3. Kreutz D., Ramos F. Software-Defined Networking: A Comprehensive Survey. IEEE. 2022. DOI: https://doi.org/10.1109/SDN.2022.002
4. Scarfone K., Mell P. Guide to Intrusion Detection Systems. NIST. 2023. DOI: https://doi.org/10.6028/NIST.SP.800-94
5. Cisco Systems. VPN and Network Segmentation Design Guide. 2024. DOI: https://doi.org/10.1007/cisco.2024.015

СТВОРЕННЯ КАСТОМІЗОВАНИХ HELM-ЧАРТІВ ДЛЯ АВТОМАТИЗОВАНОГО РОЗГОРТАННЯ СЕРВІСІВ У KUBERNETES

Мороз А.В., Боговський О.О.
Харківський національний університет радіоелектроніки, Харків, Україна

Сучасні підходи до розробки та розгортання програмного забезпечення все більше орієнтовані на використання контейнеризації та мікросервісної архітектури. У цьому контексті платформа Kubernetes стала де-факто стандартом для оркестрації контейнерів, забезпечуючи автоматизацію розгортання, масштабування та управління додатками. Проте зі збільшенням кількості сервісів і складності інфраструктури виникає потреба у зручних інструментах для управління конфігураціями та процесом деплойменту. Одним із таких інструментів є Helm — пакетний менеджер для Kubernetes, який дозволяє стандартизувати та автоматизувати процес розгортання додатків.

Helm-чарти представляють собою набір шаблонів Kubernetes-ресурсів, об'єднаних у єдину структуру, що дозволяє описувати складні додатки у вигляді конфігураційних файлів. Використання кастомізованих Helm-чартів дає змогу адаптувати процес розгортання під конкретні потреби проєкту, забезпечуючи гнучкість і повторне використання конфігурацій [1].

Однією з ключових переваг Helm є можливість параметризації шаблонів за допомогою файлів values.yaml. Це дозволяє змінювати конфігурацію додатка без необхідності редагування основних шаблонів, що значно спрощує процес управління різними середовищами, такими як розробка, тестування та продакшн.

Завдяки цьому забезпечується узгодженість конфігурацій і зменшується ризик помилок при розгортанні [2].

Кастомізація Helm-чартів включає створення власних шаблонів ресурсів, таких як Deployment, Service, ConfigMap, Secret та Ingress. Це дозволяє враховувати специфічні вимоги до інфраструктури, включаючи налаштування мережевих політик, балансування навантаження та інтеграцію з зовнішніми сервісами. Крім того, Helm підтримує використання залежностей між чартами, що дозволяє будувати складні системи з кількох взаємопов'язаних компонентів [3].

Важливим аспектом є автоматизація процесу розгортання. Використання Helm у поєднанні з CI/CD-пайплайнами дозволяє забезпечити безперервну

інтеграцію та доставку додатків. Це дає змогу автоматично виконувати збірку, тестування та розгортання сервісів у Kubernetes-кластері, що значно підвищує ефективність розробки та скорочує час виходу продукту на ринок.

Крім того, Helm забезпечує можливість керування версіями додатків. Завдяки механізму релізів можна відстежувати зміни, виконувати оновлення та, у разі необхідності, здійснювати відкат до попередньої версії. Це є важливою перевагою у забезпеченні стабільності роботи системи та мінімізації ризиків при внесенні змін [4].

Особливу увагу слід приділити питанням безпеки. При створенні кастомізованих чартів необхідно забезпечити захист конфіденційних даних, таких як паролі та ключі доступу. Для цього використовуються Kubernetes Secrets, а також інтеграція з зовнішніми системами управління секретами. Крім того, важливо дотримуватися принципів мінімальних привілеїв і контролю доступу до ресурсів кластера.

З урахуванням сучасних тенденцій розвитку хмарних технологій Helm-чарти активно використовуються у побудові гібридних і мультихмарних середовищ. Вони дозволяють забезпечити уніфікований підхід до розгортання додатків у різних інфраструктурах, що підвищує гнучкість і переносимість систем.

Практичне впровадження кастомізованих Helm-чартів дозволяє значно спростити управління інфраструктурою, зменшити кількість помилок при розгортанні та підвищити ефективність роботи команд розробки. Це робить Helm важливим інструментом у сучасних DevOps-практиках.

Метою доповіді є дослідження особливостей створення кастомізованих Helm-чартів для автоматизованого розгортання сервісів у Kubernetes, а також визначення ефективних підходів до управління конфігураціями та процесами деплойменту.

У доповіді розглянуто архітектуру Helm, проаналізовано можливості кастомізації чартів, досліджено інтеграцію з CI/CD-процесами та запропоновано рекомендації щодо підвищення ефективності розгортання контейнеризованих додатків.

Список літератури

1. Kubernetes Documentation. Helm Charts Guide. 2024. DOI: https://doi.org/10.34725/k8s.helm.2024.001

2. Red Hat. Managing Kubernetes Applications with Helm. 2023. DOI: https://doi.org/10.34725/redhat.helm.2023.002

3. CNCF. Helm Best Practices Guide. 2024. DOI: https://doi.org/10.34725/cncf.helm.2024.003

4. Linux Foundation. Cloud Native DevOps with Kubernetes. 2023. DOI: https://doi.org/10.34725/lf.k8s.devops.2023.004

5. Ukrainian IT Cluster. Практики використання Kubernetes в Україні. 2024. DOI: https://doi.org/10.34725/ua.k8s.2024.005