service mesh security, firewalls, network segmentation, intrusion detection systems, and policy enforcement.

7) Continuous Security Monitoring

It involves gathering of logs, metrics, security events, and audit trails in order to detect and investigate threats in real time. Security analytics combined with observability help to identify issues and respond to them.

8) Advantages of Cloud-Native Security

Cloud-native security increases the security and safety of applications and infrastructure through enhanced confidentiality [26], integrity, and availability. It lowers the possibility of attacks, ensures that regulations are followed, facilitates quick detection and handling of incidents, helps protect confidential information, and guarantees safe operations of distributed cloud-native systems.

Cloud-native observability allows complete visibility by gathering data from metrics, logs, traces, and events. Table I highlights the comparison of traditional monitoring and cloud-native observability on the basis of various operational features.

TABLE I. COMPARISON BETWEEN TRADITIONAL MONITORING AND CLOUD-NATIVE OBSERVABILITY

Feature	Traditional Monitoring	Cloud-Native Observability
Monitoring Approach	Rule-based	Data-driven and AI-assisted
Data Sources	Metrics only	Metrics, Logs, Traces, Events
Architecture Support	Monolithic Systems	Microservices & Kubernetes
Fault Detection	Reactive	Proactive
Root Cause Analysis	Manual	Automated
Scalability	Limited	Highly Scalable
Real-Time Visibility	Partial	Comprehensive
AI Integration	Limited	Supported
Security Monitoring	Basic	Integrated
Automation	Low	High

III. AI-ENHANCED CLOUD-NATIVE OBSERVABILITY FRAMEWORK

This section presents the proposed AI-enhanced cloud-native observability framework for improving monitoring, security, and operational resilience in modern cloud-native applications [27]. It describes the overall architecture and explains the functionality of its major components, including telemetry collection, the observability platform, AI-based analytics, security monitoring, and the decision and automation engine [28].

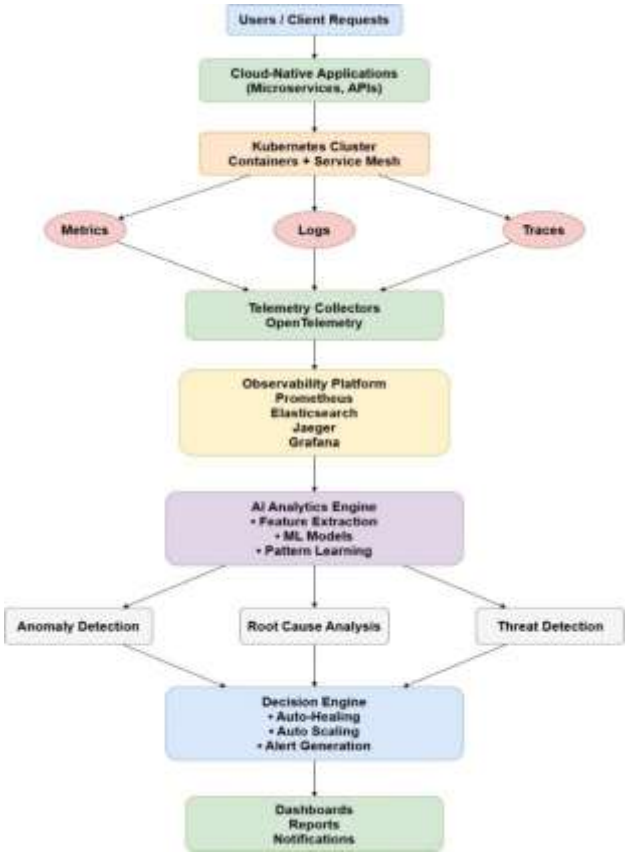

Fig. 1. AI-Enhanced Cloud-Native Observability Framework

Figure 1 illustrates the architectural workflow, demonstrating how telemetry data flows through the framework to enable anomaly detection, root cause analysis, threat detection, automated decision-making, and continuous operational monitoring for reliable and secure cloud-native environments.

