УДК 62-503.5
DOI: 10.25140/2411-5363-2021-1(23)-103-108

Олексій Сопов, Анна Цитовцева

ОСОБЛИВОСТІ МАСШТАБУВАННЯ КОНТЕЙНЕРНОГО НАВАНТАЖЕННЯ НА БАЗІ СИСТЕМИ KUBERNETES

На сьогодні технологія контейнеризації набуває широкого поширення, та проблема масштабування є однією з найбільш важливих для підвищення продуктивності систем. Kubernetes є передовим рішенням для керування контейнерами, проте проблема масштабування залишається мало описаною та дослідженою. У цій роботі досліджено особливостей масштабування контейнерного навантаження на базі системи Kubernetes. Розкрито основні операції для досягнення горизонтального та вертикального масштабування. Описані властивості масштабованості системи Kubernetes. Стаття є оглядовою.

Ключові слова: *Kubernetes; мікросервіс; контейнерне навантаження; масштабування; Docker.*
Рис.: 4. Бібл.: 6.

Актуальність теми дослідження. На сьогодні виконання програмних застосунків в умовах контейнерної віртуалізації надає певні переваги як із фінансового боку, так і з боку продуктивності, відмовостійкості та швидкості роботи. Тому сфера розробки програмного забезпечення з використанням технології контейнеризації набуває широкого поширення в багатьох сферах бізнесу

З огляду на часті зміни навантаження на програмні продукти й різні умови їх функціонування, завдання масштабування корисного навантаження стає дедалі більш актуальним. Проблема масштабування є однією з найбільш важливих для розширення обсягу виконаних задач за період часу, тому важливо використовувати правильні схеми масштабування застосунків.

Kubernetes є передовим рішенням при роботі з контейнерами та забезпечує відносно легкі для адаптації механізми керування, проте проблема масштабування залишається слабо описаною та дослідженою, що підвищує складність переходу бізнесу на контейнерні технології, зокрема, із використанням Kubernetes.

Постановка проблеми. В останній час триває перехід від класичної монолітної архітектури побудови систем до сервіс-орієнтованої, або ж, найчастіше, до мікросервісної архітектури для забезпечення високого рівня доступності та відмовостійкості. Однією з найбільших переваг використання такого підходу є можливість масштабування для досягнення адекватної реакції, з огляду на часті зміни навантаження та різні умови функціонування системи загалом.

Беручи до уваги переваги використання контейнерів, окремим мікросервісом найчастіше виступає один або група об'єднаних контейнерів, які виконують ту, чи іншу задачу. Проте керування такою системою з плином часу стає доволі складним та вимагає значних зусиль. Сервіс Kubernetes [1] виконує більшість задач із керування контейнерами, їх життєвим циклом та версіями, тому є одним із найзручніших у цій сфері.

У Kubernetes корисне навантаження та інфраструктура концептуально розділені, тому є можливість виконувати масштабування на кожній складовій окремо, що призведе до появи більше ніж двох операцій для досягнення масштабування. Хоча система Kubernetes є достатньо гнучкою, проте вона була створена нещодавно, та такі особливості масштабування із розділенням концепцій є досі не описаними.

Аналіз останніх досліджень і публікацій. З аналізу літературних джерел можна дійти висновку, що в останній час з'явилась загальна тенденція міграції до хмари та використання віртуальних машин [2]. Проблема оптимального масштабування із використанням віртуальних машин вирішена достатньо широко [3; 4]. Архітектурний стиль мікропослуг та використання контейнерного навантаження привернули значну увагу, особливо на базі системи Kubernetes.

Виділення недосліджених частин загальної проблеми. Можна зробити висновок, що Kubernetes надає значні переваги у швидкості розробки та розгортанні застосунків на базі контейнерів, проте проблема масштабування до цього часу не отримувала значної уваги з боку наукових кіл.

Мета статті є дослідження особливостей масштабування контейнерного навантаження на базі системи Kubernetes. Розкрити основні операції для досягнення горизонтального та вертикального масштабування із розділенням концепцій корисного навантаження та інфраструктури. Описати властивість масштабованості системи Kubernetes.

