链滴

配置 docker 静态 ip 地址

作者：cuijianzhe
原文链接：https://ld246.com/article/1553668284610
来源网站：链滴

一、docker的网络模式

1. Docker 有以下 4 种网络模式：

host 模式，使用--net=host 指定。

container 模式，使用--net=container:NAME_or_ID 挃定。

none 模式，使用--net=none 指定。

bridge 模式，使用--net=bridge 指定，默认就是 bridge 模式。

默认选择 bridge 的情况下，容器启动后会通过 DHCP 获取一个地址，这可能不是我们想要的，在

centos7 系统上， docker 环境下可以使用 pipework 脚本对容器分配固定 IP（这个 IP 可以是和物机同网段 IP）。

注： docker 默认是 bridge（--net=bridge）模式，相当于 VMware 中 NAT 模式。

docker 环境下可以使用 pipework 脚本对容器分配固定 IP，相当于 VMware 中桥接模式。注：Pipework 有个缺陷，容器重吭后 IP 设置会自动消失，需要重新设置。

2. 配置桥接网络

桥接本地物理网络的目的,是为了局域网内用户方便访问 docker 实例中服务,丌要需要各种端口映射即访问服务。 但是这样做,又违背了 docker 容器的安全隔离的原则,工作中辩证的选择.

创建桥设备 br0：

安装包：

http://www.rpmfind.net/linux/rpm2html/search.php?query=bridge-utils(x86-64)

```
[root@bogon ~]# rpm -ivh /mnt/Packages/bridge-utils-1.5-9.el7.x86_64.rpm

准备中...                          ################################# [100%]
   软件包 bridge-utils-1.5-9.el7.x86_64 已经安装
[root@bogon ~]# cd /etc/sysconfig/network-scripts/
[root@bogon network-scripts]# cp ifcfg-ens33 /opt/
[root@bogon network-scripts]# vim ifcfg-ens33
TYPE="Ethernet"
PROXY_METHOD="none"
BROWSER_ONLY="no"
BOOTPROTO="dhcp"
DEFROUTE="yes"
IPV4_FAILURE_FATAL="no"
IPV6INIT="yes"
IPV6_AUTOCONF="yes"
IPV6_DEFROUTE="yes"
IPV6_FAILURE_FATAL="no"
IPV6_ADDR_GEN_MODE="stable-privacy"
NAME="ens33"
UUID="d18d860b-c4ae-4b23-8851-3847cc811fa6"
DEVICE="ens33"
ONBOOT="yes"
IPV6_PRIVACY="no"
BRIDGE="br0"
```

#删除 IP 地址相关内容，删除下面 几行

IPADDR=172.17.148.238

NETMASK=255.255.240.0

文末插入BRIDGE="br0"

创建生成ifcfg-br0文件

```
[root@bogon network-scripts]# vim ifcfg-br0
DEVICE="br0"
NM_CONTROLLED="yes"
ONBOOT="yes"
TYPE="Bridge"
BOOTPROTO="dhcp"
DNS1=114.114.114.114
```

查看当前网卡状态：

```
[root@bogon ~]# ifconfig

br0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.89.209  netmask 255.255.255.0  broadcast 192.168.89.255
        inet6 fe80::a0a0:45ff:fe1b:5e0b  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:33:7b:83  txqueuelen 1000  (Ethernet)
        RX packets 77  bytes 9297 (9.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20  bytes 2357 (2.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

docker0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 0.0.0.0
        inet6 fe80::42:a1ff:fee8:1077  prefixlen 64  scopeid 0x20<link>
        ether 02:42:a1:e8:10:77  txqueuelen 0  (Ethernet)
        RX packets 33599  bytes 2213263 (2.1 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 37274  bytes 74193506 (70.7 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether 00:0c:29:33:7b:83  txqueuelen 1000  (Ethernet)
        RX packets 523587  bytes 226962732 (216.4 MiB)
        RX errors 0  dropped 2  overruns 0  frame 0
        TX packets 103893  bytes 9210892 (8.7 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 638  bytes 57348 (56.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 638  bytes 57348 (56.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

veth30787ca: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::e872:ffff:fe01:47ae  prefixlen 64  scopeid 0x20<link>
```

```
        ether ea:72:ff:01:47:ae  txqueuelen 0  (Ethernet)
        RX packets 8  bytes 648 (648.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 23  bytes 1854 (1.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

vethad3caa5: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::3cfc:d0ff:feea:45a8  prefixlen 64  scopeid 0x20<link>
        ether 3e:fc:d0:ea:45:a8  txqueuelen 0  (Ethernet)
        RX packets 8  bytes 648 (648.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 24  bytes 1944 (1.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

可以看出br0网卡已变成桥接类型网卡。

还有另一种方法：

```
[root@bogon ~]# git clone https://github.com/jpetazzo/pipework.git
```

2.使用pipework

```
[root@bogon ~]# git  clone  https://github.com/jpetazzo/pipework.git

[root@bogon pipework]# cp pipework /usr/local/bin/

[root@bogon ~]# docker run -itd --net=none --privileged=true docker.io/centos:latest /bin/ash #开启特权模式
```

pipework 网桥名 容器实例 ID 分配给容器的 IP /掩码 @网关