DOI: https://doi.org/10.5281/zenodo.21412754

A. Observability Platform

The observability platform is the middle layer of the proposed architecture for monitoring and visualizing the telemetry data of cloud-native apps, collecting, storing, and correlating it. It combines logs, metrics and traces to offer end-to-end visibility of application performance, system health, and security events, helping to streamline troubleshooting and incident management [29]. It uses Prometheus to collect metrics, Grafana to visualize them, Jaeger for distributed tracing, Elasticsearch to store logs in a centralized place, and Kibana to analyze them. Correlated telemetry data is then delivered into the AI analytics engine to help with accurate root cause analysis, anomaly detection and security monitoring in cloud-native environments.

B. Security Monitoring Module

Cloud-native application protection is ongoing through the analysis of telemetry data such as logs, metrics, traces, and network traffic to detect security threats, vulnerabilities, and policy violations. AI-powered analysis improves threat detection and speeds up incident response by intelligently correlating observability data [30]. The module also enables runtime monitoring, vulnerability assessment, compliance verification, and risk prioritization, enhancing the security posture, resilience, and response time in cloud-native environments.

C. Decision and Automation Engine

Converting AI-generated insights into automated actions reduces human intervention and speeds up incident resolution, enabling intelligent execution of operational and security tasks [31]. The engine can operate functions like alert generation, auto-healing, resource scaling, deployment recommendations, and incident prioritization based on the telemetry data, anomaly detection, and risk assessment. This automated decision-making process ensures reliability, resource optimization, and resilience in cloud-native environments.

D. AI-Based Anomaly Detection

Anomaly detection, failure forecasting, and root cause investigation are all possible with cloud-native apps that employ intelligent telemetry data analysis. Processing both real-time and historical telemetry data, the module applies AI and ML algorithms for tasks such as feature extraction, anomaly identification, pattern learning, pre-processing, root-cause analysis, and failure prediction. Retrofitting learning improves system reliability, system security, and enables predictive maintenance based on accurate and proactive analytics [32].

The suggested architecture is built to be implemented in a methodical manner, beginning with the execution of cloud-native apps and progressing to telemetry data gathering, centralised observability, security monitoring, intelligent decision-making, and continual optimisation. The operational workflow, the technologies involved and the output produced at each stage of the proposed framework is summarized in Table II.

TABLE II. OPERATIONAL WORKFLOW

Process	Technologies Used	Generated Output
Cloud-native application execution	Containers, Microservices, Kubernetes	Runtime telemetry
Telemetry collection	Open Telemetry, Prometheus Exporters	Metrics, Logs, Traces
Data aggregation	Prometheus, Elasticsearch	Centralized telemetry repository
Data visualization	Grafana, Kibana, Jaeger	Performance dashboards
AI analytics	Machine Learning, Deep Learning	Anomaly detection, failure prediction
Security monitoring	Runtime monitoring, Threat detection	Security alerts and risk scores
Decision automation	Auto-healing, Resource scaling, Alert management	Automated operational response
Continuous optimization	Feedback learning	Improved reliability, security, and performance

IV. TECHNIQUES AND TECHNOLOGIES FOR CLOUD-NATIVE SECURITY AND OBSERVABILITY

Advanced techniques and solutions are essential for secure operation, visibility and delivery of distributed cloud-native applications in dynamic computing environments. These capabilities are realized by a mixture of security mechanisms [33], observability platforms and AI-powered monitoring solutions that enable threat detection, performance analysis, fault diagnosis and automated system management. This section summarizes the major techniques and technologies that enhance the security, resilience and operational efficiency of cloud-native structures [34].

A. Cloud-Native Security Techniques

Distributed workloads need multiple layers of defense across the application lifecycle [35]. Key security practices involve container image protection, hardening Kubernetes, IAM, RBAC, runtime protection, vulnerability assessment, network segmentation, policy enforcement, secrets management, encryption and Zero Trust security model [36][37]. All of these strategies ensure that workloads and access privileges are protected, communication between services is secured, and runtime activities are monitored. Their intelligent security analytics capabilities further enhance threat detection, incident response, and infrastructure resilience.

