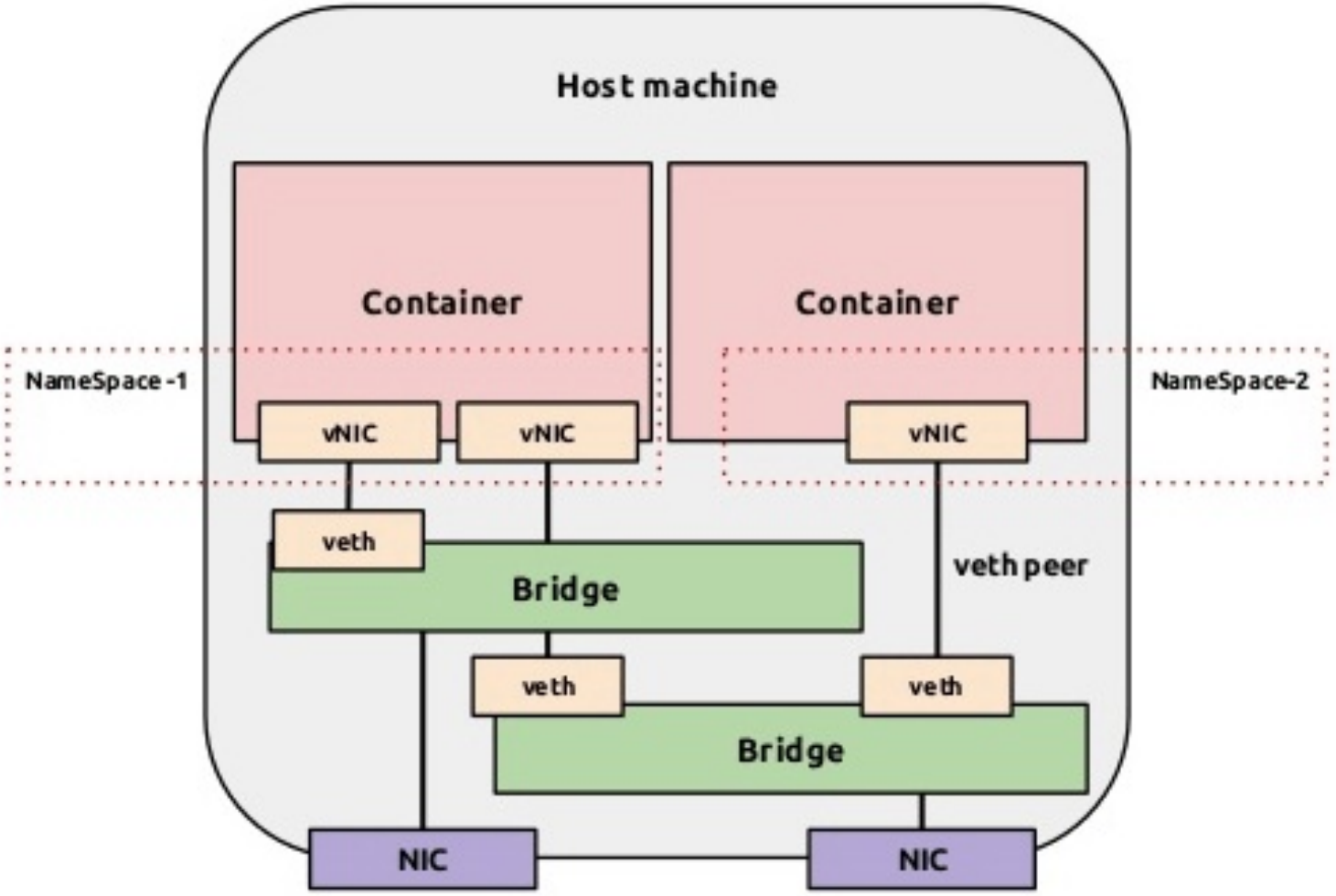
Docker 网络配置

Docker 网络配置

- Dokcer通过使用Linux桥接提供容器之间的通信，docker0桥接接口的目的就是方便Docker管理。
- 当Docker daemon启动时需要做以下操作：
- creates the docker0 bridge if not present
  - #如果docker0不存在则创建 searches for an
- IP address range which doesn't overlap with an existing route
  - #搜索一个与当前路由不冲突的ip段
- picks an IP in the selected range
  - #在确定的范围中选择
- ipassigns this IP to the docker0 bridge
  - #绑定ip到docker0

