Security of Infocommunication Systems and Internet of Things

2025 Vol 3, No 2

https://doi.org/10.31861/sisiot2025.2.02001

Received 12 June 2025; revised 16 November 2025; accepted 21 December 2025; published 30 December 2025

Method of Granular Analysis of Shared Resource Utilization in a Kubernetes Cluster

Mykola Buriak*

Department of the Computer Systems Software, Yuriy Fedkovych Chernivtsi National University, Chernivtsi, Ukraine

*Corresponding author (E-mail: buriak.mykola@chnu.edu.ua)

ABSTRACT In modern computing environments, the Kubernetes platform has become the standard for automated deployment, scaling, and management of containerized applications. With the growing popularity of Kubernetes worldwide, there is an increasing need for accurate and efficient analysis of resource consumption. This is critically important to ensure stable performance, optimal infrastructure utilization, and economically justified capacity planning. However, resource management in Kubernetes is complicated by the dynamic nature of workloads and the interdependence of applications operating within a shared environment. The discovery establishes initial conditions for the analysis, considering the main types of resources: processor, random-access memory, disk space, and network traffic. For each resource, weighting coefficients are defined that reflect its relative importance in the context of different task executions. Additionally, each resource is divided into four states: allocated, reserved, utilized, and free. Such a division allows for a more detailed picture of the actual state of infrastructure usage and supports decision-making based on both technical and economic efficiency.
The primary model focuses on evaluating the resource consumption of a single application at a specific point in time. Within this model, the relationship between reserved and actually utilized resources is examined, enabling the identification of excessive reservation or, conversely, insufficient provisioning. The model forms the foundation for a basic understanding of the application's behavior in the cluster and allows for initial diagnostics of inefficiencies. Further model refinement incorporates changes in resource usage over time. The behavior of a single application is analyzed throughout a specified period, which opens the possibility to identify long-term trends, gradual resource leakage, peak loads, or uneven consumption. This approach significantly improves the accuracy of assessing application performance and supports informed decisions regarding scaling or setting constraints. The final level of the model provides for analyzing resource usage by multiple applications simultaneously over a given period. This allows consideration of the mutual influence between applications, competition for shared resources, and overall environment load. As a result, a comprehensive picture of resource balance is formed, which serves as the basis for intelligent cluster policy planning, workload placement optimization, and service stability assurance. The proposed approach to granular analysis of resource utilization in Kubernetes clusters is a promising research direction. It opens wide opportunities for further development of forecasting models, automated resource management, and the construction of adaptive monitoring systems capable of independently responding to changing cluster loads. The conclusions of the study emphasize the importance of integrating such methods to enhance the efficiency of modern cloud infrastructure operations.

KEYWORDS Kubernetes, resources optimization, containerization, analysis of resources usage, clusterization.

I. INTRODUCTION

The Kubernetes platform has become the standard for managing containerized applications in the current era of rapid development of cloud computing and microservice architecture. However, widespread adoption of Kubernetes in production environments presents several challenges related to the efficient utilization of cluster computing resources-processor time, random access memory (RAM), network bandwidth, disk space, and others. A particularly pressing issue is the fair and productive allocation of shared resources in multi-tenant or multi-user environments without compromising the quality of service for individual applications. Despite Kubernetes' built-in scheduling and resource limitation mechanisms, these often fail to account for the actual consumption profiles, workload dynamics, and interactions between services, which can lead to inefficient infrastructure usage or performance degradation.

The relevance of this research is driven by the need to develop methods for detailed, granular analysis of resource utilization that enable a deeper investigation of individual cluster components behavior, taking into account workload context, deployment topology, priorities, and interdependencies. Such an approach is a crucial step toward improving resource optimization accuracy, analyzing consumption conflicts, and ensuring the stable operation of Kubernetes clusters.

The aim of this article is to develop and describe a method for granular analysis of shared resource utilization in a Kubernetes cluster based on system metrics and contextual analysis. To achieve this aim, the following tasks are set: to define the classification of resources and their characteristics, to formulate mathematical models for analyzing resource consumption for a single application and for multiple applications over time, and to evaluate the prospects of applying the proposed approach to optimize cluster operation.