B. Cloud-Native Observability Technologies

Distributed services require full visibility, which is only achievable through full telemetry collection and analysis [38]. Observability is built on metrics, logs, and distributed traces that give a view of the performance of the applications as well as the utilization of resources and interactions with services. Open Telemetry, Prometheus, Grafana, Jaeger, Zipkin, and the EFK stack are technologies that facilitate telemetry data collection, storage, visualization, and trace analysis. These platforms combine to enable real-time monitoring, quick troubleshooting, performance optimization, and the creation of accurate operational data that can power cutting-edge analytics and automated management.

C. Driven Intelligent Monitoring Technologies

The capabilities of modern monitoring systems have greatly surpassed traditional threshold-based methods [39] thanks to advances in artificial intelligence. Telemetry data is continuously analyzed through Machine Learning, Deep Learning, Generative AI, anomaly detection, and predictive analytics to detect abnormal behavior, predict failures, make

DOI: https://doi.org/10.5281/zenodo.21412754

automated root cause analysis, and recommend corrective actions. These smart technologies improve the accuracy of monitoring, quicker incident response time, improve resource utilization in large cloud-native environments and make security monitoring more robust, with reduced reliance on manual actions.

D. Comparative Analysis of Existing Technologies

The security and observability solutions feature differing degrees of monitoring capability, automation, scalability, and deployment complexity [27]. Traditional monitoring technologies offer reliable telemetry collection and visualization, but are typically rule-based and depend on manual investigation. AI-powered methods, on the other hand, integrate predictive analytics, intelligent automation, and adaptive learning to provide proactive fault detection, automated root cause analysis, predictive maintenance, and improved security monitoring. While these smart solutions demand more computational power and deployment work, they significantly enhance the operational efficiency, scalability, resilience, and decision-making processes within contemporary cloud-native environments.

V. Literature Review

Recent progress towards observability-driven security for cloud-native apps has been discussed in the literature, focusing on AI-based diagnostics, DevOps, root cause analysis, and eBPF-based monitoring.

Paramasivam Murugesan and Ganesan (2026), propose a self-evolving diagnostics methodology using generative AI, which uses semantic correlation engines and generative troubleshooting models to detect anomalies in the system. The system is built on top of the observability stack with generative guidance, and learns from past failures to automatically diagnose new failures without having to update the rules manually [40].

Poojary, Kamble and Mamatha (2025), provide a fresh approach that teams may use to showcase a framework that integrates automated continuous integration and continuous delivery pipelines, infrastructure as code, observability, security, optimisation based on artificial intelligence, and multi-cloud strategies. The framework enhances the scalability, resilience, and efficiency of the systems through the use of Kubernetes pods, services, and autoscaling, helping to overcome security, performance, cost, compliance, and sustainability challenges [41].

Han et al. (2024), propose HolisticRCA, a root cause analysis framework for cloud-native systems that tackles the heterogeneity of systems with an assembling building blocks strategy. The framework maps the observability features to a common vector space and uses the Graph Attention Network to model relationships between resources for comprehensive root cause analysis [42].

Feng, Zhou and Tang (2024), promote safe operations, event-driven security, and performance observability in cloud settings with the implementation of eBPF technology. The integration of eBPF improves efficiency, supports security optimization, facilitates fault localization and troubleshooting, and strengthens security event handling across development, testing, and production [43].

Soldani et al. (2023), introduce eBPF and its applications in Telco cloud environments, highlighting network observability, runtime security, and power monitoring. Their eBPF platform, Sauron, enables real-time performance monitoring, energy estimation, and detection of unauthorized access to cloud-native resources through dynamically deployed kernel programs[44].

Usman et al. (2022), aims to solve the problems with cloud computing by relocating processing power to the network's periphery, where it is closer to the consumers. Conversely, when containerised workloads are hosted as microservices, distributed edge infrastructures become more complicated, which in turn makes it more difficult to detect and fix failures for critical use cases such as industrial automation activities. Operators may better manage and run distributed infrastructures and microservices with the support of observability, which tracks performance in real-time [45].

Table III summarizes the reviewed studies by comparing their focus areas, techniques, key findings, limitations, and future research directions.

