International Journal of Computer Trends and Technology
ISSN: 2231–2803 / https://doi.org/10.14445/22312803/IJCTT-V73I2P104

Volume 73 Issue 2, 23-30, February 2025

Original Article

D-K8S: Container Orchestration Through Nodes Empowerment and Participation

Indravadan Patel

Department of Computer Science and Engineering, University of North Texas, Denton, Texas, USA.

[1]*Corresponding Author : indravadanpatel@my.unt.edu*

Received: 26 December 2024 Revised: 21 January 2025 Accepted: 14 February 2025 Published: 28 February 2025

Abstract *- With the tremendous rise in virtualization and containerization over the past few decades and in the software sector, the paradigm has shifted to reevaluate the deployment architecture and orchestration management of containers by taking decentralization into account [22]. Given the growing acceptance and use of container orchestration over the past few years and the field's ongoing development [4], this study presents the ideas of "D-K8S" (Decentralized-Kubernetes) and reveals the investigation of various facets of the decentralized system using Kubernetes. Examining the mixing of different elements of the decentralized architecture among nodes and its complexities, such as cooperating for container orchestration, goes beyond the confines of centralized realms that have existed since its invention [14]. Resilient constellations, improved safety, resource efficiency, reinforced security orbits, and performance dynamics are the main topics of the D-K8S architectural study.*

Keywords *- Containerization, Decentralization, Centralization, Kubernetes, Worker node.*

1. Introduction

Container orchestration stands as a cornerstone in the realm of managing, running, and deploying containerized applications [25]. With Kubernetes emerging as a major pathfinder in the era of container orchestration, this essential container orchestration framework provides the cornerstone for fault tolerance, scalability, and resilience [5]. Its quick and widespread acceptance in cloud-native environments is evidence of its effectiveness [9], as many sectors come to comprehend its benefits for container service management.

Although Kubernetes has a centralized architecture, its continual growth has led to a great deal of investigation into optimizing performance, making the best use of resources, reducing response times, and strengthening security at the node and network communication levels [12]. These projects aim to improve the overall security, fault tolerance, and scalability of Kubernetes. In this quest, a frontier is the combination of AI, ML, and SDN, particularly when it comes to customizing Kubernetes for the resource-constrained domains of edge computing and IoT devices [13].

Kubernetes is one of the most used open-source container orchestration frameworks among others from the beginning [18]. It was developed by Google, and now, it is maintained by CNCF under Apache Licenses. The traditional architecture has different components and interactions among those at a high level with the centralized approach, where the control plan makes decisions for the whole cluster. After growth in the cloud in the last few years, this open-source orchestration framework has been used heavily in managing different container services in a cluster-based approach [28]. The increased industrial usages motivate the researcher to think and consider Distributed systems, Artificial Intelligence, and Machine learning [8]. Many efforts have been underway to make the orchestration framework more user-friendly, secure, and efficient, enhance observability, and better resource utilization [24]. Considering the Cloud growth and Serverless approach for cost optimization creates more interest in making it more reliable, scalable, secure, and cost-efficient in the cloud-native environment [16].

As you can see, the traditional approach of Kubernetes architecture is centralized, where all the decision-making and heavy lifting occur at the Master node level. This traditional model of container orchestration creates interest in exploring the distributed controller model, which spreads at all the worker node levels [10]. This research aims to see the application of the distributed controller model at each worker node level and conduct different performance analyses over distributed complexities, assuming that the node-level operation brings down some of the overhead from the control plane and helps with better performance and response time. This decentralized system model at the worker node level helps reduce network traffic and decide the node level.

The deployment of a distributed controller at the worker node level is the new strategy that this study centers around.

The two main goals are optimizing performance and reducing traffic and network latency while considering network and node security. The fundamental principle is using the benefits of distributed systems inside the orchestration framework. This paradigm challenges traditional designs in the pursuit of optimal resource utilization and reaction time by strategically placing controllers at each worker node. It is understood that the distributed strategy has inherent complexity [23], but better outcomes are anticipated. The research paper attempts to challenge the present status quo by exposing a fresh approach to the Kubernetes orchestration framework using a decentralized architectural design. The projected consequences will result in significant improvements in reaction times and resource utilization, indicating a new step in the development of a safe, reliable, and expandable container orchestration system.

