Studying the Principles of Infocommunication Network Virtualisation Using the Docker Platform

V. M. Antonova
Moscow Technical University of Communications and Informatics
Moscow, Russia
v.m.antonova@mtuci.ru

M. A. Egorov
Moscow Technical University of Communications and Informatics
Moscow, Russia
arhangelmihail2000@mail.ru

V. P. Blinov
National Research Nuclear University MEPHI
Moscow, Russia
bl1nov.victor@yandex.ru

E. E. Malikova
Moscow Technical University of Communications and Informatics
Moscow, Russia
e.e.malikova@mtuci.ru

A. Y. Malikov
Moscow Technical University of Communications and Informatics
Moscow, Russia
amalikov2000@gmail.com

2024 Systems of Signals Generating and Processing in the Field of on Board Communications | 979-8-3503-8361-4/24/$31.00 ©2024 IEEE | DOI: 10.1109/IEEECONF60226.2024.10496739

***Abstract*—The paper considers the use of virtualization technology, and specifically containerization, for building Future Networks. Containerization technology is a type of virtualization that enables software code to be packaged into a single executable file, ensuring its proper execution. These files are referred to as containers. Docker, an open source containerization platform, greatly simplifies and automates the process of creating and deploying various applications. Multiple containers can be quickly deployed on a single machine using the Docker platform. It is a valuable tool for students at the higher education level to gain hands-on experience with the latest technologies. This platform can be utilized for laboratory assignments focused on studying virtualization technologies. It provides an interactive environment for university students to learn a variety of modern networking technologies, including configuration of new servers and running applications on them. The platform can be used by students and professors to explore modern infocommunication network structures and databases for research. Familiarity with Docker is becoming more and more important in the IT sector, which makes it a valuable skill for students looking for a job.**

Keywords—Networks Functions Virtualization (NFV); Future Networs (FN); container; Docker platform; virtual machine; hypervisor; containerization technolog; ITU-T Recommendations

I. INTRODUCTION

Telecommunication systems have undergone significant development. One of the most popular concepts nowadays is the concept of Future Networks, which is described in ITU-T Y.3000 series Recommendations [1,2]. According to the concept future networks should support Network Functions Virtualization (NFV) technology [3], which requires new principles for building telecommunication networks. Virtualization technology is gaining popularity in various fields, particularly in telephony [4]. It improves the efficiency of hardware resource utilization and reduces the cost of telephone services [5].

Personnel responsible for maintaining modern telecommunication networks must possess the necessary knowledge and practical skills to handle new technologies. As early as the higher education stage, it is crucial to gain practical experience of the latest virtualization technologies [16-20].

This paper discusses how to learn about infocommunication networks using the Docker platform that makes it possible for students to learn a range of modern networking technologies in an interactive way. Docker is an open-source containerization platform that automates application creation, delivery, and management [6,7]. It allows for rapid testing of applications and running the required number of containers on a single machine. Docker enables the use of microservice architecture for applications, allowing updates to be released without impacting the performance of the entire system.

II. OVERVIEW OF THE DOCKER PLATFORM

Containerization is a technique that packages software code into a single executable file, known as a container, to ensure proper execution [8,9]. Containers can be deployed in different environments in which they can manage their activities.

When code is developed in a specific computing environment, errors related to the code's settings can often occur. However, by using containerization, such problems can

979-8-3503-8361-4/24/$31.00 ©2024 IEEE

be reduced. A container is independent of the underlying operating system settings. A container is universal as it can be deployed on any system, regardless of its configuration. A container is a deployed and running process that has a number of features [10]:

- Containers have a short lifecycle and are easy to stop, restart, and destroy. It is recommended to not store data in the container (i.e. 'Stateless') to avoid data loss upon destruction.
- Containers take up a small amount of resources as only the software required for operation is installed in them. This reduces startup time and saves disc space.
- Each container represents one process. It does not significantly affect the entire system and can always be debugged or updates can be installed.
- When using containers, security is improved by keeping them isolated; at the same time containers do not have access to the main operating system.
- For large projects, containers are convenient as they allow for quick deployment to new hosts.
- Since a container is a single running process, the transition to a microservice architecture is made, which in turn speeds up application development.
- Containers are more efficient than virtual machines as they consume fewer system resources. This allows more containers to be run on the same hardware.

