УДК 004.032:621.391 DOI https://doi.org/10.30837/IYF.PDICIMT.2024.148

АНАЛІЗ ПРИНЦИПІВ КОНТЕЙНЕРНОГО УПРАВЛІННЯ НА БАЗІ СИСТЕМИ KUBERNETES

Приходько О.С.
Науковий керівник – доц. Колтун Ю.М.
Харківський національний університет радіоелектроніки, каф. ІМІ,
м. Харків, Україна
e-mail: oleksii.prykhodko@nure.ua

A key solution used for service orchestration is Kubernetes, an open source software for automating the deployment, scaling and management of containerized applications. At the core of Kubernetes is container technology, which is made up of Docker containers. A container is a small virtual machine that performs one simple task, meaning it actually implements a single microservice. The aggregate of microservices forms a microservice platform. The paper makes a detailed analysis of Kubernetes system architecture, its components and services. The options of traffic management between multiple microservices of Kubernetes container platform with the help of Istio are analyzed.

Нинішнє согодення все гостріше потребує розв'язання задач, що пов'язані з реалізацією та розгортанням сучасних мережних і хмарних сервісів, а також забезпечення можливостей, щодо їх автоматизації, управління, маршрутизації, виявлення і усунення проблем, і т. ін. Програмні системи та платформи, що реалізують такі підходи, вже досить широко використовуються сучасними інформаційними платформами (такими як Google, Amazon, Ebay, Facebook, YouTube, тощо) завдяки значній гнучкості та економічності в реалізації своїх рішень [1].

Серед рішень, розроблених для оркестрації сервісів, виділяється система Kubernetes - відкрите ПЗ для автоматизації розгортання, масштабування та управління контейнеризованими додатками. Це ПЗ з відкритим вихідним кодом розміщене на серверах Cloud Native Computing Foundation (CNCF). Основою системи Kubernetes є котейнерна технологія, яку утворюють docker контейнери, що є базовими для створення сервісів. Контейнер по суті є міні операційною системою з необхідним функціоналом тільки для виконання певної задачі. Вони займають дуже мало дискового простору, а їх запуск триває невеликий час. Іншими словами - це класична віртуальна машина дуже маленького розміру, яка виконує одну досить просту задачу. Виходячи з цього, контейнер фактично реалізує один мікросервіс, а їх сукупність формує мікросервісну платформу або, інакше, котейнерну технологію, що лежить в основі Kubernetes, де кожен сервіс характеризується незалежністю один від одного [1].

Метою доповіді є аналіз можливостей застосування Kubernetes для автоматизації розгортання, управління та моніторингу сервісів з використанням контейнерів. У процесі аналізу такої складної системи як

Kubernetes треба відстежувати сервіси в контейнерах, збирати та систематизувати метрики, налаштувати безпеку, враховувати аспекти мережного управління трафіком у процесі взаємодії мікросервісів, тощо.

Традиційно сервіси розгортали безпосередньо на вузлах із вихідних кодів або інсталяційних пакетів. З появою систем класу CMT (Configuration Management Tools), таких як puppet, ansible, chef, - цей процес став більш автоматизованим, але все ще потребував участі фахівця в плануванні інфраструктури сервісної платформи і написанні конфігураційних файлів для автоматизації процесів, що пов'язані із виконанням рутинних задач. Головним недоліком такого підходу є низька продуктивність [2].

У разі застосування Kubernetes підхід щодо розгортання сервісів буде відмінним від традиційного, де контейнери подано як об'єкти, що не піддаються змінам (Immutable infrastructure). Контейнери розташовуються між вузлами кластера виходячи з політик розподілу (за замовчуванням розташовуються рівномірно між робочими вузлами кластера). Ізоляція між сервісами виконується на рівні контейнерної віртуалізації та мережевих модулів з можливістю шифрування до кінцевого пункту призначення. На рівні кластера існує безліч абстракцій щодо ізоляції об'єктів [2].

У роботі зроблено детальний аналіз архітектури системи Kubernetes, її компонентів і служб. Проаналізовано варіанти управління трафіком між мікросервісами за допомогою Istio, що представляє собою виділений рівень інфраструктури, який називається Service Mesh. Він допомагає обробляти зв'язок між сервісами, повторні запити, тайм-аути та автоматично шифрувати з'єднання і забезпечує можливості з управління трафіком між сервісами, збір статистики, моніторинг і безпеку в складних розгортаннях [3]. Зокрема в роботі аналізується управління трафіком на основі маршрутизації запитів, балансування навантаження, A/B-тестування та інші.

Всі варіанти управління трафіком мають свої особливості, переваги та недоліки. Тому вибір найкращого варіанта управління трафіком в мікросистемній платформі Kubernetes буде залежати від її налаштувань, структури та потреб сервісної інфраструктури.

Список використаних джерел:

1. Kubernetes Documentation: Overview of Kubernetes [Електронний ресурс] // KubeCon + CloudNativeCon Europe 2024. – Режим доступу до ресурсу: https://kubernetes.io/docs/concepts/overview/.

2. Kelsey Hightower, Brendan Burns, Joe Beda “Kubernetes: Up and Running: Dive into the Future of Infrastructure 1st Edition” – O’Reilly Media, 2017. 272 p.

3. Lee Calcote, Zack Butcher “Istio: Up and Running: Using a Service Mesh to Connect, Secure, Control, and Observe 1st Edition” – O’Reilly Media, 2019. 272 p.