Федеральное государственное бюджетное образовательное учреждение высшего образования

Московский государственный технический университет

им. Н. Э. Баумана

Национальный исследовательский университет (МГТУ им. Н.Э. Баумана)

Практическое задание:

«Облачные решения. Основы контейнеризации Docker»

Выполнил: Шарафутдинов Э. М.

Группа: РЛ6-61Б

Москва 2023 г.

Содержание

Цель работы

Ознакомиться с запуском сервисов в системе контейнеризации Docker. Научиться создавать и проводить базовые операции управления контейнерами в системе контейнеризации Docker.

Краткие теоретические сведения

Контейнеризация — метод виртуализации, при котором ядро операционной системы поддерживает несколько изолированных экземпляров пространства пользователя вместо одного. Эти экземпляры (обычно называемые контейнерами или зонами) с точки зрения выполняемых в них процессов идентичны отдельному экземпляру операционной системы. Для систем на базе Unix эта технология похожа на улучшенную реализацию механизма chroot. Ядро обеспечивает полную изолированность контейнеров, поэтому программы из разных контейнеров не могут воздействовать друг на друга.

В отличие от аппаратной виртуализации, при которой эмулируется аппаратное окружение и может быть запущен широкий спектр гостевых операционных систем, в контейнере может быть запущен экземпляр операционной системы только с тем же ядром, что и у хостовой операционной системы (все контейнеры узла используют общее ядро). При этом при контейнеризации отсутствуют дополнительные ресурсные накладные расходы на эмуляцию виртуального оборудования и запуск полноценного экземпляра операционной системы, характерные при аппаратной виртуализации

Порядок выполнения работы

1. Подготовка системы разрешения имён ВМ-2 путём редактирования файла /etc/hosts.

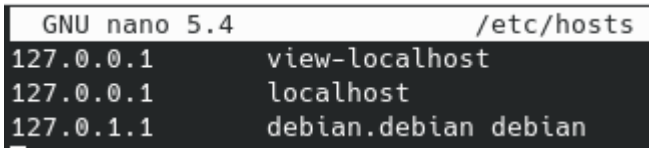

Рисунок 1 - изменение файла hosts ВМ-2

2. Убеждаемся, что демон Docker запущен на ВМ-1.

```
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; preset: enabled)
     Active: active (running) since Fri 2023-06-23 23:50:58 MSK; 1min 0s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 691 (dockerd)
      Tasks: 8
     Memory: 99.4M
        CPU: 3.646s
     CGroup: /system.slice/docker.service
             └─691 /usr/sbin/dockerd -H fd:// --containerd=/run/containerd/containerd.>

Jun 23 23:50:56 lol dockerd[691]: time="2023-06-23T23:50:56.179811523+03:00" level=inf>
Jun 23 23:50:56 lol dockerd[691]: time="2023-06-23T23:50:56.179845889+03:00" level=inf>
Jun 23 23:50:56 lol dockerd[691]: time="2023-06-23T23:50:56.322904126+03:00" level=inf>
Jun 23 23:50:56 lol dockerd[691]: time="2023-06-23T23:50:56.484219310+03:00" level=inf>
Jun 23 23:50:57 lol dockerd[691]: time="2023-06-23T23:50:57.975232419+03:00" level=inf>
Jun 23 23:50:58 lol dockerd[691]: time="2023-06-23T23:50:58.174845723+03:00" level=inf>
Jun 23 23:50:58 lol dockerd[691]: time="2023-06-23T23:50:58.710115621+03:00" level=inf>
Jun 23 23:50:58 lol dockerd[691]: time="2023-06-23T23:50:58.714266659+03:00" level=inf>
Jun 23 23:50:58 lol systemd[1]: Started docker.service - Docker Application Container >
Jun 23 23:50:58 lol dockerd[691]: time="2023-06-23T23:50:58.875702623+03:00" level=inf>
```

Рисунок 2 - проверка, запущен ли демон Docker на ВМ-1

3. Добавляем основного пользователя в группу docker.

```
emil@lol:~$ sudo usermod -aG docker emil
emil@lol:~$ newgrp docker
emil@lol:~$ groups
docker sudo users emil
```

Рисунок 3 - добавление пользователя в группу docker

4. Подготавливаем системные переменные окружения. Для этого в файл `/etc/environment` добавляем строку: `export GITLAB_HOME=/var/GitlabVolumes`.