A review of recent studies shows that previous work has largely focused on global monitoring and basic resource scheduling [1-6], while the issue of deep,

contextual analysis considering the dynamics and interactions of cluster components remains underdeveloped. The method proposed in this article aims to fill this gap by integrating system monitoring with analytical models to achieve greater flexibility and accuracy in Kubernetes resource management.

II. INITIAL CONDITIONS

To conduct a granular analysis of the usage of shared resources in a Kubernetes cluster, a baseline set of characteristics was defined to reflect the state of the infrastructure at the initial stage of the study. The main focus was placed on the following types of resources: central processing unit (CPU), RAM, disk space (Disk), and network resources (Network) [7]. For each resource, four key indicators are considered: Allocated – resources allocated by the cloud provider, the actual resources allocated to nodes – *A*, known immediately after receiving resources from the cloud service provider; Reserved – reserved resources, i.e., the amount of resources explicitly reserved for pods according to the requests specified in the corresponding declarations – *R*, known at the time of application launch; Used – actually used resources at the moment of metric collection – *U*, known during application operation; Free – free resources that remain unused due to reservation and have not yet been utilized – *F*, unknown, calculated based on all previous indicators Fig. 1 [8].

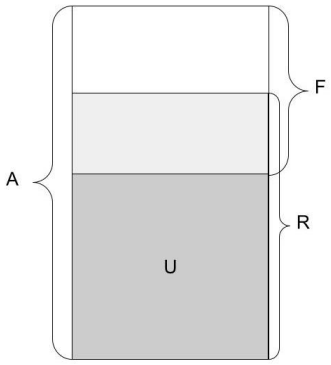

FIG. 1. Schematic representation of key resource metrics.

The concepts of allocated, reserved, used, and free units of CPU – *Ac*, *Rc*, *Uc*, *Fc*, RAM – *Ar*, *Rr*, *Ur*, *Fr*, Disk – *Ad*, *Rd*, *Ud*, *Fd*, and Network – *An*, *Rn*, *Un*, *Fn* have been introduced accordingly Fig. 2.

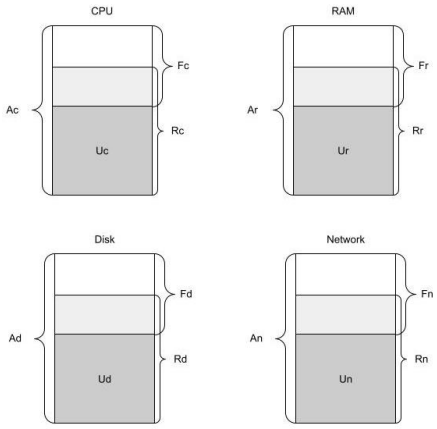

FIG. 2. Schematic representation of key metrics corresponding to resources.

All types of resources are measured in their specific units (for example, CPU – in cores or millicores, RAM – in megabytes or gigabytes, Disk – in gigabytes, Network – in megabits per second), which directly affects the accuracy and completeness of the overall infrastructure resource analysis. Additionally, among cloud computing providers, the ratio of allocated resources varies depending on the instance type, selected configuration, or the provider's business model. For example, some types of virtual machines may have a higher share of CPU per unit of RAM, or conversely, a larger amount of memory with lower computing power [9, 10]. In this regard, the concept of a weighting coefficient is introduced for each type of resource: *c* – CPU, *r* – RAM, *d* – disk, *n* – network.

These coefficients allow for the normalization of heterogeneous resources to a common comparison or calculation scale, which is especially important when constructing aggregated load metrics, performing resource balancing, and analyzing resource usage.

III. PRIMARY MATHEMATICAL MODEL FOR ANALYZING RESOURCES OF A SINGLE APPLICATION

Given the initial conditions – namely, the set of data reviewed in the previous section, including allocated, reserved, used, and free resource units, as well as their weighting coefficients – an initial equation can be formulated to determine the used resources Eq. (1).

$$U = Uc * c + Ur * r + Ud * d + Un * n, \qquad (1)$$

where c, r, d, n – weighting coefficients, Uc, Ur, Ud, Un – utilized resources.