Docker 网络配置


Host machine
Container
Container
NameSpace-1
NameSpace-2
vNIC
vNIC
vNIC
veth
Bridge
veth peer
veth
veth
Bridge
NIC
NIC

Docker 网络配置

- Docker run创建Docker容器时，可以用—netx选项指定容器的网络模式，Docker有以下四种网络模式：
  - Host模式，使用--net=host指定
  - Container模式，使用--net=container:Name_or_Id指定
  - None模式，使用--net=none指定
  - Bridge模式，使用--net=bridge指定，该模式为默认模式。

Docker 网络配置

- Host模式
  - 如果启动容器的时候使用host模式，那么这个容器将不会获得一个独立的Network Namespace，而是和宿主机公用一个Network Namespace。容器将不会虚拟出自己的网卡，配置自己的IP等，而是使用宿主机的IP和端口。
  - 例如，我们在10.10.101.105/24的机器上用host模式启动一个含有web应用的Docker容器，监听tcp80端口。当我们在容器中执行任何类似ifconfig命令查看网络环境时，看到的都是宿主机上的信息。而外界访问容器中的应用，则直接使用10.10.101.105:80即可，不用任何NAT转换，就如直接跑在宿主机中一样。但是，容器其他方面，如文件系统、进程列表等还是和宿主机隔离的。

Docker 网络配置

- Container模式
  - 这个模式指定新创建的容器和已经存在的一个容器共享一个Network Namespace，而不是和宿主机共享。新创建的容器不会创建自己的网卡，配置自己的IP，而是和一个指定的容器共享IP、端口范围等。同样，两个容器除了网络方面，其他如文件系统、进程列表还是隔离的。两个容器的进程可以通过lo网卡设备通信。

