

РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций
«Работа в Docker с сетью контейнеров и томами»

Отчет по лабораторной работе по
дисциплине «Анализ данных»

Выполнил студент группы ИВТ-б-о-21-1

Стригалов Дмитрий.

«20» ноября 2023г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил Воронкин Р.А. _____
(подпись)

Ставрополь 2023

Цель работы: научиться использовать Docker для управления томами и сетями.

Порядок выполнения работы:

Задача 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.

```
root@ReviOS:~# docker network create my_custom_network
a9de2b7c2d5959f39b022ab84b26ae2b6edc474eedad8b89541778b564f2a8da
root@ReviOS:~# docker run --network=my_custom_network -d --name web_container nginx
52524c491a050e9009a6f15a663a457224352aa37a9ecada93fc45690bcfc185
```

Рисунок 1 – Создание сети

```
root@ReviOS:~# docker run --network=my_custom_network -d --name db_container -e POSTGRES_PASSWORD=123 postgres
```

```

root@ReviOS:~# docker network inspect my_custom_network
[
  {
    "Name": "my_custom_network",
    "Id": "a9de2b7c2d5959f39b022ab84b26ae2b6edc474eedad8b89541778b564f2a8da",
    "Created": "2023-12-09T11:26:58.493947885Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.18.0.0/16",
          "Gateway": "172.18.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "52524c491a050e9009a6f15a663a457224352aa37a9ecada93fc45690bcfc185": {
        "Name": "web_container",
        "EndpointID": "cacf2101b23bef611922a18f660101766d9cd9cc302fa69bb1e70b132be73f5b",
        "MacAddress": "02:42:ac:12:00:02",
        "IPv4Address": "172.18.0.2/16",
        "IPv6Address": ""
      },
      "6f2c65c82cf5d84785a1cba9b4c6386ebc4c3ebd1665b2af3b8b38c059a32985": {
        "Name": "db_container",
        "EndpointID": "3910f39c69db29bb6fea9bbf1641fc142b0385acd09473ff14919017f4bd9fbd",
        "MacAddress": "02:42:ac:12:00:03",
        "IPv4Address": "172.18.0.3/16",
        "IPv6Address": ""
      }
    }
  }
]

```

Рисунок 2 - Проверка наличия контейнеров в сети

Задача 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.

```

root@ReviOS:~# docker volume create shared_data
shared_data
root@ReviOS:~# docker run -d -v shared_data:/data --name container1 nginx
f031bd2a6cfa2f1279ffb1427c3bb954d7b88791d506113b0e6a6297f05dc26d
root@ReviOS:~# docker exec -it container1 bash
root@f031bd2a6cfa:/# echo "Hello from container1" > /data/data_file.txt
root@f031bd2a6cfa:/# exit
exit
root@ReviOS:~# docker run -d -v shared_data:/data --name container2 nginx
7bc870ce85d492406a0c583e9cc702bc79b113137272e7ee9bf1507d2689306a
root@ReviOS:~# docker exec -it container2 bash
root@7bc870ce85d4:/# cat /data/data_file.txt
Hello from container1
root@7bc870ce85d4:/#

```

Рисунок 3 – Передача данных через том

Задача 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.

```

root@ReviOS:~# docker swarm init
Swarm initialized: current node (nhog98hxcg40em5otucxvizmyi) is now a manager.

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

    docker swarm join --token SWMTKN-1-0jel9ze6izx99hemwuc5jn6fmlfoadjapwh47iqlxusw5ph5f
p-5o4czyapl6wua4vxa7u39jo2h 192.168.65.3:2377

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

root@ReviOS:~# docker network create -d overlay --attachable my_overlay_network
hctdj968zxd8jlbvg7crlveqp
root@ReviOS:~# docker run --network=my_overlay_network -d --name sw1 nginx
23f21ff25fb43389b579b806ee7e628afc4d45a6692995586344a76b1f6fa44c
root@ReviOS:~# docker run --network=my_overlay_network -d --name sw2 nginx
75d65b77750f2b4dd4c86a06d1f9888bd74a9cc2757cf6be8d94b5925df68de7

```

Рисунок 4 – Создание сети overlay

Задача 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.

```

root@ReviOS:~# docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' web_container
172.18.0.2
root@ReviOS:~# docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' db_container
172.18.0.3
root@ReviOS:~# docker exec -it web_container bash

```

```

root@52524c491a05:/# ping 172.18.0.3
PING 172.18.0.3 (172.18.0.3) 56(84) bytes of data.
64 bytes from 172.18.0.3: icmp_seq=1 ttl=64 time=0.053 ms
64 bytes from 172.18.0.3: icmp_seq=2 ttl=64 time=0.091 ms
64 bytes from 172.18.0.3: icmp_seq=3 ttl=64 time=0.077 ms
64 bytes from 172.18.0.3: icmp_seq=4 ttl=64 time=0.068 ms
^C
--- 172.18.0.3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3094ms
rtt min/avg/max/mdev = 0.053/0.072/0.091/0.013 ms

```

Рисунок 5 - Связь контейнеров по IP-адресу

Задача 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`.

```

root@ReviOS:~# docker run --name mysql_container -e MYSQL_ROOT_PASSWORD=my -d mysql
Unable to find image 'mysql:latest' locally
latest: Pulling from library/mysql
e9f2695d7e5b: Pull complete
c041cd0148ec: Pull complete
27c9fbf7aa29: Pull complete
62fc1efc1f1f: Pull complete
e1d25a6611c2: Pull complete
5846de7fe479: Pull complete
faf13e3256e8: Pull complete
2217ed684a4f: Pull complete
45bfd3acf105: Pull complete
5b68afdb04ae: Pull complete
Digest: sha256:6057dec95d87a0d7880d9cfc9b3d9292f9c11473a5104b906402a2b73396e377
Status: Downloaded newer image for mysql:latest
ee8596aed786ed1a353eb5d9ba8eb2f569f95f681e314b3a71fa850ed182d002
root@ReviOS:~# docker run -d --name ngi_container --link mysql_container:db -p 8080:80 nginx
50c595b3f68edcabe84181c8078e0f4dab2d3db4795d7e3f09e2e38c469756ac