Виклад основного матеріалу. У класичному розумінні масштабування застосунків поділяється на вертикальне та горизонтальне. Ці вектори масштабування є основоположними та мають різні варіації в різних системах.

Горизонтальне масштабування полягає у збільшенні кількості одиниць, що містять у собі додаток. Виконується розбиття системи на більш дрібні структурні компоненти та рознесення їх по окремих обчислювальних одиницях, або їх групах, і збільшення кількості одиниць, що паралельно виконують ту ж саму функцію.

Вертикальне масштабування полягає у збільшенні ресурсів одиниці, на якій виконується застосунок. Тобто збільшення продуктивності кожного компонента системи з метою підвищення загальної продуктивності.

При роботі в умовах контейнерної віртуалізації необхідно розуміти основні змінні, які можуть використовуватися для горизонтального та вертикального масштабування. У цьому випадку одиницею, що виконує застосунок є, безпосередньо, контейнер, який розгорнуто із деякого зображення контейнера, тобто деяка ізольована оболонка вказаного програмного коду та його залежностей.

Горизонтальне масштабування у випадку із контейнерами полягає в наступному: додається новий вузол, а саме, додається новий контейнер, який розпочато із того самого зображення, з якого розпочато і перший контейнер. Контейнер може бути розгорнуто на будь-якій фізичній або віртуальній машині, де досягнуті необхідні умови, наприклад, встановлено Docker daemon при роботі із системою контейнерної віртуалізації Docker [5].

Вертикальне масштабування полягає в зміні мінімальної/максимальної кількості ресурсів для контейнера. Можливість обмежити використання ресурсів окремим контейнером ж є однією з найважливіших складових контейнерної віртуалізації. Під ресурсами мається на увазі: процесорний час (CPU), кількість оперативної пам’яті (RAM), ресурси диску (iops). Обмеження ресурсів є необхідністю, адже необхідно контролювати максимальну кількість, яку використовує контейнер, щоб не допустити ситуації, коли один контейнер займає більшу частину ресурсів. За замовчуванням кількість ресурсів у контейнера необмежена, проте є можливість це налаштовувати за допомогою параметрів --cpus, --memory, --device-read-bps тощо.

Незважаючи на простоту розгортання контейнера на будь-якій обчислювальній одиниці за допомогою спеціалізованих програмних засобі, таких як Docker, перед розробниками зазвичай стають більш складні задачі: підтримка великої кількості таких контейнерів, підтримка контейнерів різного розміру, різного навантаження та їх горизонтальне та вертикальне масштабування.

Для вирішення більшості вищенаведених задач були розроблені платформи, такі як Kubernetes, що значно полегшують роботу із великою кількістю контейнерів та їх контролем. Основні структурні компоненти Kubernetes зображено на рис. 1 (для простоти більшу частину, яка не приймає участь у масштабуванні було опущено).

Тобто контейнери виконуються у середині так званих под (англ. Pod), що є найменшими одиницями керування у Kubernetes. Найчастіше, один под містить у собі лише один контейнер. Тобто у Kubernetes контейнер знаходиться всередині окремої найменшої одиниці, що може бути розгорнута, пода. Саме Pod є об’єктом для керування при масштабуванні в Kubernetes.

У Kubernetes є можливість керувати ресурсами контейнера в середині поду. Аналогічно, як і у Docker можливо керувати основними ресурсами, а саме: процесорний час (CPU), кількість оперативної пам'яті (RAM), ресурси диску (iops) [6]. Проте особливість роботи Kubernetes полягає в тому, що необхідно вказувати два параметри для ресурсів. Перший — ліміт, тобто, скільки максимально ресурсів буде використано контейнером. Другий — запит, тобто скільки мінімально ресурсів необхідно для роботи контейнера та без яких ресурсів його не буде розгорнуто на кластері.

