



МИНОБРНАУКИ РОССИИ

**Федеральное государственное бюджетное образовательное учреждение
высшего образования**

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт комплексной безопасности и специального приборостроения

Кафедра КБ-14 «Интеллектуальные системы информационной безопасности»

Администрирование баз данных

Практическая работа № 2

ОТЧЁТ

Выполнил студент группы

БСБО-07-20

Любовский С.В.

Москва 2023

Выполнение задания 1.

1. Установим docker на локальной машине
2. Используя Docker Hub загрузим образ nginx командой **docker pull nginx:latest**
3. Запустим контейнер **nginx** в фоновом режиме командой **docker run -d nginx**
4. Выведем список работающих контейнеров командой **docker ps**

```
→ practice2 (main) ✓ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
4e5f13d64c84	nginx	"/docker-entrypoint..."	2 minutes ago	Up 2 minutes	80/tcp	gifted_euclid

5. Остановим контейнер используя команду **docker stop 4e5f**
6. Переименуем контейнер в webhost используя команду **docker rename 4e5f webhost**

```
→ practice2 (main) ✓ docker rename 4e5f webhost
→ practice2 (main) ✓ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
4e5f13d64c84	nginx	"/docker-entrypoint..."	25 minutes ago	Exited (0) About a minute ago		webhost

7. Запустим контейнер с именем webhost используя команду **docker start webhost**
8. Выведем логи контейнера webhost используя команду **docker logs webhost**

```
→ practice2 (main) ✓ docker logs webhost
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/02/26 12:21:21 [notice] 1#1: using the "epoll" event method
2023/02/26 12:21:21 [notice] 1#1: nginx/1.23.3
2023/02/26 12:21:21 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
2023/02/26 12:21:21 [notice] 1#1: OS: Linux 5.15.83.1-microsoft-standard-WSL2
2023/02/26 12:21:21 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/02/26 12:21:21 [notice] 1#1: start worker processes
2023/02/26 12:21:21 [notice] 1#1: start worker process 29
2023/02/26 12:21:21 [notice] 1#1: start worker process 30
2023/02/26 12:21:21 [notice] 1#1: start worker process 31
2023/02/26 12:21:21 [notice] 1#1: start worker process 32
2023/02/26 12:21:21 [notice] 1#1: start worker process 33
```

Выполнение задания 2.

1. Загрузим образ Ubuntu из Docker hub командой **docker pull ubuntu:latest**
2. Запустим контейнер используя команду **docker run --name ubuntu -d ubuntu /bin/bash -c "sleep 1000000"** и подключимся к нему используя команду **docker exec -it ubuntu /bin/bash**

3. Установим пакеты **vim, curl, http, ping** используя команду **apt update && apt upgrade -y && apt install vim curl http inetutils-ping -y**
4. Выйдем из контейнера и остановим его командой **docker stop ubuntu**
5. Запустим контейнер заново используя команду **docker start ubuntu** и убедимся, что все пакеты установлены используя команду **docker exec -it ubuntu /bin/bash -c "apt list --installed | grep 'vim\\|http\\|curl\\|ping'"**

Выполнение задания 3.

1. Используя утилиту **jq** создадим запрос на получения информации о сети полученной из команды **docker inspect -f json webhost ubuntu**. Итоговая команда будет выглядеть так **docker container inspect --format json ubuntu webhost | jq '[] | {ContainerName: .Name,**

```

MIREA_database_administration (main) ✓ docker container inspect --format json ubuntu webhost | jq '[] | {ContainerName: .Name, ContainerNetwork: .NetworkSettings.Networks}'
{
  "ContainerName": "/ubuntu",
  "ContainerNetwork": {
    "bridge": {
      "IPAMConfig": null,
      "Links": null,
      "Aliases": null,
      "NetworkID": "036afaa3066533d144605b10992ae4a87cb762026f3a2c4e29c3b77bb59c64609",
      "EndpointID": "8b0ed8965192a58a71fcfb7ee84cb260f05b7e887b11f98a1a7c444795dbbd51",
      "Gateway": "172.17.0.1",
      "IPAddress": "172.17.0.2",
      "IPPrefixLen": 16,
      "IPv6Gateway": "",
      "GlobalIPv6Address": "",
      "GlobalIPv6PrefixLen": 0,
      "MacAddress": "02:42:ac:11:00:02",
      "DriverOpts": null
    }
  }
}
{
  "ContainerName": "/webhost",
  "ContainerNetwork": {
    "bridge": {
      "IPAMConfig": null,
      "Links": null,
      "Aliases": null,
      "NetworkID": "036afaa3066533d144605b10992ae4a87cb762026f3a2c4e29c3b77bb59c64609",
      "EndpointID": "03edd216e1eca9b29ccf58898a45d71c309cdd0a8a6359313615a4503b75abb7",
      "Gateway": "172.17.0.1",
      "IPAddress": "172.17.0.3",
      "IPPrefixLen": 16,
      "IPv6Gateway": "",
      "GlobalIPv6Address": "",
      "GlobalIPv6PrefixLen": 0,
      "MacAddress": "02:42:ac:11:00:03",
      "DriverOpts": null
    }
  }
}

```

ContainerNetwork: .NetworkSettings.Networks}'

2. Используя команду **docker network ls** выведем все доступные сети.

```

MIREA_database_administration (main) ✓ docker network ls

```

NETWORK ID	NAME	DRIVER	SCOPE
036afaa30665	bridge	bridge	local
6d5230fdd8ca	host	host	local
e7c7df05e089	mirea_database_administration_default	bridge	local
5820976db770	mirea_database_security_policies_default	bridge	local
a4039e7a7b89	none	null	local

