Using Agenticai With Kubernetes For Faster Development, Deployment and Delivery in Production Environments.

[1] Hardeep Singh Tiwana
[1] Enterprise Support, Amazon Web Services, USA

Received: 02th Nov 2025 | Received Revised Version: 12th Nov 2025 | Accepted: 28th Nov 2025 | Published: 24th Dec 2025

Volume 07 Issue 12 2025 | Crossref DOI: 10.37547/tajet/Volume07Issue12-07

Abstract

AgenticAIs makes autonomous decisions, optimize, and organize workflows on their own are becoming a disruptive layer in cloud-native software delivery. Karpenter collaborated with Kubernetes to enable intelligent node provisioning, simplify resource allocation, and provide production-grade reliability at scale. As explored in this paper, Agentic AI enhances the following Kubernetes-based DevOps processes: optimizing CI/CD orchestration, forecasting resource demand, accelerating fault recovery, and improving application lifecycle management. Continuous monitoring and control of cluster behavior, advanced diagnostics, and targeted corrective actions are all aspects of the Integrated Agentic AI models that makes great independent operations with limited human control a possibility. The paper presents key technical underpinnings, including multi-agent systems (for example MCP server), reinforcement learning, operator-based AI control loops, and AI-based policy enforcement. Practical examples are examined, such as automated scaling, self-healing clusters, smart canary rollouts, drift detection, and cost-constrained resource allocation. Issues such as model reliability, governance, interpretability, and production-grade security are addressed with mitigating solutions. Combining existing practices and fresh innovations, this article can serve as a comprehensive guide for leaders in the engineering community, DevOps teams, and platform designers who want to use Agentic AI in Kubernetes-driven environments. The insights highlight how automated intelligence can greatly reduce development cycles, minimize operational friction, and enable continuous, dependable delivery in the evolving, dynamic ecosystem of production.

Keywords: Agentic AI, Kubernetes, Autonomous DevOps, AI-Driven Deployment, Cloud-Native Engineering, MCP server, Kapenter.

Cite This Article: Hardeep Singh Tiwana. (2025). Using Agenticai With Kubernetes For Faster Development, Deployment and Delivery in Production Environments. The American Journal of Engineering and Technology, 7(12), 73–82. https://doi.org/10.37547/tajet/Volume07Issue12-07

1. Introduction

The history of autonomous development systems has been marked by their emergence. Software engineering has shifted from manual, script-based DevOps to more recent AI-powered development systems that can make autonomous decisions (Cois et al., 2015). This change has been driven by the increased complexity of cloud-native systems where the existing automation is no longer able to scale and remain dynamic with the scale and dynamism of dispersed micro-services (Kratzke & Quint, 2017).

The next frontier in automation is agentic AI or the use of AI systems that can independently reason, plan and execute multi-step tasks. Compared to previous rule-based systems, Agentic AI can comprehend the situation, streamline development processes, and restructure workflows on the fly (Martinez-Fernandez et al., 2021; Sundar, 2020). At the same time, Kubernetes has been adopted as the new cloud-native orchestration platform, providing declarative configuration, self-healing, and resilient workload scheduling (Carrion, 2023; Poniszewska-Maranda and Czechowska, 2021). At the intersection of Agentic AI and Kubernetes, it is possible to establish an environment where intelligent agents can coordinate development and delivery with limited human involvement.

1.1 Problem Statement

Engineering teams are increasingly struggling to scale microservices, even though DevOps tooling has improved. Distributed services are complex, leading to operational overhead, inconsistent deployments, and prolonged debugging (Cui et al., 2021). Manual or semi-automated CI/CD operations slow down the release pace, especially in high-paced product projects where rapid iteration is crucial (Atouani et al., 2021). Combining all these issues, it is clear that intelligent, adaptable, and autonomous development and delivery frameworks are required.

2. Underlying Principles and System Design.

2.1. Features of Agentic Artificial Intelligence in Cloud-native Systems.

The agentic AI adds autonomous planning, situational reasoning, and the ability to execute multiple actions simultaneously, enabling systems to exhibit machine agency beyond conventional automation systems (Sundar, 2020). Such agents can monitor system conditions, predict necessary actions, and make decisions in accordance with the postulates of responsible autonomous behavior (Martinez-Fernandez et al., 2021). In a cloud-native environment, this enables optimisation across development, deployment, and runtime processes.

2.2. Kubernetes Essentials of Smart Automation.

Kubernetes provides an orchestration system that Agentic AI can use to perform actions. It consists of its core components, including clusters, pods, deployments, services, and operators, and is a declarative, self-healing environment well suited to microservices (Poniszewska-Maranda & Czechowska, 2021). The loosely coupled services that constitute the platform's architecture support scalability and resilience patterns that underpin modern cloud systems (Kratzke & Quint, 2017). In addition, sophisticated scheduling systems and resource distribution plans provide good opportunities for AI-based optimization (Carrion, 2023; Senjab et al., 2023).

