

TP1-Virtualização de Redes-MIEI

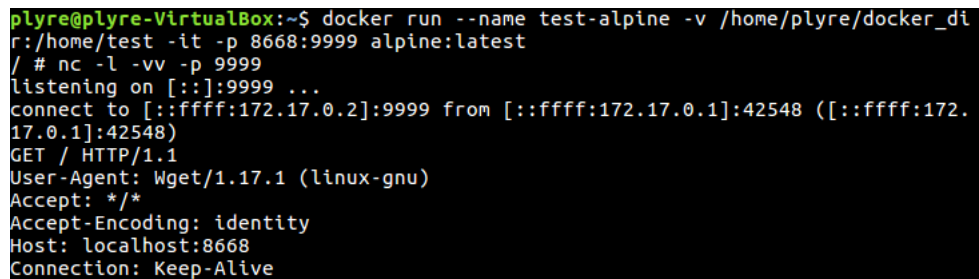
João Bernardo Freitas a74814

João Mendes a71862

27 Fevereiro 2020

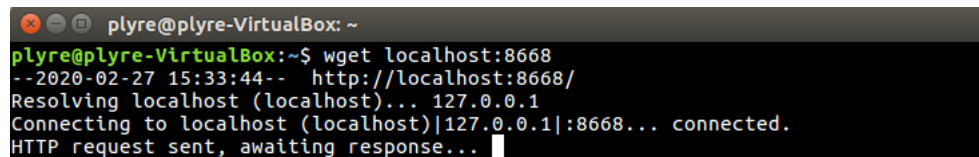
1 How to create a container from a Linux alpine that has the container 9999 port mapped in the host 8668= This should also have a bind mount, mounting the container /home/internaldir in the hosts /home/user/dockerdir/.

Em primeiro lugar é necessário fazer o download da imagem Linux alpine usando o comando **docker pull alpine:latest**. De seguida foi utilizado o comando **docker run --name test-alpine -v /home/plyre/dockerdir:/home/test -it -p 8668:9999 alpine:latest** de forma a criar o container de nome test-alpine com o mount pedido e o port 9999 mapeado no host 8668.



```
plyre@plyre-VirtualBox:~$ docker run --name test-alpine -v /home/plyre/docker_dir:/home/test -it -p 8668:9999 alpine:latest
/# nc -l -vv -p 9999
listening on [::]:9999 ...
connect to [::ffff:172.17.0.2]:9999 from [::ffff:172.17.0.1]:42548 ([::ffff:172.17.0.1]:42548)
GET / HTTP/1.1
User-Agent: Wget/1.17.1 (linux-gnu)
Accept: */*
Accept-Encoding: identity
Host: localhost:8668
Connection: Keep-Alive
```

Figure 1: Criação do container test-alpine



```
plyre@plyre-VirtualBox: ~
plyre@plyre-VirtualBox:~$ wget localhost:8668
--2020-02-27 15:33:44-- http://localhost:8668/
Resolving localhost (localhost)... 127.0.0.1
Connecting to localhost (localhost)|127.0.0.1|:8668... connected.
HTTP request sent, awaiting response...
```

Figure 2: Verificação de conexão

2 How to create the volume “my-volume-1”?

Para criar o volume “my-volume-1” utiliza-se o comando **docker volume create my-volume-1**

2.1 What is the volume mountpoint?

Através do comando **docker volume inspect my-volume-1** conseguimos verificar que o mountpoint deste volume é `/var/lib/docker/volumes/my-volume-1/_data`.

2.2 Which driver is being used?

Através do comando utilizado na alínea anterior é também possível verificar o driver, neste volume o driver utilizado é **local**.

3 Create two basic containers (They should not be attached to any manually created network).

Os dois containers foram criados através do seguinte comando. **docker run --name container1 --mount source=my-volume-1,target= /home/test -it alpine:latest** e **docker run --name container2 --mount source=my-volume-1,target= /home/test -it alpine:latest**

3.1 Is it possible to inspect the bridge network and find their IP addresses?

É possível inspecionar a bridge network através do comando **docker network inspect bridge**, sendo que é possível também descobrir os endereços IP dos dois containers.