TABLE III. Summary of Related Work on Cloud-Native Security and Observability

Reference	Focus Area	Technologies/Techniques	Key Findings	Limitations	Future Work
Paramasivam Murugesan & Ganesan (2026)	AI-driven self-evolving diagnostics for observability	Generative AI, semantic correlation, troubleshooting models	Improved automated anomaly diagnosis through self-evolving diagnostics.	Limited validation in large-scale cloud-native environments.	Investigate scalability and real-time deployment in production systems.
Poojary, Kamble & Mamatha (2025)	Modern DevOps and cloud-native CI/CD	CI/CD, Kubernetes, IaC, AI optimization, observability	Enhanced scalability, resilience, and operational efficiency	Limited evaluation of runtime security effectiveness	Develop adaptive security and intelligent observability mechanisms
Han et al. (2024)	Root Cause Analysis (RCA) in cloud-native systems	HolisticRCA, Graph Attention Network, vector embeddings	Improved root cause identification in cloud-native systems	High computational complexity for dynamic environments	Optimize real-time RCA for large-scale distributed applications
Feng, Zhou & Tang (2024)	Secure cloud operation and performance monitoring	eBPF, event-driven monitoring, fault localization	Strengthened security monitoring and performance observability.	Focused mainly on transaction systems	Extend eBPF support to diverse cloud-native workloads
Soldani et al. (2023)	eBPF for cloud-native observability and security	eBPF, runtime security, network observability, Sauron platform, energy monitoring	Enabled real-time monitoring, energy estimation, and runtime security	Limited validation across heterogeneous cloud platforms	Improve interoperability and cross-platform deployment

DOI: https://doi.org/10.5281/zenodo.21412754

Usman et al. (2022)	Edge computing observability	Microservices, edge computing, runtime instrumentation	Enhanced monitoring of distributed infrastructures	Limited emphasis on integrated security mechanisms	Develop unified observability and security frameworks for edge-cloud systems

VI. Conclusion and Future Work

AI-enabled observability system for intelligent monitoring and security of cloud-native applications. In this paper, have proposed an architecture of AI-based observability system, which can be used for monitoring and analysis of cloud-native applications. This framework uses cloud-native technologies along with AI in order to provide centralized observability, intelligent detection of anomalies, automated root cause analysis, prediction of failure, and intelligent security monitoring. Through use of telemetry data collected from logging, metrics, and tracing data, the framework ensures real-time visibility, proactive incident handling, better resource management, and decreased operational complexity. As a whole, the architecture provides a scalable, secure, and efficient solution for management of cloud-native applications.

Future research work would include development and validation of the proposed framework in practical cloud-native scenarios. Further improvements can be achieved through implementation of advanced AI methods such as LLMs, Generative AI, XAI, and Reinforcement Learning for enhancing anomaly detection and automated decision making. Also, integration of Zero Trust security, AIOps, and multi-cloud capabilities will make the framework more intelligent, automated, and secure.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data supporting the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest: The author declares no conflict of interest.

How to cite this article

C. K. Patel, "Review of Observability Driven Security Engineering in Modern Cloud Native Applications," *Journal of Artificial Intelligence and Semiconductor Integrated Hardware*, vol. 1, no. 2, pp. 07-13 Apr.–Jun. 2026, Doi: 10.5281/zenodo.21412754

References

[1] J. B. Mehta, "Ai-Driven Test Engineering For Cloud-Native Systems," *Int. J. Data Sci. IoT Manag. Syst.*, vol. 5, no. 1, Jan. 2026, doi: 10.64751/ijdim.2026.v5i1.297.

[2] A. Mahida, "Enhancing Observability in Distributed Systems-A Comprehensiv Review," *J. Math. Comput. Appl.*, vol. 2, no. 3, p. 1, Sep. 2023, doi: 10.47363/JMCA/2023(2)135.

[3] S. R. Chanthati, "Architecting Next-Gen Financial Systems with AI and Cloud-Native Microservice," in *Proceedings of the 4th IEEE World Conference on Applied Intelligence and Computing (AIC-2025)*, Jul. 2025. doi: 10.5281/zenodo.20281424.

[4] P. Kale, "Architecting Autonomous Testing Pipelines for Cloud-Native Systems Using AI-Driven Fault Injection and Predictive Analytics," *Int. J. AI, BigData, Comput. Manag. Stud.*, vol. 6, pp. 207–216, 2025, doi: 10.63282/3050-9416.IJAIBDCMS-V6I1P121.