This equation serves as a fundamental tool for the formalized assessment of the current system load. Such an approach enables a systematic and quantitative characterization of the infrastructure's load level, identification of potential bottlenecks or inefficient resource usage. The resulting equation serves as a starting point for further analysis – both for operational monitoring and for forecasting, resource optimization, and making informed decisions regarding scaling, redistribution, or updating the architecture of the computing environment.

The next step, based on the same information, is to formulate similar equations for reserved Eq. (2) and allocated Eq. (3) resources.

$$R = Rc * c + Rr * r + Rd * d + Rn * n, \qquad (2)$$

where Rc, Rr, Rd, Rn – reserved resources.

$$A = Ac * c + Ar * r + Ad * d + An * n, \qquad (3)$$

where Ac, Ar, Ad, An – allocated resources.

These equations allow for a quantitative description of the amount of resources reserved for guaranteed use, as well as those administratively or systemically assigned to specific tasks or components. Formalizing these relationships is important for a comprehensive understanding of the overall resource distribution in the system and for ensuring accurate comparison between actual usage, planning, and resource allocation.

The concept of free resources was defined in the previous section; however, they are unknown and depend on the actually used resources and the allocated resources.

Let us formulate equations that allow the calculation of free resources for CPU Eq. (4), RAM Eq. (5), Disk Eq. (6), Network Eq. (7), as well as an overall free resources metric Eq. (8).

$$Fc = Ac - Uc, \tag{4}$$

$$Fr = Ar - Ur, \tag{5}$$

$$Fd = Ad - Ud, \tag{6}$$

$$Fn = An - Un, \tag{7}$$

$$F = Fc * c + Fr * r + Fd * d + Fn * n, \tag{8}$$

where Fc, Fr, Fd, Fn – free resources.

At this stage, it is already possible to perform a real-time analysis of the efficiency of resource usage by a single containerized application. For example, if the overall metric of used resources exceeds the metric of reserved resources Eq. (9) Fig. 3, this indicates either insufficient reserved resources or a suboptimal application algorithm that consumes more resources than expected, requiring a detailed analysis of the application's algorithm.

Conversely, if the overall metric of used resources is significantly lower than the metrics of reserved and allocated resources Eq. (10) Fig. 4, this indicates resource underutilization, and it is likely possible to reduce the level of reserved resources, which would in turn decrease the amount of allocated resources.

$$U > R. \tag{9}$$

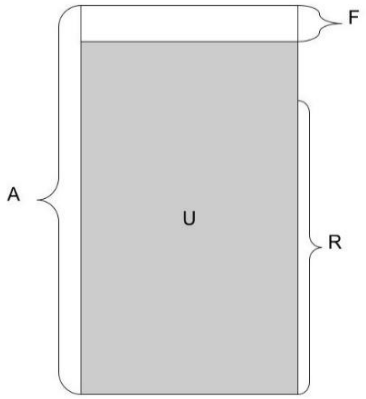

FIG. 3. Schematic representation of suboptimal resource usage.

$$U < R < A. \tag{10}$$

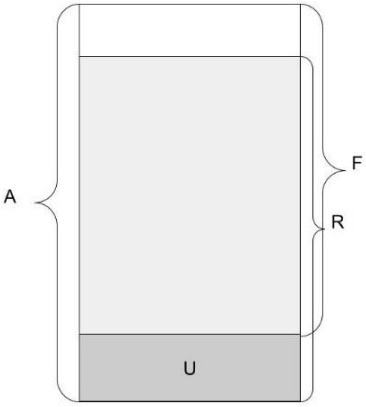

FIG. 4. Schematic representation of resource underutilization.

IV. MATHEMATICAL MODEL FOR ANALYZING RESOURCES OF A SINGLE APPLICATION OVER A PERIOD OF TIME

In the previous section, a mathematical model was developed for analyzing resource usage by a single containerized application at a fixed point in time. However, this approach has limited practical value since application load is dynamic and can vary significantly depending on the time of day, external events, user activity, and other factors. At one moment, the application may consume minimal resources, while at another it may reach peak loads. To improve the reliability and stability of analysis results, it is advisable to introduce into the mathematical model the concept of numerical series describing resource usage changes over time. This allows analysis not only at a specific moment but also tracking the dynamics of resource usage over a certain period – both short-term (e.g., several minutes) and long-term (hours, days, or even weeks). Thus, we obtain a time-based model of resource usage – *T*. This significantly increases the analytical value of the model and makes it suitable for real-world scenarios such as monitoring, automatic scaling, and optimization of computing infrastructure Fig. 5.

