Дмитро Кирьяков, Ірина Борецька*,Олександр Сторожук*
[1] НЛТУ України, Львів, Україна, ORCID: 0009-0009-2948-5965,
122.24kyriakov.d@nltu.lviv.ua
[2] НЛТУ України, Львів, Україна, ORCID: 0000-0002-6767-104X,
boretska@nltu.edu.ua
[3] НЛТУ України, Львів, Україна. ORCID: 0000-0001-6566-5271,
storozhuk@nltu.edu.ua

Інфраструктура як код (iac) та ci/cd для хмарних сервісів на базі Azure та Kubernetes

Анотація. *У публікації розглянуто практичне застосування підходу Infrastructure as Code (IaC) та практик CI/CD для автоматизованого розгортання контейнерних сервісів у Microsoft Azure. Запропоновано шаблон рішення, який поєднує модульну структуру Terraform, конвеєри Azure DevOps для безперервної інтеграції та розгортання в Azure Kubernetes Service (AKS), а також стек спостережуваності Prometheus / Grafana / Loki. Показано, що використання IaC скорочує час початкового розгортання інфраструктури та зменшує вплив людського фактора, а впровадження DevOps-метрик (Lead Time, Deployment Frequency, MTTR) дає змогу кількісно оцінювати ефективність процесу постачання змін. Отримані результати демонструють підвищення відтворюваності середовищ і прозорості експлуатації хмарних сервісів.*

Ключові слова *– Infrastructure as Code, Terraform, Azure DevOps, Kubernetes, AKS, CI/CD, Prometheus, Grafana, Loki.*

Вступ. Швидке зростання масштабів і складності хмарних платформ висуває високі вимоги до повторюваності, надійності та швидкості випуску змін. Декларативний підхід Infrastructure as Code (IaC) у поєднанні з CI/CD-конвеєрами дає змогу описувати та ідемпотентно застосовувати зміни інфраструктури, а керований сервіс Azure Kubernetes Service (AKS) мінімізує операційні витрати на підтримку control-plane. У роботі узагальнено досвід побудови відтворюваної інфраструктури Azure з Terraform [1] і розгортання контейнерних сервісів у AKS за допомогою Azure DevOps і Helm [3-5].

Матеріали та методи. Інфраструктура описана в Terraform із розділенням на модулі (мережа, ACR, AKS). Стан зберігається у віддаленому backend

* науковий керівник

(Azure Storage) із блокуванням операцій. Безперервна інтеграція й розгортання реалізовані в Azure DevOps (YAML-pipelines): стадія Build збирає Docker-образ та завантажує його в Azure Container Registry; стадія Deploy застосовує Terraform-план і виконує Helm-розгортання у кластер AKS [2]. Конфіденційні дані керуються через Azure Key Vault і Managed Identity. Спостережуваність забезпечено стеком Prometheus/Grafana/Loki (метрики, логи, алерти) [6-8].

Архітектура рішення. На рисунку 1 наведено спрощену архітектурну схему конвеєра від внесення змін у код до середовища розгортання.

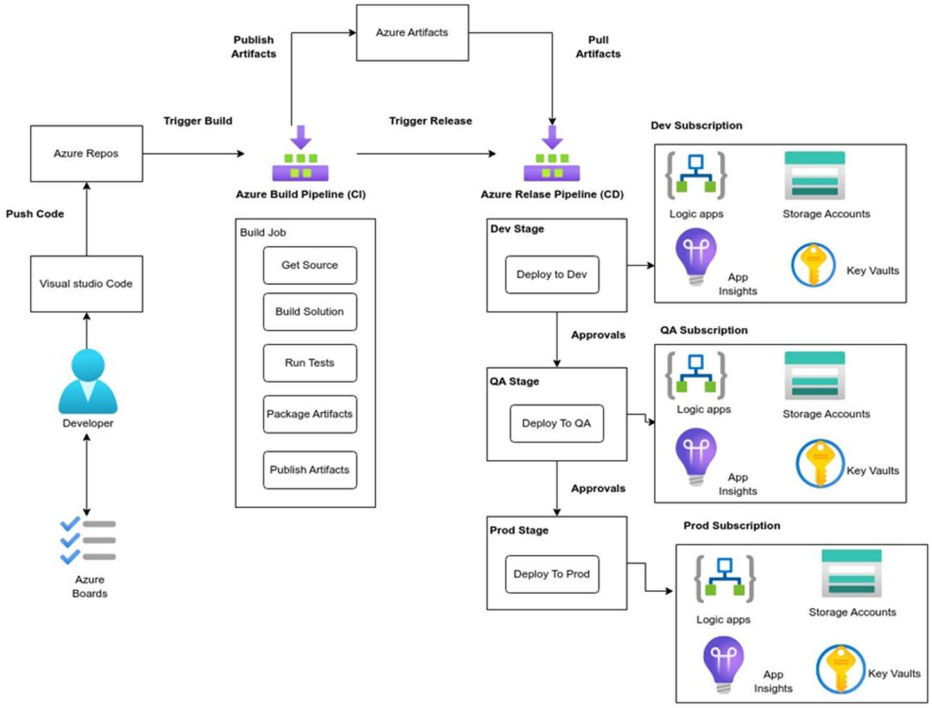

Рисунок 1. Приклад інтеграції Terraform та Azure DevOps для розгортання інфраструктури в Azure.

Результати. У тестовому середовищі повний цикл початкового розгортання (Resource Group, VNet, ACR, AKS) за підходу IaC зайняв близько 8 хвилин проти ~45 хвилин ручних операцій. За зібраними DevOps-метриками середній Lead Time for Changes становив ≈ 0,7 доби, Deployment Frequency – близько п’яти релізів на тиждень, а Mean Time to Recovery – близько 25 хвилин. Фактична доступність ключового HTTP-сервісу становила ≈ 99,92 %, що відповідає заданому SLO 99,9 %.