[5] J. E. Kofi, "Monitoring Cloud Performance Metrics Utilizing AI to Estimate the Efficiency of Cloud Operations," in *2025 7th International Symposium on Advanced Electrical and Communication Technologies (ISAECT)*, IEEE, Dec. 2025, pp. 1–6. doi: 10.1109/ISAECT68904.2025.11318827.

[6] S. R. Chanthati, "Architecting-Next-Gen-Financial-Systems-with-AI-and-Cloud-Native-Microservices," Zenodo.

[7] M. Wurster, U. Breitenbücher, A. Brogi, F. Leymann, J. Soldani, and others, "Cloud-native Deploy-ability: An Analysis of Required Features of Deployment Technologies to Deploy Arbitrary Cloud-native Applications.," in *CLOSER*, 2020, pp. 171–180.

[8] P. H. Desai, P. Mahalle, and P. Chandre, "Intelligent Access Control Schemes for the Internet of Everything: A Survey of Techniques, Challenges, and Future Directions," in *Information Systems for Intelligent Systems*, 2026, pp. 95–105. doi: 10.1007/978-3-032-13196-6_9.

[9] I. Kohyarnejadfard, D. Aloise, M. R. Dagenais, and M. Shakeri, "A Framework for Detecting System Performance Anomalies Using Tracing Data Analysis," *Entropy*, vol. 23, no. 8, p. 1011, Aug. 2021, doi: 10.3390/e23081011.

[10] R. K. Kanneganti, "Security and Compliance Issues in Cloud-Based Deployments of Content and Workflow Management Systems," *Eastasouth J. Inf. Syst. Comput. Sci.*, vol. 02, no. 01, pp. 131–138, Aug, 2024, doi: https://doi.org/10.58812/esiscs.v2i01.1122.

[11] J. Shah and D. Dubaria, "Building Modern Clouds: Using Docker, Kubernetes & Google Cloud Platform," in *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, IEEE, Jan. 2019, pp. 0184–0189. doi: 10.1109/CCWC.2019.8666479.

[12] P. Naayini and S. Kamatala, "Automating Infrastructure Platforms with Cloud , Kubernetes , and Site Reliability Engineering," *Int. J. Comput. Tech.*, vol. 8, no. 6, pp. 1–9, 2021, doi: 10.5281/zenodo.15634257.

[13] T. I. Okpong, "Data Flow Modeling Framework for Cloud-Native Systems Engineering: An Intelligent Approach for Microservices and Secure Data Pipelines," *J. Appl. Phys. Sci. Int.*, vol. 18, no. 2, pp. 1–18, Jul. 2026, doi: 10.56557/japsi/2026/v18i210796.

[14] H. P. Cyril and S. Kumara, "DevSecOps-Driven Security Integration in the Software Development Lifecycle Using CI/CD Pipelines," in *2026 IEEE 5th International Conference on AI in Cybersecurity (ICAIC)*, IEEE, Feb. 2026, pp. 1–6. doi: 10.1109/ICAIC67076.2026.11395737.

[15] M. C. Borges, J. Bauer, and S. Werner, "OXN - Automated Observability Assessments for Cloud-Native Applications," in *2024 IEEE 21st International Conference on Software Architecture Companion (ICSA-C)*, IEEE, Jun. 2024, pp. 167–170. doi: 10.1109/ICSA-C63560.2024.00035.

[16] N. Kratzke, "Cloud-Native Observability: The Many-Faceted Benefits of Structured and Unified Logging—A Multi-Case Study," *Futur. Internet*, vol. 14, no. 10, p. 274, Sep. 2022, doi: 10.3390/fi14100274.

[17] K. K. Mohammed, "Leadership Practices of Data Engineering for AI and Machine Learning," *Int. J. Sci. Res. Eng. Trends*, vol. 12, no. 1, 2026, doi: 10.5281/zenodo.18677279.

[18] U. Faseeha, H. Jamil Syed, F. Samad, S. Zehra, and H. Ahmed, "Observability in Microservices: An In-Depth Exploration of Frameworks, Challenges, and Deployment Paradigms," *IEEE Access*, vol. 13, pp. 72011–72039, 2025, doi: 10.1109/ACCESS.2025.3562125.