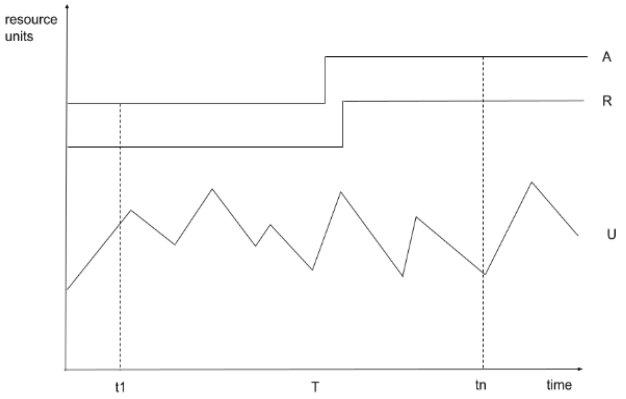

FIG. 5. Schematic representation of resources on a timeline.

To use time series in the model, it is necessary to have historical data on resource consumption over a certain period, which must be collected at consistent intervals and maintain a high level of consistency. Such series are formed separately for each type of resource: CPU Eq. (11), RAM Eq. (12), Disk Eq. (13), and Network Eq. (14). Obtaining this data is possible thanks to metric exporters integrated with the Kubernetes cluster. These can be optionally installed depending on the needs of the user or system administrator. The most common solutions include Prometheus Node Exporter, kube-state-metrics, cAdvisor, and others. They enable automatic and regular collection of system resource status information both at the instance level and at the level of individual containers or pods. Thus, having a metric collection infrastructure is a necessary prerequisite for building and applying time series in the analysis model, which opens up opportunities for deeper and more accurate monitoring, load optimization, and data-driven decision-making.

Having obtained all the necessary time-dependent equations, it is possible to construct a mathematical model incorporating time series Eq. (15).

$$U_t = Uc_t * c + Ur_t * r + Ud_t * d + Un_t * n, \quad t \in [1;T], \tag{11}$$

where T – period.

$$R_t = Rc_t * c + Rr_t * r + Rd_t * d + Rn_t * n, \quad t \in [1;T]. \tag{12}$$

$$A_t = Ac_t * c + Ar_t * r + Ad_t * d + An_t * n, \quad t \in [1;T]. \tag{13}$$

$$\begin{gathered} F_t = Fc_t * c + Fr_t * r + Fd_t * d + Fn_t * n, \\ Fc_t = Ac_t - Uc_t, \\ Fr = Ar_t - Ur_t, \\ Fd_t = Ad_t - Ud_t, \\ Fn_t = An_t - Un_t, \\ t \in [1;T]. \end{gathered} \tag{14}$$

$$\begin{gathered} \sum_{t=1}^{T} U_t, \\ \sum_{t=1}^{T} R_t, \\ \sum_{t=1}^{T} A_t, \\ \sum_{t=1}^{T} F_t, \\ t \in [1;T]. \end{gathered} \tag{15}$$

With the established current mathematical model integrating time series of resource usage, it is possible to conduct a consistent and comprehensive analysis of the load generated by a single containerized application over a selected time period. This approach allows taking into account not only instantaneous metrics but also changes in resource consumption over time, enabling the identification of cyclical fluctuations, peak loads, and idle periods.

V. MATHEMATICAL MODEL FOR ANALYZING RESOURCES OF MULTIPLE APPLICATIONS OVER A PERIOD OF TIME

Typically, a Kubernetes cluster is not designed to run just a single application, as such an approach is economically inefficient and impractical considering infrastructure and management costs. Instead, Kubernetes is built as a platform for scalable management of a large number of applications simultaneously – P, enabling the most efficient use of available computing resources, load distribution, and ensuring high availability and fault tolerance. Considering this, the next logical step in the development of the mathematical model is to introduce parameters that reflect the dependence of used Eq. (16) and reserved Eq. (17) resources on a specific application (pod). This will allow modeling and analyzing resource allocation among many concurrently running applications, taking into account their individual characteristics and requirements.