2.3. Karpenter-Directed Intelligent Autoscaling.

Karpenter makes Kubernetes more efficient by enabling real-time, event-driven node provisioning and adaptable workloads with AgenticAI pipelines. It has a rapid, cost-conscious scheduler that minimizes latency within the infrastructure and shortens deployment times, enabling AI agents to execute resource-intensive jobs without throughput capacity constraints (Gawande & Gorthi, 2024). Karpenter autonomous scaling is useful in a production setup to maintain performance consistency under unpredictable application loads, much like the patterns of intelligent autoscaling seen in e-commerce applications on cloud-native systems (Nerella, 2025). This is an adaptive type of elasticity that provides multi-agent and computational workloads, complementary to the development of agentic orchestration frameworks (Liu et al., 2025).

2.4. High-Level Architecture of Integrating Agentic AI and Kubernetes.

It is generally a strong integration architecture comprising:

- A reasoning, planning, and action sequencing AI orchestration layer.
- An operational layer that is Kubernetes and handles the workloads and cluster resources.
- Cluster-telemetry-interpreting agent ingestion pipes.
- Policies and models of governance to control safe and compliant interventions.

This architectural model aligns with generative and adaptive system models that leverage artifact references and structured metadata to enable intelligent automation (Atouani et al., 2021). The layered design makes Agentic AI highly integrated into Kubernetes processes, enabling autonomous updates,

Layer	Function / Role	Key Features / Capabilities	Reference / Notes
Agentic AI Orchestration Layer	Reasoning, planning, and sequencing of actions across development and operations workflows	Autonomous task scheduling, multi-step execution, workflow optimization	Aligns with generative and adaptive AI models leveraging artifact references and structured metadata (Atouani et al., 2021)
Operational (Kubernetes) Layer	Handles workloads, manages cluster resources, and ensures high availability	Pod scheduling, deployments, autoscaling, resource reconciliation	Provides the foundation for cloud-native orchestration and declarative system management (Carrión, 2023; Poniszewska-Marańda & Czechowska, 2021)
Cluster Telemetry & Agent Ingestion Layer	Ingests, interprets, and feeds cluster metrics and logs to AI agents	Real-time monitoring, anomaly detection, predictive alerts	Enables closed-loop feedback between AI agents and cluster operations (Hrusto, Runeson & Engström, 2021)
Policy & Governance Layer	Ensures safe, compliant, and auditable agent actions	Policy-as-code enforcement, RBAC controls, security guardrails	Supports trustworthy autonomous interventions and compliance management (Thiebes, Lins & Sunyaev, 2021)
Karpenter Intelligent Autoscaling Layer	Dynamic provisioning and optimization of compute nodes for AI-driven workloads	Fast node creation, cost-aware scaling, adaptation to workload changes, declarative configuration generation	Enhances elasticity and performance for AI-intensive pipelines by enabling responsive cluster expansion/contraction (Gawande & Gorthi, 2024; Nerella, 2025; Liu et al., 2025)

Table 1: Architecture for AgenticAI + Kubernetes Deployment (Including Karpenter Intelligent Autoscaling)

2.5. Event-Driven Triggers, APIs, GitOps, and Control Loops.

Independent workflows occur due to events involving AI agents and Kubernetes. To manage cluster states, e.g., pod failures, resource saturation, or configuration drift, agents constantly monitor cluster state and respond through Kubernetes APIs or Operators. GitOps pipelines provide a versioned source of truth, meaning agents can identify desired and actual states and ensure they are reliable and traceable (Hrusto et al., 2021).

Control loops are the fundamental building blocks of the Kubernetes design and, as such, the operational components that allow agents to monitor for deviations, analyze their causes, and take corrective actions in real time.

2.6. RBAC, Trust and Security of Agent Permissions.

The issue of security is fundamental when it comes to granting AI systems the right to alter production environments. Kubernetes Role-Based Access Control (RBAC) must be configured to grant agents the minimum necessary permissions to do their jobs and minimize the risk of misconfigurations or increased privileges (Rahman et al., 2023).

This can align with the larger principles of trustworthy AI, which focus on transparency, control, and accountability when autonomous actors affect working frameworks (Thiebes et al., 2021). The requisite conditions for building safe, reliable, and governed automation are clear audit trails, policy-as-code frameworks, and continuous validation mechanisms.

