链滴

10、使用网络

作者：sunjvhui
原文链接：https://ld246.com/article/1562740645005
来源网站：链滴

Docker 允许通过外部访问容器或容器互联的方式来提供网络服务

外部访问容器

容器中可以运行一些网络应用，要让外部也可以访问这些应用，可以通过 -P 或 -p 参数来指定端口映
。

当使用 -P 标记时，Docker 会随机映射一个 49000~49900 的端口到内部容
开放的网络端口。

```
$ docker run -d -P training/webapp python app.py
$ docker container ls -l
CONTAINER ID  IMAGE                COMMAND       CREATED        STATUS        PORTS
   NAMES
bc533791f3f5  training/webapp:latest  python app.py 5 seconds ago  Up 2 seconds  0.0.0.0:49
55->5000/tcp  nostalgic_morse
```

使用 docker container ls 可以看到，本地主机的 49155 被映射到了容器的 5000 端口。此时访问本
的 49155 端口即可访问容器内 web 应用提供的界面。

同样的，可以通过 ==docker logs== 命令来查看应用的信息。

```
$docker logs -f bc533791f3f5
* Running on http://0.0.0.0:5000/
10.0.2.2 - - [23/May/2014 20:16:31] "GET / HTTP/1.1" 200 -
10.0.2.2 - - [23/May/2014 20:16:31] "GET /favicon.ico HTTP/1.1" 404 -
```

-p 则可以指定要映射的端口，并且，在一个指定端口上只可以绑定一个容器。
持的格式有

```
ip:hostPort:containerPort | ip::containerPort | hostPort:containerPort。
```

映射所有接口地址

使用 ==hostPort:containerPort== 格式本地的 5000 端口映射到容器的 5000 端口，可以执行

```
$ docker run -d -p 5000:5000 training/webapp python app.py
```

此时默认会绑定本地所有接口上的所有地址。

映射到指定地址的指定端口

可以使用 ==ip:hostPort:containerPort== 格式指定映射使用一个特定地址，比如 localhost 地址 1
7.0.0.1

```
$ docker run -d -p 127.0.0.1:5000:5000 training/webapp python app.py
```

映射到指定地址的任意端口

使用 ==ip::containerPort== 绑定 ==localhost== 的任意端口到容器的 5000 端口，本地主机会自

分配一个端口。

```
$ docker run -d -p 127.0.0.1::5000 training/webapp python app.py
```

还可以使用 ==udp== 标记来指定 ==udp== 端口

```
$ docker run -d -p 127.0.0.1:5000:5000/udp training/webapp python app.py
```

查看映射端口配置

使用 ==docker port== 来查看当前映射的端口配置，也可以查看到绑定的地址

```
$ docker port nostalgic_morse 5000
127.0.0.1:49155.
```

注意：

- 容器有自己的内部网络和 ip 地址（使用 docker inspect 可以获取所有的变量，Docker 还可以有个可变的网络配置。）
- -p 标记可以多次使用来绑定多个端口

```
$ docker run -d \
    -p 5000:5000 \
    -p 3000:80 \
    training/webapp \
    python app.py
```

容器互联

将容器加入自定义的 Docker 网络来连接多个容器

新建网络

```
$ docker network create -d bridge my-net
```

-d 参数指定 Docker 网络类型，有 bridge overlay。其中 overlay 网络类型用于 Swarm mode，在小节中你可以忽略它。

连接容器

运行一个容器并连接到新建的 my-net 网络

```
$ docker run -it --rm --name busybox1 --network my-net busybox sh
```

打开新的终端，再运行一个容器并加入到 my-net 网络

```
$ docker run -it --rm --name busybox2 --network my-net busybox sh
```

再打开一个新的终端查看容器信息

```
$ docker container ls
```

```
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PO
TS                  NAMES
b47060aca56b        busybox             "sh"                11 minutes ago      Up 11 minutes
                    busybox2
8720575823ec        busybox             "sh"                16 minutes ago      Up 16 minutes
                    busybox1
```

下面通过 ping 来证明 busybox1 容器和 busybox2 容器建立了互联关系。

在 busybox1 容器输入以下命令

```
/ # ping busybox2
PING busybox2 (172.19.0.3): 56 data bytes
64 bytes from 172.19.0.3: seq=0 ttl=64 time=0.072 ms
64 bytes from 172.19.0.3: seq=1 ttl=64 time=0.118 ms
```

用 ping 来测试连接 busybox2 容器，它会解析成 172.19.0.3。

同理在 busybox2 容器执行 ping busybox1，也会成功连接到。

```
/ # ping busybox1
PING busybox1 (172.19.0.2): 56 data bytes
64 bytes from 172.19.0.2: seq=0 ttl=64 time=0.064 ms
64 bytes from 172.19.0.2: seq=1 ttl=64 time=0.143 ms
```

这样，busybox1 容器和 busybox2 容器建立了互联关系。

配置DNS

定义配置容器的主机名和 DNS ,Docker 利用虚拟文件来挂载容器的 3 个相关配置文件。

在容器中使用 mount 命令可以看到挂载信息：

```
$ mount
/dev/disk/by-uuid/1fec...ebdf on /etc/hostname type ext4 ...
/dev/disk/by-uuid/1fec...ebdf on /etc/hosts type ext4 ...
tmpfs on /etc/resolv.conf type tmpfs ...
```

这种机制可以让宿主主机 DNS 信息发生更新后，所有 Docker 容器的 DNS 配置通过 /etc/resolv.con
文件立刻得到更新。

配置全部容器的 DNS ， 也可以在 /etc/docker/daemon.json 文件中增加以下内容来设置。

```
{
  "dns" : [
    "114.114.114.114",
    "8.8.8.8"
  ]
}
```

这样每次启动的容器 DNS 自动配置为 114.114.114.114 和 8.8.8.8。使用以下命令来证明其已经生
。

```
$ docker run -it --rm ubuntu:17.10  cat etc/resolv.conf
nameserver 114.114.114.114
```

nameserver 8.8.8.8

如果用户想要手动指定容器的配置，可以在使用 docker run 命令启动容器时加入如下参数：

-h HOSTNAME 或者 --hostname=HOSTNAME 设定容器的主机名，它会被写到容器内的 /etc/hostname 和 /etc/hosts。但它在容器外部看不到，既不会在 docker container ls 中显示，也不会在其的容器的 /etc/hosts 看到。

--dns=IP_ADDRESS 添加 DNS 服务器到容器的 /etc/resolv.conf 中，让容器用这个服务器来解析所不在 /etc/hosts 中的主机名。

--dns-search=DOMAIN 设定容器的搜索域，当设定搜索域为 .example.com 时，在搜索一个名为 host 的主机时，DNS 不仅搜索 host，还会搜索 host.example.com。

注意：

如果在容器启动时没有指定最后两个参数，Docker 会默认用主机上的 /etc/resolv.conf 来配置容器。