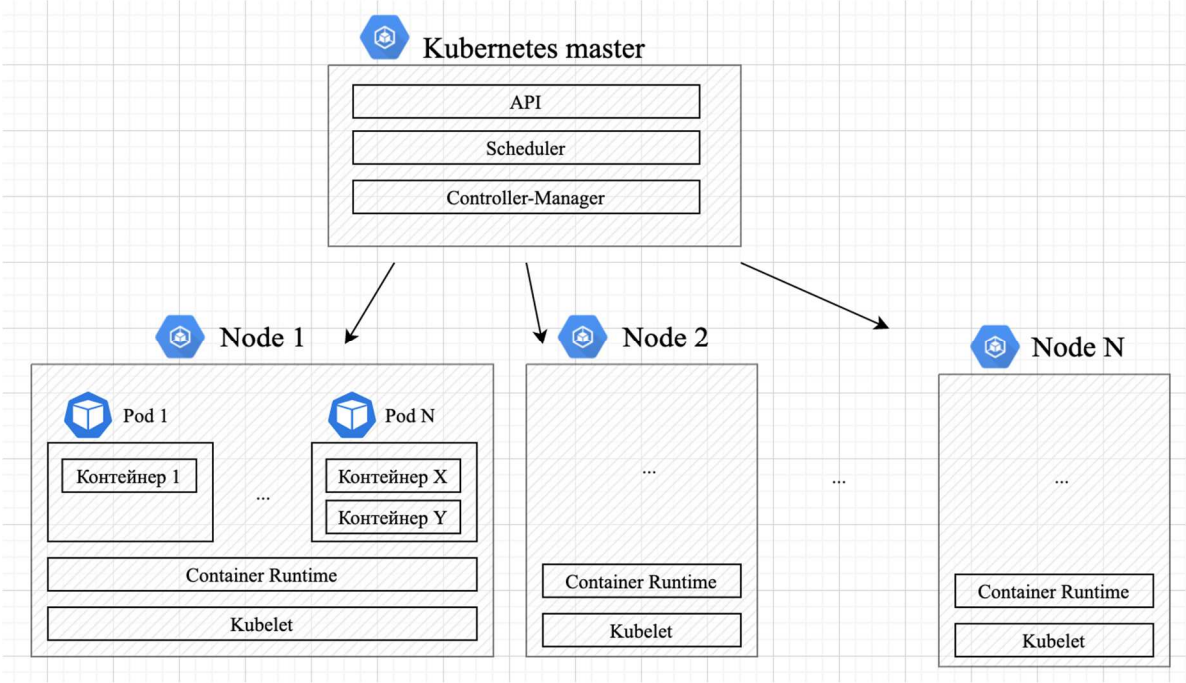

Рис. 1. Схема роботи Kubernetes

Однією з одиниць керування є ресурси контейнера всередині под: процесорний час, кількість оперативної пам'яті та ресурси диску, що можливо використовувати при виконанні масштабування.

З іншого боку, поди розгортаються на так званих нодах. Нодами можуть виступити фізичні або віртуальні машини, залежно від конфігурації кластера. Фізичні та віртуальні машини мають власну ресурсну ємність та визначають максимальну кількість под, що може бути розгорнута на них. Основними ресурсами керування для под є класичні технічні характеристики фізичної або віртуальної машини: процесорний час, кількість оперативної пам'яті та ресурси диску.

Ресурси контейнера та ресурси ноди описують необхідні ресурси для заданого навантаження та ресурсну ємність кластера та є основними змінними в роботі із кластером. Тому для масштабування використано саме ці змінні.

Тобто підсумовуючи, завдяки концепції Kubernetes у розділенні інфраструктури та додатку, два класичних типи масштабування: горизонтальне та вертикальне розширюються у двох розрізах: у розрізах інфраструктури (зміна ресурсів нод у кластері та їх кількості) та корисного навантаження (зміна ресурсів контейнерів та їх кількості).

Горизонтальне масштабування корисного навантаження в Kubernetes полягає у зміні кількості под із тим самим додатком, тобто його реплікація. Тобто той самий додаток буде виконуватися паралельно, у різних подах, завдяки чому підвищується продуктивність та максимальна пропускна здатність.

З боку інфраструктури горизонтальне масштабування полягає в додаванні нових нод до кластера, на яких можливе буде виконання нових под. Тобто збільшується кількість ресурсів, на яких можуть бути розташовані поди.