* науковий керівник

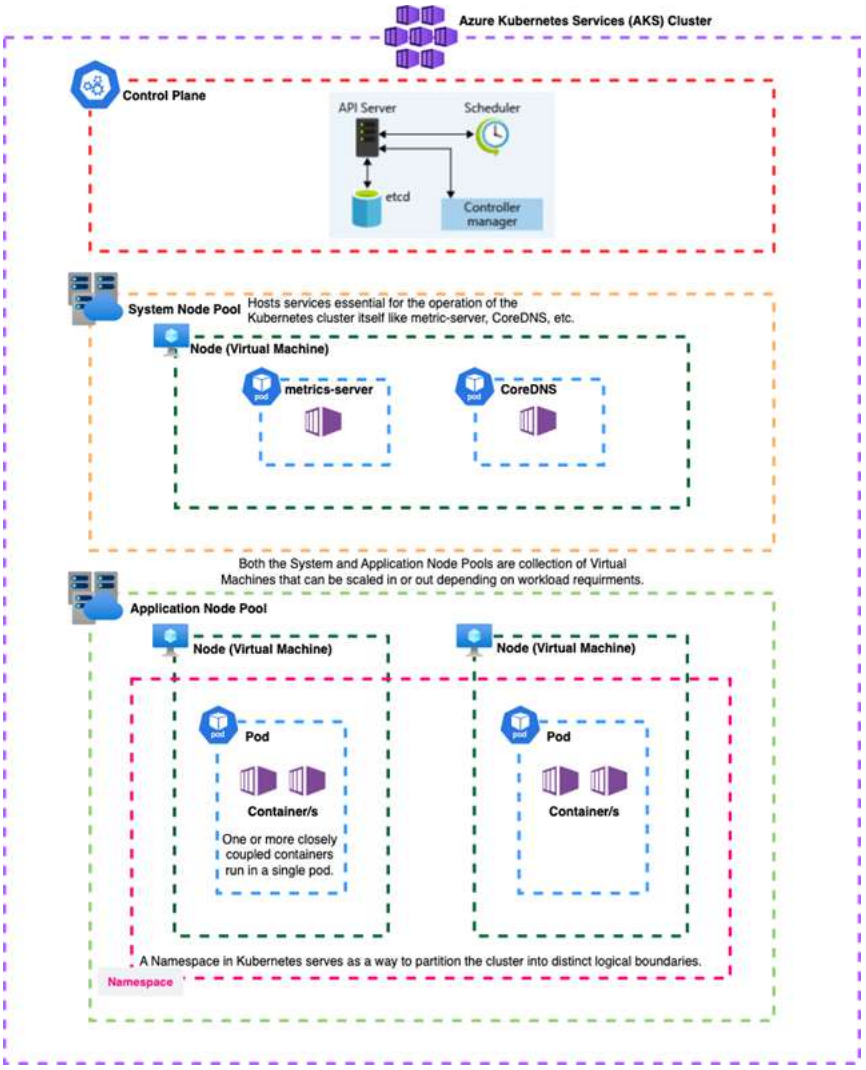

Рисунок 2. Спрощена архітектура кластера Kubernetes / AKS

Практика підтвердила критичність коректного керування станом Terraform і дотримання блокування на рівні віддаленого сховища. GitOps-оператор (наприклад, Flux CD) може додатково автоматизувати рекон-силяцію стану. Для конфіденційних даних доцільно розглянути інтеграцію з HashiCorp Vault. Оптимізація витрат можлива за рахунок Spot-вузлів AKS або віртуальних нод.

Висновки. Запропонований шаблон IaC + CI/CD для Azure/AKS забезпе-чує відтворюваність, керованість і прозору спостережуваність контейнерних навантажень. Отримані метрики свідчать про скорочення часу постачання змін та швидке відновлення після збоїв. Подальші роботи передбачають впрова-дження GitOps-оператора, секрет-менеджера Vault і економічних стратегій у AKS.

* науковий керівник

Список використаних літературних джерел

[1] HashiCorp. *Terraform documentation*, 2025, [Електронний ресурс].

[2] Microsoft. *Azure Kubernetes Service (AKS) documentation*, 2025, [Електронний ресурс].

[3] Microsoft. *Azure DevOps Pipelines Documentation*, 2025, [Електронний ресурс].

[4] The Helm Authors, *Helm Documentation*, 2025, [Електронний ресурс].

[5] Forsgren N., Humble J., Kim G. *Accelerate: State of DevOps Report, 2023* [Електронний ресурс]. – DORA, 2023.

[6] Turnbull J. *Prometheus: The Definitive Guide*, 2023.

[7] Grafana Labs. *Loki Documentation*, 2024.

[8] Microsoft. *Azure Architecture Framework,* 2024.

Infrastructure as Code (IaC) and CI/CD for cloud services based on Azure and Kubernetes

Dmytro Kyriakov, Iryna Boretska, Oleksandr Storozhuk

The paper presents a practical template for automating cloud infrastructure and containerized workloads in Microsoft Azure using Infrastructure as Code (IaC) and CI/CD practices. The proposed solution combines a modular Terraform design, Azure DevOps pipelines for continuous integration and deployment to Azure Kubernetes Service (AKS), and an observability stack based on Prometheus, Grafana and Loki. Experimental results show that IaC significantly reduces the time of initial environment provisioning and minimizes human error, while DevOps metrics such as Lead Time, Deployment Frequency and MTTR provide a quantitative view of delivery performance. The approach improves environment reproducibility, operational transparency and creates a solid baseline for further optimization of cloud-native platforms.

* науковий керівник