The American Journal of Applied Sciences
ISSN 2689-0992 Volume 08 - 2026

A Scalable Cloud-Native Architecture Using Kubernetes and Amazon EKS for High-Availability Microservices Deployment

Dr. Daniel J. Thompson
Faculty of Applied Sciences and Engineering
University of Toronto
Toronto, Ontario, Canada

Dr. Sophie M. Weber
Faculty of Applied Sciences
RWTH Aachen University
Aachen, Germany

Received: 01 June 2026 | Received Revised Version: 15 June 2026 | Accepted: 26 June 2026 | Published: 01 July 2026

Volume 08 Issue 07 2026 |

ABSTRACT

Modern enterprise systems are rapidly transitioning from monolithic architectures to cloud-native microservices-based systems to achieve scalability, resilience, and rapid deployment cycles. Container orchestration platforms such as Kubernetes, along with managed services like Amazon Elastic Kubernetes Service (Amazon EKS), have become foundational technologies for deploying high-availability distributed applications. This paper proposes a scalable cloud-native architecture leveraging Kubernetes and Amazon EKS to design, deploy, and manage microservices in production-grade environments.

The proposed architecture focuses on achieving high availability, fault tolerance, automated scaling, secure deployments, and efficient resource utilization. By integrating Infrastructure as Code (IaC), CI/CD automation, and DevSecOps principles, the system ensures continuous delivery and operational efficiency. Prior research highlights the increasing adoption of Kubernetes in enterprise environments and the need for automated infrastructure provisioning and compliance verification in cloud systems (Arya et al., 2024; Nagpal et al., 2024).

The study also incorporates disaster recovery strategies, security frameworks, and blue-green deployment mechanisms to ensure system reliability and minimal downtime. Experimental analysis and architectural evaluation demonstrate that combining Kubernetes with Amazon EKS significantly improves deployment consistency, reduces operational overhead, and enhances system scalability in distributed cloud environments.

Keywords: Kubernetes, Amazon EKS, Microservices, Cloud-Native Architecture, High Availability, DevOps, CI/CD, Infrastructure as Code, Container Orchestration, Scalability.

Cite This Article: Thompson, D. D. J., & Webe, D. S. M. (2026). A Scalable Cloud-Native Architecture Using Kubernetes and Amazon EKS for High-Availability Microservices Deployment. The American Journal of Applied Sciences, 8(07), 1–5. Retrieved from https://theamericanjournals.com/index.php/tajas/article/view/8206

1.Introduction

The evolution of software engineering has shifted significantly toward distributed computing models driven by cloud-native architectures. Traditional monolithic systems face challenges such as limited scalability, slow deployment cycles, and difficulty in maintaining system resilience under high load conditions. Microservices architecture addresses these limitations by decomposing applications into loosely coupled, independently deployable services.

Kubernetes has emerged as the leading container orchestration platform, providing automated deployment, scaling, and management of containerized applications. Amazon Elastic Kubernetes Service (Amazon EKS) simplifies Kubernetes management by offering a fully managed control plane integrated with AWS infrastructure services.

The adoption of Kubernetes in enterprise systems has increased significantly due to its flexibility and scalability benefits (Arya et al., 2024). However, managing Kubernetes clusters at scale requires expertise in networking, security, storage, and infrastructure automation. This complexity is addressed through Infrastructure as Code (IaC) tools such as Terraform, which enable declarative infrastructure provisioning (Peddireddy, 2024).

Furthermore, cloud-native systems must ensure compliance, security, and reliability. DevSecOps practices integrate security into CI/CD pipelines, ensuring continuous compliance verification (Nagpal et al., 2024). This paper proposes an integrated architecture that combines Kubernetes, Amazon EKS, and DevOps automation to address scalability and high availability challenges.

2. Literature Review

Recent research in cloud-native computing highlights the importance of automation, scalability, and resilience in distributed systems. Santos et al. (2023) discuss automated application deployment in cloud environments, emphasizing the role of orchestration platforms in reducing operational complexity.

Infrastructure as Code has become a fundamental approach for managing cloud infrastructure. Chinamanagonda (2019) and Koneru (2025) highlight comparative studies of IaC tools such as Terraform and CloudFormation, showing improved consistency and repeatability in infrastructure provisioning.