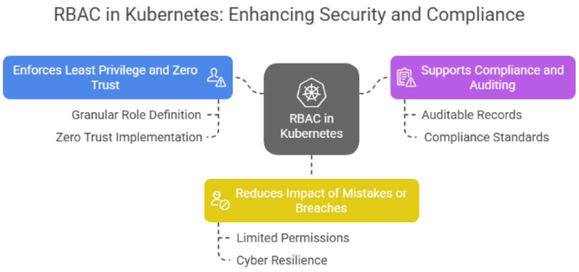
RBAC in Kubernetes: Enhancing Security and Compliance
Enforces Least Privilege and Zero Trust
Granular Role Definition
Zero Trust Implementation
RBAC in Kubernetes
Supports Compliance and Auditing
Auditable Records
Compliance Standards
Reduces Impact of Mistakes or Breaches
Limited Permissions
Cyber Resilience

Figure 1: **Tips for Kubernetes Role-Based Access Control (RBAC)**

3. The Accelerating Development with Agentic AI.

The introduction of Agentic AI into the Kubernetes-based development space accelerates software development by automating cognitively intensive and routine tasks. The agentic systems, which can reason, plan, and perform multi-step actions, improve the development lifecycle by going beyond conventional automation (Sundar, 2020). These abilities enable developers to redirect their attention to making architectural choices and solving higher-value problems, while leaving normal engineering tasks to self-directed agents.

3.1 Automation of testing using AI: Unit, Integration, and Regression Testing.

Software development is one of the activities that consumes a lot of testing resources. The process is optimized in agentic AI, which independently creates test cases, identifies edge conditions, and runs full test suites in Kubernetes environments. This reflects the general movement toward autonomous operation feedback loops within DevOps, in which machine intelligence actively bridges quality loops (Hrusto et al., 2021). The AI detects services with unreliable behavior, regressions, and potential performance bottlenecks earlier in the development cycle through real-time analysis.

3.2 Smart Code Reviews, Pull Request Enhancement and Static Analysis.

The agentic AI is an intelligent reviewer in the team development processes. It checks the pull requests for logic errors, security gaps, performance vulnerabilities, and architectural mismatches. Studies on efficient interactions within DevOps systems demonstrate the essential role of high-fidelity, short-timeframe feedback loops in enhancing team throughput (Cois et al., 2015). In addition to traditional linters, the AI can intelligently assess code intent and ensure compliance with organizational policies, Kubernetes best practices, and cloud-native patterns (Kratzke and Quint, 2017).

3.3 Automated Dependency Checks, Dependency Upgrades, and Dependency Vulnerabilities.

Microservice dependencies change rapidly, leading to frequent security risks and incompatibilities. The agentic AI helps to identify outdated libraries, assess breakage, create upgrade patches and apply vulnerability fixes independently. The ability is crucial given the growing complexity of containerized workloads and the number of misconfigurations in code involving Kubernetes (Rahman et al., 2023). The AI minimizes exposure to supply chain threats and ensures system stability through continuous scanning and remediation.

3.4 The reason Agentic AI is reducing the number of development cycles and decreasing the load on developers.

A combination of these abilities significantly reduces lead time, speeds iteration, and reduces context switching for development teams. It is highlighted that AI supplements and improves the use of cognitive resources in humans by consuming monotonous tasks and allowing them to make decisions more quickly (Johri, 2022; Markauskaite et al., 2022). In Kube-based systems, where microservices, manifests, operators, and CI/CD pipelines are constantly changing, the autonomic-assisted nature of Agentic AI reduces operational friction and improves overall development throughput. This change will enable engineering groups to shift their focus from manual implementation to strategic management, leading to faster, more reliable software delivery.

Capability of Agentic AI	Impact on Development Process	Supporting Evidence
Automates repetitive tasks and code generation	Frees developers from monotonous work, reducing context switching	Johri, 2022; Markauskaite et al., 2022
Intelligent orchestration of CI/CD pipelines	Speeds iteration and reduces lead time	Kratzke & Quint, 2017; Hrusto, Runeson & Engström, 2021
Autonomic management of Kubernetes resources (microservices, manifests, operators)	Reduces operational friction and prevents bottlenecks in deployment	Carrión, 2023; Poniszewska-Marańda & Czechowska, 2021
Continuous feedback and self-correcting mechanisms	Improves overall development throughput and reliability	Hrusto, Runeson & Engström, 2021; Martínez-Fernández et al., 2021
Supports cognitive augmentation	Allows engineers to focus on strategic decision-making instead of manual implementation	Johri, 2022; Markauskaite et al., 2022

Table 2: How Agentic AI Reduces Development Cycles and Developer Workload

4. Kubernetes AI-Augmented Deployment and Delivery.

The agentic AI is an improvement over Kubernetes-based deployment pipelines by enabling autonomous decision-making, continuous validation, and smart orchestration across all delivery phases. This change will align with the current DevOps philosophy, which focuses on streamlined system communication and faster delivery times (Cois et al., 2015). Adaptive intelligence enables agents to optimise the deployment workflow, minimise human interaction, and increase confidence in cloud-native processes.