3.2 Is it possible for the containers to communicate among themselves using their names?

Não é possível visto que os containers não fazem parte da mesma network.

3.3 And with their IPs?

É possível que os dois containers comuniquem um com o outro através dos seus endereços IP.

4 What is the command to create the network “my-network-1”?

Para criar a network “my-network-1” utilizamos o seguinte comando: **docker create network my-network-1**.

4.1 How to attach two containers to that network?

Para ligarmos dois containers à network utilizamos os comandos **docker run --dit --name container1 --network my-network-1 alpine:latest** e **docker run --dit --name container2 --network my-network-1 alpine:latest**

4.2 Which relevant parameter is possible to found about that network?

Através do comando **docker network inspect my-network-1** podemos obter diversos parametros, nomeadamente os MAC/IP's/nomes dos containers nessa network, o nome da network, o endereço da gateway, a gama de endereços e o seu driver.

4.3 Can the containers ping one another using their name?

Após correr o comando **docker attach container1** ou **docker attach container2** que permitem aceder ao terminal dos containers respetivos é possível fazer ping de um container para o outro.

5 Which were the results obtained from the commands **docker network ls**, **docker network inspect**, **docker volume ls** and **docker network inspect** in section 2 of this document? What conclusions is possible to take from the result of these commands?

```
root@kali:~# docker volume inspect my-volume-1
[
  {
    "CreatedAt": "2020-02-27T06:22:18-05:00",
    "Driver": "local",
    "Labels": {},
    "Mountpoint": "/var/lib/docker/volumes/my-volume-1/_data",
    "Name": "my-volume-1",
    "Options": {},
    "Scope": "local"
  }
]
```

Figure 3: Resultado do comando **docker volume inspect my-volume-1**

```
root@kali:~# docker volume ls
DRIVER          VOLUME NAME
local           0a054f122bb99ba47da8f8eebad38bbeba55b5499241a5622aff402fd7513c2c
local           my-volume-1
local           root_httpd-vol
```

Figure 4: Resultado do comando **docker volume ls**

```

root@kali:~# docker network inspect my-network-1
[
  {
    "Name": "my-network-1",
    "Id": "e88d515d49aea542f2042be478d0282f529805f7df868af9315a5d1f8adf9c6e",
    "Created": "2020-02-27T08:38:44.803743595-05:00",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": {},
      "Config": [
        {
          "Subnet": "172.20.0.0/16",
          "Gateway": "172.20.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": false,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "d2b96bde5bf09f21c99272972beed351db413df875d800a3755873bc4bf83418": {
        "Name": "container2",
        "EndpointID": "303cbcfc2bf0e8cfaa2468cf8e6af3788ce6028bf2f6aabb1295f1a7bcbd44aed",
        "MacAddress": "02:42:ac:14:00:02",
        "IPv4Address": "172.20.0.2/16",
        "IPv6Address": ""
      },
      "e33d32ed63e5726389ddadcb011e9c146d5be150e87ff9db07020ca6e86f6314": {
        "Name": "container1",
        "EndpointID": "9b8f7d3b80f16113982b251c78acc85e7021f9e21280b92d6dbc08402e227436",
        "MacAddress": "02:42:ac:14:00:03",
        "IPv4Address": "172.20.0.3/16",
        "IPv6Address": ""
      }
    },
    "Options": {},
    "Labels": {}
  }
]

```

Figure 5: Resultado do comando `docker network inspect my-network-1`

```

root@kali:~# docker network ls
NETWORK ID          NAME                DRIVER             SCOPE
92fdff5c2002        bridge              bridge              local
ac6681d8cc5c        host                host                local
e88d515d49ae        my-network-1        bridge              local
bbacc8c6360d        none                null                local

```

Figure 6: Resultado do comando `docker network ls`

As conclusões retiradas por estes comandos já foram mencionadas na pergunta 4.2.