5. Подготовка системы разрешения имён ВМ-1 путём редактирования файла /etc/hosts.

```
 GNU nano 7.2                                  /etc/hosts *
127.0.0.1       view-localhost
127.0.0.1       localhost
127.0.1.1       lol.myguest.virtualbox.org      lol
```

Рисунок 4 - изменение файла hosts ВМ-1

```
 GNU nano 7.2                        /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).
source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet static
    address 127.0.0.1
    netmask 255.0.0.0
```

Рисунок 5 - указываем статический ip-адрес ВМ-1

6. Создаём папку `GitlabService` в домашнем каталоге и в ней создаём файл `docker-compose.yml`. Вносим в yml файл каркас основного контейнера.

Open ▾ ⊞ docker-compose.yml ~/GitlabService

```yaml
version: '3.6'
services:
  web:
    image: 'gitlab/gitlab-ce:latest'
    restart: always
    hostname: 'gitlab.localdomain'
    environment:
      GITLAB_OMNIBUS_CONFIG: |
        external_url 'http://gitlab.localdomain'
        gitlab_rails['gitlab_shell_ssh_port'] = 2224
    ports:
      - '80:80'
      - '443:443'
      - '2224:22'
    volumes:
      - '$GITLAB_HOME/config:/etc/gitlab'
```

Рисунок 6 - содержимое файла docker-compose.yml

7. Подготавливаем папку хранения постоянных данных службы Gitlab. Создаём от имени пользователя `root` директорию `/var/GitlabVolumes` и устанавливаем её права доступа в маску: rwxrwx---.

```
emil@lol:~/GitlabService$ sudo su
root@lol:/home/emil/GitlabService# mkdir -p /var/GitlabVolumes
root@lol:/home/emil/GitlabService# chmod 770 /var/GitlabVolumes
```

Рисунок 7 - создание директории /var/GitlabVolumes

8. Запускаем службу Gitlab перейдя в папку хранения описания проекта службы Gitlab и выполняем команду: `docker-compose up -d`.

```
root@lol:/home/emil/GitlabService# docker-compose up -d
Pulling web (gitlab/gitlab-ce:latest)...
latest: Pulling from gitlab/gitlab-ce
3f94e4e483ea: Pull complete
b62b47c46004: Pull complete
3159a18fca6b: Pull complete
eaf6b7858194: Pull complete
a97b3855c0e5: Pull complete
2603cbaa89cc: Pull complete
509d5c765ac8: Pull complete
572befffc62a: Pull complete
Digest: sha256:3b9a92461bf8a03d32e92dacea8740eeec97ed560fdcba39470d490c9c4f80c2
Status: Downloaded newer image for gitlab/gitlab-ce:latest
Creating gitlabservice_web_1 ... done
```

Рисунок 8 - выполнение команды docker-compose up -d

9. Просмотр списка запущенных контейнеров командой: `docker container ls`.

```
emil@lol:~$ sudo docker container ls
[sudo] password for emil:
CONTAINER ID   IMAGE                      COMMAND             CREATED         STATUS
                        PORTS
                                                  NAMES
3887d7b0c281   gitlab/gitlab-ce:latest    "/assets/wrapper"   2 minutes ago   Up 2 minut
es (health: starting)   0.0.0.0:80->80/tcp, :::80->80/tcp, 0.0.0.0:443->443/tcp, :::443
->443/tcp, 0.0.0.0:2224->22/tcp, :::2224->22/tcp   gitlabservice_web_1
```

Рисунок 9 - просмотр списка запущенных контейнеров docker

10. Проверяем доступность службы Gitlab, запустив веб-браузер на ВМ-2 и перейдя по адресу: https://gitlab.localdomain.

11. Останавливаем службу Gitlab на ВМ-1 командой: `docker-compose down`.

```
root@lol:/home/emil# cd GitlabService
root@lol:/home/emil/GitlabService# docker-compose down
Stopping gitlabservice_web_1 ... done
Removing gitlabservice_web_1 ... done
Removing network gitlabservice_default
```

Рисунок 10 - остановка службы Gitlab

Выводы

В ходе практической работы были получены следующие результаты:

1. Освоены основы работы с системой контейнеризации Docker. Сюда входит установка и установка Docker, а также запуск и остановка сервисов в контейнерах.

2. Получены навыки создания контейнеров в Docker. Для создания контейнеров и управления ими использовались базовые методы, включая команду запуска, остановки, извлечения и просмотра состояния контейнеров.

3. Осуществлены основные операции по управлению контейнерами.