4.1 AI-Enhanced CI/CD Automation

CI/CD pipelines that are agent-driven add smart build automation, dynamic quality gate and real-time policy enforcement. These are not just the capabilities of static automation reasoning but context-based decision-making is used all along the pipeline. Hrusto et al. (2021) note that the role of autonomous operational monitors is crucial to closing the feedback loop, enabling issues to be identified and addressed quickly. Artifact-aware models are also used by AI agents to assess dependency graphs, version integrity, and configuration accuracy, based on the principles of artifact intelligence introduced by Atouani et al. (2021). This will guarantee consistency, compliance, and security standards for every build released to the deployment stage.

4.2 Optimization of AI-Disciplined Deployment Strategy.

The Kubernetes platform supports various deployment models, such as blue-green, canary, and rolling updates; however, to choose the most suitable one, it is necessary to conduct a situational evaluation of system load, traffic inflow and outflow, and resource availability. The agentic AI models use these variables on-the-fly to determine the safest, most efficient deployment pathway.

Carrion (2023) and Senjab et al. (2023) also point to the complexity of scheduling in a multi-tenant cluster environment, as it is inherently challenging in a dynamic mode of operation. These difficulties can be mitigated significantly when agentic reasoning is in place, in which deployment reasoning is continuously recalibrated on the basis of the projected performance results and real-time cluster telemetry.

The resource needs of an imminent pod are also fulfilled through Karpenter's intelligent autoscaling facilities, which Agentic AI uses to make fast, ad hoc decisions. Once deployment workloads exceed existing capacity, Karpenter will automatically create optimized compute nodes based on AI-based forecasts to ensure the chosen deployment strategy is implemented without delays or resource contention. This synergy enables a fully adaptive deployment process, with infrastructure and deployment logic developed in real time.

4.3 Real-Time Detection of Risk and Autonomous Rollback.

The AI-powered pipelines monitor anomalies, misconfigurations, and performance regressions throughout and after their deployment. These strikes trigger automatic rollback processes before the impact on the end user. This is compatible with the tenets of credible autonomy argued by Martinez-Fernandez et al. (2021), who affirm that continuous assessment and safe automation of intelligent systems should be regarded.

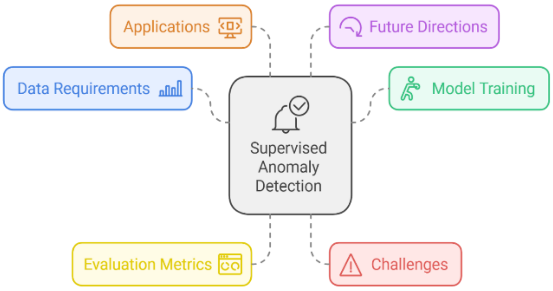

Figure 2: AI-Powered Anomaly Detection

4.4 Configuration and Infrastructure Updates by AI.

The agentic AI is a useful enhancement to configuration management, enabling autonomic updates to deployment manifests, Helm charts, and Dockerfiles. On an ongoing basis, comparing live performance patterns with the desired cluster states and analyzing real-time telemetry enables AI agents to initiate configuration changes to maintain optimal operational conditions. These updates are resource rightsizing, environment variable tuning, HPA recalibration, and security configuration reinforcement.

Besides configuration-level adoption, AI also works with Karpenter to enable intelligent, on-the-fly decisions about infrastructure. The scaling of AI agents implemented by Karpenter provides fast provisioning and workload-driven scaling, spinning up nodes as needed for pending pods or deprovisioning underutilized instances to ensure efficiency (Gawande & Gorthi, 2024). This adaptive elasticity reflects the latest production-scale autoscaling models, which have provided assurance that workloads are performant and cost-efficient even with changing demand (Nerella, 2025).