3. Используя команду **docker network create -d bridge my-bridge-network** создадим свою сеть типа **bridge**
4. Подключим созданные контейнеры к нашей сети используя команду **docker network connect bridge my-bridge-network [ИМЯ КОНТЕЙНЕРА]**

5. При помощи утилиты ping проверим, что контейнеры видят друг друга

```
→ MIREA_database_administration (main) ✓ docker exec -it ubuntu /bin/bash
root@54dac3f82900:/# ping webhost
PING webhost (172.22.0.2): 56 data bytes
64 bytes from 172.22.0.2: icmp_seq=0 ttl=64 time=0.546 ms
64 bytes from 172.22.0.2: icmp_seq=1 ttl=64 time=0.131 ms
64 bytes from 172.22.0.2: icmp_seq=2 ttl=64 time=0.142 ms
64 bytes from 172.22.0.2: icmp_seq=3 ttl=64 time=0.136 ms
64 bytes from 172.22.0.2: icmp_seq=4 ttl=64 time=0.145 ms
64 bytes from 172.22.0.2: icmp_seq=5 ttl=64 time=0.128 ms
```

Выполнение задания 4.

1. Создадим Dockerfile по условиям

```
FROM ubuntu:22.04

RUN apt update -y \
    && apt upgrade -y \
    && apt install nano apache2 -y

RUN echo 'Hello, World!' > index.html

WORKDIR /usr/local/apache2/htdocs/

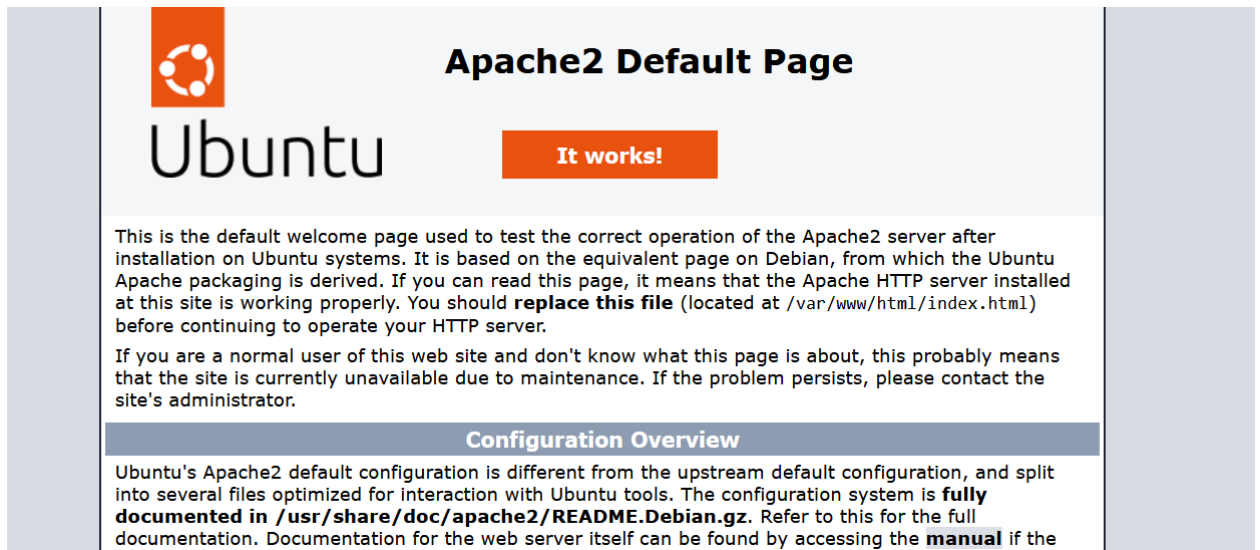
RUN cp /index.html .

EXPOSE 80

CMD apachectl -D FOREGROUND
```

2. Соберем образ командой **docker build -t ubuntu-apache .**
3. Запустим контейнер на основе собранного образа командой **docker run --name ubuntu-apache -d -p 80:80 ubuntu-apache**

4. Проверим работу



5. Остановим и удалим контейнер командой `docker rm -f ubuntu-apache`

6. Добавим установку пакета curl в Dockerfile

```
FROM ubuntu:22.04

RUN apt update -y \
    && apt upgrade -y \
    && apt install nano apache2 curl -y

RUN echo 'Hello, World!' > index.html

WORKDIR /usr/local/apache2/htdocs/

RUN cp /index.html .

CMD /usr/sbin/apache2ctl -D FOREGROUND
```

7. Пересоберем образ и запустим контейнер заново

8. Проверим что curl установлен внутри контейнера

```
→ practice2 (main) X docker exec -it ubuntu-apache curl localhost:80
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <!--
    Modified from the Debian original for Ubuntu
    Last updated: 2022-03-22
    See: https://launchpad.net/bugs/1966004
  -->
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Apache2 Ubuntu Default Page: It works</title>
    <style type="text/css" media="screen">
      * {
        margin: 0px 0px 0px 0px;
        padding: 0px 0px 0px 0px;
      }
    </style>
  </head>
  <body>
    <div style="text-align: center;">
      <img alt="Ubuntu logo" data-bbox="232 98 279 142" style="height: 40px; width: 40px;"/>
      <h1 style="margin: 0 0 0 0;">Apache2 Default Page
      <div style="background-color: #a52a2a; color: white; padding: 5px 10px; display: inline-block;">It works!
    </div>
    <p>This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should replace this file (located at /var/www/html/index.html) before continuing to operate your HTTP server.
    <p>If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.
    <h3>Configuration Overview</h3>
    <p>Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is fully documented in /usr/share/doc/apache2/README.Debian.gz. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the manual if the
  </body>
</html>
```