一、网络端口映射

1、案例

runoob@runoob:~$ docker run -d -P training/webapp python app.py

fce072cc88cee71b1cdceb57c2821d054a4a59f67da6b416fceb5593f059fc6d

runoob@runoob:~$ docker ps

CONTAINER ID IMAGE COMMAND ... PORTS NAMES

fce072cc88ce training/webapp "python app.py" ... 0.0.0.0:32768->5000/tcp grave\_hopper

2、参数绑定

-P：容器内部端口随机映射到主机高端口

-p：容器内部端口绑定到指定主机端口

runoob@runoob:~$ docker run -d -p 5000:5000 training/webapp python app.py

33e4523d30aaf0258915c368e66e03b49535de0ef20317d3f639d40222ba6bc0

runoob@runoob:~$ docker ps

CONTAINER ID IMAGE COMMAND ... PORTS NAMES

33e4523d30aa training/webapp "python app.py" ... 0.0.0.0:5000->5000/tcp berserk\_bartik

fce072cc88ce training/webapp "python app.py" ... 0.0.0.0:32768->5000/tcp grave\_hopper

绑定指定网络地址，默认绑定tcp端口

runoob@runoob:~$ docker run -d -p 127.0.0.1:5001:5000 training/webapp python app.py

95c6ceef88ca3e71eaf303c2833fd6701d8d1b2572b5613b5a932dfdfe8a857c

runoob@runoob:~$ docker ps

CONTAINER ID IMAGE COMMAND ... PORTS NAMES

95c6ceef88ca training/webapp "python app.py" ... 5000/tcp, 127.0.0.1:5001->5000/tcp adoring\_stonebraker

33e4523d30aa training/webapp "python app.py" ... 0.0.0.0:5000->5000/tcp berserk\_bartik

fce072cc88ce training/webapp "python app.py" ... 0.0.0.0:32768->5000/tcp grave\_hopper

绑定到udp端口，在端口后添加 /udp

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

6779686f06f6204579c1d655dd8b2b31e8e809b245a97b2d3a8e35abe9dcd22a

runoob@runoob:~$ docker ps

CONTAINER ID IMAGE COMMAND ... PORTS NAMES

6779686f06f6 training/webapp "python app.py" ... 5000/tcp, 127.0.0.1:5000->5000/udp drunk\_visvesvaraya

95c6ceef88ca training/webapp "python app.py" ... 5000/tcp, 127.0.0.1:5001->5000/tcp adoring\_stonebraker

33e4523d30aa training/webapp "python app.py" ... 0.0.0.0:5000->5000/tcp berserk\_bartik

fce072cc88ce training/webapp "python app.py" ... 0.0.0.0:32768->5000/tcp grave\_hopper

docker port：查看端口绑定情况

runoob@runoob:~$ docker port adoring\_stonebraker 5000

127.0.0.1:5001

二、容器互联：多个容器互联，共享资源，连接创建父子关系

1、容器命名：--name 标识命名容器，docker ps查看命名容器

runoob@runoob:~$ docker run -d -P --name runoob training/webapp python app.py

43780a6eabaaf14e590b6e849235c75f3012995403f97749775e38436db9a441

runoob@runoob:~$ docker ps -l

CONTAINER ID IMAGE COMMAND ... PORTS NAMES

43780a6eabaa training/webapp "python app.py" ... 0.0.0.0:32769->5000/tcp runoob

2、新建网络

a）创建docker网络：-d 指定网络类型，bridge、overlay（用于Swarm mode）

$ docker network create -d bridge test-net

b）查看网络：docker network ls

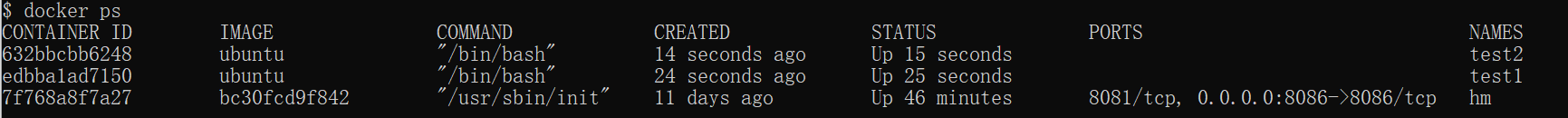
3、连接容器

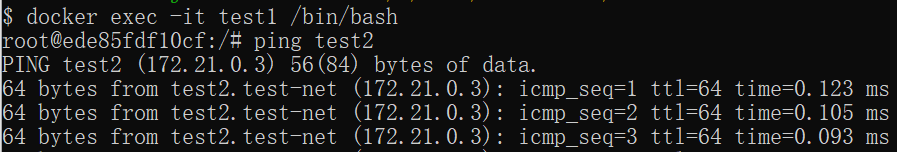
a）运行容器连接到新建的test-net网络

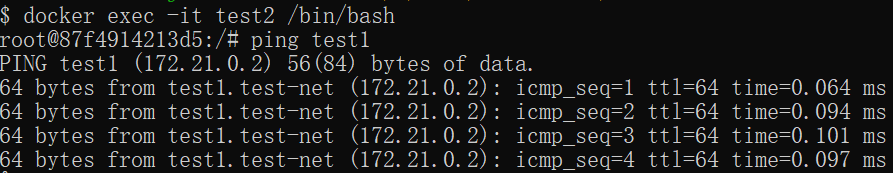
$ docker run -itd --name test1 --network test-net ubuntu /bin/bash

新终端运行

$ docker run -itd --name test2 --network test-net ubuntu /bin/bash







4、配置dns

a）配置文件：宿主机/etc/docker/daemon.json文件中加入如下内容，重新启动后会自动配置114.114.114.114、8.8.8.8

{

"dns" : [

"114.114.114.114",

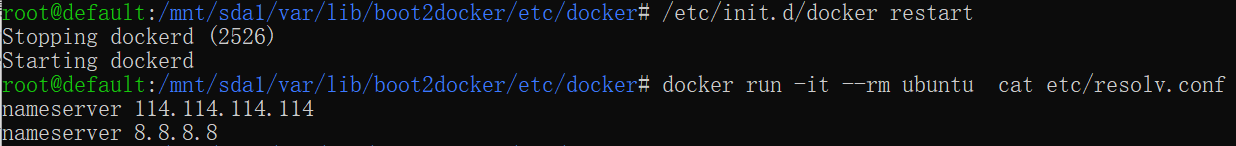
"8.8.8.8"

]

}

b）查看配置效果

$ docker run -it --rm ubuntu cat etc/resolv.conf



c）手动配置

$ docker run -it --rm host\_ubuntu --dns=114.114.114.114.114 --dns-search=test.com ubuntu

参数说明：

**-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。

**--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有不在 /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