As we venture into the distributed frontier of Kubernetes orchestration, the research aims to contribute to both the technical discourse and the pragmatic growth of orchestration paradigms, paying particular consideration to the benefits and obstacles that come with distributed systems. The study contributes to the theoretical discussion and the practical development of orchestration paradigms as we explore the distributed frontier of Kubernetes orchestration, taking special note of the advantages and challenges associated with distributed systems.

2. Problem Definition and Motivation

Despite the effectiveness of the traditional Kubernetes, the centralized control plane creates risks of a single point of failure and, along with it, scalability bottlenecks [1]. The increasing size of the clusters creates more security concerns, impedes the performance and response time, and creates questions about resource utilization and associated costs. The increased usage of containerized applications within industries and considering the growth of cloud and edge computing strives for enhanced security, fault tolerance, and robust autonomy across clusters [2].

The motivation for this research work is the limitation of centralized Kubernetes architectures and the need to cope with the evolving modern technological landscape and its dynamics.

With respect to Architectural Resilience, the centralized, traditional Kubernetes, as aforesaid, poses a single point of failure and hinders high availability and scalabilities [19], affecting the orchestration resiliency. The decentralized model will help with failure, introducing redundancy, achieving high availability, fault tolerance, and resiliency, and improving the overall container orchestration robustness.

Along with a single point of failure, a single-point target creates a major risk for introducing and impacting vulnerabilities within the cluster [21]. In the case of a decentralized model, the paradigms around decentralized security will help with threat mitigation, enhancing network security along identity management.

The centralized, traditional Kubernetes hinders operational flexibility with varying workloads and imposes rigidness with dynamical scalability [7]. Instigating scalability challenges and respective operation flexibility with dynamic workloads around the adaptive strategies of the decentralized model helps with overcoming challenges and dynamic resource optimization.

In fact, the motivation of this research focuses on the Kubernetes capabilities and efficiencies imperative elevation considering some of the bottlenecks of centralized, traditional Kubernetes architecture with respect to the different evolving computing needs and demands of emerging new technologies landscape [8]. Through this research work, we aim to contribute to the evolving needs of container orchestration and lay down the roots of decentralized architecture with respect to container orchestration that embodies enhanced characteristics like scalability, security, operational efficiency, and resilience.

3. Literature Survey

The research paper published by Larsson et al. [15] introduced the federation using the distributed CRDT-powered database, which was created at each node level, and the federation was achieved through a consistent algorithm. The said approach is very complex compared to the centralized current architecture, considering network latency may create slowness for the decision-making process. The security aspect of the said approach is not explained considerably and requires practical implementation of the said architecture with different use cases.

The literature survey conducted by Watada et al. [27] has explored the containerization issues with different orchestration tools and their usage with edge computing and IoT in cloud-native environments. They also discussed different health care security issues and respective IoT environments that consider containerization approaches. The microservices with respect to their implementation with stateful and stateless architecture, along with communication among those through different network topologies, have been discussed, along with the different issues concerning securities, observability, and log aggregation. The comparison between different virtualization and containers has been discussed thoroughly and explained in detail, and the benefits of using containers over virtualization have stood out. The growth in cloud computing increases the usage of containers, so orchestration framework usage increases year after year. The literature survey also talked about different approaches suggested by the research community on artificial intelligence

and machine learning. There is no specific discussion about the distributed system model for the orchestration framework, and its applicability is not explicitly elaborated in the research direction section.

As emphasised in the above survey paper, artificial intelligence and Machine Learning usage will help with resource utilization. Along that line, Theodoropoulos et al. [26] suggested the GreenKube framework for helping with energy utilization by considering the QoS best practices. The said prototype had been compared with horizontal pod scaling and showed better results. This paper only talks about the prototype but does not talk about what will be implicated in considering scalability and implementation issues with complex multi-cluster environments.