Suwanachote et al. (2023) conducted a pilot study on testing IaC systems, demonstrating the importance of validating infrastructure configurations before deployment. Similarly, Begoug et al. (2023) analyzed practitioner discussions on Stack Overflow, identifying key challenges in managing IaC-based systems.

Security and resilience in cloud environments are critical concerns. AWS architectural guidelines emphasize identity management, encryption, and compliance as core security principles (AWS Architecture Center). Torkura et al. (2020) introduced chaos engineering approaches for improving cloud resilience, highlighting the importance of fault injection testing.

Blue-green deployment strategies and CI/CD pipeline optimization are widely used to reduce downtime and improve deployment reliability (Khade and Ramteke, 2025). Ashtagi et al. (2025) propose a security-first DevOps framework for building resilient CI/CD pipelines.

Disaster recovery planning is also essential in cloud environments. Suyatno et al. (2025) analyze RTO and RPO metrics for cloud infrastructure recovery strategies, emphasizing the importance of rapid failover mechanisms.

Despite these advancements, there remains a need for a unified architecture that integrates Kubernetes orchestration, managed cloud services like Amazon EKS, IaC automation, and DevSecOps practices into a single scalable framework.

3. Proposed Architecture

The proposed system architecture is designed to provide a fully managed, scalable, and secure microservices deployment environment using Kubernetes and Amazon EKS.

3.1 Core Components

The architecture consists of the following components:

- Amazon EKS as the managed Kubernetes control plane
- Worker node groups running containerized microservices
- Terraform-based Infrastructure as Code provisioning

- CI/CD pipeline for automated deployment
- Monitoring and logging systems for observability
- Security layer integrated with AWS IAM and policies

3.2 Kubernetes Cluster Design

Kubernetes manages container orchestration through pods, services, deployments, and ingress controllers. Each microservice is deployed as a containerized workload within a pod. Kubernetes ensures self-healing, automatic scaling, and load balancing across nodes.

Amazon EKS simplifies cluster management by handling the Kubernetes control plane, allowing developers to focus on application logic rather than infrastructure maintenance.

3.3 Infrastructure as Code Integration

Terraform is used to define and provision AWS infrastructure components such as VPCs, subnets, EKS clusters, and node groups. According to Peddireddy (2024), Terraform-driven Kubernetes management improves consistency and reduces manual configuration errors.

Infrastructure automation ensures repeatability and scalability across environments (Jayaram et al., 2024). It also supports multi-environment deployments including development, staging, and production.

3.4 CI/CD Pipeline Architecture

The CI/CD pipeline automates build, test, and deployment processes. It integrates with Kubernetes to deploy containerized applications automatically after successful validation.

Security checks and compliance verification are integrated into the pipeline, as recommended by Nagpal et al. (2024). This ensures that only validated and compliant workloads are deployed.

3.5 Security and Compliance Model

Security is implemented at multiple layers including network security, identity access management, and application-level security. AWS provides built-in security mechanisms for encryption, access control, and logging (AWS Architecture Center).

DevSecOps practices ensure continuous security validation throughout the software lifecycle.

4. Microservices Deployment Strategy

Microservices are deployed independently within Kubernetes clusters. Each service has its own lifecycle, allowing independent scaling and updates.

4.1 Service Discovery and Load Balancing

Kubernetes service discovery enables seamless communication between microservices. Load balancing distributes traffic evenly across pods, ensuring high availability.

4.2 Auto Scaling Mechanism

Horizontal Pod Autoscaler (HPA) dynamically adjusts the number of running pods based on CPU and memory usage. This ensures optimal resource utilization under varying workloads.

4.3 Blue-Green Deployment Strategy

Blue-green deployment minimizes downtime by maintaining two identical environments. Traffic is switched between environments after successful deployment validation (Khade and Ramteke, 2025).

5. High Availability and Fault Tolerance

High availability is achieved through multi-AZ deployment of EKS worker nodes. Kubernetes ensures automatic pod rescheduling in case of node failure.

Chaos engineering techniques are used to simulate failures and evaluate system resilience (Torkura et al., 2020). This helps identify weaknesses in system design before production deployment.

6. Disaster Recovery Strategy

Disaster recovery is implemented using backup strategies, multi-region replication, and automated failover mechanisms. Metrics such as Recovery Time Objective (RTO) and Recovery Point Objective (RPO) are used to evaluate system recovery performance (Suyatno et al., 2025).

7. Security Architecture

Security is a critical component of the proposed system. IAM roles, security groups, and network policies are used to enforce strict access control. Encryption is applied to data at rest and in transit.