As Rahman et al. (2023) point out, misconfigurations are currently considered one of the most important sources of Kubernetes instability, which is why the concept of AI-driven systems that will automatically fix them as the cluster configuration and infrastructure are continuously brought to the target operational state is a solid argument

4.5 Improved Deployment Performance and Reliability.

All the above, plus AI-based automation, smart deployment policies, early risk identification, and

The agents scan logs, metrics, and service behavior to spot early indicators of degradation, e.g., latency spikes, memory leaks, or pod failures, and take timely corrective actions. This proactive stance is quite effective in reducing the likelihood of production incidents and minimizing recovery time.

dynamic configuration, yield quantifiable deployment performance. There is an increased rate of release, reduced change-failure rates, and much more stable production rollouts in organizations. These advancements represent the broader progress of cloud-native engineering, as characterized by Kratzke and Quint (2017), in which automation and smarter systems are the key drivers of faster, more resilient software delivery. Incorporating Agentic AI into Kubernetes enables development teams to take a step toward actual autonomous DevOps, including faster iteration, continuous delivery, and higher production reliability.

5. Smart Production Operations and Optimization (AIOps).

The level of production is enhanced by agentic AI, which enables autonomous decisions based on data across Kubernetes clusters. Real-time monitoring, interpretation, and action-taking based on system signals help solve various operational issues that have repeatedly occurred in cloud-native environments.

5.1 Real-time monitoring/anomaly detection.

Telemetry, logs, and distributed traces can be continuously analyzed by agentic AI, which can identify irregular patterns before they affect service performance. This is consistent with new studies on autonomous monitoring systems that create a feedback loop between DevOps, minimize the human factor, and shorten system restoration time (Hrusto et al., 2021). This smart detection enhances reliability, particularly in big microservice systems (Cui et al., 2021).

5.2 Predictive scaling and optimization of resources.

In addition to the common auto-scaling models (HPA, VPA, and KEDA) that help predict capacity based on load trends, user behavior, and past patterns, Agentic AI enables capacity planning in advance based on traffic trends, user behavior, and past patterns. This trend aligns with the innovations in intelligent Kubernetes scheduling and resource management that Carrion (2023) and Phuc et al. (2022) describe.

5.3 Cost saving and efficiency in infrastructure.

Smart rightsizing and the removal of overprovisioning can help organizations sustain performance and reduce operational costs. The mentioned optimizations align with the results of cloud-native architecture research, which place greater emphasis on resource efficiency and container lifecycle management (Kratzke & Quint, 2017; Kratzke, 2018).

Optimization Strategy	Description / Implementation	Expected Benefit	Supporting References
Smart Rightsizing	Dynamically adjusting CPU/memory allocations based on workload demand	Reduced idle resource costs; improved cluster efficiency	Kratzke & Quint, 2017; Phuc, Phan & Kim, 2022
Removal of Overprovisioning	Identifying and terminating underutilized pods, nodes, or clusters	Lower operational expenses; improved resource utilization	Kratzke, 2018; Truyen et al., 2020
Autoscaling Policies	Horizontal and vertical pod autoscaling based on real-time metrics	Maintain performance while minimizing unnecessary resource allocation	Phuc, Phan & Kim, 2022; Tuli et al., 2023
Container Lifecycle Optimization	Efficient management of container creation, deletion, and updates	Reduced infrastructure overhead; faster deployment cycles	Kratzke & Quint, 2017; Poniszewska-Marańda & Czechowska, 2021
Predictive Resource Allocation	Using AI/agentic models to forecast workload spikes and allocate resources proactively	Prevent resource shortages; maintain SLA compliance	Tuli et al., 2023; Hrusto, Runeson & Engström, 2021

Table 3: Strategies for Cost Saving and Efficiency in Kubernetes Infrastructure

5.4 Operating system security and compliance.

The use of agentic AI enhances security by self-identifying misconfigurations, evaluating compliance, and identifying suspicious workloads. This holds especially true given the high rate of attacks on Kubernetes manifests (Rahman et al., 2023) and the growing demand for reliable, transparent AI systems (Thiebes et al., 2021). Altogether, Agentic AI would turn Kubernetes operations into proactive, predictive, and autonomous optimization, achieving better availability, safer workloads, and much faster production responsiveness.

6. Conclusion and Future Perspective.

Kubernetes-integrated agentic AI is already redefining the production spaces of various industries. Autonomous agents in fintech aid in maintaining ongoing policy execution, enabling almost-real-time anomaly detection and automatic rollback of risky deployments to increase reliability in stringent regulatory contexts (Cois, Yankel, and Connell, 2015). Telecom operators leverage AI-based orchestration to manage dynamic traffic patterns and optimize pod scheduling under high load, thereby extending existing studies on Kubernetes scheduling (Carrion, 2023; Senjab et al., 2023). Agentic AI can be used to speed up deployments in multi-tenant SaaS by automatically generating manifests, autoscaler configurations, and drift-fixing features that implement cloud-native architecture principles (Kratzke & Quint, 2017). The MLOps pipelines also feature autonomous model deployment, versioning, monitoring, and drift detection, reflecting the need for reliable, sustainable AI systems in the production process (Martinez-Fernandez et al., 2021).