The combination of hardware and software for maintaining resource management has been explored by Monaco et al. [20]. The paper aims to better utilize resources using dual-mode in a real-time environment, considering resource demands on a time scale. The related implementation intricacies have not been mentioned anywhere and are not considered different use cases. It has been clear that the implementation requires hardware and software interconnection, creating more troubleshooting and maintenance complexities.

4. Methodology

The research methodology and design approach encompasses a deeper dive into decentralized architectural model incorporation and investigation for containerized or orchestration with respect to a security evaluation, architecture analysis, and performance assessments.

The abstract conceptualization of a decentralized controller architecture within the Kubernetes ecosystem signifies and incorporates a paradigm shift in managing orchestrating containerized workloads within clusters for different infrastructures established using different topologies. The core tenet of this architectural approach revolves around with consideration the empowerment of individual independent worker nodes with different configurations, each having distributed controllers, fostering independent decision-making and task execution based on gathered decentralized cluster-level data and isolated node-level data.

The intricate architectural design seeks to reinvent resource management techniques, reduce network dependencies through reducing network traffic, and curtail the cost of operations through sophisticated distributed local governance.

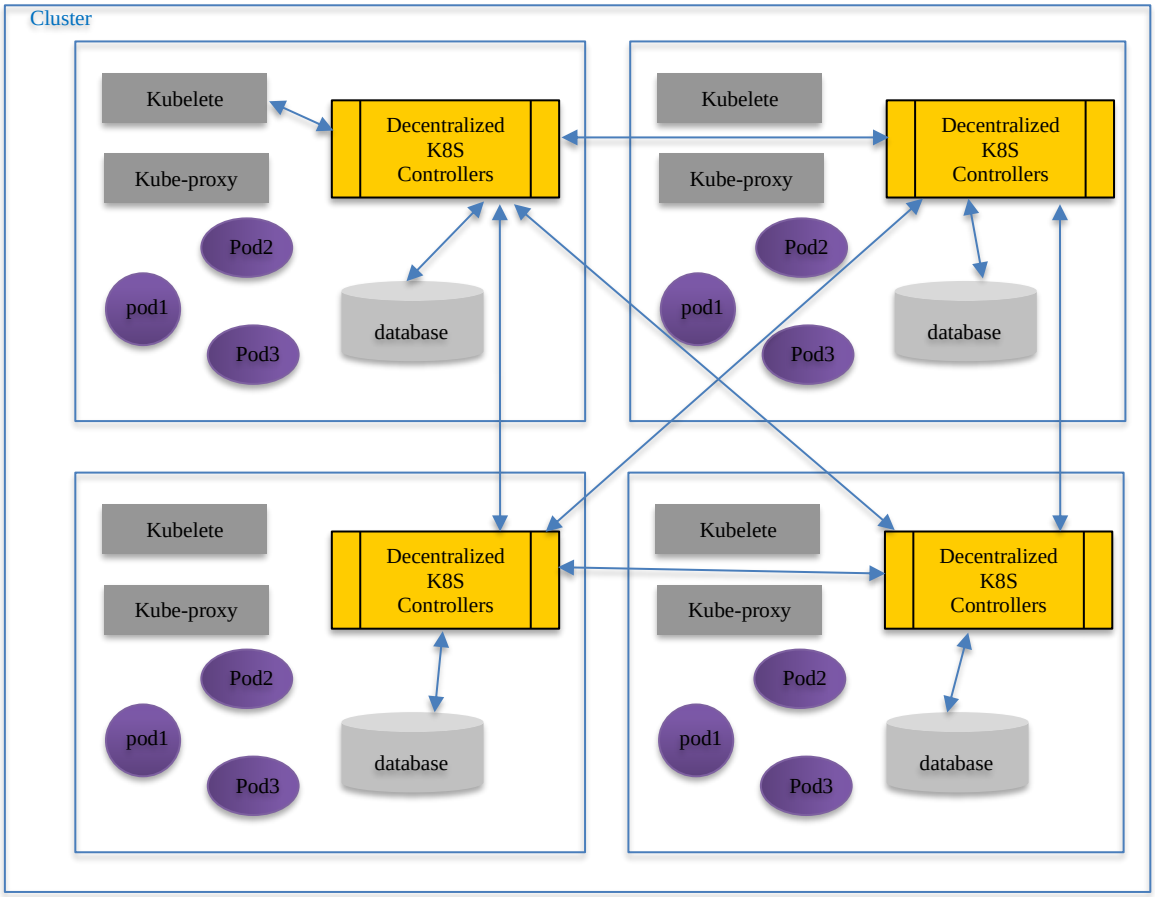

Fig. 1 Distributed Kubernetes (D-K8S) Architecture

