

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное
образовательное учреждение высшего образования
«Северо-Кавказский федеральный университет»**

Кафедра инфокоммуникаций

**Отчет по лабораторной работе № 2
«Работа в Docker с сетью контейнеров и томами»
по дисциплине «Курсы DevOps»**

Выполнил студент группы ИВТ-б-о-21-1

Богадуров В.И. « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил доцент Кафедры Инфокоммуни-
каций, старший преподаватель

Воронкин Р.А. _____

(подпись)

Ставрополь 2023

Цель: познакомить студентов с использованием Docker для управления томами и сетями.

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.

```
PS C:\Users\Admin> docker network create my_custom_network
2ff63ace14c24283492d3483bc0286a934449a3c8539c0d22d61b0bd3ca0a5a6
PS C:\Users\Admin>
```

Рисунок 1 – Создание пользовательской сети

```
PS C:\Users\Admin> docker run --network=my_custom_network -d nginx
a9213b7f7053de3c6e01604a51891da1e573687a398a08481cfd600423320c9
PS C:\Users\Admin> docker run --network=my_custom_network -d postgres
3534d90fa4e60410e08629b787fb7eb9cabd4da43bfed4171a3afddd49d66573
PS C:\Users\Admin>
```

Рисунок 2 – Запуск контейнера с присоединением к пользовательской сети

```
PS C:\Users\Admin> docker network inspect my_custom_network
[
  {
    "Name": "my_custom_network",
    "Id": "2ff63ace14c24283492d3483bc0286a934449a3c8539c0d22d61b0bd3ca0a5a6",
    "Created": "2023-12-07T11:35:24.721354766Z",
    "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": {
      "36f525d25539db81a500d9775258652dbfbf6568f4ab828fc1ad7b15010ef5d5": {
        "Name": "vigorous_mahavira",
        "EndpointID": "8737c930e0a07d194b9ea8ee1a7ae382972316d822ea9d1752f7b328dfca1d1",
        "MacAddress": "02:42:ac:12:00:03",
        "IPv4Address": "172.18.0.3/16",
        "IPv6Address": ""
      },
      "a9213b7f7053de3c6e01604a51891da1e573687a398a08481cfd600423320c9": {
        "Name": "vigilant_gates",
        "EndpointID": "a952eed66dab2d52594f3e5508c8f9766e64b2457a845e405315175938a1bea4",
        "MacAddress": "02:42:ac:12:00:02",
        "IPv4Address": "172.18.0.2/16",
        "IPv6Address": ""
      }
    },
    "Options": {},
    "Labels": {}
  }
]
PS C:\Users\Admin> docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS        NAMES
36f525d25539   postgres  "docker-entrypoint.s..." 16 minutes ago Up 16 minutes 5432/tcp     vigorous_mahavira
a9213b7f7053   nginx     "/docker-entrypoint..." 22 minutes ago Up 22 minutes 80/tcp       vigilant_gates
```

Рисунок 3 – Оба контейнера находятся в 1-ой сети

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.

```
PS C:\Users\Admin> docker volume create my_volume
my_volume
PS C:\Users\Admin>
```

Рисунок 4 – Создание тома

```
PS C:\Users\Admin> docker run -d -v my_volume:/data --name container1 ubuntu
623553f9c0b0348c43b23a2347ced7075f57fe363b3aadb793942e4c0eab860b
```

Рисунок 5 – Создание 1 контейнера и присоединение его к общему тому

```
PS C:\Users\Admin> docker run -d -v shared_data:/data --name container2 ubuntu
3b8e41ef3827365bce0772be0440cb037d1f44db94649d01233ceec95580cd5d
```

Рисунок 6 – Создание 2-го контейнера и присоединение его к общему тому

```
PS C:\Users\Admin> docker run -it -v my_volume:/data --name container11 ubuntu
root@0cd45784f1e4:/# echo "Hello from container11" > /data/data_file.txt
root@0cd45784f1e4:/# exit
exit
PS C:\Users\Admin> docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS   NAMES
a9213b7f7053   nginx    "/docker-entrypoint..." 52 minutes ago Up 52 minutes 80/tcp   vigilant_gates
0912a47b6cc6   nginx    "/docker-entrypoint..." 56 minutes ago Up 56 minutes 80/tcp   eb_container
PS C:\Users\Admin> docker run -it -v my_volume:/data --name container22 ubuntu
root@5bfad16e2fbc:/# cat /data/data_file.txt
Hello from container11
root@5bfad16e2fbc:/#
```

Рисунок 7 – Просмотр файла, хранящегося в общем томе, в контейнере 2

3. Создание сети overlay для распределенного приложения:

Задача: 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.

```
PS C:\Users\Admin> docker swarm init
Swarm initialized: current node (oy2dfs5wp81zb071hm70pyzi3) is now a manager.

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

    docker swarm join --token SWMTKN-1-49vh73gysv469bgff1kf77ygf7x372fna8ugc1v6se1b53t5nc-86f1
r50tqc54kz3kxobs89cye 192.168.65.3:2377

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

PS C:\Users\Admin>
```