Containers are similar to virtual machines [11] in terms of function and purpose, but there are some differences. Unlike virtual machines, containers are not installed inside other operating systems and do not have their own kernel or hypervisor as well as other resources.

Containers use fewer resources than virtual machines and take up less memory. They are almost independent of the operating system cluster and each container module runs a single program. A small number of virtual machines can be run on the same server, while the number of containers running simultaneously will be much larger (Fig. 1). Thus, in a cloud infrastructure, containerization technology can result in lower costs compared to virtual machines.

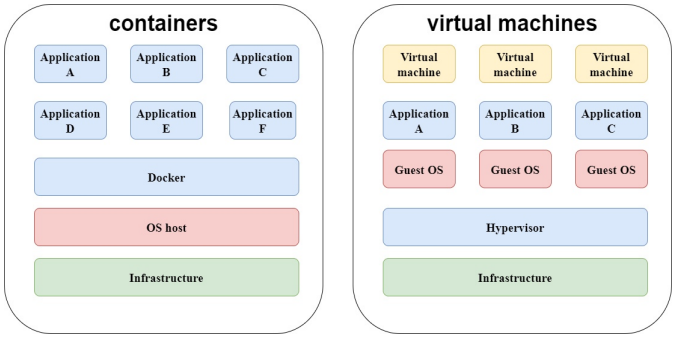

Fig. 1. Container and virtual machone comparison.

III. Installing the Docker Platform on a Virtual Machine

To investigate the network infrastructure using the Docker platform, you can use a virtual machine (VM) with the Linux operating system (OS) (Ubuntu 22.04) [12]. By default, the virtual machine runs in NAT (Network Address Translation) mode, and all traffic should pass through a personal computer. In order to carry out this research, it is necessary to switch to the Bridge mode in the network settings so that the virtual machine is directly connected to the router.

The structure of the network under investigation prior to the installation of the Docker platform is shown in Fig. 2. The IP addresses of the network components are visible. The virtual machine with 192.168.0.19 IP address is directly connected to the router, since it uses the Bridge connection type.

To install the Docker platform on a virtual machine, it is necessary to run a series of commands using the sudo utility and to test Docker. For this purpose, the hello-world container, which is intended for testing the system, is used. Once Docker is installed, a new interface (172.17.0.1) will appear on the virtual machine, which is used by the Docker platform (Fig. 3).

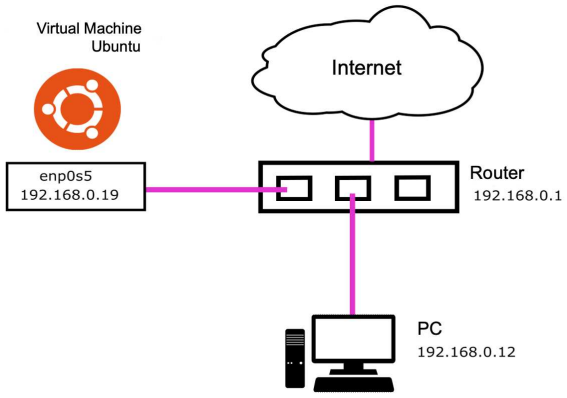

Fig. 2. Structure of the network prior to installing the Docker platform.

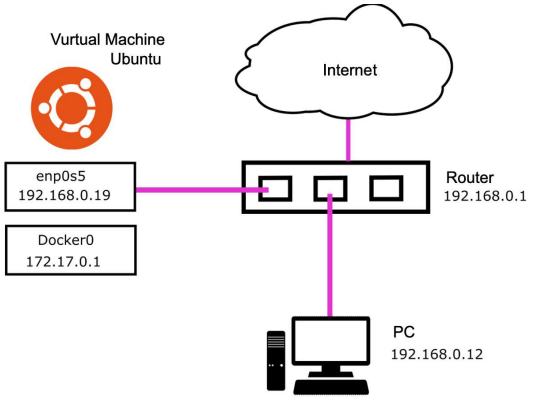

Fig. 3. Structure of the network after installing the Docker platform.

