

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №17
дисциплины «Основы программной инженерии»

Выполнила:
Ламская Ксения Вячеславовна
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка и
сопровождение программного
обеспечения», очная форма обучения

(подпись)

Доцент кафедры инфокоммуникаций
Воронкин Роман Александрович

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Тема: работа в Docker с сетью контейнеров и томами.

Цель работы: познакомиться с использованием Docker для управления томами и сетями.

Порядок выполнения работы

1. Создание пользовательской сети: создайте пользовательскую сеть в Docker с именем "my_custom_network". Запустите два контейнера, присоединенных к этой сети, например, с использованием образов Nginx и PostgreSQL. Убедитесь, что они могут взаимодействовать друг с другом.

```
C:\Ksen\docker>docker network create my_custom_network
c078b83d86488b91ccce7f654b5add7240596073d6d61d66c4be70a9f2ac1bba
```

Рисунок 1.1 – Создание пользовательской сети

```
C:\Ksen\docker>docker network create my_custom_network
c078b83d86488b91ccce7f654b5add7240596073d6d61d66c4be70a9f2ac1bba

C:\Ksen\docker>docker run --network=my_custom_network -d --name web_container nginx
3b54088536b71c4126f9a4eef0c54e91cef6992e33d769b08929ec3022f77b57

C:\Ksen\docker>docker run --network=my_custom_network --name db_container -e POSTGRES_PASSWORD=123 -d postgres
Unable to find image 'postgres:latest' locally
latest: Pulling from library/postgres
af107e978371: Already exists
4dab593eebe3: Pull complete
4998fa695fba: Pull complete
68722367c502: Pull complete
f94fde538ad8: Pull complete
083cda9930f9: Pull complete
d17e28f1e487: Pull complete
60abce37aea7: Pull complete
dc71bc844158: Pull complete
8af67c1d8689: Pull complete
a3a37d60b464: Pull complete
a28cd92dbadf: Pull complete
ebba832273a7: Pull complete
ca09208e18c7: Pull complete
Digest: sha256:b09f2562ab14fcae750cfc5ae457cd97e90c37679f520bc4a84180913de90261
Status: Downloaded newer image for postgres:latest
634eadee32d895000ac13d3bc210782e87586e3cbf26e7b9a4b2dba57fb930d9

C:\Ksen\docker>docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS        NAMES
634eadee32d8   postgres  "docker-entrypoint.s..." 57 seconds ago Up 30 seconds 5432/tcp     db_container
3b54088536b7   nginx     "/docker-entrypoint...." 5 minutes ago  Up 5 minutes  80/tcp       web_container
868d01e2185e   alpine    "/bin/sh"                41 minutes ago Up 41 minutes                container1
```

Рисунок 1.2 – Запуск двух контейнеров, присоединенных к этой сети

```
C:\Ksen\docker>docker inspect -f "{{.NetworkSettings.Networks}}" web_container
map[my_custom_network:0xc000000000]

C:\Ksen\docker>docker inspect -f "{{.NetworkSettings.Networks}}" db_container
map[my_custom_network:0xc000458240]
```

Рисунок 1.3 – Сети контейнеров

2. Передача данных через тома: создайте Docker-контейнер с использованием тома. Запишите данные в том из одного контейнера, а затем прочитайте их из другого контейнера, используя тот же том. Обеспечьте, чтобы данные сохранялись после перезапуска контейнеров.

```
C:\Ksen\docker>docker volume create shared_data
shared_data
```

Рисунок 2.1 – Создание общего тома

```
C:\Users\vladi\OneDrive\Рабочий стол\Основы программной инженерии\docker>docker run -itd -v shared_data:/data --name container1 ubuntu
13988c5bb4ec6642017bd6d70085ff3e9e1cb5c19679b92a9f8e875b36e7375a
```

Рисунок 2.2 – Запуск первого контейнера с указанным общим томом

```
C:\Ksen\docker>docker exec -it container4 /bin/bash
root@458c22c62c15:/# cd data
root@458c22c62c15:/data# touch data_file.txt
root@458c22c62c15:/data# ls
data_file.txt
root@458c22c62c15:/data# echo "Hello from container4" > data_file.txt
root@458c22c62c15:/data#
```

