Самостоятельная практическая работа №2. Облачные решения. Основы контейнеризации Docker.

Цель работы

Ознакомиться с системой контейнеризации Docker. Научиться создавать, устанавливать и проводить базовые операции управления контейнерами в системе контейнеризации Docker.

Необходимые условия

1. Виртуальная машина, далее ВМ, удовлетворяющая требованиям:
   * ОС Linux Debian 11 в базовой конфигурации («свежеустановленная»).
   * Установленный текстовый редактор (в примерах mcedit).
   * Свободное место на жёстком диске не менее 10 GB.
   * Оперативная память ВМ не менее 8 ГБ
   * Наличие доступа ВМ к сети Internet.
   * Статический IP-адрес ВМ во внутренней сети, доступный для окружения.

Общее описание работы

В качестве практической работы мы рассмотрим создание контейнеризуемого веб-приложения на базе веб-сервера nginx работающего на локальном сервере поддержки контейнерных приложений Docker.

Порядок выполнения работы

1. Выполнить установку ПО Docker на ВМ.

**Примечание:** Установка возможна в двух вариантах:

* 1. Последняя версия, выпущенная компанией Docker. Процесс установки описан на странице документации : <https://docs.docker.com/engine/install/debian/>.
  2. Версия, проверенная на совместимость с ОС Dedbian 11.

В практической работе мы будем использовать вариант, проверенный на совместимость с ОС Debian 11.

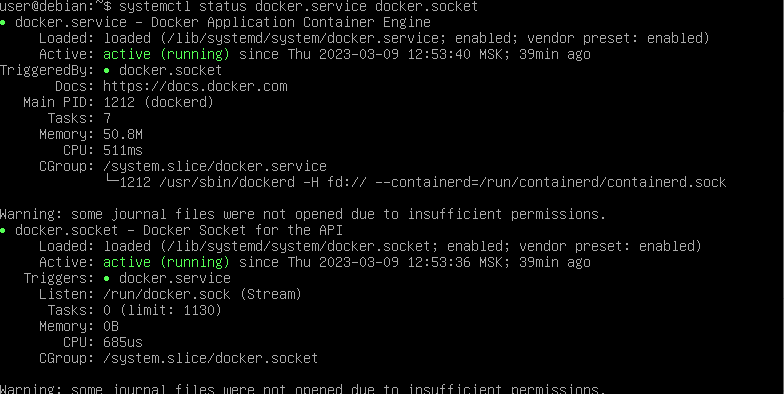
Для выполнения установки необходимо в терминале управления ОС выполнить следующие действия от имени пользователя **root** или от имени пользователя, имеющего право **sudo** (в текущем примере используется пользователь **root** ) :

* + Обновить базу пакетного репозитория apt : **apt-get update** ;
  + Выполнить установку базовых пакетов:
  + Убедиться в том, что демон Docker запущен:

1 apt-get install -y ca-certificates curl gnupg lsb-release docker.io docker-compose mcedit

1 systemctl status docker.service docker.socket

Ожидаемый вывод может выглядеть примерно следующим образом:



1. Подготовить проект описания контейнеризуемого приложения:
   * Создать папку **MyContainer** в домашнем каталоге текущего пользователя.
   * Создать файлы **index.html** и **Dockerfile** в каталоге **MyContainer** .
   * В текстовом редакторе отредактировать файл **Dockerfile** и внести в него каркас основного контейнера:

**Примечание:** Основная документация на формат и возможные варианты применения формата Dockerfile доступна по ссылке: <https://docs.docker.com/engine/reference/builder/>

В текстовом редакторе отредактировать файл **index.html** и внести в него содержимое главной страницы веб-сервера. Например,

1 FROM nginx:alpine

2 COPY index.html /usr/share/nginx/html/index.html

1 <!DOCTYPE html>

2 <html lang="ru">

3 <head>

4 <meta charset="UTF-8">

5 <meta name="viewport" content="width=device-width, initial-scale=1.0">

6 <title> Привет от Nginx в Docker-е</title>

7 <style>

8 h1{

9 font-weight:lighter;

10 font-family: Arial, Helvetica, sans-serif;

11 }

12 </style>

13 </head>

14 <body>

15

16 <h1>

17 Привет!