Using the Docker platform requires applying some basic commands [13]. These commands include:

- *docker network ls*, which allows viewing networks;
- *docker inspect*, which lets you inspect networks and see more detailed information about network containers;
- *docker ps*, which displays running containers, their identifiers, creation date and port statuses; and
- *docker run*, which runs containers.

IV. Overview of Laboratory Assignments

In order to study the Docker platform, students are required to complete four lab assignments to investigate different types of networks [14]. The following labs are included:

1) exploring a Bridge core network,
2) investigating a Host type network,
3) studying a MACVLAN type network,
4) studying an IPVLAN (L3) type network of the third layer.

The first laboratory assignment introduces students to the principles of the Docker platform as they explore a core network of the Bridge type. Essentially, this network acts as a bridge between a container and a host machine. Three containers are created, each with its own virtual interface. Each container is assigned an IP address using the DHCP protocol. The resulting network scheme is illustrated in Fig. 4.

Bridged networking schemes are commonly used when applications run in separate containers that require communication with each other. In this networking scheme, Docker0 acts as a switch, allowing containers to communicate with each other. One of the containers hosts a web server (nginx). Once a port is opened to the external network, students can test the Internet access of any container.

In the second lab assignment, the Host type of network is studied. In this type of network, the container connects directly to the virtual machine and utilizes resources from the host. Container ports do not need to be opened. This type of network can be useful for optimizing network performance, or when a large number of ports need to be opened, as no network address translation is required (Fig. 5).

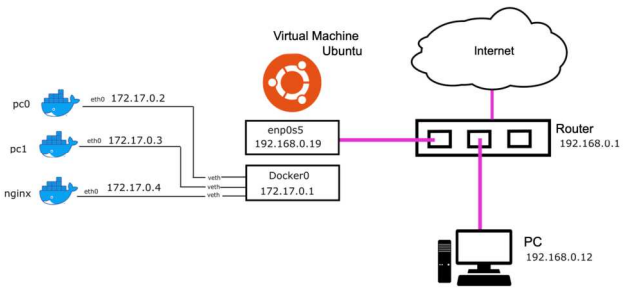

Fig. 4. Bridge Core Network Scheme.

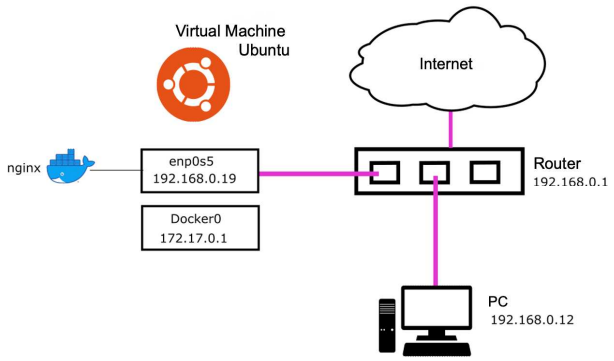

Fig. 5. Host Network Scheme.

When the web server is accessed, a connection is successfully established (Fig. 6).

In some cases, such as outdated applications or applications that monitor network traffic, a direct connection to a physical network is required. To achieve this, a MACVLAN network [15] can be utilized, and it is investigated **in the third lab assignment**. This network assigns a unique MAC address to each network interface of the container to make it look like a physical interface. To ensure MACVLAN network isolation, distinct physical interfaces must be employed (Fig. 7).

Docker directs traffic to containers based on their MAC addresses. Typically, if the MAC address of the packet being received does not match the MAC address of the network interface, the packet is simply discarded. Macvlan-type networks can be used in some cases to enable applications that require a direct connection to the physical network.