Visualized in Figure 1, the distributed controller architecture model envisages moving away from the paradigm of conventional centralized control. Instead, it promotes the proliferation of controllers at every worker node, imbuing them with the capacity to function and make decisions independently using the data collected and stored at the node and cluster levels. The proposed divide-and-conquer strategy emphasizes the autonomy of every node, fostering a self-sufficient approach to decision-making with respect to different current states and desired states. The final objective is the optimization of local performance metrics in order to reduce the strain on network bandwidth utilization and accentuate resource efficiency.

The development of the distributed controller architecture, along with communication among nodes for decision-making, is required to comprehend the different states and events involved and how those can be used for different tasks and their interdependencies. The controllers are primarily involved in achieving the desired states from the current states of the Kubernetes objects and maintaining those at the node level. As you can see in the diagram below in Figure 2, different components are involved in building the distributed controller, which will help at the worker node level and customize development considering different jobs performed at the node level and cluster level based on the custom-specific need and which can help with making the best decision based on local conditions and local data which are easily available for decision making. It will also help reduce network traffic and so forth, allowing the best performance to be achieved at its most.

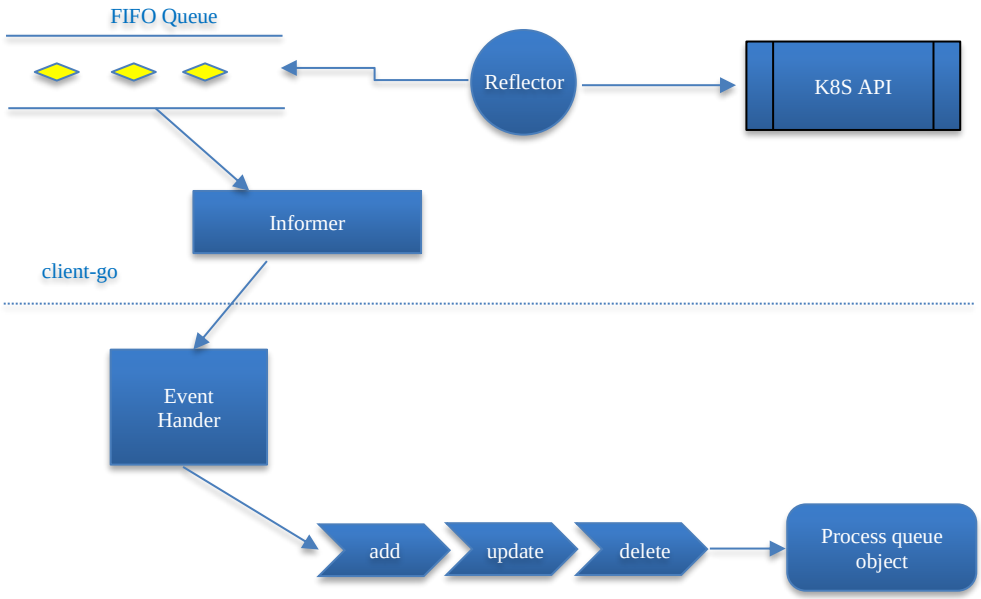

Fig. 2 Distributed controller event-driven design

As shown in Figure 2, two main and major tasks are parts of controller flow and are involved in developing decentralized controllers, which are implemented at all worker nodes. The informer and worker queue are two main components.

