A Ilica 242, HR-10000 Zagreb
T (01) 2222 182
E student@algebra.hr
www.algebra.hr

ALGEBRA

DEPARTMENT OF OPERATING SYSTEMS

Intro to DevOps

Lab 04 – Advanced container concepts

REV 3.3

© Josip Stanešić, University Algebra
Zagreb, 2024.

A Ilica 242, HR-10000 Zagreb
T (01) 2222 182
E student@algebra.hr
www.algebra.hr

ALGEBRA

Contents

Introduction 3

Tasks 3

Additional reading 4

© Josip Stanešić, University Algebra
Zagreb, 2024.

A Ilica 242, HR-10000 Zagreb
T (01) 2222 182
E student@algebra.hr
www.algebra.hr

ALGEBRA

Introduction

In this lab we will learn network, volumes, bind mounts, and docker/podman compose.

Container networking allows containers to communicate with each other and with external systems, enabling seamless connectivity within distributed applications.

Volumes and bind mounts provide mechanisms for persisting data in containers, with volumes offering independent storage and bind mounts linking host file system directories into containers.

Docker Compose and Podman Compose are tools for defining and managing multi-container applications through simple configuration files, streamlining the deployment and orchestration of complex container environments. They both follow the Compose Specification.

Tasks

1. Log in into workstation VM in RH academy.
2. By using **podman network** command list all podman networks. It should only show default network called **podman** with the **bridge** driver.
3. Inspect the network configuration of the default podman network. What is the value of dns_enabled setting?
4. Create a custom network which will have dns enabled. Name it labnet.
5. Create a mysql database container by using official mysql image(https://hub.docker.com/_/mysql). It should randomize the root password, create a database called wordpress and a user called student with password DB15secure!. It should be started in the background and connected to labnet network. Name the container mysql.
6. Create a wordpress container by using official wordpress image(https://hub.docker.com/_/wordpress). It should be started in the background, connected to the labnet network and it should publish port 80 from the container to port 8080 on the host. Configure environment variables to connect to database container created in the previous step. Wen referencing the database host use mysql(name of the previous container) as the host.
7. If you repeated previous two steps by using default network, would it work?
8. Create a default web page in the wordpress application.
9. In case you needed to delete containers to replace them with updated images or for any other reason, all the files in the read write layer of the image would be lost. To persist the data bind mounts and volumes are used.
   Delete the old mysql container, and create a new one which will bind mount the container directory /var/lib/mysql to host directory /home/student/mysql. Wait for the container to start and examine the host directory.
10. Recreate the previously created container, this time using volume instead of a bind mout.
11. Install podman-compose package.
12. Take a look at samples of compose found in https://github.com/containers/podman-compose/tree/main/examples and https://github.com/docker/awesome-compose.

© Josip Stanešić, University Algebra
Zagreb, 2024.

A Ilica 242, HR-10000 Zagreb
T (01) 2222 182
E student@algebra.hr
www.algebra.hr

ALGEBRA

13. Create a compose file which you can use to deploy wordpress and mysql.
14. Deploy them by using the compose file you created.

Additional reading

- https://github.com/containers/podman/blob/main/docs/tutorials/basic_networking.md
- https://www.redhat.com/sysadmin/container-networking-podman
- https://docs.podman.io/en/latest/markdown/podman-network-create.1.html
- https://github.com/containers/podman-compose
- https://github.com/docker/awesome-compose
- https://github.com/compose-spec/compose-spec/blob/master/spec.md
- https://docs.docker.com/compose/compose-application-model/
- https://docs.podman.io/en/v4.4/volume.html
- https://www.redhat.com/sysadmin/behind-scenes-podman
- https://sysdig.com/learn-cloud-native/

© Josip Stanešić, University Algebra
Zagreb, 2024.