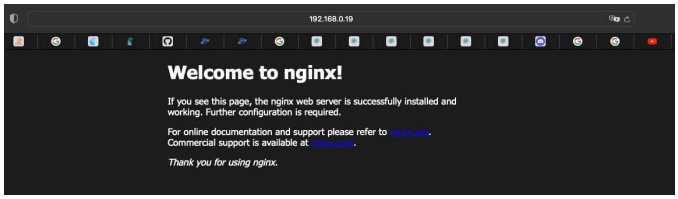

Fig. 6. Result of Connecting to the Web Server.

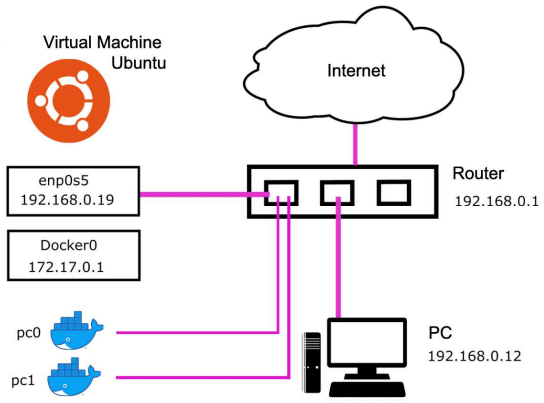

Fig. 7. MACVLAN Network Scheme.

After configuring the network in this manner, the containers can successfully receive data packets. Since the container is connected to this network, it will not need to open ports. It is also possible to connect to the web server.

The disadvantages of this network include the problem with MAC addresses and the lack of a DHCP server; therefore manual assignment of IP addresses is required.

Furthermore, utilizing the Docker application allows for the examination of second and third layer VLANs.

The fourth laboratory assignment investigates the IPVLAN (L3) type network of the third level (Fig. 8). In this case, the investigated networks treat hosts as routers. At Layer 3, there are more opportunities to control the network, while broadcast traffic is eliminated, and network devices can be more easily isolated.

Packets do not pass to the Internet since the router does not have a route to the projected subnet. To enable container access to the Internet, a static route needs to be added to the router.

V. CONCLUSION

This article provides an overview of the Docker containerization platform, which can be used to automate the creation of various applications. The main types of networks covered in the article, which are available for research using Docker, highlight the versatility of the platform in setting up communications between applications. The lab workshop described in the article not only demonstrates Docker's wide range of capabilities but also emphasizes the importance of testing network performance. Connecting to the Internet provides hands-on experience in integrating networks with external resources, which is essential in real-world scenarios.

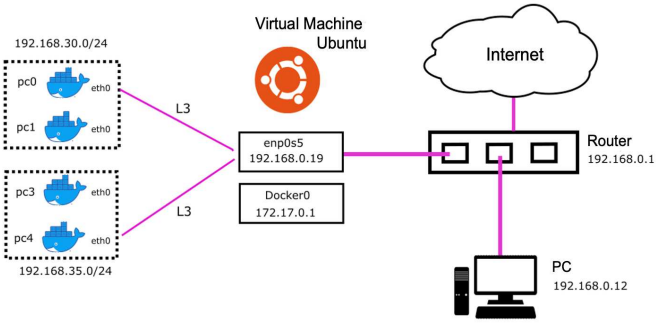

Fig. 8. IPVLAN (L3) Network Scheme.

Working with Docker develops students' technical skills and the ability to manage infrastructure in a changing digital environment. Thus, the paper highlights the technical aspects of working with Docker as well as emphasizes its strategic importance in today's IT industry. Students who have been trained to master this tool are not only prepared for efficient project implementation but also for adapting to a dynamic professional environment. This makes them highly sought-after candidates in the job market.

REFERENCES

[1] ITU-T Recommendations Y.3000-Y.3499: Future networks. URL: http://www.itu.int/ ITU-T/recommendations/index.aspx?ser=Y.