18 </h1>

19

20 </body>

21 </html>

1. Просмотреть доступные образы командой:

Вывод будет содержать полные имена контейнеров в поле REPOSYTORY и некоторые данные. Пример вывода приведён ниже:

1 docker images ls

REPOSITORY                       TAG       IMAGE ID       CREATED          SIZE

nginx-container                  latest    b790e75566eb   40 minutes ago   40.7MB

nginx-container                  v1        b790e75566eb   40 minutes ago   40.7MB

1. Выполнить сборку основного контейнера командой:

Ожидаемый вывод может выглядеть примерно следующим образом:

1 docker build -t nginx-container:v1 -t nginx-container:latest .

**Примечание:** Для выполнения команд docker пользователь должен иметь права группы docker. Альтерантивой является выполнение команд **docker** от имени пользователя root или используя sudo (пользователь должен быть в группе **sudo**).

user@debian:~/MyContainer$ docker build -t nginx-container:v1 -t nginx-container:latest .

Sending build context to Docker daemon 3.072kB

Step 1/2 : FROM nginx:alpine

---> 2bc7edbc3cf2

Step 2/2 : COPY index.html /usr/share/nginx/html/index.html

---> Using cache

---> b790e75566eb

Successfully built b790e75566eb

Successfully tagged nginx-container:v1

Successfully tagged nginx-container:latest

**Примечание 2:** Основная документация по команде docker build доступна по ссылке: <https://docs.docker.com/engine/reference/commandline/build/>

**Примечание3:** Рассмотрим пример полной схемы именования контейнеров, используемых в частном проекте, в закрытой от Internet среде. Типичное имя контейнера выглядит следующим образом:

**docker-registry.localdomain/project/part/application:version**

* + 1. Путь к узлу сервера реестра контейнеров **docker-registry.localdomain**
    2. Путь к репозиторию приложения **project/part**
    3. Имя контейнера  **application**

Версия контейнера в **version** . В случае обращения к контейнеру «последней актуальной» версии используется тэг latest как при сборке, так и при любом другом доступе.

1. Запустить контейнер с привязкой к портам виртуальной машины

**Примечание:** Формула запуска контейнера зависит от контейнеризованного приложения. Команда docker run обладает большим перечнем параметров, необходимых для этого действия. С полным перечнем параметров можно ознакомиться по ссылке: <https://docs.docker.com/engine/reference/run/>

1 docker run --name bmstu-practice-nginx -p 80:80 -d nginx-container

1. Просмотрите журнал приложения, работающего в контейнере, командой:

**Примечание:** Обратите внимание, что на обращение к контейнеру идёт по имени, присвоенному в параметре **--name**.

1 docker logs bmstu-practice-nginx

1. Войдите в командный терминал нашего контейнера и просмотрите конфиуграционный файл сервера nginx **/etc/nginx/nginx.conf** и завершите работу с контйенером следующей последовательностью команд:

**Примечание:** Обратите внимание, что в работающем контейнере минимизируется установленное программное обеспечение. При этом, получив доступ к консоли по умолчанию мы получаем терминал привилегированного пользователя и можем доустановить необходимо ПО, внести некоторые изменения в конфигурационные файлы. После перезапуска контейнера все внесённые изменения будут сброшены, если к контейнеру не применена операция **docker commit**.

docker exec -it bmstu-practice-nginx /bin/sh  
#cat /etc/nginx/nginx.conf  
#exit

1. Просмотрите перечень работающих контейнеров командой:

**Примечание:** Обратите внимание на вывод команды. Особенно на уникальные идентификаторы контейнеров. Данные идентификаторы могут использоваться для обращения к контейнерам как альтернатива именам.

1 docker ps

1. Остановите работу контейнера командой:

1 docker stop bmstu-practice-nginx

1. Убедитесь, что контейнер действительно остановлен при помощи команды:
2. Перезапустите контейнер при помощи команды:

1 docker ps docker restart bmstu-practice-nginx

1 docker restart bmstu-practice-nginx

Вопросы для самопроверки

1. В случае нашего контейнера, какие записи журнала вы видите?
2. Как найти контейнер, если при запуске не был указан параметр **-- name**?
3. Что будет, если после остановки попробовать повторно запустить командой из п. 5.
4. Что будет, если в строке запуска контейнера убрать параметр **-d**?
5. Запустите контейнер nginx таким образом, чтобы он использовал конфигурацию из файла в папке /var/nginx/nginx.conf.
6. Создайте несколько версий контейнера с разными тэгами и запустите их одновременно. Для этого можно использовать различные порты основного узла в опции **-p**.