[19] S. D. Veeravalli, S. B. Venkata, A. Ashirova, R. MS, S. A. D. Poovaiah, and D. Turaev, "Breaking the Swarm Paradigm: Single Candidate Optimization for Cloud Task Allocation," in *2025 International Conference on Innovations in Intelligent Systems: Advancements in Computing, Communication, and Cybersecurity (ISAC3)*, Bhubaneswar, India: IEEE, 2025, pp. 1–7, July.

[20] S. Karumuri, F. Solleza, S. Zdonik, and N. Tatbul, "Towards Observability Data Management at Scale," *ACM SIGMOD Rec.*, vol. 49, no. 4, pp. 18–23, Mar. 2021, doi: 10.1145/3456859.3456863.

[21] H. Ravilla, "Predictive Analytics for Customer Churn in Salesforce Service Cloud," in *Learning and Analytics in Intelligent Systems*, 2026, pp. 14–24. doi: 10.1007/978-3-032-05377-0_2.

[22] T. Theodoropoulos *et al.*, "Security in Cloud-Native Services: A Survey," *J. Cybersecurity Priv.*, vol. 3, no. 4, pp. 758–793, 2023, doi: 10.3390/jcp3040034.

[23] K. Chen, H. Lu, Y. Yao, B. Fang, Y. Liu, and Z. Tian, "Enhancing Container Security Through Phase-Based System Call Filtering," *IEEE Trans. Cloud Comput.*, vol. 13, no. 3, pp. 983–994, Jul. 2025, doi: 10.1109/TCC.2025.3583414.

[24] M. C. Borges and S. Werner, "Continuous Observability Assurance in Cloud-Native Applications," in *2025 IEEE 22nd International Conference on Software Architecture Companion (ICSA-C)*, IEEE, Mar. 2025, pp. 182–185. doi: 10.1109/ICSA-C65153.2025.00036.

[25] S. Tarakampet, R. Koilakonda, and S. Tatavarthi, "From Vision to Victory: Best Practices for Architecting Enterprise-Wide Compliance Ecosystems," *Int. J. Comput. Appl.*, vol. 187, no. 87, pp. 31–37, 2026.

[26] S. I. Abbas and A. Garg, "AIOps in DevOps: Leveraging Artificial Intelligence for Operations and Monitoring," in *2024 3rd International Conference on Sentiment Analysis and Deep Learning (ICSADL)*, IEEE, Mar. 2024, pp. 64–70. doi: 10.1109/ICSADL61749.2024.00016.

[27] S. Deng *et al.*, "Cloud-Native Computing: A Survey from the Perspective of Services," Jun. 2023. doi: 10.36227/techrxiv.23500383.

[28] W. Pourmajidi, L. Zhang, J. Steinbacher, T. Erwin, and A. Miranskyy, "A Reference Architecture for Governance of Cloud Native Applications," *IEEE Trans. Cloud Comput.*, vol. 13, no. 3, pp. 935–952, Jul. 2025, doi: 10.1109/TCC.2025.3578557.

[29] A. Nikhare, "Performance Analysis and Benchmarking of Cloud-Native Observability Frameworks: A Comparative Study of OpenTelemetry and Commercial Solutions," 2025.

[30] R. Chandramouli, "Implementation of devsecops for a microservices-based application with service mesh," *NIST Spec. Publ.*, 2022.

[31] S. Tatineni, "AIOps in Cloud-native DevOps: IT Operations Management with Artificial Intelligence," *J. Artif. Intell. Cloud Comput.*, vol. 2, pp. 1–7, 2023, doi: 10.47363/JAICC/2023(2)154.

[32] A. M. Abdallah, A. Saif Rashed Obaid Alkaabi, G. Bark Nasser Douman Alameri, S. H. Rafique, N. S. Musa, and T. Murugan, "Cloud Network Anomaly Detection Using Machine and Deep Learning Techniques— Recent Research Advancements," *IEEE Access*, vol. 12, pp. 56749–56773, 2024, doi: 10.1109/ACCESS.2024.3390844.