- 6 Create a docker-compose that starts at least two services:

```
version: '3'
services:
  service1:
    image: httpd:latest
    ports:
      - "8080:80"
    volumes:
      - httpd-vol:/usr/local/apache2/
    networks:
      - container-net
  service2:
    image: httpd:latest
    ports:
      - "8888:9999"
    volumes:
      - httpd-vol:/usr/local/apache2/
    networks:
      - container-net
volumes:
  httpd-vol:
networks:
  container-net:
```

Figure 7: Ficheiro docker-compose.yml

```

plyre@plyre-VirtualBox:~/VR$ docker-compose up -d
Creating network "vr_container-net" with the default driver
Creating volume "vr_httpd-vol" with default driver
Creating vr_service2_1 ... done
Creating vr_service1_1 ... done
plyre@plyre-VirtualBox:~/VR$ docker volume ls
DRIVER          VOLUME NAME
local           5f1bda2ce5f8914f436d209e1db2f9b967bc5b5ba93ebdb59204afed3af06025
local           new-test-vol
local           plyre_httpd-vol
local           vr_httpd-vol
plyre@plyre-VirtualBox:~/VR$ docker volume inspect vr_httpd-vol
[
  {
    "CreatedAt": "2020-02-29T19:47:12Z",
    "Driver": "local",
    "Labels": {
      "com.docker.compose.project": "vr",
      "com.docker.compose.version": "1.25.4",
      "com.docker.compose.volume": "httpd-vol"
    },
    "Mountpoint": "/var/lib/docker/volumes/vr_httpd-vol/_data",
    "Name": "vr_httpd-vol",
    "Options": null,
    "Scope": "local"
  }
]
plyre@plyre-VirtualBox:~/VR$

```

Figure 8: Resultado do comando docker network ls

```

plyre@plyre-VirtualBox:~/VR$ docker network inspect vr_container-net
[
  {
    "Name": "vr_container-net",
    "Id": "92604d4d3b06ff58429da11feff36c8d4f1eadc7ba07122dd421c694cea8dad",
    "Created": "2020-02-29T19:47:10.356625295Z",
    "Scope": "local",
    "Driver": "bridge",
    "EnableIPv6": false,
    "IPAM": {
      "Driver": "default",
      "Options": null,
      "Config": [
        {
          "Subnet": "172.19.0.0/16",
          "Gateway": "172.19.0.1"
        }
      ]
    },
    "Internal": false,
    "Attachable": true,
    "Ingress": false,
    "ConfigFrom": {
      "Network": ""
    },
    "ConfigOnly": false,
    "Containers": {
      "13359e6b6e5ee1b9222dc6f139d32f6d12041bc276919e1c6fb3ec148b0145db": {
        "Name": "vr_service2_1",
        "EndpointID": "c6066c096ec7ce60ed6b5d657622934dc687fb12156d16d65a84ca29e7cf46be",
        "MacAddress": "02:42:ac:13:00:03",
        "IPv4Address": "172.19.0.3/16",
        "IPv6Address": ""
      },
      "8a9a7e672b758dc7c5c71eea5986a7cceb126da770daa321b1619de8b7c2eb30": {
        "Name": "vr_service1_1",
        "EndpointID": "b22833c28a03951066bd800a02d482b4a83fa64b177d6e8656954b52b421e7c7",
        "MacAddress": "02:42:ac:13:00:02",
        "IPv4Address": "172.19.0.2/16",
        "IPv6Address": ""
      }
    },
    "Options": {},
    "Labels": {
      "com.docker.compose.network": "container-net",
      "com.docker.compose.project": "vr",
      "com.docker.compose.version": "1.25.4"
    }
  }
]

```

Figure 9: Resultado do comando `docker network inspect` a `vr-container-net`

6.1 Using the inspect and ls commands, what conclusions can you take about their results?

Através destes comandos podemos confirmar que ambos os serviços estão a funcionar corretamente, sendo que estão na mesma rede.