Рисунок 8 – Использование Docker Swarm для создания кластера

```
PS C:\Users\Admin> docker network create -d overlay my_overlay_network
we2mzsu3qlmigit6o8iwptoez
PS C:\Users\Admin> docker run --network=my_overlay_network -d nginx
40d794de4d8f713592e16e60208845231d25bed401188e9ac66c5c7b3ca410bc
```

Рисунок 9 – Создание overlay-сети и запуск контейнеров

4. Связь контейнеров по IP-адресу:

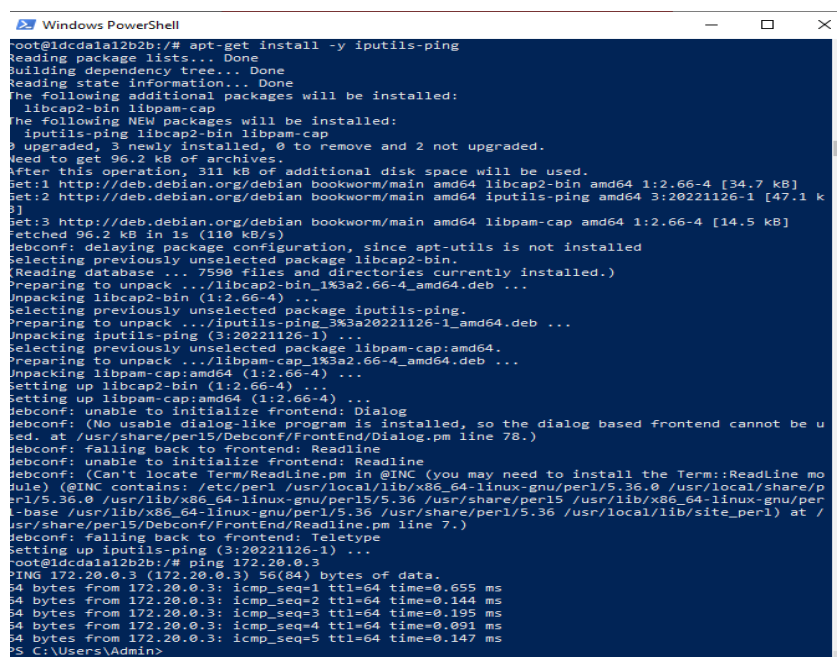
Задача: Запустите два контейнера и присвойте им IP-адреса из одной пользовательской сети. Обеспечьте взаимодействие между контейнерами по их IP адресам.

```
PS C:\Users\Admin> docker network create my_network
92602d71d9c5623c4160d7a77eab379fb6240bf67d7ee5c6aa9e32df7166f968
PS C:\Users\Admin> docker run --network=my_network -d --name web_container nginx
docker: Error response from daemon: Conflict. The container name "/web_container" is already i
n use by container "0912a47b6cc6cbfbf22b72f54b938d9ec8aa16f7b083b277c144121e80d4d376". You hav
e to remove (or rename) that container to be able to reuse that name.
See 'docker run --help'.
PS C:\Users\Admin> docker run --network=my_network -d --name web_container11 nginx
1dcda1a12b2bf69f60601e917d33ee4eeef778d46c7e2b835e4e2bf32056ed8b
PS C:\Users\Admin> docker run --network=my_network -d --name db_container postgres
dabdf2bb4ac076c9023cdd2c6b4d98f37ea0e88a5303f4e0651e2b537e677265
PS C:\Users\Admin>
```

Рисунок 10 – Запуск двух контейнеров в одной сети

```
PS C:\Users\Admin> docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}'
web_container11
172.20.0.2
PS C:\Users\Admin> docker run --network=my_network -e POSTGRES_PASSWORD=123qwe -d --name db_co
ntainer2 postgres
39b824a746354d9a23b07881073f257615680ccea15ba09cab3edd2f297a06e
PS C:\Users\Admin> docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}'
db_container2
172.20.0.3
PS C:\Users\Admin>
```