[33] Z. Na, W. Liu, and K. Li, "Implementation of Cloud Component for Security Monitoring and Comprehensive Guarantee of Identifier Resolution System," in *2022 3rd Information Communication Technologies Conference (ICTC)*, IEEE, May 2022, pp. 167–172. doi: 10.1109/ICTC55111.2022.9778775.

[34] D. Pathak, M. Verma, A. Chakraborty, and H. Kumar, "Self Adjusting Log Observability for Cloud Native Applications," in *2024 IEEE 17th International Conference on Cloud Computing (CLOUD)*, IEEE, Jul. 2024, pp. 482–493. doi: 10.1109/CLOUD62652.2024.00061.

[35] J. E. Kofi, "Data-Driven Cloud Workload Optimization Using Machine Learning Modeling for Proactive Resource Management," *Int. J. Artif. Intell. Data Sci. Mach. Learn.*, vol. 6, no. 4, pp. 27–37, 2025, doi: 10.63282/3050-922x.ijeret-v6i4p104.

[36] A. P. Perumal, "Cloud-Native Architecture Observability and Compliance Challenges: A Comprehensive Reference Architecture Approach," *Libr. Prog.*, vol. 44, pp. 25718–25723, 2024.

[37] J. B. Mehta, "Autonomous Patch Validation For Zero-Day Exploits In Enterprise Clouds," *Int. J. Appl. Math.*, vol. 38, no. 4s, pp. 1270–1285, Oct. 2025, doi: 10.12732/ijam.v38i4s.685.

[38] P. R. Seknametla, "Observability Strategies for Multi-Cloud DevOps Deployments: Challenges in Unified Monitoring and Telemetry Aggregation Pruthvi Raj Seknametla Independent Researcher," *Int. J. Comput. Sci. Eng. Tech.*, vol. 8, p. 7, 2024, doi: 10.5281/zenodo.20560048.

[39] Z. Yu *et al.*, "MonitorAssistant: Simplifying Cloud Service Monitoring via Large Language Models," in *Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering*, New York, NY, USA, NY, USA: ACM, Jul. 2024, pp. 38–49. doi: 10.1145/3663529.3663826.

[40] S. B. Paramasivam Murugesan and D. Ganesan, "AI-Native Observability and Self-Evolving Diagnostics: Generative Troubleshooting, Semantic Correlation, and Embedded Guidance for Large-Scale Cloud Systems," in *2026 6th International Conference on Image Processing and Capsule Networks (ICIPCN)*, IEEE, Jan. 2026, pp. 1167–1172. doi: 10.1109/ICIPCN67432.2026.11439026.

[41] S. S. Poojary, S. Kamble, and G. S. Mamatha, "A Scalable DevOps Framework Using Kubernetes-Orchestrated Microservices for Cloud-Native Infrastructure," in *2025 9th International Conference on Computational System and Information Technology for Sustainable Solutions (CSITSS)*, IEEE, Nov. 2025, pp. 1–6. doi: 10.1109/CSITSS67709.2025.11294211.

[42] Y. Han *et al.*, "Holistic Root Cause Analysis for Failures in Cloud-Native Systems Through Observability Data," *IEEE Trans. Serv. Comput.*, vol. 17, no. 6, pp. 3789–3802, Nov. 2024, doi: 10.1109/TSC.2024.3478759.

[43] M. Feng, J. Zhou, and Y. Tang, "Enhancing Cloud-Native Security Through eBPF Technology," in *2024 IEEE 11th International Conference on Cyber Security and Cloud Computing (CSCloud)*, IEEE, Jun. 2024, pp. 165–168. doi: 10.1109/CSCloud62866.2024.00036.

[44] D. Soldani *et al.*, "eBPF: A New Approach to Cloud-Native Observability, Networking and Security for Current (5G) and Future Mobile Networks (6G and Beyond)," *IEEE Access*, vol. 11, pp. 57174–57202, 2023, doi: 10.1109/ACCESS.2023.3281480.

[45] M. Usman, S. Ferlin, A. Brunstrom, and J. Taheri, "A Survey on Observability of Distributed Edge & Container-Based Microservices," *IEEE Access*, vol. 10, pp. 86904–86919, 2022, doi: 10.1109/ACCESS.2022.3193102.