2.3.5 网络管理

讲师：汪洋

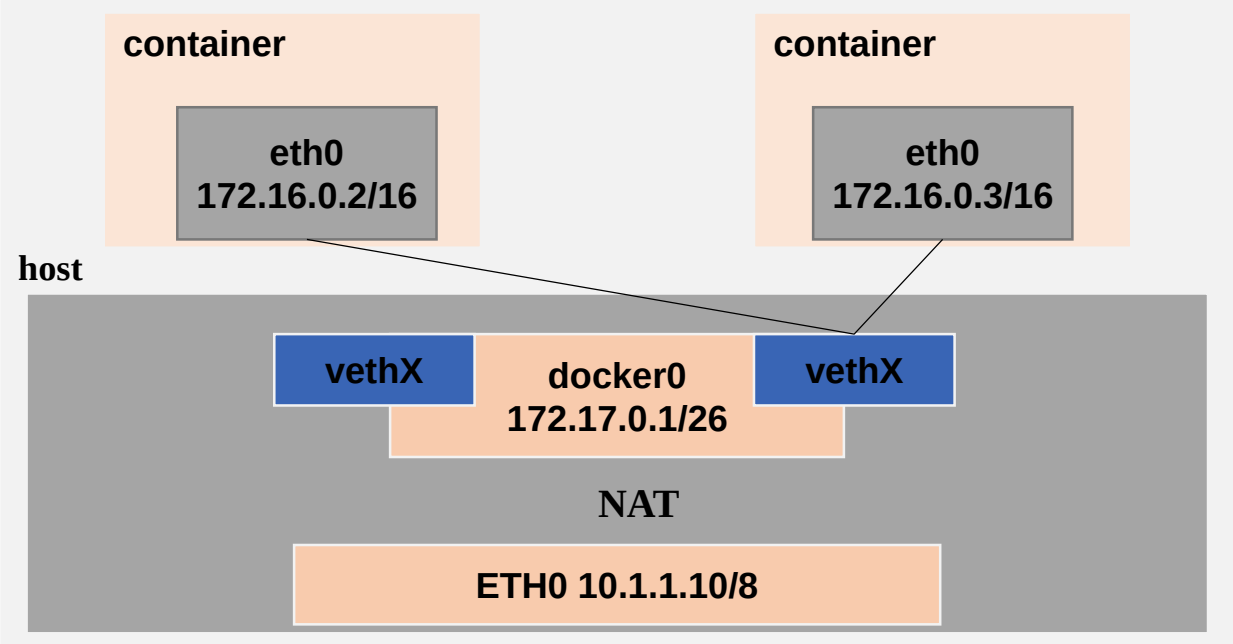
目录

让天下没有难学的技术

1 Docker 网络通讯

让天下没有难学的技术

在通常情况下，Docker 使用网桥（ Bridge ）与 NAT 的通信模式



让天下没有难学的技术

(1) 容器访问外部网络

```
iptables -t nat -A POSTROUTING -s 172.17.0.0/16 -o docker0 -j MASQUERADE
```

(2) 外部网络访问容器

```
docker run -d -p 80:80 apache
```

```
iptables -t nat -A  PREROUTING -m addrtype --dst-type LOCAL -j DOCKER
iptables -t nat -A DOCKER ! -i docker0 -p tcp -m tcp --dport 80 -j  DNAT --to-destination 172.17.0.2:80
```

让天下没有难学的技术

2 Docker 网络模式修改

让天下没有难学的技术

-b, --bridge=”” 指定 Docker 使用的网桥设备，默认情况下 Docker 会自动创建和使用 docker0 网桥设备，通过此参数可以使用已经存在的设备

--bip 指定 Docker0 的 IP 和掩码，使用标准的 CIDR 形式，如 10.10.10.10/24

--dns 配置容器的 DNS，在启动 Docker 进程是添加，所有容器全部生效

让天下没有难学的技术

--dns 用于指定启动的容器的 DNS

--net 用于指定容器的网络通讯方式，有以下四个值

- bridge：Docker 默认方式，网桥模式
- none：容器没有网络栈
- container：使用其它容器的网络栈，Docker容器会加入其它容器的 network namespace
- host：表示容器使用 Host 的网络，没有自己独立的网络栈。容器可以完全访问 Host 的网络，不安全

让天下没有难学的技术

-p/P 选项的使用格式

> -p :<ContainerPort> 将制定的容器端口映射至主机所有地址的一个动态端口

> -p <HostPort>:<ContainerPort>：映射至指定的主机端口

> -p <IP>::<ContainerPort>：映射至指定的主机的 IP 的动态端口

> -p <IP>:<HostPort>:<ContainerPort>：映射至指定的主机 IP 的主机端口

> -P（大）：暴露所需要的所有端口

* docker port ContainerName 可以查看容器当前的映射关系

让天下没有难学的技术

修改 /etc/docker/daemon.json 文件

```
{
    "bip": "192.168.1.5/24",
    "fixed-cidr": "10.20.0.0/16",
    "fixed-cidr-v6": "2001:db8::/64",
    "mtu": "1500",
    "default-gateway": "10.20.1.1",
    "default-gateway-v6": "2001:db8:abcd::89",
    "dns": ["10.20.1.2","10.20.1.3"]
}
```

让天下没有难学的技术

3 常见隔离方式

让天下没有难学的技术

docker network ls　　查看当前可用的网络类型

docker network create -d 类型 网络空间名称

类型分为：

overlay network

bridge network

让天下没有难学的技术

```
docker network create -d bridge --subnet "172.26.0.0/16" --gateway "172.26.0.1" my-bridge-network

docker run -d --network=my-bridge-network --name test1  hub.c.163.com/public/centos:6.7-tools

docker run -d --name test2  hub.c.163.com/public/centos:6.7-tools
```

让天下没有难学的技术

```
[root@localhost network-scripts]# vi ifcfg-eth0
                                 DEVICE=eth0
                                 HWADDR=00:0C:29:06:A2:35
                                 TYPE=Ethernet
                                 UUID=34b706cc-aa46-4be3-91fc-d1f48c301f23
                                 ONBOOT=yes
                                 BRIDGE=br0
                                 NM_CONTROLLED=yes
                                 BOOTPROTO=static

[root@localhost network-scripts]# vi ifcfg-br0
                                 //改成这样
                                 DEVICE=br0
                                 TYPE=Bridge
                                 ONBOOT=yes
                                 BOOTPROTO=static
                                 IPADDR=192.168.216.131
                                 NETMASK=255.255.255.0
                                 GATEWAY=192.168.216.2
                                 DNS=8.8.8.8
```

让天下没有难学的技术

```
[root@localhost network-scripts]# yum install -y git
[root@localhost network-scripts]# git clone https://github.com/jpetazzo/pipework
[root@localhost network-scripts]# cp pipework/pipework /usr/local/bin/
[root@localhost network-scripts]# docker run -itd --net=none --name=ff centos-6-x86 bash
[root@localhost network-scripts]# pipework br0 f1 192.168.216.135/24
```

让天下没有难学的技术

Over

让天下没有难学的技术

namespace	系统调用参数	隔离内容	内核版本
UTS	CLONE_NEWUTS	主机名和域名	2.6.19
IPC	CLONE_NEWIPC	信号量、消息队列和共享内存	2.6.19
PID	CLONE_NEWPID	进程编号	2.6.24
NetWork	CLONE_NEWNET	网络设备、网络栈、端口等	2.6.29
Mount	CLONE_NEWNS	挂载点（文件系统）	2.4.19
User	CLONE_NEWUSER	用户和用户组	3.8

让天下没有难学的技术