Рисунок 2.3 – Запись в файл из первого контейнера

```
C:\Ksen\docker>docker run -itd -v shared_data:/data --name container5 ubuntu
7c2b5c6d228f9dd837e60c242aad54110f1238ff77f07c63ed3daf07c3153726
```

Рисунок 2.4 – Запуск второго контейнера

```
C:\Ksen\docker>docker exec -it container5 bash
root@7c2b5c6d228f:/# cat /data/data_file.txt
Hello from container4
```

Рисунок 2.5 – Чтение из второго контейнера

3. Создание сети overlay для распределенного приложения: используйте Docker Swarm или Kubernetes (в зависимости от предпочтений) для создания кластера. Создайте overlay-сеть и запустите несколько контейнеров, которые могут взаимодействовать через эту сеть.

```
C:\Ksen\docker>docker swarm init
Swarm initialized: current node (c8y26mgp0wb7vpqdp7e6u3rf) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-3anuttevmbpnk225cw6prd7amas17o6svz71vrp864glovrt8e-4i7wgnlnisac6m21mmc5tjuj0 192.168.65.3:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.
```

Рисунок 3.1 – Инициализация Swarm-кластера

```
C:\Ksen\docker>docker network create -d overlay --attachable my_overlay_network
vvol5m7brtuibfteuaaigk5n
```

Рисунок 3.2 – Создание Overlay-сети

```
C:\Ksen\docker>docker run --network=my_overlay_network -d nginx
96e77f6d96e92ebf7f0d9d37ced0a72f0e3a4cc12754f93b7574725823e8f5af

C:\Ksen\docker>docker run --network=my_overlay_network -itd ubuntu
510ce535a3d70ed9e0374cfece0f3509ef8ddc60bfc1cb45e9d89a63601e5ac9
```

Рисунок 3.3 – Запуск двух контейнеров

```
C:\Ksen\docker>docker inspect -f "{{.NetworkSettings.Networks}}" 510ce535a3d7
map[my_overlay_network:0xc000536300]

C:\Ksen\docker>docker inspect -f "{{.NetworkSettings.Networks}}" 96e77f6d96e9
map[my_overlay_network:0xc0003e6180]
```

Рисунок 3.4 – Контейнеры находятся в одной сети

4. Связь контейнеров по IP-адресу: запустите два контейнера и присвойте им IP-адреса из одной пользовательской сети. Обеспечьте взаимодействие между контейнерами по их IP-адресам.

```
C:\Ksen\docker>docker run -itd --name cont1 --network=my_custom_network --ip 172.18.0.2 ubuntu
4857f83c0a04c8dfef5758a205b80395f88174300f2b1757f67d5b6a1f77b845
```

Рисунок 4.1 – Запуск первого контейнера

```
C:\Ksen\docker>docker run -itd --name cont2 --network=my_custom_network --ip 172.18.0.2 ubuntu
8bd3c6bac590ed73f446bda50a46834762d53670bb37131243d9dd9cd9cbe810
```

Рисунок 4.2 – Запуск второго контейнера

5. Использование ссылок для связи контейнеров: используя устаревшую опцию `--link`, создайте два контейнера (например, с Nginx и MySQL) и свяжите их между собой. Убедитесь, что контейнер с Nginx может успешно обращаться к контейнеру с MySQL через имя контейнера, указанное при использовании опции `--link`.

```
C:\Ksen\docker>docker run -d --name dbb_container -e POSTGRES_PASSWORD=123 postgres
963c2083b5d6d83eb98bb4f2c62aa065dae871b12f8ba6a35a3aae1c26a914a4
```

Рисунок 5.1 – Запуск первого контейнера

```
C:\Ksen\docker>docker run -d --name webb_container --link dbb_container:postgres -p 8080:80 nginx
e447d8490b2520403eba81cc73b49b096ca06c71e5c427256b3939b8e4951e57
```

Рисунок 5.2 – Запуск второго контейнера

```

C:\Ksen\docker>docker exec -it webb_container bash
root@e447d8490b25:/# cat etc
cat: etc: Is a directory
root@e447d8490b25:/# cd etc
root@e447d8490b25:/etc# cat hosts
127.0.0.1        localhost
::1            localhost ip6-localhost ip6-loopback
fe00::0        ip6-localnet
ff00::0        ip6-mcastprefix
ff02::1        ip6-allnodes
ff02::2        ip6-allrouters
172.17.0.5      postgres 963c2083b5d6 dbb_container
172.17.0.6      e447d8490b25
root@e447d8490b25:/etc#