$$\begin{gathered} U_t(pod) = Uc_t(pod) * c + Ur_t(pod) * r \\ + Ud_t(pod) * d + Un_t(pod) * n, \\ t \in [1;T], \\ pod \in [1;P], \end{gathered} \tag{16}$$

where $Uc(pod), Ur(pod), Ud(pod), Un(pod)$ – utilized resources, P – pods count.

$$\begin{gathered} R_t(pod) = Rc_t(pod) * c + Rr_t(pod) * r \\ + Rd_t(pod) * d + Rn_t(pod) * n, \\ t \in [1;T], \\ pod \in [1;P], \end{gathered} \tag{17}$$

where $Rc(pod), Rr(pod), Rd(pod), Rn(pod)$ – reserved resources.

The obtained equations will be presented in a form to be applied in the mathematical model Eq. (18).

$$\begin{gathered} US = \sum_{pod=1}^{P} \sum_{t=1}^{T} U_t(pod), \\ RS = \sum_{pod=1}^{P} \sum_{t=1}^{T} R_t(pod), \\ t \in [1;T], \\ pod \in [1;P]. \end{gathered} \tag{18}$$

Allocated and free resources do not have a direct dependency on specific applications, since the same instance allocated by the cloud provider can serve multiple applications simultaneously. This means that allocated resources represent a shared pool available to all deployed applications, while free resources are those remaining available after distribution and actual consumption. Therefore, the equations describing the mathematical model in the context of allocated Eqs. (19) and (20) and free Eqs. (21) and (22) resources should not include parameters dependent on individual applications (pods). Instead, these equations reflect the overall availability and residual resources within a given instance.

$$A_t = Ac_t * c + Ar_t * r + Ad_t * d + An_t * n, \quad t \in [1;T]. \tag{19}$$

$$\begin{gathered} P \sum_{t=1}^{T} A_t, \\ t \in [1;T]. \end{gathered} \tag{20}$$

$$\begin{gathered} F_t = Fc_t * c + Fr_t * r + Fd_t * d + Fn_t * n, \\ Fc_t = Ac_t - \sum_{pod=1}^{P} Uc_t\,(pod), \\ Fr_t = Ar_t - \sum_{pod=1}^{P} Ur_t\,(pod), \\ Fd_t = Ad_t - \sum_{pod=1}^{P} Ud_t\,(pod), \\ Fn_t = An_t - \sum_{pod=1}^{P} Un_t(pod), \\ t \in [1;T], \\ pod \in [1;P]. \end{gathered} \tag{21}$$

$$FS = \sum_{t=1}^{T} F_t, \quad t \in [1;T]. \tag{22}$$

Having obtained the equations, the final mathematical model can be constructed, enabling the calculation of resource usage and the analysis of their utilization efficiency Eq. (23).

$$\begin{gathered} US = \sum_{pod=1}^{P} \sum_{t=1}^{T} U_t(pod), \\ RS = \sum_{pod=1}^{P} \sum_{t=1}^{T} R_t(pod), \\ AS = \sum_{t=1}^{T} A_t, \\ FS = \sum_{t=1}^{T} F_t, \\ t \in [1;T], \\ pod \in [1;P]. \end{gathered} \tag{23}$$

VI. EXAMPLES OF MODEL USAGE

The developed mathematical model can be applied to various types of resource usage analysis, significantly expanding the capabilities for monitoring and optimizing computing infrastructure.

First, the model enables an overall assessment of resource usage within the entire Kubernetes cluster or a specific instance, allowing determination of system load levels, availability of free resources, and scaling potential. For example, if the total resource usage by a group of applications approaches the allocated resources Eq. (24), this indicates that resources will soon need to be increased or the application algorithms reviewed for potential inefficiencies in resource utilization.

$$\sum_{pod=1}^{P} \sum_{t=1}^{T} U_t(pod) > \sum_{t=1}^{T} A_t. \tag{24}$$

Second, the model allows evaluation with respect to specific applications (pods), enabling identification of which services or applications consume the largest share of resources and supporting decisions regarding prioritization of resource allocation or optimization of their operation. For example, if Application 3 uses more resources than reserved for it Eq. (25), this indicates that resource consumption was incorrectly forecasted or there are issues with the application's algorithm.

$$\sum_{t=1}^{T} U_t(3) > \sum_{t=1}^{T} R_t(3). \tag{25}$$

Third, integrating time series into the model opens up the possibility to analyze resource usage dynamics over time. This enables tracking changes in load during different periods, identifying peak and minimum values, and forecasting future resource needs – an especially important capability for effective planning and automatic scaling. For example, if it is observed that resource usage significantly fluctuates at certain intervals, this indicates the regular occurrence of some internal or external event, such as user activity or a scheduled task within the application.

