УДК 621.391:004.7

CONTAINERLAB ТА VRNETLAB: ЕФЕКТИВНІ ІНСТРУМЕНТИ ДЛЯ ЕМУЛЯЦІЇ МЕРЕЖНИХ СЕРЕДОВИЩ В ІНФОКОМУНІКАЦІЙНІЙ ІНЖЕНЕРІЇ

Савченко Р.О.
e-mail: roman.savchenko1@nure.ua
Харківський національний університет радіоелектроніки,
каф. ІКІ ім. В.В. Поповського
м. Харків, Україна

This paper examines the integration of Containerlab and vrnetlab for efficient network topology emulating. Containerlab facilitates rapid deployment of complex network scenarios using containers, while vrnetlab enables virtualization of network devices. Using Cisco IOL firmware, the study demonstrates containerized network device creation and highlights the benefits of these tools for scalable, realistic testing. By optimizing topology preparation and management, these technologies enhance flexibility and efficiency, making them ideal for modern network engineering, research, and development.

У сучасних мережних технологіях зростає увага до створення гнучких та ефективних середовищ для тестування та емуляції мережних топологій. Одним із перспективних інструментів для цього є Containerlab [1, 2], який забезпечує швидке розгортання складних мережних сценаріїв за допомогою контейнерів. Важливу роль у цьому контексті відіграє vrnetlab – інструмент для інтеграції віртуальних мережних пристроїв у контейнери, що дозволяє емулювати реальне обладнання. У цій роботі досліджується поєднання Containerlab і vrnetlab як ефективного інструмента для мережних інженерів, дослідників і розробників.

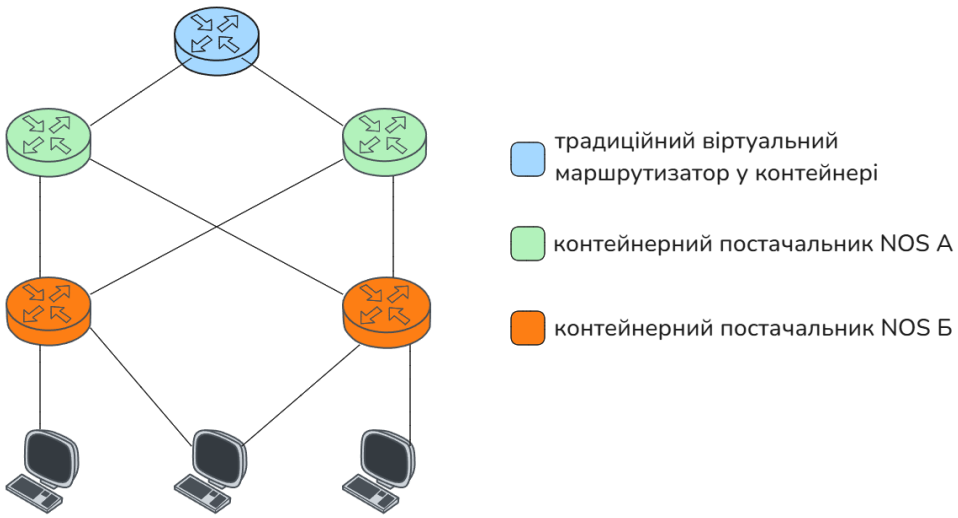

Рис. 1 – Реалізація концепції «Lab as Code», створеної за допомогою Containerlab

Vrnetlab дозволяє контейнеризувати віртуальні мережні пристрої, що точно імітують роботу реального обладнання. Цей інструмент підтримує різні типи мережних пристроїв, зокрема маршрутизатори, комутатори та інші компоненти, що сприяє створенню реалістичних тестових середовищ для моделювання складних мережних сценаріїв. Інтеграція віртуальних пристроїв у контейнери спрощує процес розгортання та управління мережними топологіями, роблячи їх доступнішими для тестування, навчання та розробки. Крім того, vrnetlab забезпечує масштабованість тестових середовищ, що робить його ефективним рішенням для проєктів будь-якого рівня складності.

Для демонстрації можливостей vrnetlab розглянуто процес створення контейнера мережного пристрою на основі прошивки Cisco IOL (IOS on Linux). Cisco IOL є легковаговим образом, який працює у вигляді бінарного файлу без додаткового шару віртуалізації, що забезпечує його швидке розгортання. Виділяють два типи образів Cisco IOL:

- IOL – призначений для роботи на рівні Layer 3 (маршрутизатори);
- IOL-L2 – використовується для рівнів Layer 2/Layer 2+ (комутатори).