[2] A.V. Roslyakov, S.V. Vanyashin, Future Networks. Samara, PSUTI Publ., 2015, 274 p.

[3] A.P. Pshenichnikov, E.E. Malikova, Future networks: the role of information, conceptual basics, textbook for universities. Moscow, Goryachaya Linia–Telecom, 2021, 100 p.

[4] S. A. Grushko, A. P. Pshenichnikov, E. E. Malikova and A. Y. Malikov, "Virtual Asterisk IP-PBX Operation Studying and Exploring at the University," *2022 Systems of Signals Generating and Processing in the Field of on Board Communications*, Moscow, Russian Federation, 2022, pp. 1-5.

[5] M.G. Kanischeva, E.E. Malikova, I.I. Pelevin and A.P. Pshenichnikov, The basics of working with the virtual telephone ASTERISK IP-PBX, textbook. Moscow: Media Publisher, 2021, 100 p.

[6] Docker.Docs. URL: https://docs.docker.com

[7] A. Mouat, Using Docker, O'Reilly Media, Inc., 2016. 355p.

[8] R. Felani, M. Azam, D.P. Adi, A. Widodo, A.B. Gumelar, "Optimizing virtual resources management using Docker on cloud applications," *Indonesian Journal of Computing and Cybernetics Systems*, 14, 2020, pp.319-330.

[9] D.Morris, S. Voutsinas, N. Hambly, R.G. Mann, "Use of Docker for deployment and testing of astronomy software," *Astronomy and Computing*, vol. 20, 2017, pp.105-119.

[10] N. Poulton, Docker Deep Dive, Packt Publishing, 2020, 249 p.

[11] R.L. Smelyanskiy, V.A. Antonenko, SDN and NFV Principles in Computer Networks, Moscow, KURS, 2020, 160p.

[12] A.H. Sayers, I. Miell, Docker in Practice, Manning, 2019. — 384 p.

[13] P.S. Kocher, Microservices and containers, Addison-Wesley Professional, 2018, 304 p.

[14] E. Gkatziouras, A Developer`s Essebtial Guide to Docker Compose, Packt Publishing, 2022, 264 p.

[15] E. Waud, Docker Quick Start Guide: Learn Docker like a boss, and finally own your applications, Packt Publishing, 2018, 230p.

[16] V.M. Antonova, E.E. Malikova, A.E. Panov, I.V. Spichek, A.Y. Malikov, "Implementation of IoT technology for data monitoring via cloud services," *T-Comm*, vol. 15, no.2, pp. 46-53, 2021.

[17] I.G. Buzhin, V.M. Antonova, E.A. Gaifutdinov, Yu.B. Mironov, "Methodology for a comprehensive assessment of the telecommunication services quality of transport networks using SDN/NFV technologies," *T-Comm*, vol. 16, no.12, pp. 40-45, 2022. DOI: 10.36724/2072-8735-2022-16-12-40-45.

[18] I.G. Buzhin, A. Yu. Derevyankin, V.M. Antonova, A.P. Perevalov, Yu.B. Mironov, "The effectiveness of frameworks for transmitting information in a virtualized communication network with a microservice architecture," *H&ES Reserch*. 2023. Vol. 15. No 2, pp. 23-32. doi: 10.36724/2409-5419-2023-15-2-23-32.

[19] I.G. Buzhin, A.Yu. Derevyankin, V.M. Antonova, A.P. Perevalov, Yu.B. Mironov, "Comparative Analysis of the REST and gRPC Used in the Monitoring System of Communication Network Virtualized Infrastructure," *T-Comm*, vol. 17, no.4, pp. 50-55, 2023. DOI: 10.36724/2072-8735-2023-17-4-50-55.

[20] V.M. Antonova, E.E. Malikova, M.S. Stepanov, "Ways of the ASTERISK Software PBX Protection," *T-Comm*, vol. 17, no.10, pp. 52-58, 2023/ DOI: 10.36724/2072-8735-2023-17-10-52-58.