In the future, the trend is toward complete autonomy for cloud-native systems, where Agentic AI handles the entire application lifecycle, from code generation to deployment to runtime optimization. DevOps' functions will evolve as infrastructure becomes more of an adaptive, intelligent ecosystem (Kratzke and Siegfried, 2021), requiring less manual operation and supervision of AI-based processes while restricting higher-level system design. Such a development, in line with global trends in AI-enhanced learning, operations, and service management (Kamalov, Santandreu Calonge & Gurrib, 2023; Wang, Skeete and Owusu, 2022), implies that the future may be characterized by self-governing production environments becoming the rule, rather than the exception.

References

1. Atouani, A., Kirchhof, J. C., Kusmenko, E., & Rumpe, B. (2021). Artifact and reference models for generative machine learning frameworks and build systems. In GPCE 2021 - Proceedings of the 20th ACM SIGPLAN International Conference on Generative Programming: Concepts and Experiences, co-located with SPLASH 2021 (pp. 55–68). Association for Computing Machinery, Inc. https://doi.org/10.1145/3486609.3487199

2. Carrión, C. (2023). Kubernetes Scheduling: Taxonomy, Ongoing Issues and Challenges. ACM Computing Surveys, 55(7). https://doi.org/10.1145/3539606

3. Cois, C. A., Yankel, J., & Connell, A. (2015). Modern DevOps: Optimizing software development through effective system interactions. In IEEE International Professional Communication Conference (Vol. 2015-January). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/IPCC.2014.7020388

4. Cui, H. T., Zhang, C., Ding, X., Cao, L. L., & Yang, Y. (2021, May 1). Evaluation Framework for Development Organizations' Adaptability to Micro-services Architecture. Ruan Jian Xue Bao/Journal of Software. Chinese Academy of Sciences. https://doi.org/10.13328/j.cnki.jos.006232

5. Gawande, S., & Gorthi, A. (2024). Containerization and Kubernetes: Scalable and Efficient Cloud-Native Applications. International Journal of Innovative Science and Research Technology (IJISRT), 435–439. https://doi.org/10.38124/ijisrt/ijisrt24nov314

6. Hong, J. W., Cruz, I., & Williams, D. (2021). AI, you can drive my car: How we evaluate human drivers vs. self-driving cars. Computers in Human Behavior, 125. https://doi.org/10.1016/j.chb.2021.106944

7. Hrusto, A., Runeson, P., & Engström, E. (2021). Closing the Feedback Loop in DevOps Through Autonomous Monitors in Operations. SN Computer Science, 2(6). https://doi.org/10.1007/s42979-021-00826-y

8. Johri, A. (2022). Augmented sociomateriality: implications of artificial intelligence for the field of learning technology. Research in Learning Technology, 30. https://doi.org/10.25304/rlt.v30.2642

9. Kamalov, F., Santandreu Calonge, D., & Gurrib, I. (2023). New Era of Artificial Intelligence in Education: Towards a Sustainable Multifaceted Revolution. Sustainability (Switzerland), 15(16). https://doi.org/10.3390/su151612451

10. Kratzke, N. (2018, August 14). A brief history of cloud application architectures. Applied Sciences (Switzerland). MDPI AG. https://doi.org/10.3390/app8081368

11. Kratzke, N., & Quint, P. C. (2017). Understanding cloud-native applications after 10 years of cloud computing - A systematic mapping study. Journal of Systems and Software, 126, 1–16. https://doi.org/10.1016/j.jss.2017.01.001

12. Kratzke, N., & Siegfried, R. (2021). Towards cloud-native simulations – lessons learned from the front-line of cloud computing. Journal of Defense Modeling and Simulation, 18(1), 39–58. https://doi.org/10.1177/1548512919895327

13. Li, H., Xu, Y., & Hong, T. (2025). EnergyPlus-MCP: A model-context-protocol server for ai-driven building energy modeling. SoftwareX, 32. https://doi.org/10.1016/j.softx.2025.102367

14. Liu 刘, J. 家轩, Zhu 朱, T. 天念, Ye 叶, C. 财渊, Fang 方, Z. 忠, Weng 翁, H. 红明, & Wu 吴, Q. 泉生. (2025). VASPilot: MCP-facilitated multi-agent intelligence for autonomous VASP simulations. Chinese Physics B, 34(11), 117106. https://doi.org/10.1088/1674-1056/ae0681

15. Markauskaite, L., Marrone, R., Poquet, O., Knight, S., Martinez-Maldonado, R., Howard, S., … Siemens, G. (2022). Rethinking the entwinement between artificial intelligence and human learning: What capabilities do learners need for a world with AI? Computers and Education: Artificial Intelligence, 3. https://doi.org/10.1016/j.caeai.2022.100056