Масштабованість у цьому контексті означає можливість додавати до системи нові вузли: ноди та поди для збільшення загальної продуктивності. Це найпростіший спосіб масштабування, оскільки не вимагає ніяких змін в прикладних програмах, що працюють на таких системах. Схему горизонтального масштабування у Kubernetes у двох розрізах показано на рис. 2. Як видно, основними компонентами для горизонтального масштабування є поди (у розрізі горизонтального масштабування корисного навантаження) та ноди (у розрізі горизонтального масштабування інфраструктури).

Отже, горизонтальне масштабування у Kubernetes може проводитися у двох розрізах: корисного навантаження та інфраструктурного та у двох сегментах: зміні кількості под та зміні кількості нод.

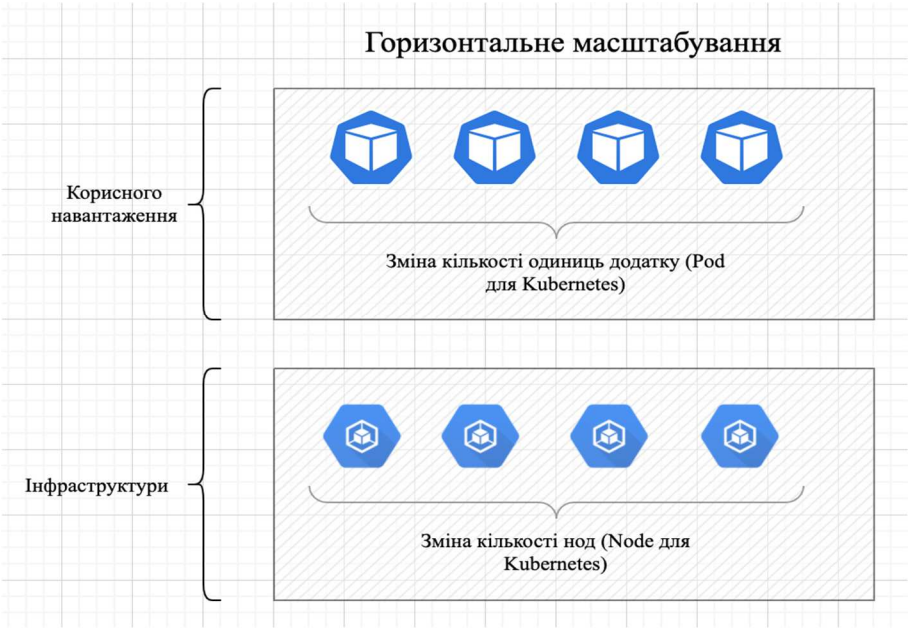

Рис. 2. Схема горизонтального масштабування у Kubernetes

Вертикальне масштабування корисного навантаження в Kubernetes полягає у зміні кількості ресурсів, які може зайняти контейнер. Тобто, у зміні кількості дозволеного використання процесорного часу, оперативної пам'яті, мережевих та дискових ресурсів.

Вертикальне масштабування в розрізі інфраструктури полягає у зміні типу віртуальної машини, на якій виконуються контейнери, тобто, зміні потужності та кількості оперативної пам'яті на цій віртуальній машині, що дозволить змінити кількість контейнерів, що можуть бути розгорнуті.

Масштабованість у цьому контексті означає можливість замінювати в існуючій системі автоматичного компоненти більш потужними і швидкими в міру зростання вимог і розвитку технологій. Цей спосіб масштабування може вимагати внесення змін до програми, щоб програми могли повною мірою користуватися дедалі більшій кількості ресурсів. Схему вертикального масштабування у Kubernetes у двох розрізах показано на рис. 3.

Отже, вертикальне масштабування у Kubernetes може проводитися у двох розрізах: корисного навантаження та інфраструктурного у двох сегментах: зміні ресурсів одиниці поду та зміні ресурсів одиниці ноди.

Такі схеми масштабування є основою для планування ресурсної ємності під задане навантаження в умовах контейнерної віртуалізації на платформі Kubernetes, тому можуть бути операціями для стратегій планування ресурсів та складовими у різних алгоритмах планування.