The informer helps with all the events generated for the Kubernetes resources, and those pushed using the event handler developed at the distributed controller with the interaction with the worker queue and added tasks to this queue are processed as per actions and accomplish the respective required jobs and preserve those objects as current states within the database.

The experiment setup for this study is considered the MaC machine with 8 Core CPU and 10 Core GPU, 16GM RAM (Minimum requirement 8 GB), 1 TB hard disk (Minimum requirement 256 GB) has been used for setting up the Kubernetes using the source code pulled from GitHub - “https://github.com/kubernetes/kubernetes” as Kubernetes is open-source container orchestration framework developed using GO language.

5. Results and Discussion

The quantitative analysis undertaken in this research is a nuanced investigation and exploration into the implication of implementing a decentralized architectural design paradigm throughout the spectrum of all worker nodes within the Kubernetes ecosystem. The juxtaposition of response time and resource consumption metrics reveals a meticulous endeavor, shedding light on the detailed performance differences between the decentralized and centralized orchestration approaches.

Fig. 3 Centralized Approach – CPU Resource Utilization

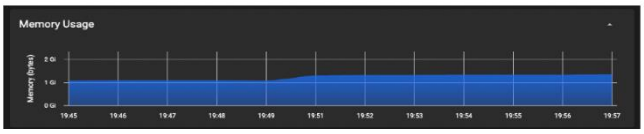

Fig. 4 Centralized Approach – Memory Resource Utilization

2.1. Resource Utilization Metrics

The extraction of CPU Utilization (Figures 3 & 5) and Memory Utilization (Figures 4 & 6) data, made possible by the Resource Matrix API, turns an expedition into the complexities of computational resource dynamics, primarily usage and performance.

The visual contrast between the decentralized and centralized approaches materializes as a captivating and enthralling narrative, articulating the performance distinctions etched into the fabric of every operational node within the Kubernetes cluster.

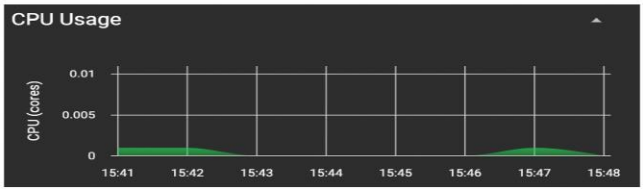

Fig. 5 Decentralized Approach – CPU Resource Utilization

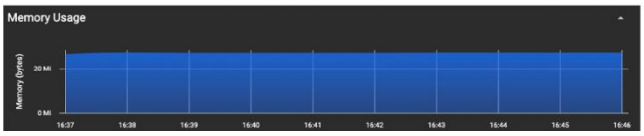

Fig. 6 Decentralized Approach – Memory Resource Utilization

Within the graph presented for respective metrics (Figures 3 & 5, 4 & 6), the X-axis emerges as an elaborate tableau, illustrating the varied workloads at different timescales. At the same time, the Y-axis articulates the intricacies of resource consumption across various operational contexts at the node level and cluster level.

The presented details in the respective figures serve as a crucible for determining performance differences, providing a comparative narrative between the decentralized architecture and centralized counterparts—the control plan and control manager. A decentralized model strategy produces advantageous positive results if specific workloads are very sensitive to resource changes and require rapid adaptability and flexibility. A decentralized strategy excels for highly dynamic varied workloads, whereas a centralized approach performs better compared to a decentralized approach for predictable workloads.

The ensuing results using the different varied workloads emanate a compelling narrative, painting a vivid portrait that unequivocally extols the positive characteristics of the proposed decentralized architectural model. Scrupulous scrutiny of CPU and Memory Utilization metrics across nodes illuminates the ascendancy of distributed controllers at different node levels regarding operational efficiency. The salient feature of lightweight controllers, characterized by a minimal memory footprint, emerges as the lodestar guiding and directing high availability, scalability, and, within the cluster, discernible mitigation in network traffic among nodes.