Для створення контейнера на основі Cisco IOL застосовується файл *cisco_iol-17.15.01.bin*, який можна отримати з офіційного рішення Cisco Modeling Lab (CML) [3]. Цей файл містить образ операційної системи для віртуального мережного пристрою та використовується як основа для створення контейнера. Після отримання файлу *cisco_iol-17.15.01.bin* його копіюють у робочу директорію, де здійснюється процес створення Docker-образу. Для цього використовується команда *make docker-image*, яка автоматично формує Docker-образ на основі вказаного файлу прошивки.

```
root@containerlab:~/containerlab_dir/vrnetlab/cisco/iol# make docker-image
for IMAGE in cisco_iol-l2.17.15.01.bin; do \
        echo "Making $IMAGE"; \
        make IMAGE=$IMAGE docker-build; \
        make IMAGE=$IMAGE docker-clean-build; \
done
Making cisco_iol-l2.17.15.01.bin
make[1]: Entering directory '/root/containerlab_dir/vrnetlab/cisco/iol'
--> Cleaning docker build context
rm -f docker/*.qcow2* docker/*.tgz* docker/*.vmdk* docker/*.iso docker/*.xml docker/*.bin
rm -f docker/healthcheck.py docker/vrnetlab.py
Building docker image using cisco_iol-l2.17.15.01.bin as vrnetlab/cisco_iol:l2.17.15.01
echo "ok"
ok
make IMAGE=$IMAGE docker-build-image-copy
make[2]: Entering directory '/root/containerlab_dir/vrnetlab/cisco/iol'
cp cisco_iol-l2.17.15.01.bin* docker/
make[2]: Leaving directory '/root/containerlab_dir/vrnetlab/cisco/iol'
(cd docker; docker build --build-arg http_proxy= --build-arg HTTP_PROXY= --build-arg https_proxy= --b
uild-arg HTTPS_PROXY= --build-arg IMAGE=cisco_iol-l2.17.15.01.bin --build-arg VERSION=l2.17.15.01 --l
abel "vrnetlab-version=$(git log -1 --format=format:"Commit: %H from %aD")" -t vrnetlab/cisco_iol:l2.
17.15.01 .)
```

Рис. 2 – Створення контейнеру на основі файлу прошивки Cisco

У процесі створення контейнера файл *cisco_iol-17.15.01.bin* інтегрується в контейнер та отримує відповідний тег, наприклад, *vrnetlab/cisco_iol:17.15.01*. Після завершення генерації образу його коректність можна перевірити за допомогою команди *docker images*, яка відображає список усіх доступних Docker-образів. Створений образ *vrnetlab/cisco_iol:17.15.01* готовий до використання в топологіях Containerlab. Для його інтеграції в мережну топологію необхідно внести відповідні налаштування у файл конфігурації топології [4].

Таким чином, поєднання Containerlab і vrnetlab відкриває нові можливості для швидкого та ефективного створення віртуальних мережних середовищ [5]. Ці інструменти дають змогу емулювати мережні топології різної складності – від простих лабораторних сценаріїв до ієрархічних мереж, забезпечуючи при цьому високу точність відтворення роботи реального обладнання. Завдяки можливості швидкого розгортання та масштабування вони є незамінними для тестування нових мережних рішень, підготовки фахівців і проведення досліджень у сфері інфокомунікаційних технологій.

Крім того, використання Containerlab і vrnetlab відповідає концепції «Lab as Code» (LaC), що передбачає програмно визначене управління лабораторними середовищами [6]. Це дає змогу автоматизувати розгортання, налаштування та тестування мережних інфраструктур за допомогою декларативних конфігурацій. Завдяки цьому підхід LaaC підвищує гнучкість і повторюваність експериментів, а також сприяє інтеграції тестових середовищ у CI/CD-процеси. Використання таких інструментів значно скорочує час на підготовку тестових середовищ, що робить їх оптимальним вибором для мережного проєктування.

Список використаних джерел:

1. Шестопалов С. С. Containerlab як інструмент моделювання та симуляції мереж на основі контейнеризації. Інформаційно-комунікаційні технології та кібербезпека (ІКТК-2024) : матеріали Міжнародної науково-технічної конференції, 2024. Харків : ХНУРЕ, 2024. С. 140-141.

2. Containerlab. URL: https://containerlab.dev/ (дата звернення: 04.03.2025).

3. Introduction - Cisco Modeling Labs v2.8 - Cisco DevNet. Cisco DevNet. URL: https://developer.cisco.com/docs/modeling-labs/ (дата звернення: 04.03.2025).

4. Roman S. Containerlab configs. Zenodo. URL: https://doi.org/10.5281/zenodo.14916309 (дата звернення: 04.03.2025).

5. Vina S. Containerlab - Creating Network Labs Can't be Any Easier. Packetswitch. URL: https://www.packetswitch.co.uk/containerlabs-intro/ (дата звернення: 04.03.2025).

6. ESNOG – Grupo de Operadores de Red Españoles. URL: https://esnog.net/gore32/archivos/esnog32-victor_serrano-containerlab.pdf (дата звернення: 04.03.2025).