16. Martínez-Fernández, S., Franch, X., Jedlitschka, A., Oriol, M., & Trendowicz, A. (2021). Developing and Operating Artificial Intelligence Models in Trustworthy Autonomous Systems. In Lecture Notes in Business Information Processing (Vol. 415 LNBIP, pp. 221–229). Springer Science and Business Media Deutschland GmbH. https://doi.org/10.1007/978-3-030-75018-3_14

17. Nerella, H. (2025). Intelligent Autoscaling for E-Commerce Applications in Public Cloud Kubernetes Environment. In Lecture Notes in Networks and Systems (Vol. 1356 LNNS, pp. 143–155). Springer Science and Business Media Deutschland GmbH. https://doi.org/10.1007/978-981-96-5238-9_14

18. Phuc, L. H., Phan, L. A., & Kim, T. (2022). Traffic-Aware Horizontal Pod Autoscaler in Kubernetes-Based Edge Computing Infrastructure. IEEE Access, 10, 18966–18977. https://doi.org/10.1109/ACCESS.2022.3150867

19. Poniszewska-Marańda, A., & Czechowska, E. (2021). Kubernetes cluster for automating software production environment. Sensors, 21(5), 1–24. https://doi.org/10.3390/s21051910

20. Rahman, A., Shamim, S. I., Bose, D. B., & Pandita, R. (2023). Security Misconfigurations in Open Source Kubernetes Manifests: An Empirical Study. ACM Transactions on Software Engineering and Methodology, 32(4). https://doi.org/10.1145/3579639

21. Ray, P. P., & Pratim, P. R. (2025). A Survey on Model Context Protocol: Architecture, State-of-the-art, Challenges and Future Directions. Authorea Preprints, 1–44. Retrieved from https://www.techrxiv.org/doi/pdf/10.36227/techrxiv.174495492.22752319/v1

22. Saw, S. N., & Ng, K. H. (2022, August 1). Current challenges of implementing artificial intelligence in medical imaging. Physica Medica. Associazione Italiana di Fisica Medica. https://doi.org/10.1016/j.ejmp.2022.06.003

23. Schwalbe, N., & Wahl, B. (2020, May 16). Artificial intelligence and the future of global health. The Lancet. Lancet Publishing Group. https://doi.org/10.1016/S0140-6736(20)30226-9

24. Senjab, K., Abbas, S., Ahmed, N., & Khan, A. ur R. (2023, December 1). A survey of Kubernetes scheduling algorithms. Journal of Cloud Computing. Springer Science and Business Media Deutschland GmbH. https://doi.org/10.1186/s13677-023-00471-1

25. Shank, D. B., North, M., Arnold, C., & Gamez, P. (2021). Can Mind Perception Explain Virtuous Character Judgments of Artificial Intelligence? Technology, Mind, and Behavior, 2(2). https://doi.org/10.1037/tmb0000047

26. Sundar, S. S. (2020). Rise of Machine Agency: A Framework for Studying the Psychology of Human-AI Interaction (HAII). Journal of Computer-Mediated Communication, 25(1), 74–88. https://doi.org/10.1093/jcmc/zmz026

27. Thiebes, S., Lins, S., & Sunyaev, A. (2021). Trustworthy artificial intelligence. Electronic Markets, 31(2), 447–464. https://doi.org/10.1007/s12525-020-00441-4

28. Truyen, E., Kratzke, N., Van Landuyt, D., Lagaisse, B., & Joosen, W. (2020). Managing Feature Compatibility in Kubernetes: Vendor Comparison and Analysis. IEEE Access, 8, 228420–228439. https://doi.org/10.1109/ACCESS.2020.3045768

29. Tuli, S., Mirhakimi, F., Pallewatta, S., Zawad, S., Casale, G., Javadi, B., … Jennings, N. R. (2023, July 1). AI augmented Edge and Fog computing: Trends and challenges. Journal of Network and Computer Applications. Academic Press. https://doi.org/10.1016/j.jnca.2023.103648

30. Wang, Y., Skeete, J. P., & Owusu, G. (2022). Understanding the implications of artificial intelligence on field service operations: a case study of BT. Production Planning and Control, 33(16), 1591–1607. https://doi.org/10.1080/09537287.2021.1882694