2.2. Response Time Dynamics

The data unfolds and extends as a dynamic chronicle, highlighting the varying reaction times in response to workloads. The X-axis unfurls a narrative of varying workloads at different timescales, while the Y-axis quantifies the orchestration complexities by depicting the average response time.

The whole depiction over different timescales serves and presents as a dynamic scorecard, revealing and signifying the detailed performance differences between the decentralized and centralized models. Decentralized controllers provide local decision-making authority to individual worker nodes within the cluster. The technique results in faster response times in the case of a decentralized approach compared to the centralized, traditional architectural container orchestration approach for the tasks handled independently at the worker node level.

2.3. Enhanced Security

The decentralized nature of the components deployed at different nodes within the cluster makes it harder to compromise the whole cluster at once. Apart from that, heterogeneity among nodes within a cluster helps create isolation at the infrastructure level to improve security and minimize the attack surface within the cluster. The heterogeneity creates additional challenges and obstacles to attackers for exploiting the vulnerabilities throughout the cluster compared to the centralized traditional approach.

We observed improved network securities in the case of decentralized architecture as most of the network traffic within nodes and regulations through policies created at nodes and pods level with consideration of the decentralized approach, which restricts the vulnerable activities traveling up the ladder. This movement, in the case of the centralized, traditional approach, is fast and spreads across the whole cluster quickly compared to the decentralized approach.

Security through diversity means that the diverse infrastructure and hardware of participating nodes in the cluster make it more difficult for the attackers to use a single strategy or one approach against the exposed vulnerabilities of different nodes.

2.4. Data Protection

The decentralized system helps with data protection because even though one node is compromised or hacked, the system will be intact and operational without impacting overall performance.

2.5. Fault Isolation and Tolerance

The impact of malfunction at one part of the cluster can be easily isolated without impacting the other part of the cluster. The overall functionalities stay healthy and serve the different services simultaneously. This improves the fault tolerance of the cluster as nodes can be removed easily without impacting the overall functionalities, and new nodes can be added easily as well.

2.6. Self-Healing

The decentralized Kubernetes model has the ability to recover from any failure easily and increase resiliency and availability. This characteristic promotes the automated fail-over and dynamic scaling. The decentralized approach concerning distribution, automation, and redundancy helps with self-healing capabilities.

2.7. Network Latency

The decentralized approach significantly reduces network traffic, and most actions can be performed near the user. This characteristic of the decentralized model helps in edge computing, where data should be near the end users, and this helps with enhancing and strengthening security, performance, affordability, and response time.

2.8. Other Key Observations

In terms of operational efficiency and performance, the decentralized architectural approach outperforms its centralized counterparts, the control plan and control manager, by collaborating in a decentralized manner within the cluster.

The distributed controllers are the pinnacle of operational and performance efficiency due to their minimal memory footprint at every node level, which opens up a hitherto unexplored world of high availability and scalability.

The decentralized architectural approach's reliance on the symphony of event-driven communication between and within nodes takes center stage, and performance is improved by showcasing a strong ability to orchestrate transformative transformations from the current states of objects to their intended states.

To sum up, the above-synchronized results and graphical depictions provide an engaging story that unquestionably verifies and affirms the superiority and benefits of the proposed decentralized architecture. This paradigm is an excellent illustration of operational elegance that may be expanded and made publicly accessible, going beyond just being a resource-efficient use. By placing the decentralized orchestration model at the forefront of creating efficiency, scalability, and resilience inside the Kubernetes ecosystem's intricate structure and communication fabric, this thorough analysis and exploration open the door for a paradigm shift.

3. Limitations and Future Directions

Although the research approach used in the study gives insightful, valuable information about deploying the D-K8S at every worker node level in a Kubernetes environment, it's necessary to recognize some of the limitations that might affect how the interpretation of the respective findings related discussions:

3.1. Simplification of Workloads

The experiment employed synthetic and simplified workloads at different selected time scales, which might not fully and accurately represent real-world applications' intricacies and usage patterns.