Рисунок 11 – IP адреса контейнеров



```
Windows PowerShell
root@1dcda1a12b2b:/# apt-get install -y iputils-ping
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcap2-bin libpam-cap
The following NEW packages will be installed:
  iputils-ping libcap2-bin libpam-cap
3 upgraded, 3 newly installed, 0 to remove and 2 not upgraded.
Need to get 96.2 kB of archives.
After this operation, 311 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bookworm/main amd64 libcap2-bin amd64 1:2.66-4 [34.7 kB]
Get:2 http://deb.debian.org/debian bookworm/main amd64 iputils-ping amd64 3:20221126-1 [47.1 kB]
Get:3 http://deb.debian.org/debian bookworm/main amd64 libpam-cap amd64 1:2.66-4 [14.5 kB]
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package libcap2-bin.
(Reading database ... 7590 files and directories currently installed.)
Preparing to unpack .../libcap2-bin_1%3a2.66-4_amd64.deb ...
Unpacking libcap2-bin (1:2.66-4) ...
Selecting previously unselected package iputils-ping.
Preparing to unpack .../iputils-ping_3%3a20221126-1_amd64.deb ...
Unpacking iputils-ping (3:20221126-1) ...
Selecting previously unselected package libpam-cap:amd64.
Preparing to unpack .../libpam-cap_1%3a2.66-4_amd64.deb ...
Unpacking libpam-cap:amd64 (1:2.66-4) ...
Setting up libcap2-bin (1:2.66-4) ...
Setting up libpam-cap:amd64 (1:2.66-4) ...
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based frontend cannot be u
sed. at /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 78.)
debconf: falling back to frontend: Readline
debconf: unable to initialize frontend: Readline
debconf: (Can't locate Term/Readline.pm in @INC (you may need to install the Term::Readline mo
dule) (@INC contains: /etc/perl /usr/local/lib/x86_64-linux-gnu/perl/5.36.0 /usr/local/share/p
erl/5.36.0 /usr/lib/x86_64-linux-gnu/perl5/5.36 /usr/share/perl5 /usr/lib/x86_64-linux-gnu/per
l-base /usr/lib/x86_64-linux-gnu/perl/5.36 /usr/share/perl/5.36 /usr/local/lib/site_perl) at /
usr/share/perl5/Debconf/FrontEnd/Readline.pm line 7.)
debconf: falling back to frontend: Teletype
Setting up iputils-ping (3:20221126-1) ...
root@1dcda1a12b2b:/# ping 172.20.0.3
PING 172.20.0.3 (172.20.0.3) 56(84) bytes of data.
64 bytes from 172.20.0.3: icmp_seq=1 ttl=64 time=0.655 ms
64 bytes from 172.20.0.3: icmp_seq=2 ttl=64 time=0.144 ms
64 bytes from 172.20.0.3: icmp_seq=3 ttl=64 time=0.195 ms
64 bytes from 172.20.0.3: icmp_seq=4 ttl=64 time=0.091 ms
64 bytes from 172.20.0.3: icmp_seq=5 ttl=64 time=0.147 ms
PS C:\Users\Admin>
```

Рисунок 12 – Проверка взаимодействия контейнеров по их IP адресам

```
root@1dcda1a12b2b:/# ping 172.20.0.3
PING 172.20.0.3 (172.20.0.3) 56(84) bytes of data.
64 bytes from 172.20.0.3: icmp_seq=1 ttl=64 time=0.655 ms
64 bytes from 172.20.0.3: icmp_seq=2 ttl=64 time=0.144 ms
64 bytes from 172.20.0.3: icmp_seq=3 ttl=64 time=0.195 ms
64 bytes from 172.20.0.3: icmp_seq=4 ttl=64 time=0.091 ms
64 bytes from 172.20.0.3: icmp_seq=5 ttl=64 time=0.147 ms
PS C:\Users\Admin>
```

Рисунок 13 – Проверка взаимодействия

5. Использование ссылок для связи контейнеров:

Задача: 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`.

```
PS C:\Users\Admin> docker run -d --name web_container12111 --link sql:db -p 80:80 nginx
41ceff585d123b2e45aa0d56c51714792d19ecf0f72d278cb058241384779347
```

Рисунок 14 – Связывание двух контейнеров с опцией `--link`

Вывод: в результате выполнения лабораторной работы были получены практические навыки необходимые для использования Docker, чтобы управлять томами и сетями.

Ответы на контрольные вопросы:

1. Как создать новый том в Docker?

Ответ: `docker volume create название_тома`

2. Как удалить существующий том в Docker?

Ответ: `docker volume rm название_тома`

3. Как просмотреть список всех созданных томов в Docker?

Ответ: `docker volume ls`

4. Как создать том с определенным именем?

Ответ: `docker volume create название_тома`

5. Как присоединить том к контейнеру при его запуске?

Ответ: `docker run -d -v shared_data:/data --name container1 image1`

В этом примере контейнер `container1` использует том `shared_data` и монтирует его внутри контейнера по пути `/data`.

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

Ответ: `docker volume inspect my_volume`

7. Как создать новую сеть в Docker?

Ответ: `docker network create название_сети`

8. Как удалить существующую сеть в Docker?

Ответ: `docker network rm название_сети`

9. Как просмотреть список всех созданных сетей в Docker?

Ответ: `docker network ls`

10. Как создать пользовательскую сеть с определенным именем?

Ответ: `docker network create название_сети`

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

Ответ: `docker network connect my_custom_network container_id`

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

Ответ: `docker network inspect название_сети`

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

Ответ: `docker run --network=название_сети -d nginx`

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

Ответ: по умолчанию, без явного указания сети, контейнер в Docker будет подключен к сети по умолчанию. В Docker по умолчанию существует три сети: bridge, host и none.

bridge: Это сеть по умолчанию для контейнеров. Она позволяет контейнерам общаться друг с другом и с хостовой машиной. Каждый контейнер, созданный без указания другой сети, будет подключен к этой сети.

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

Ответ: При запуске контейнера в Docker вы можете присоединить его к нескольким сетям одновременно, указав список сетей через запятую. Для этого используется параметр --network (или его короткая форма -net).

```
docker run -d --network=network1,network2 my-image
```

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

Ответ: docker network ls

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

Ответ: docker run --network=bridge -d nginx

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

Ответ: docker run --network=host -d nginx