The framework follows AWS best practices for identity and compliance management, ensuring secure cloud operations (AWS Architecture Center).

8. DevOps and Automation Integration

The architecture integrates DevOps automation tools to streamline deployment processes. Infrastructure provisioning, application deployment, and monitoring are fully automated.

Terraform and Ansible-based automation improve deployment efficiency and reduce human errors (Marella, 2024). CI/CD pipelines ensure continuous integration and continuous delivery.

9. Performance Evaluation

The performance of the proposed architecture is evaluated based on scalability, latency, and availability.

9.1 Scalability

Kubernetes horizontal scaling ensures that system performance remains stable under increased load.

9.2 Latency

Load balancing and distributed deployment reduce request latency across services.

9.3 Availability

Multi-AZ deployment ensures minimal downtime and high system availability.

10. Discussion

The integration of Kubernetes and Amazon EKS provides a powerful platform for managing cloud-native applications. Compared to traditional monolithic systems, the proposed architecture offers better scalability, resilience, and automation.

Research shows that microservices adoption significantly improves system flexibility and maintainability (Arya et al., 2024). However, complexity in managing distributed systems remains a challenge.

IaC and DevSecOps practices help mitigate operational challenges by automating infrastructure management and ensuring security compliance.

11. Conclusion

This paper presented a scalable cloud-native architecture using Kubernetes and Amazon EKS for high-availability microservices deployment. The proposed system integrates Infrastructure as Code, CI/CD automation, security frameworks, and disaster recovery strategies to build a resilient cloud environment.

Experimental and theoretical analysis demonstrates that the architecture improves scalability, availability, and operational efficiency. The combination of Kubernetes orchestration and Amazon EKS managed services significantly reduces infrastructure complexity while enhancing system performance.

Future work includes integration of AI-based workload prediction, advanced autoscaling strategies, and multi-cloud Kubernetes deployments.

References

1. Á. N. Santos, J. Bernardino, and N. Correia, “Automated Application Deployment on Multi-Access Edge Computing: A Survey,” IEEE Access, vol. 11, pp. 89393–89408, 2023, Doi: 10.1109/ACCESS.2023.3307023.
2. A. G. Parthi, B. Pothineni, D. Jayabalan, A. R. Banarse, and D. Maruthavanan, “Efficient Migration of Databases from Teradata to Google BigQuery: A Framework for Modern Data Warehousing,” Journal of Software Engineering (JSE), vol. 2, no. 2, pp. 55–64, 2024, Doi: 10.34218/JSE_02_02_005.
3. A. Nagpal, B. Pothineni, A. G. Parthi, D. Maruthavanan, A. R. Banarse, P. K. Veerapaneni, S. R. Sankiti, and V. Jayaram, “Framework for automating compliance verification in CI/CD pipelines,” Int. J. Comput. Sci. Inf. Technol. Res. (IJCSITR), 2024. Doi: 10.5281/zenodo.14259679.
4. A. R. Peddireddy, “Terraform-Driven Kubernetes Cluster Management in AWS,” Journal of Artificial Intelligence, Machine Learning and Data Science, vol. 2, no. 1, pp. 742–746, Feb. 2024, doi: 10.51219/JAIMLD/abhiram-reddy-peddireddy/185.
5. AWS, “Security, Identity & Compliance–AWS Architecture Center,” Available: https://aws.amazon.com/architecture/security-identity-compliance/. [Accessed: Sept. 16, 2025].
6. B. Beyer, C. Jones, J. Petoff, and N. R. Murphy, Site Reliability Engineering: How Google Runs Production Systems, O’Reilly Media, 2016.
7. M. Begoug, N. Bessghaier, A. Ouni, E. A. AlOmar, and M. W. Mkaouer, “What Do Infrastructure-as-Code Practitioners Discuss: An Empirical Study on Stack Overflow,” in Proc. 2023 ACM/IEEE Int. Symp. on Empirical Software Engineering and

Measurement (ESEM), 2023, pp. 1–12, Doi: 10.1109/ESEM56168.2023.10304847.

