

**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

Богдашов Артём Владимирович.

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@DESKTOP-V640UPV:~# docker network create my_customaze_network
045eadc05415049e18bf66e8af85e2155d5168b3625efbee5d66c484b04a3761
root@DESKTOP-V640UPV:~# docker run --network=my_customaze_network -d --name web_container nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
1f7ce2fa46ab: Pull complete
9b16c94bb686: Pull complete
9a59d19f9c5b: Pull complete
9ea27b074f71: Pull complete
c6edf33e2524: Pull complete
84b1ff10387b: Pull complete
517357831967: Pull complete
Digest: sha256:10d1f5b58f74683ad34eb29287e07dab1e90f10af243f151bb50aa5dbb4d62ee
Status: Downloaded newer image for nginx:latest
643ab3c05507e4748d696c0d1d19e4dad9c7bcbc1866b64780f4a0d7569ecb36
root@DESKTOP-V640UPV:~#
```

Рисунок 1 – Создание сети

```

root@DESKTOP-V640UPV:~# docker network inspect my_customize_network
[
  {
    "Name": "my_customize_network",
    "Id": "045eadc05415049e18bf66e8af85e2155d5168b3625efbee5d66c484b04a3761",
    "Created": "2023-12-07T12:19:50.785505657Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.19.0.0/16",
          "Gateway": "172.19.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "3274b69331cd68c6e84bc0c2f19e375d49666f5a1643496f1e1ebf51f4e4d777": {
        "Name": "db_container",
        "EndpointID": "4ae6f7da9b765765daa090fdf9377896531a2d5c2d102955be1c695642122af3",
        "MacAddress": "02:42:ac:13:00:03",
        "IPv4Address": "172.19.0.3/16",
        "IPv6Address": ""
      },
      "643ab3c05507e4748d696c0d1d19e4dad9c7bcb1866b64780f4a0d7569ecb36": {
        "Name": "web_container",
        "EndpointID": "9a5c25c6ffa86f84f6d477f277110008a3deae988141edd5ff95b8b07ec4c1a5",
        "MacAddress": "02:42:ac:13:00:02",
        "IPv4Address": "172.19.0.2/16",
        "IPv6Address": ""
      }
    },
    "Options": {},
    "Labels": {}
  }
]

```

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@DESKTOP-V640UPV:~# docker volume create shared_data
shared_data
root@DESKTOP-V640UPV:~# docker run -d -v shared_data:/data --name container1 nginx
4aa3bb80e69d91369538f12b1d53e125af76e11e969796d0b7bb433e9f146a4d
root@DESKTOP-V640UPV:~# docker exec -it container1" > /data/data_file.txt
> ^C
root@DESKTOP-V640UPV:~# docker exec ~it container1 bash
Error response from daemon: No such container: ~it
root@DESKTOP-V640UPV:~# docker exec -it container1 bash
root@4aa3bb80e69d:/# echo "Hello from container1" > /data/data_file.txt
root@4aa3bb80e69d:/# exit
exit
root@DESKTOP-V640UPV:~# docker run -d -v shared_data:/data --name container2 nginx
1e19eeebae48ef7c8f4573e16a8f1c5c0ec99eb138dea33b4b577756aed63505
root@DESKTOP-V640UPV:~# docker exec -it container2 bash
root@1e19eeebae48:/# cat /data/data_file.txt
Hello from container1

```

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@DESKTOP-V640UPV:~# docker swarm init
Swarm initialized: current node (th5ugjml7fo3i50k69opnni3y) is now a manager.

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

    docker swarm join --token SWMTKN-1-61dt2w5mlcgtpmfmkeoql4t0wgqb6i07qnxblul5tuardniipjt-0594n6oh903d24cs8cxutd30b 192.168.65.3:2377