```

Рисунок 5.3 – Проверка связи контейнеров

Вопросы для самопроверки:

1. Чтобы создать новый том в Docker, нужно выполнить команду `docker volume create <имя_тома>`. Например, чтобы создать том с именем `my_volume`, нужно выполнить команду `docker volume create my_volume`.
2. Чтобы удалить существующий том в Docker, нужно выполнить команду `docker volume rm <имя_тома>`. Например, чтобы удалить том с именем `my_volume`, нужно выполнить команду `docker volume rm my_volume`.
3. Чтобы просмотреть список всех созданных томов в Docker, нужно выполнить команду `docker volume ls`.
4. Чтобы создать том с определенным именем, нужно выполнить команду `docker volume create <имя_тома>`. Например, чтобы создать том с именем `my_volume`, нужно выполнить команду `docker volume create my_volume`.
5. Чтобы присоединить том к контейнеру при его запуске, нужно использовать опцию `-v` или `--mount` при запуске контейнера. Например, чтобы присоединить том с именем `my_volume` к контейнеру, нужно выполнить команду `docker run -v my_volume:/path/to/mount <имя_образа>`.
6. Чтобы просмотреть подробную информацию о конкретном томе в Docker, нужно выполнить команду `docker volume inspect <имя_тома>`. Например, чтобы просмотреть информацию о томе с именем `my_volume`, нужно выполнить команду `docker volume inspect my_volume`.

7. Чтобы создать новую сеть в Docker, нужно выполнить команду `docker network create <имя_сети>`. Например, чтобы создать сеть с именем `my_network`, нужно выполнить команду `docker network create my_network`.

8. Чтобы удалить существующую сеть в Docker, нужно выполнить команду `docker network rm <имя_сети>`. Например, чтобы удалить сеть с именем `my_network`, нужно выполнить команду `docker network rm my_network`.

9. Чтобы просмотреть список всех созданных сетей в Docker, нужно выполнить команду `docker network ls`.

10. Чтобы создать пользовательскую сеть с определенным именем, нужно выполнить команду `docker network create <имя_сети>`. Например, чтобы создать сеть с именем `my_network`, нужно выполнить команду `docker network create my_network`.

11. Чтобы присоединить контейнер к пользовательской сети при его запуске, нужно использовать опцию `--network` при запуске контейнера. Например, чтобы присоединить контейнер к сети с именем `my_network`, нужно выполнить команду `docker run --network my_network <имя_образа>`.

12. Чтобы просмотреть подробную информацию о конкретной сети в Docker, нужно выполнить команду `docker network inspect <имя_сети>`. Например, чтобы просмотреть информацию о сети с именем `my_network`, нужно выполнить команду `docker network inspect my_network`.

13. Чтобы указать определенную сеть при запуске контейнера с использованием `docker run`, нужно использовать опцию `--network`. Например, чтобы запустить контейнер на сети с именем `my_network`, нужно выполнить команду `docker run --network my_network <имя_образа>`.

14. Если не указана конкретная сеть, то контейнер будет подключен к сети “bridge” по умолчанию.

15. Чтобы присоединить контейнер к нескольким сетям сразу при его запуске, нужно использовать опцию `--network` несколько раз. Например, чтобы присоединить контейнер к сетям с именами `my_network1` и `my_network2`,

нужно выполнить команду `docker run --network my_network1 --network my_network2 <имя_образа>`.

16. Чтобы просмотреть список сетей, доступных на хосте Docker, нужно выполнить команду `docker network ls`.

17. Чтобы создать контейнер, подключенный к сети “bridge”, нужно выполнить команду `docker run <имя_образа>`. По умолчанию, контейнер будет подключен к сети “bridge”.

18. Чтобы создать контейнер, подключенный к сети “host”, нужно выполнить команду `docker run --network host <имя_образа>`. При использовании сети “host”, контейнер использует сетевые настройки хоста, а не свои собственные.