3.2. Assumed Uniformity in Node Capabilities

The experiments and related test cases were carried out with the heterogeneous nodes in mind, considering probable variances in hardware or resource limits. Node variety and diversity are common and widespread in real-world Kubernetes clusters [3], and considering that, covering all the real-world scenarios is challenging; therefore, findings and conclusions might not fully represent situations and circumstances in which nodes have different capacities and resources.

3.3. Security Scope

It is conceivable that not every security concern related to the distributed architecture design has been thoroughly covered, investigated, and explored by the conducted research. Although the experiment may create security concerns, it is conceivable that not all potential weaknesses and defences have been adequately explored.

3.4. Limited Exploration of Failure Scenarios

The system's resilience to nodes within cluster or controller failures and other failure scenarios were not fully explored and assessed by the different experiments considering limitations concerning different real-world extensive use cases and their associated respective varying infrastructure setup.

4. Conclusion

The findings of this study demonstrate a paradigm-shifting model approach focused on creating and deploying decentralized controllers at every worker node in the Kubernetes ecosystem, enabling communication between and among nodes. With a focus on network traffic reduction and increased security concerns, this unique new methodology offers many advantages. Notable improvements and noteworthy advancements in cost efficiency and resource use

further show the transformative potential of this technology. This decentralized approach heralds a new age in Kubernetes orchestration as we grow technologically and work to increase efficiency while considering different aspects and features. When taken as a whole, the potential advantages of less network traffic, improved security, better use of resources, increased safety, and cost-effectiveness present a strong argument. The flexibility and robustness of this innovative architectural approach are highly advantageous for industries situated at the intersection of edge and cloud computing. The research study illustrates and discusses a strategic and technological accomplishment. The decentralized architecture model is a lighthouse guiding the development and implementation of the Kubernetes ecosystem. With all the advantages described and expanded in this research, it is well-positioned to shape and advance the features of cloud-native and edge computing environments that are secure, effective, and economical.

References

[1] Sisay Tadesse Arzo et al., "MSN: A Playground Framework for Design and Evaluation of MicroServices-Based SdN Controller," *Journal of Network and Systems Management*, vol. 30, pp. 1-31, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[2] Showkat Ahmad Bhat, Ishfaq Bashir Sofi, and Chong- Yung Chi, "Edge Computing and Its Convergence with Blockchain in 5G and Beyond: Security, Challenges, and Opportunities," *IEEE Access*, vol. 8, pp. 205340-205373, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[3] Carmen Carrión, "Kubernetes Scheduling: Taxonomy, Ongoing Issues and Challenges," *ACM Computing Surveys*, vol. 55, no. 7, pp. 1-37, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[4] Nathan F. Saraiva de Sousa et al., "Network Service Orchestration: A Survey," *Computer Communications*, vol. 142-143, pp. 69-94, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[5] Gerard Finol, and Aitor Arjona, "*Study of the Feasibility of Serverless Access Transparency for Python Multiprocessing Applications*," Bachelor Thesis, Rovira i Virgili University (URV) and Open University of Catalonia, pp. 1-89, 2021. [Google Scholar] [Publisher Link]

[6] Kubernetes, The Linux Foundation, 2023. [Online]. Available: https://kubernetes.io/

[7] Silvery Fu et al., "From Kubernetes to Knactor: A State-Centric Rethink of Service Integration," *arXiv*, pp. 1-8, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[8] Sukhpal Singh Gill et al., "Transformative Effects of IoT, Blockchain and Artificial Intelligence on Cloud Computing: Evolution, Vision, Trends and Open Challenges," *Internet of Things*, vol. 8, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[9] Gido´falvy Gordon Zsolt, "*Automating Deployments of Trusted Execution Environments*," Master Thesis, KTH Royal Institute Of Technology, pp. 1-159, 2023. [Google Scholar] [Publisher Link]