VII. CONCLUSIONS

The current method significantly expands the capabilities for analyzing resource usage in a Kubernetes cluster, enabling more accurate, flexible, and adaptive management of computing resources. This approach scales well and can be fully automated through a Kubernetes operator that handles the collection, processing, and real-time updating of metrics. The calculation results – such as aggregated indicators, efficiency assessments, and residual resource evaluations – can be presented in an intuitive graphical format using Grafana, greatly simplifying load monitoring and decision-making. Moreover, the mathematical model can be extended to generate automatic recommendations that take into account the current and forecasted system state. Such recommendations, for example regarding scaling, redistribution, or resource optimization, can also be visualized in Grafana, making the system not only analytical but also practically oriented.

The next stage in the development of the proposed method is the introduction of the concept of a heterogeneous environment into the model, which is characteristic of most real cloud infrastructures. This will allow accounting for the diversity of instance types provided by cloud providers, each with different performance profiles, resource capacities, and costs. An important component will also be the modeling of application distribution mechanisms across instances within the Kubernetes cluster, taking into account placement policies, resource constraints, and scaling logic. Thus, the model will move closer to representing a complete and realistic picture of the operation of a complex, dynamic cloud infrastructure.

AUTHOR CONTRIBUTIONS

M.B. – conceptualization, methodology, investigation; writing (original draft preparation), writing (review and editing).

COMPETING INTERESTS

The author declare no competing interests.

REFERENCES

[1] G. Turin, A. Borgarelli, S. Donetti, F. Damiani, E. B. Johnsen, and S. L. Tapia Tarifa, "Predicting resource consumption of Kubernetes container systems using resource models," *Journal of Systems and Software*, vol. 203, p. 111750, Sep. 2023, doi: 10.1016/j.jss.2023.111750.

[2] V. Medel, R. Tolosana-Calasanz, J. Á. Bañares, U. Arronategui, and O. F. Rana, "Characterising resource management performance in Kubernetes," *Computers & Electrical Engineering*, vol. 68, pp. 286–297, May 2018, doi: 10.1016/j.compeleceng.2018.03.041.

[3] M.-N. Tran and Y. Kim, "Optimized resource usage with hybrid auto-scaling system for knative serverless edge computing," *Future Generation Computer Systems*, vol. 152, pp. 304–316, Mar. 2024, doi: 10.1016/j.future.2023.11.010.

[4] H. Guo, H. Cao, J. He, X. Liu, and Y. Shi, "POBO: Safe and optimal resource management for cloud microservices," *Performance Evaluation*, vol. 162, p. 102376, Nov. 2023, doi: 10.1016/j.peva.2023.102376.

[5] B. Jeong, S. Baek, S. Park, J. Jeon, and Y.-S. Jeong, "Stable and efficient resource management using deep neural network on cloud computing," *Neurocomputing*, vol. 521, pp. 99–112, Feb. 2023, doi: 10.1016/j.neucom.2022.11.089.
[6] G. Marques, C. Senna, S. Sargento, L. Carvalho, L. Pereira, and R. Matos, "Proactive resource management for cloud of services environments," *Future Generation Computer Systems*, vol. 150, pp. 90–102, Jan. 2024, doi: 10.1016/j.future.2023.08.005.
[7] "Resource Management for Pods and Containers," *Kubernetes*, Jan. 28, 2025. [Online]. Available: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/#resource-types. Accessed: Apr. 12, 2025.
[8] "Resource Management for Pods and Containers," *Kubernetes*, Jan. 28, 2025. [Online]. Available: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/#pod-level-resource-specification. Accessed: Apr. 10, 2025.
[9] J. Barr, "Choosing the Right EC2 Instance Type for Your Application | Amazon Web Services," *Amazon Web Services*, May 14, 2013. [Online]. Available: https://aws.amazon.com/blogs/aws/choosing-the-right-ec2-instance-type-for-your-application/. Accessed: Apr. 10, 2025.
[10] "It just got easier to discover and compare EC2 instance types | Amazon Web Services," *Amazon Web Services*, Nov. 26, 2019. [Online]. Available: https://aws.amazon.com/blogs/compute/it-just-got-easier-to-discover-and-compare-ec2-instance-types/. Accessed: Apr. 10, 2025.