Рис. 3. Схема вертикального масштабування у Kubernetes

Підсумовуючи вищенаведені особливості масштабування, можна дійти висновку, що класичне масштабування (вертикальне та горизонтальне) у Kubernets виконується за допомогою чотирьох незалежних операцій (схематично такі операції показано на рис. 4):

- зміна кількості под (горизонтальне);
- зміна кількості нод (горизонтальне);
- зміна розміру контейнера, та, відповідно поду (вертикальне);
- зміна розміру машини, на якій підіймається контейнери (вертикальне).

Масштабованість Kubernetes проявляється у двох розрізах: корисного навантаження та інфраструктури. Масштабованість корисного навантаження полягає у можливості додати нові вузли, тобто, поди та ноди, а вертикального – у відповідному збільшенні ресурсів вузлів.

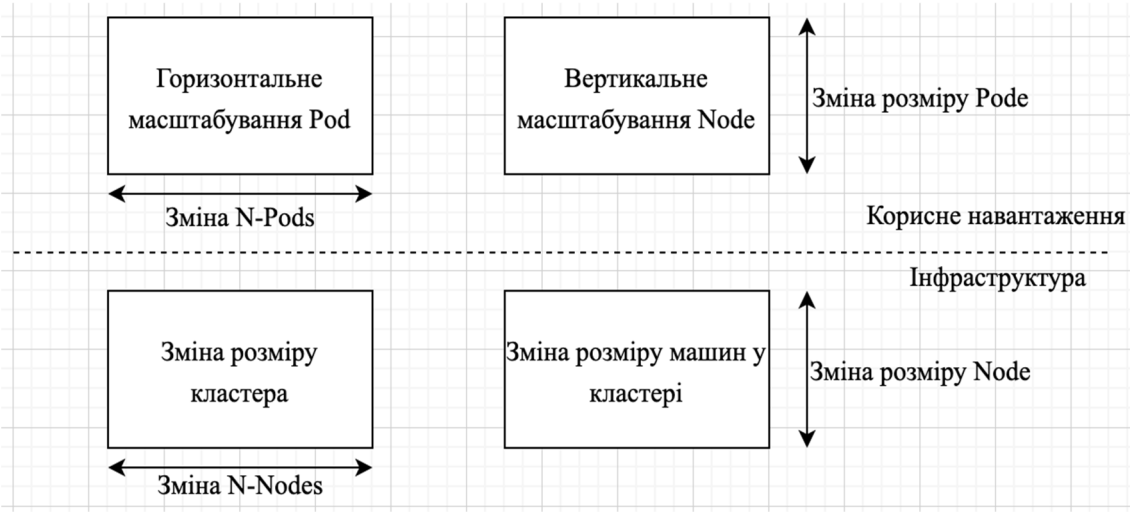

Рис. 4. Операції для досягнення масштабування у Kubernetes

Висновки. У цій статті були досліджені та описані особливості масштабування контейнерного навантаження на базі системи Kubernetes із розділенням концепцій масштабування інфраструктури та корисного навантаження.

Досліджено чотири основних операції масштабування у Kubernetes: у двох розрізах (корисного навантаження та інфраструктури) та у двох напрямках (горизонтально та вертикально). Горизонтальне масштабування досягається шляхом зміни кількості контейнерів, та, відповідно, под та зміні кількості машин у кластері, на яких можуть бути розгорнуті поди. Вертикальне масштабування полягає у зміні кількості ресурсів як для под, так і для машин кластера.

Список використаних джерел

1. Офіційний портал Kubernetes. URL: https://kubernetes.io/.
2. Sam Newman. Building Microservices: Designing Fine-Grained System. O'Reilly, 2015. 251 с.

3. Теленик С. Ф., Ролик А. І., Жаріков Е. В. Управление распределением виртуальных машин в цод. *Вісник НТУУ «КПІ» Інформатика, управління та обчислювальна техніка*. 2016. № 64. С. 90-99. URL: https://ela.kpi.ua/bitstream/123456789/20434/1/64-13-Telenyk.pdf.

