Docker — ?????? ? ?????? (Networks)

Эта страница объясняет, как Docker управляет сетями и как контейнеры взаимодействуют друг с другом. Здесь разобраны основные команды для просмотра сетей, диагностики сетевых проблем и проверки портов сервисов. Материал ориентирован на начинающих и помогает понять, **как контейнеры находят друг друга внутри Docker-проекта и как сервисы становятся доступными снаружи**.

??? ???????? ???? ? Docker

По умолчанию каждый контейнер подключается к одной или нескольким Docker-сетям. Сеть позволяет контейнерам:

- общаться друг с другом по имени сервиса
- передавать данные между сервисами
- изолировать проекты друг от друга

Например, в одном проекте могут работать контейнеры:

- `nginx`
- `php`
- `mysql`
- `redis`

Все они находятся в одной сети и могут обращаться друг к другу по имени сервиса:

```
php -> mysql
php -> redis
nginx -> php
```

Пример подключения к базе внутри контейнера:

```
mysql:3306
```

Где `mysql` — имя сервиса из `docker-compose.yml`.

?????? Docker-?????

???????? ??? ????

```
docker network ls
```

Пример вывода:

```
NETWORK ID     NAME               DRIVER
9a8b7c6d5e     bridge             bridge
8f7e6d5c4b     host               host
7e6d5c4b3a     project_default    bridge
```

??? ???????? ?????? ????

Сеть	Назначение
bridge	стандартная сеть Docker
host	контейнер использует сеть хоста
project_default	сеть Docker Compose проекта

???? Docker Compose

Когда запускается проект через:

```
docker compose up -d
```

Docker автоматически создаёт сеть:

```
projectname_default
```

Например:

```
site1_default
```

Все сервисы из `docker-compose.yml` автоматически подключаются к этой сети.

Это позволяет контейнерам обращаться друг к другу **по имени сервиса**.

???????? ???? ???????

??????????? ??????????? ? ????

```
docker network inspect <network_name>
```

Пример:

```
docker network inspect site1_default
```

Эта команда показывает:

- подключенные контейнеры
- IP-адреса контейнеров
- настройки сети

?????? ??????

Внутри результата можно увидеть блок:

```
"Containers": {
  "php_container": {
    "Name": "site1_php",
    "IPv4Address": "172.20.0.3"
  },
  "mysql_container": {
    "Name": "site1_mysql",
    "IPv4Address": "172.20.0.4"
  }
}
```

Это означает, что контейнеры подключены к сети и могут взаимодействовать.

???????? ?????? ????????

Контейнер может быть доступен:

- **внутри Docker-сети**
- **снаружи сервера**

Для доступа извне используются **проброшенные порты**.

?????????? ???? ???????

```
docker compose port <service> <port>
```

Пример:

```
docker compose port nginx 80
```

Результат:

```
0.0.0.0:8080
```

Это означает, что контейнерный порт `80` доступен на сервере через порт `8080`.

???????? ?????? ????? docker ps

Можно также проверить порты так:

```
docker ps
```

Пример вывода:

```
CONTAINER ID   IMAGE      PORTS
ab12cd34       nginx      0.0.0.0:8080->80/tcp
```

Это означает:

```
сервер:8080 -> контейнер:80
```

?????????? ?????????? ? ????

Иногда контейнер нужно подключить к существующей сети.

?????????? ??????????

```
docker network connect <network> <container>
```

Пример:

```
docker network connect site1_default redis_container
```

После этого контейнер сможет взаимодействовать с сервисами сети.

?????????? ?????????? ?? ????

```
docker network disconnect <network> <container>
```

Пример:

```
docker network disconnect site1_default redis_container
```

Контейнер больше не сможет общаться с сервисами этой сети.

???????? ???????? ??????? ?????? ???????????

Если сервисы не могут взаимодействовать, нужно проверить соединение.

????????????? ? ???????????

```
docker compose exec php sh
```

????????? ?????? ? ??????? ???????

Например, проверить базу данных:

```
ping mysql
```

или

```
nc -zv mysql 3306
```

Если соединение работает — сеть настроена правильно.

???????? DNS ?????? Docker

Docker автоматически создаёт DNS внутри сети.

Это позволяет обращаться к сервисам **по имени**.

Например:

```
mysql
redis
php
```

```
nginx
```

????????? DNS

Внутри контейнера:

```
nslookup mysql
```

или

```
ping mysql
```

Если DNS работает, контейнер получит IP.

???????? ??????? ????????

?????????? ?? ????? ???? ?????

Возможные причины:

- контейнеры находятся в разных сетях
- неправильное имя сервиса
- сервис не запущен

Проверить:

```
docker network inspect <network>
```

???????? ??? ???????

В Docker Compose нужно использовать **имя сервиса**, а не имя контейнера.

Пример:

```
mysql
```

а не:

```
project_mysql_1
```

???? ?? ???????? ???????

Проверить `docker-compose.yml`.

Пример правильного проброса:

```
ports:
  - "8080:80"
```

Это означает:

```
сервер:8080 -> контейнер:80
```

?????????? ???????, ?? ???? ?? ???????????

Проверить:

```
docker ps
```

Если порт не указан — он не проброшен наружу.

???????? ???????? ???????????? ????

Если сервис не работает:

1 ????????? ???????????

```
docker compose ps
```

2 ????????? ????

```
docker network ls
```

3 ????????? ????????????? ???????????

```
docker network inspect <network>
```

4 ????????? DNS ?????? ???????????

```
docker compose exec php ping mysql
```

??????? ???????

```
docker network ls                       # список сетей
docker network inspect <network>        # информация о сети

docker compose port <service> <port>    # проверить проброс порта

docker network connect <network> <container>     # подключить контейнер
docker network disconnect <network> <container>  # отключить контейнер

docker ps                               # посмотреть проброшенные порты
```

Revision #1
Created 2026-03-11 08:03:25 UTC by Crimson
Updated 2026-03-11 08:03:45 UTC by Crimson