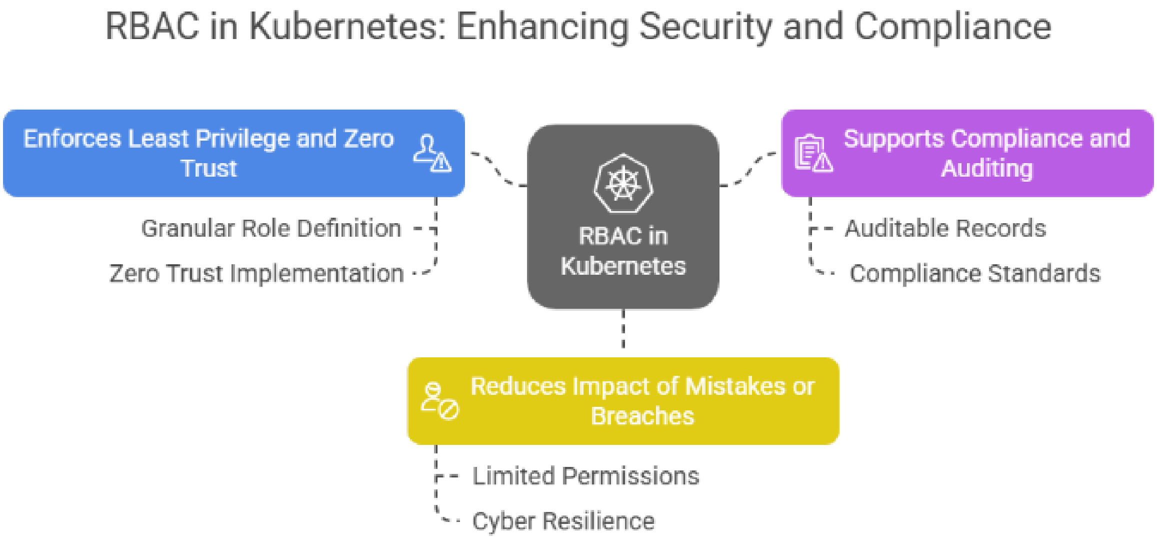
RBAC in Kubernetes: Enhancing Security and Compliance
Enforces Least Privilege and Zero Trust
Granular Role Definition
Zero Trust Implementation
RBAC in Kubernetes
Supports Compliance and Auditing
Auditable Records
Compliance Standards
Reduces Impact of Mistakes or Breaches
Limited Permissions
Cyber Resilience

Figure 1: **Tips for Kubernetes Role-Based Access Control (RBAC)**

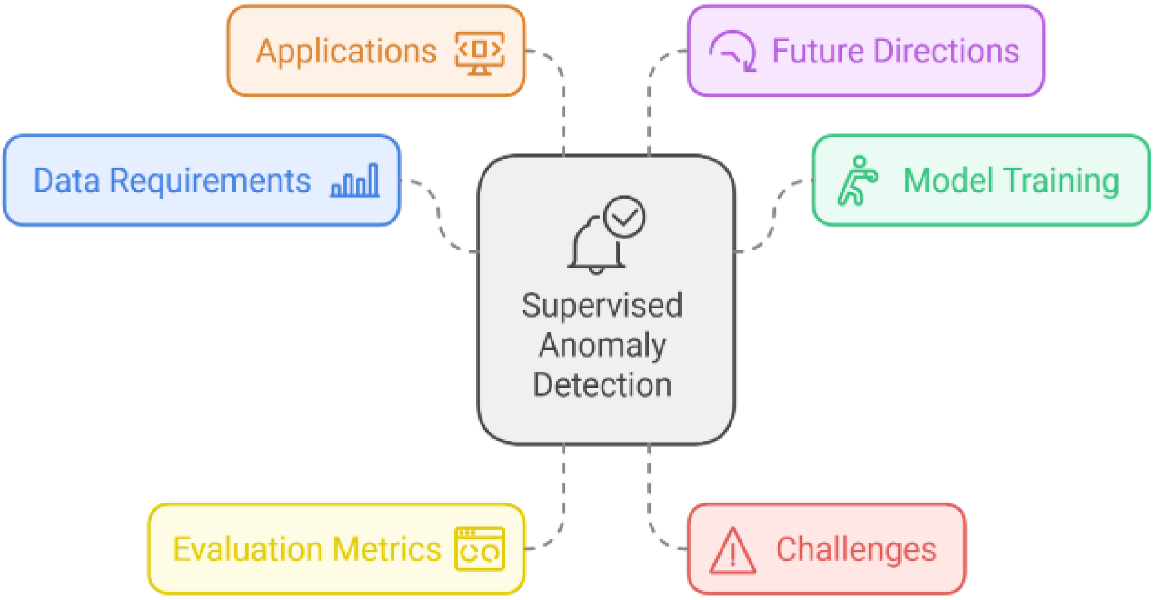

Figure 2: AI-Powered Anomaly Detection