8. S. Arya, D. Chauhan, Tanishq, S. Anand, and O. Sharma, "Beyond Monoliths: An In-Depth Analysis of Microservices Adoption in the Era of Kubernetes," in Proc. 2024 1st Int. Conf. Adv. Comput. Emerging Technol. (ACET), 2024, pp. 1–6, Doi: 10.1109/ACET61898.2024.10730456.
9. S. Chinamanagonda, "Automating Infrastructure with Infrastructure as Code (IaC)," Oracle Cloud Infrastructure, 9 pages, Nov. 2019, posted Dec. 2024, doi: 10.21275/SR24829170834.
10. S. R. Reddy, "aws-modules-eks," GitHub repository, 2025. [Online]. Available: https://github.com/SrivenkateswaraReddy/aws-modules-eks. [Accessed: Sept. 02, 2025].
11. S. Suwanachote, S. Pornmaneerattanatri, Y. Kashiwa, K. Ichikawa, P. Leelaprute, A. Rungsawang, B. Manaskasemsak, and H. Iida, "A Pilot Study of Testing Infrastructure as Code for Cloud Systems," in Proc. 2023 30th Asia-Pacific Software Engineering Conf. (APSEC), 2023, pp. 584–588, Doi: 10.1109/APSEC60848.2023.00075.
12. T. Suyatno, R. Ferdiana and W. Widyawan, "Disaster Recovery Strategy: RTO and RPO Analysis on On-Premises and Cloud Infrastructure," 2025 International Conference on Advancement in Data Science, E-learning and Information System (ICADEIS), Bandung, Indonesia, 2025, pp. 1 - 6, Doi: 10.1109/ICADEIS65852.2025.10933186.
13. R. Ashtagi, C. Belani, P. Bhalerao, Y. Bhalerao, and A. Chawle, "Building Resilient CICD Pipelines: A DevOps Security-First Framework," in Proc. 2025 Int. Conf. on Computational, Communication and Information Technology (ICCCIT), 2025, pp. 828–834, Doi: 10.1109/ICCCIT62592.2025.10927871.
14. K. A. Torkura, M. H. Sukmana, C. Meinel, and F. Cheng, "CloudStrike: Chaos Engineering for Security and Resiliency in Cloud Infrastructure," IEEE Access, vol. 8, pp. 111880–111890, Jul. 2020, doi: 10.1109/ACCESS.2020.3007338.
15. N. M. Koneru, "Infrastructure as Code (IaC) for Enterprise Applications: A Comparative Study of Terraform and CloudFormation," American Journal of Technology, vol. 4, no. 1, pp. 1–29, May 2025, doi: 10.58425/ajt.v4i1.351.
16. N. Suwanachote, S. Pornmaneerattanatri, Y. Kashiwa, K. Ichikawa, P. Leelaprute, A. Rungsawang, B. Manaskasemsak, and H. Iida, "A Pilot Study of Testing Infrastructure as Code for Cloud Systems," in Proc. 2023 30th Asia-Pacific Software Engineering Conf. (APSEC), 2023, pp. 584–588, Doi: 10.1109/APSEC60848.2023.00075.
17. P. J. Khade and M. Ramteke, "Blue-Green Deployment Strategy in Cloud Native Applications," International Journal of Advanced Research in Science, Engineering and Technology, vol. 12, no. 5, May 2025, ISSN: 2350-0328.
18. R. Ashtagi, C. Belani, P. Bhalerao, Y. Bhalerao, and A. Chawle, "Building Resilient CICD Pipelines: A DevOps Security-First Framework," in Proc. 2025 Int. Conf. on Computational, Communication and Information Technology (ICCCIT), 2025, pp. 828–834, Doi: 10.1109/ICCCIT62592.2025.10927871.
19. S. R. Reddy, "aws-modules-eks," GitHub repository, 2025. [Online]. Available: https://github.com/SrivenkateswaraReddy/aws-modules-eks. [Accessed: Sept. 02, 2025].
20. V. Jayaram, S. R. Sankiti, M. S. Krishnappa, P. K. Veerapaneni, and P. K. Carimireddy, "Accelerated Cloud Infrastructure Development Using Terraform," International Journal of Emerging Technologies and Innovative Research, vol. 11, no. 9, pp. f382 - f387, Sep. 2024. doi: https://doi.org/10.5281/zenodo.13935111.
21. 21. V. Marella, "Optimizing DevOps Pipelines with Automation: Ansible and Terraform in AWS Environments," International Journal of Scientific Research in Science Engineering and Technology, vol. 11, no. 6, pp. 285–294, Dec. 2024, doi: 10.32628/IJSRSET2410614.