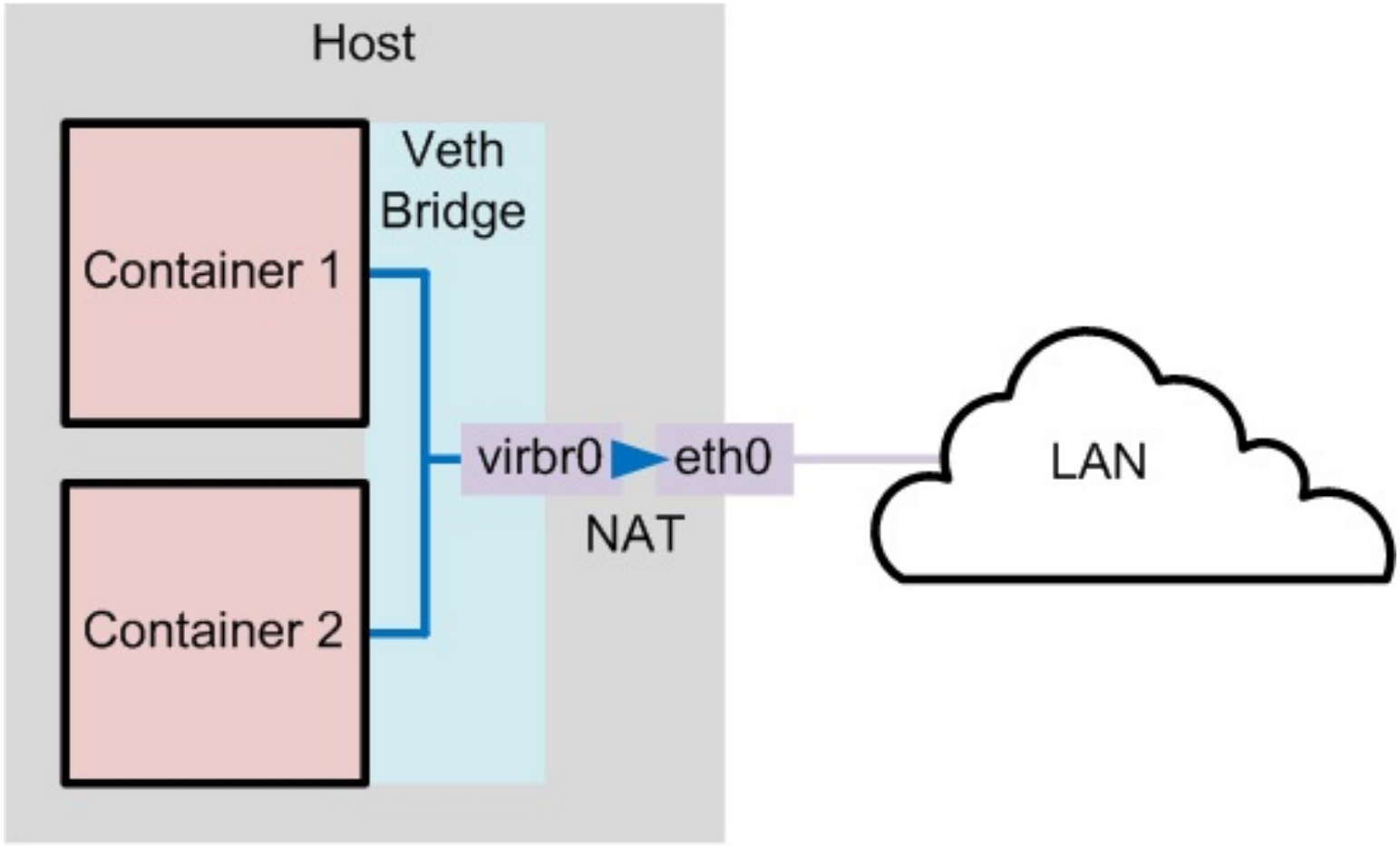
Docker 网络配置

- None模式
  - 这个模式和前两个不同。在这种模式下，Docker容器拥有自己的Network Namespace，但是，并不为Docker容器进行任何网络配置。也就是说，这个Docker容器没有网卡、IP、路由等信息。需要我们自己为Docker容器添加网卡、配置IP等。

Docker 网络配置

- Bridge模式
  - Bridge模式是Docker默认的网络设置，此模式会为每一个容器分配Network Namespace、设置IP等，并将一个主机上的Docker容器链接到一个虚拟网桥上。当Docker server启动时，会在主机上创建一个名为docker0的虚拟网桥，此主机上启动的Docker容器会链接到这个虚拟网桥上。虚拟网桥的工作方式和物理交换机类似，这样主机上的所有容器就通过交换机连接在了一个二层网络中。接下来就是要为容器分配IP了，Docker会从RFC1918所定义的私有Ip网段中，选择一个和宿主机不同的Ip地址和子网分配给docker0，连接到docker0的容器就从这个子网中选择一个未被占用的Ip使用。如一般Docker会使用172.17.0.0/16这个网段，并将172.17.42.1/16分配给docker0网桥（在宿主机上使用ifconfig命令是可以看到docker0的，可以认为他是网桥的管理接口，在宿主机上作为一块虚拟网卡使用）

Docker 网络配置


Host
Veth
Bridge
Container 1
Container 2
virbr0
eth0
NAT
LAN

Docker 网络配置

- 列出当前主机网桥

  $ sudo brctl show

- 查看当前docker0 ip

  $ sudo ifconfig docker0

  在容器运行时，每个容器都会分配一个特定的虚拟机口并桥接到docker0.每个容器都会配置与docker0 ip相同网段的私有IP地址，dockr0的Ip地址被用于所有容器的默认网关。

- 运行一个容器

  $ sudo docker run -t -i ubuntu /bin/bash

  $ sudo brctl show

  对比一下容器创建后虚拟网桥上的端口变化。

Docker 网络配置

- 使用特定范围的IP
  - Docker会尝试寻找没有被主机使用的IP段，尽管它适用于大多数情况下，但是它不是万能的，有时候我们还是需要对IP进行进一步规划。Docker允许你管理docker0桥接或者通过-b选项自定义桥接网卡，需要安装bridge-utils软件包。
- 基本步骤如下：
  - Ensure Docker is stopped
  - Crate your own bridge(bridge0 for example)
  - Assign a specific IP to this bridge
  - Start Docker with the -b=bridge0 parameter

Docker 网络配置

- 操作示例：

```
$ sudo systemctl stop docker
$ sudo ip link set dev docker0 down
$ sudo brictl delbr docker0
$ sudo brctl addbr bridge0
$ sudo ip addr add 192.168.5.1/24 dev bridge0
$ sudo ip link set dev bridge0 up
$ ip addr show bridge0
$ echo ‘DOCKER_OPTS=“-b=bridge0”” >> /etc/default/docker
$ sudo systemctl start docker
```

Docker 网络配置

- 不同主机之间容器的通信
  - 不同主机之间的通信可以借助于pipwork这个工具：

    $ git clone https://github.com/jpetazzo/pipwork.git

    $ sudo cp -rp pipwork/pipwork /usr/local/bin
  - 安装相应依赖软件

    $ sudo apt-get install iputils-arping bridge-utils

Docker 网络配置

- pipwork