[10] Kavya Govindarajan, Chander Govindarajan, and Mudit Verma, "Network Aware Container Orchestration for Telco Workloads," *IEEE 15th International Conference on Cloud Computing*, Barcelona, Spain, pp. 397-406, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[11] Lara Lorna Jiménez, and Olov Schelén, "Docma: A Decentralized Orchestrator for Containerized Microservice Applications," *IEEE Cloud Summit*, Washington, DC, USA, pp. 45-51, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[12] Buket Karakas, "*Enhancing Security in Communication Applications Deployed on Kubernetes: Best Practices and Service Mesh Analysis*," Master's Thesis, Aalto University, pp. 1-61, 2023. [Google Scholar] [Publisher Link]

[13] Deepak Kataria et al., "Artificial Intelligence and Machine Learning," *IEEE Future Networks World Forum*, pp. 1-70, 2022. [Google Scholar] [Publisher Link]

[14] Anders Köhler, "Evaluation of MLOps Tools for Kubernetes: A Rudimentary Comparison between Open Source Kubeflow, Pachyderm and Polyaxon," *Master Programme in Computational Science*, pp. 1-87, 2022. [Google Scholar] [Publisher Link]

[15] Lars Larsson et al., "Decentralized Kubernetes Federation Control Plane*," IEEE/ACM 13th International Conference on Utility and Cloud Computing*, Leicester, UK, pp. 354-359, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[16] Changyuan Lin, and Hamzeh Khazaei, "Modeling and Optimization of Performance and Cost of Serverless Applications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 3, pp. 615-632, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[17] Marko Luksa, *Kubernetes in Action*, Simon and Schuster, pp. 1-624, 2017. [Google Scholar] [Publisher Link]

[18] Anshita Malviya, and Rajendra Kumar Dwivedi, "A Comparative Analysis of Container Orchestration Tools in Cloud Computing," *9th International Conference on Computing for Sustainable Global Development*, New Delhi, India, pp. 698-703, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[19] Enrico Martin, "*Virtualization and Containerization: A New Concept for Data Center Management to Optimize Resources Distribution*," Final Thesis, Ca' Foscari University of Venice, pp. 1-71, 2022. [Google Scholar] [Publisher Link]

[20] Gabriele Monaco, Gautam Gala, and Gerhard Fohler, "Shared Resource Orchestration Extensions for Kubernetes to Support Real-Time Cloud Containers," *IEEE 26th International Symposium on Real-Time Distributed Computing*, Nashville, TN, USA, pp. 97-106, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[21] Mytilinakis Panagiotis, "*Attack Methods and Defenses on Kubernetes*," PhD thesis, University of Piraeus, pp. 1-91, 2020. [Google Scholar] [Publisher Link]

[22] Petter Njerve Rostrup, " *A Distributed-to-Centralized Architectural Model for Smart City Applications and Services through Container Orchestration*," Master's Thesis, NTNU, 2019. [Google Scholar] [Publisher Link]

[23] Mehmet Söylemez, Bedir Tekinerdogan, and Ayça Kolukısa Tarhan, "Challenges and Solution Directions of Microservice Architectures: A Systematic Literature Review," *Applied Sciences*, vol. 12, no. 11, pp. 1-40, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[24] Naeem Firdous Syed et al., "Zero Trust Architecture (ZTA): A Comprehensive Survey," *IEEE Access*, vol. 10, pp. 57143-57179, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[25] Tarik Taleb et al., "Toward Supporting XR Services: Architecture and Enablers," *IEEE Internet of Things Journal*, vol. 10, no. 4, pp. 3567-3586, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[26] Theodoros Theodoropoulos et al., "GreenKube: Towards Greener Container Orchestration using Artificial Intelligence," *IEEE International Conference on Service-Oriented System Engineering*, Athens, Greece, pp. 135-139, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[27] Junzo Watada et al., "Emerging Trends, Techniques and Open Issues of Containerization: A Review," *IEEE Access*, vol. 7, pp. 152443-152472, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[28] Saqing Yang et al., "KubeHICE: Performance-Aware Container Orchestration on Heterogeneous-ISA Architectures in Cloud- Edge Platforms," *IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking*, New York City, NY, USA, pp. 81-91, 2021. [CrossRef] [Google Scholar] [Publisher Link]