4. Жаріков Е. В., Коваль А. А., Терентьєв Р. А. Динамічне розміщення віртуальних машин на основі навчання з підкріпленням в хмарних центрах обробки даних. *Наукові вісті Далівського університету*. 2017. № 13. http://nvdu.snu.edu.ua/wp-content/uploads/2020/03/2017_13_4.pdf.

5. Офіційний портал Docker. URL: https://www.docker.com/.

6. Конфігурація ресурсів для Docker. URL: https://docs.docker.com/config/containers/resource_constraints/.

References

1. Kubernetes official portal. https://kubernetes.io/.

2. Sam, Newman. (2015). Building Microservices: Designing Fine-Grained System. O'Reilly (pp. 251).

3. Telenyk, S., Rolik, A., Jarikov, E. (2016). Managing the distribution of virtual machines in the data center. *Bulletin of NTUU "KPI" Informatics, management and computer engineering*, (64), pp. 90–99. https://ela.kpi.ua/bitstream/123456789/20434/1/64-13-Telenyk.pdf.

4. Jarikov, E., Koval, A., Terentiev, R. (2017). Dynamic deployment of virtual machines based on training with reinforcement in cloud data centers. *Scientific news of Daliv University,* (13). http://nvdu.snu.edu.ua/wp-content/uploads/2020/03/2017_13_4.pdf.

5. Docker official portal. https://www.docker.com/.

6. Docker resources configuration. https://docs.docker.com/config/containers/resource_constraints/.

UDC 62-503.5

Oleksii Sopov, Anna Tsytovtseva

FEATURES OF SCALING CONTAINER LOAD IN KUBERNETES SYSTEM

Today, the field of software development using container technology is becoming widespread in many areas of business, and the problem of scaling is one of the most important to achieve the expansion of the volume of tasks performed over time, so it is important to use the right scaling schemes. Kubernetes requires relatively easy-to-adapt container management mechanisms, but the problem of scaling them remains poorly described and researched, which increases the complexity of the business transition to container technologies and, in particular, the Kubernetes system.

Thanks to the separation of infrastructure and payload concepts in Kubernetes, the classic scaling methods – horizontal and vertical – are revealed in each component separately. Although the Kubernetes system is quite flexible, it has only recently been developed, and the features of scaling with concept separation have not yet been described.

Actual scientific researches and issues analysis showed that there is a general trend towards cloud migration recently. In this context, the architectural style of microservices and the use of container load has attracted considerable attention, especially based on the Kubernetes system. It can be concluded that Kubernetes has significant advantages in the speed of development and deployment of container-based applications, but the problem of scaling and features of horizontal and vertical scaling in Kubernetes has not received much attention from academia so far.

The aim of the work is to study the features of scaling the container load based on the Kubernetes system. To explain the basic operations to achieve horizontal and vertical scaling. To describe the scalability property of the Kubernetes system.

This work reveals the main architectural features of Kubernetes, the principles of working with applications in container virtualization. The main methods of regulating the resource capacity of containers are shown. The concepts of horizontal and vertical scaling in the Kubernetes system with an indication of scalability properties are revealed. The main operations for scaling in the Kubernetes system in the context of separation of payload and infrastructure concepts are given.

In this article the features of container load scaling based on the Kubernetes system with a separation of the concepts of payload and infrastructure scaling were investigated and described. This article is a review.

Keywords: *Kubernetes; microservice; application containers; scaling; docker.*

Fig.: 4. References.: 6.

Сопов Олексій Олександрович — студент-магістрант, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського» (просп. Перемоги, 37, м. Київ, 03056, Україна).
Sopov Oleksii — master student, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute" (37 Peremohy Av., 03056 Kyiv, Ukraine).
E-mail: sopov.alax.a@gmail.com
ORCID: https://orcid.org/0000-0003-0389-3070

Цитовцева Анна Сергіївна — студентка-магістрантка, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського» (просп. Перемоги, 37, м. Київ, 03056, Україна).
Tsytovtseva Anna — master student, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute" (37 Peremohy Av., 03056 Kyiv, Ukraine).

Сопов О., Цитовцева А. Особливості масштабування контейнерного навантаження на базі системи Kubernetes. *Технічні науки та технології.* 2021. № 1(23). С. 103-108.