```

Рисунок 6 – Связь контейнеров с помощи ссылки

Индивидуальное задание

Создать контейнер `ignite_container_1` с образом `apacheignite`, таблицу в базе данных и несколько записей в ней. Сделать возможным доступ к созданной таблице в новом контейнере `ignite_container_2`, а также выполнить к ней запрос после удаления контейнера `ignite_container_1` и образа `apacheignite`.

```
root@ReviOS:~# docker volume create for_ignite
for_ignite
```

```
root@ReviOS:~# docker run -d --name ignite_container_2 --network ignite_network \
-v ignite_data:/opt/ignite/work \
  apacheignite/ignite
53841308d9f173fcaa8e8174ffad9382d77539d1eb3a45d89939f692a44c6346
root@ReviOS:~# docker exec -it ignite_container_2 /opt/ignite/apache-ignite/bin/sqlline.sh --color=true -u jdbc:ignite:t
hin://127.0.0.1/ -n ignite -p ignite
sqlline version 1.9.0
```

```
0: jdbc:ignite:thin://127.0.0.1/> create table users (Id LONG PRIMARY KEY, name VARCHAR) WITH "template=replicated";
No rows affected (0.032 seconds)
0: jdbc:ignite:thin://127.0.0.1/> INSERT INTO users (id, name) VALUES (1, 'Alex');
1 row affected (0.029 seconds)
0: jdbc:ignite:thin://127.0.0.1/> INSERT INTO users (id, name) VALUES (2, 'Jackson');
1 row affected (0.002 seconds)
0: jdbc:ignite:thin://127.0.0.1/> select * from users;
+-----+-----+
| ID | NAME |
+-----+-----+
| 1 | Alex |
| 2 | Jackson |
+-----+-----+
2 rows selected (0.026 seconds)
```

```
root@ReviOS:~# docker rmi -f apacheignite/ignite
Untagged: apacheignite/ignite:latest
Untagged: apacheignite/ignite@sha256:43c9273e85682f62ca0d2a2ae00bb21f13ccbc9c6a3dd17bbc2097ee262fd0a2
root@ReviOS:~# docker rm -f ignite_container_2
ignite_container_2
```

Рисунок 7 – Работа с первым контейнером

```
root@ReviOS:~# docker run -d --name ignite_container_3 --network ignite_network \
-v ignite_data:/opt/ignite/work \
  apacheignite/ignite
Unable to find image 'apacheignite/ignite:latest' locally
latest: Pulling from apacheignite/ignite
Digest: sha256:43c9273e85682f62ca0d2a2ae00bb21f13ccbc9c6a3dd17bbc2097ee262fd0a2
Status: Downloaded newer image for apacheignite/ignite:latest
272991350f9f0dbcb125c7dc7663190a1b422a393b459c3c221c291c99cb73a53
root@ReviOS:~# docker exec -it ignite_container_3 /opt/ignite/apache-ignite/bin/sqlline.sh --color=true -u
jdbc:ignite:thin://127.0.0.1/ -n ignite -p ignite
sqlline version 1.9.0
0: jdbc:ignite:thin://127.0.0.1/> SELECT * FROM users;
+-----+-----+
| ID | NAME |
+-----+-----+
| 1 | Alex |
| 2 | Jackson |
+-----+-----+
2 rows selected (0.062 seconds)
```

Рисунок 8 - Работа со вторым контейнером

Контрольные вопросы:

1. Как создать новый том в Docker?

docker volume create

2. Как удалить существующий том в Docker?

docker volume rm

3. Как просмотреть список всех созданных томов в Docker?

docker volume ls

4. Как создать том с определенным именем?

`docker volume create my_volume`

5. Как присоединить том к контейнеру при его запуске?

`docker run -v /путь/на/хосте:/путь/в/контейнере -d image_name`

6. Как просмотреть подробную информацию о конкретном томе в Docker?

`docker volume inspect my_volume`

7. Как создать новую сеть в Docker?

`docker network create my_custom_network`

8. Как удалить существующую сеть в Docker?

`docker network rm my_custom_network`

9. Как просмотреть список всех созданных сетей в Docker?

`docker network ls`

10. Как создать пользовательскую сеть с определенным именем?

`docker network create my_custom_network`

11. Как присоединить контейнер к пользовательской сети при его запуске?

`docker run --network=my_custom_network -d nginx`

12. Как просмотреть подробную информацию о конкретной сети в Docker?

`docker network inspect my_network`

13. Как указать определенную сеть при запуске контейнера с использованием `docker run` ?

`docker run --network=my_custom_network -d nginx`

14. Какие сети будут доступны по умолчанию для контейнера, если не указана конкретная сеть?

bridge, host и none.

15. Как присоединить контейнер к нескольким сетям сразу при его запуске?

```
docker run --network=my_custom_network -d nginx
```

16. Как просмотреть список сетей, доступных на хосте Docker?

```
docker network ls
```

17. Как создать контейнер, подключенный к сети "bridge"?

```
docker run --network=bridge -d nginx
```

18. Как создать контейнер, подключенный к сети "host"?

```
docker run --network=host -d nginx
```

Вывод: были изучены способы использования Docker для управления томами и сетями.