Mykola Buriak

PhD student at Computer Systems Software Department, Yuriy Fedkovych Chernivtsi National University. Practicing DevOps Engineer with over 7 years of experience. Master of Computer Sciences.

ORCID ID: 0009-0006-7887-0091

Метод гранульованого аналізу використання спільних ресурсів Kubernetes кластера

Микола Буряк*

Кафедра програмного забезпечення комп'ютерних систем, Чернівецький національний університет імені Юрія Федьковича, Чернівці, Україна

*Автор-кореспондент (Електронна адреса: buriak.mykola@chnu.edu.ua)

АНОТАЦІЯ У сучасних обчислювальних середовищах платформа Kubernetes стала стандартом для автоматизованого розгортання, масштабування та управління контейнеризованими застосунками. Зі зростанням популярності Kubernetes у світі постає дедалі гостріша потреба в точному й ефективному аналізі споживання ресурсів. Це критично важливо для забезпечення стабільної продуктивності, оптимального використання інфраструктури та економічно обґрунтованого планування потужностей. Однак керування ресурсами в Kubernetes ускладнюється через динамічну природу навантажень і взаємозалежність застосунків, які функціонують у спільному середовищі. У дослідженні встановлюються початкові умови для проведення аналізу, де розглядаються основні типи ресурсів: процесор, оперативна пам'ять, дисковий простір і мережевий трафік. Для кожного ресурсу визначаються вагові коефіцієнти, що відображають його відносну значущість у контексті виконання завдань різних типів. Крім того, кожен ресурс поділяється на чотири стани: виділені, зарезервовані, використані та вільні. Такий поділ дозволяє отримати більш деталізовану картину фактичного стану використання інфраструктури та приймати рішення з урахуванням як технічної, так і економічної ефективності. Первинна модель аналізу зосереджується на оцінці ресурсного споживання одного окремого застосунку в конкретний момент часу. У рамках цієї моделі розглядається співвідношення між зарезервованими та фактично використаними ресурсами, що дозволяє виявити надлишкове резервування або навпаки – недостатню забезпеченість. Модель формує основу для базового розуміння поведінки застосунку у кластері та дає змогу проводити початкову діагностику неефективностей. Подальше ускладнення моделі дозволяє враховувати зміну використання ресурсів у часі. Поведінка одного застосунку аналізується протягом заданого періоду, що відкриває можливість виявляти довгострокові тренди, поступове витікання ресурсів, пікові навантаження чи нерівномірність споживання. Такий підхід значно підвищує точність оцінки ефективності роботи застосунку та дозволяє приймати обґрунтовані рішення щодо масштабування або налаштування обмежень. Останній рівень моделі передбачає аналіз використання ресурсів багатьма застосунками одночасно протягом певного періоду. Це дозволяє враховувати взаємний вплив між застосунками, конкуренцію за спільні ресурси та загальну завантаженість середовища. В результаті формується цілісна картина ресурсного балансу, яка є основою для розумного планування кластерних політик, оптимізації розміщення робочих навантажень та забезпечення стабільності обслуговування. Запропонований підхід до гранульованого аналізу використання ресурсів у Kubernetes-кластері є перспективним напрямом дослідження. Він відкриває широкі можливості для подальшого розвитку моделей прогнозування, автоматизованого керування ресурсами та побудови адаптивних систем моніторингу, здатних самостійно реагувати на зміну навантажень у кластері. Висновки дослідження підкреслюють важливість інтеграції таких методів для підвищення ефективності експлуатації сучасних хмарних інфраструктур.

КЛЮЧОВІ СЛОВА Kubernetes, оптимізація ресурсів, контейнеризація, аналіз використання ресурсів, кластеризація.

This article is licensed under a Creative Commons Attribution 4.0 International License. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.