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

root@DESKTOP-V640UPV:~# docker network create -d overlay --attachable my_overlay_network
6cqs6ue9715cxmqb16t3en7f
root@DESKTOP-V640UPV:~# docker run --network=my_overlay_network -d --name sw1 nginx
987816ddd07501462c44521db7c00c0be61fc9c20e2a15a15c8260cc9879c33f
root@DESKTOP-V640UPV:~# docker run --network=my_overlay_network -d --name swarm1 nginx
ad29f9a4dc85c7056277df888c0b09f98bc25ca347b80e7ea455ac59d5e25204
root@DESKTOP-V640UPV:~# docker run --network=my_overlay_network -d --name swarm2 nginx
30053ea90124c6a59eddc13606a1e77c46d280421b5aaca6669d07fe11e7eccf

```

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@DESKTOP-V640UPV:~# docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' db_container
172.19.0.3
root@DESKTOP-V640UPV:~# docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' web_container
172.19.0.2
root@DESKTOP-V640UPV:~# docker exec -it web_container bash
root@643ab3c05507:/# ping 172.19.0.3
PING 172.19.0.3 (172.19.0.3) 56(84) bytes of data.
64 bytes from 172.19.0.3: icmp_seq=1 ttl=64 time=0.134 ms
64 bytes from 172.19.0.3: icmp_seq=2 ttl=64 time=0.110 ms
64 bytes from 172.19.0.3: icmp_seq=3 ttl=64 time=0.092 ms
64 bytes from 172.19.0.3: icmp_seq=4 ttl=64 time=0.114 ms
64 bytes from 172.19.0.3: icmp_seq=5 ttl=64 time=0.114 ms
64 bytes from 172.19.0.3: icmp_seq=6 ttl=64 time=0.098 ms
^C
--- 172.19.0.3 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5181ms
rtt min/avg/max/mdev = 0.092/0.110/0.134/0.013 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@DESKTOP-V640UPV:~# docker run --name mysql_container -e MYSQL_ROOT_PASSWORD=my -d mysql
Unable to find image 'mysql:latest' locally
latest: Pulling from library/mysql
8e0176adc18c: Pull complete
2d2c52718f65: Pull complete
d88d03ce139b: Pull complete
4a7d7f11aa1e: Pull complete
ce5949193e4c: Pull complete
f7f024dfb329: Pull complete
5fc3c840facc: Pull complete
509068e49488: Pull complete
cbc847bab598: Pull complete
942bef62a146: Pull complete
Digest: sha256:1773f3c7aa9522f0014d0ad2bbdaf597ea3b1643c64c8ccc2123c64afd8b82b1
Status: Downloaded newer image for mysql:latest
a6b5d061116d6a840b0d348c37077cb70caa494efe7b621f8d2892ef3605c8c9
root@DESKTOP-V640UPV:~# docker run -d --name ng_container --link mysql_container:db -p 8080:80 nginx
24892cc26e7a427527fcdad7d8bb8624e5f6b7cdf564e3fd03cf4b8336a75bbca
root@DESKTOP-V640UPV:~#

```

Рисунок 6 – Связь контейнеров с помощи ссылки

## Индивидуальное задание

Создать контейнер `post_2` с образом `firebird`, таблицу в базе данных и несколько записей в ней. Сделать возможным доступ к созданной таблице в новом контейнере `post_1`, а также выполнить к ней запрос после удаления контейнера `post_2` и образа `postgres`.

```

root@DESKTOP-V640UPV:~# docker volume create for_direbird
root@DESKTOP-V640UPV:~# docker run --name post_2 -v for_firebird:/firebird/data -e ISC_USER=sysdba -e ISC_PASSWORD=masterkey -p 3053:3050 -d firebird_copy
be477a893ef1c8bb5f93b610e326db762d422a718be57416af6754d3284fdd9
root@DESKTOP-V640UPV:~# docker exec -it post_2 /bin/bash
root@be477a893ef1:/# /usr/local/firebird/bin/isql -z
ISQL Version: LI-V4.0.2.2816 Firebird 4.0
Use CONNECT or CREATE DATABASE to specify a database
SQL> CONNECT '/firebird/data/new_base.fdb' USER 'SYSDBA' PASSWORD 'masterkey';
Server version:
LI-V4.0.2.2816 Firebird 4.0
Database: '/firebird/data/new_base.fdb', User: SYSDBA
SQL> DROP TABLE users;
SQL> CREATE TABLE users (
CON> ID INTEGER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
CON> NAME VARCHAR(30),
CON> Age INTEGER
CON> );
SQL> INSERT INTO users (NAME, Age) VALUES ('Tom', 25);
SQL> INSERT INTO users (NAME, Age) VALUES ('Jarry', 20);
SQL> COMMIT;
SQL> SELECT * FROM users;

      ID NAME                                AGE
-----
      1 Tom                                25
      2 Jarry                             20

SQL> ^Z
[1]+  Stopped                  /usr/local/firebird/bin/isql -z
root@be477a893ef1:/# exit
exit
There are stopped jobs.
root@be477a893ef1:/#
exit
root@DESKTOP-V640UPV:~# docker rm -f post_2
post_2
root@DESKTOP-V640UPV:~#

```

Рисунок 7 – Работа с первым контейнером

```

root@DESKTOP-V640UPV:~# docker run --name post_1 -v for_firebird:/firebird/data -e ISC_USER=sysdba -e ISC_PASSWORD=masterkey -p 3052:3050 -d firebird_copy
441e4fa89130da3e5df2db6fea8526ccddc3df999eb68c78f3b013b6233e9f90
root@DESKTOP-V640UPV:~# docker exec -it post_1 /bin/bash
root@441e4fa89130:/# /usr/local/firebird/bin/isql -z
ISQL Version: LI-V4.0.2.2816 Firebird 4.0
Use CONNECT or CREATE DATABASE to specify a database
SQL> CONNECT '/firebird/data/new_base.fdb' USER 'SYSDBA' PASSWORD 'masterkey';
Server version:
LI-V4.0.2.2816 Firebird 4.0
Database: '/firebird/data/new_base.fdb', User: SYSDBA
SQL> SELECT * FROM users;

      ID NAME                                AGE
-----
      1 Tom                                25
      2 Jarry                             20

SQL>

```

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.