7 Create a Docker file that:

7.1 Have some tool installed. Any that is chosen by the student. Extra points for any application that is listening from outside requests (web server, for example);

7.2 Has a volume, for later persistence;

Foi criado um ficheiro Dockerfile com todas as especificações pedidas:

```
FROM ubuntu:latest
RUN apt-get update
RUN apt-get install -y apache2
RUN apt-get install -y apache2-utils
RUN apt-get clean
EXPOSE 80
CMD ["apache2ctl", "-D" , "FOREGROUND"]
VOLUME /home/output
```

Figure 10: Ficheiro dockerfile para correr servidor web apache2

Foram usados os comandos **sudo docker build -t="mywebserver"** . e **sudo docker run -d -p 80:80 mywebserver** para criar a imagem com nome **mywebserver**, e de seguida criar um container com essa imagem.

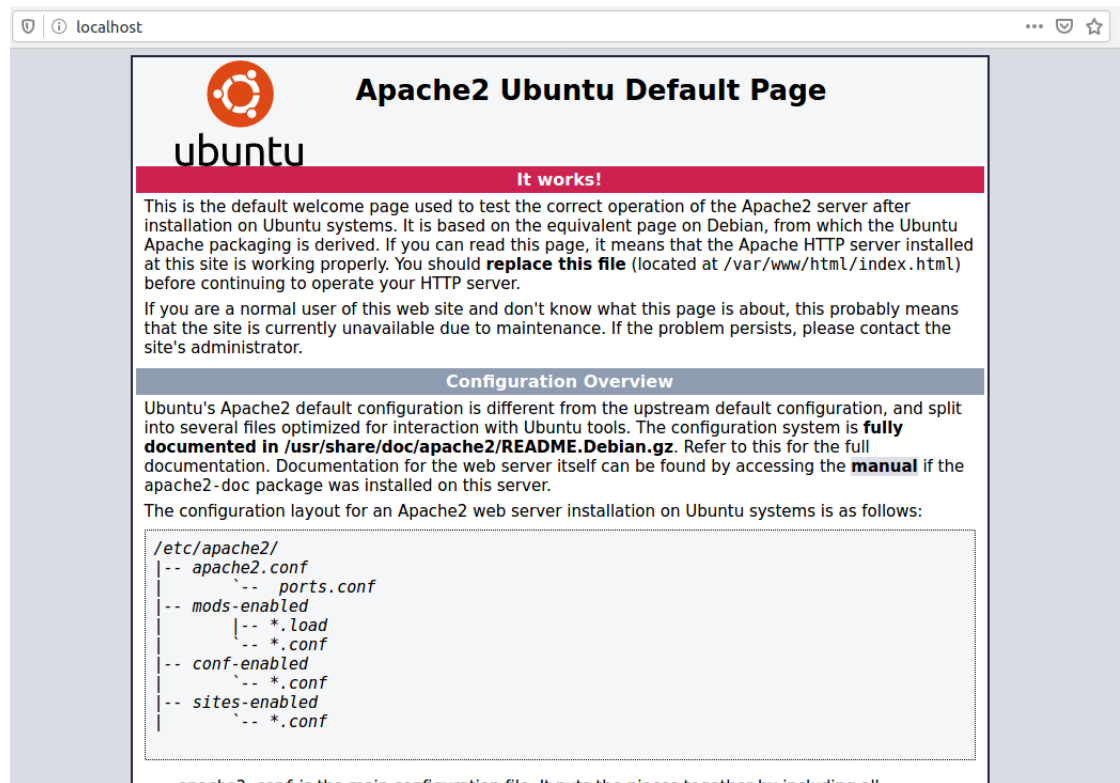


Figure 11: Servidor apache2 a correr no localhost

8 Create an automatic build. The docker file built in the previous exercise may be used.

O repositório github encontra-se neste link <https://github.com/JMendes95/VirRedes> e dockerhub <https://hub.docker.com/repository/docker/plyre/vr/general>