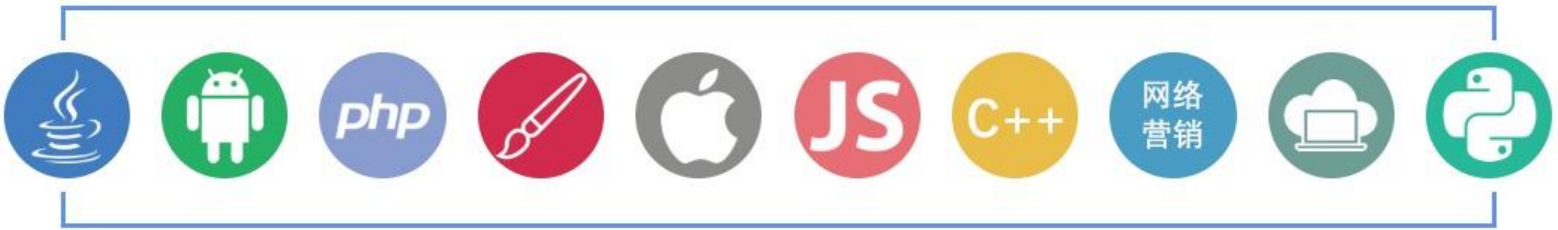
Docker核心技术之网络管理



Docker核心技术之网络管理-课程概要

- Docker 网络管理简介
- Docker 网络管理命令
- Docker 网络模式简介
- 总结

Docker核心技术之网络管理

Docker 网络管理简介

为什么需要Docker网络管理

容器的网络默认与宿主机、与其他容器都是相互隔离。

- 容器中可以运行一些网络应用(如nginx、web应用、数据库等)，如果要让外部也可以访问这些容器内运行的网络应用，那么就需要配置网络来实现。
- 有可能有的需求下，容器不想让它的网络与宿主机、与其他容器隔离。
- 有可能有的需求下，容器根本不需要网络。
- 有可能有的需求下，容器需要更高的定制化网络（如定制特殊的集群网络、定制容器间的局域网）。
- 有可能有的需求下， 容器数量特别多，体量很大的一系列容器的网络管理如何
- ……

因此容器的网络管理是非常重要的

Docker中有哪些网络驱动模式

Docker有五种网络驱动模式

- bridge network 模式（网桥）：默认的网络模式。类似虚拟机的nat模式
- host network 模式（主机）：容器与宿主机之间的网络无隔离，即容器直接使用宿主机网络
- None network 模式：容器禁用所有网络。
- Overlay network 模式（覆盖网络）： 利用VXLAN实现的bridge模式
- Macvlan network 模式：容器具备Mac地址，使其显示为网络上的物理设备

Docker核心技术之网络管理

Docker网络管理命令

查看网络 – docker network ls

- 作用：

  查看已经建立的网络对象

- 命令格式：

  docker network ls [OPTIONS]

- 命令参数(OPTIONS)：

-f, --filter filter	**过滤条件(如 'driver=bridge’)**
--format string	**格式化打印结果**
--no-trunc	**不缩略显示**
-q, --quiet	**只显示网络对象的ID**

- 注意：

  默认情况下，docker安装完成后，会自动创建bridge、host、none三种网络驱动

查看网络 – docker network ls

- 命令演示：

```
[root@centos-linux ~]# docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
62f268b886d5        bridge              bridge              local
6f3d8fb80ce2        host                host                local
ac7a20a911b6        none                null                local
[root@centos-linux ~]# docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
62f268b886d5        bridge              bridge              local
6f3d8fb80ce2        host                host                local
ac7a20a911b6        none                null                local
[root@centos-linux ~]# docker network ls --no-trunc
NETWORK ID                                                         NAME                DRIVER              SCOPE
62f268b886d54428842434b2f20d7454dcdbc61bee9919ddb8edf6d8f5f89dfa   bridge              bridge              local
6f3d8fb80ce2db868d7737253d5e6e15d716a041a209c4d0e6c0cddf9d514d8f   host                host                local
ac7a20a911b6d10ad84619cce8c762c0ab24d4045fabc87af3f29eebf2e68a6f   none                null                local
[root@centos-linux ~]# docker network ls -f 'driver=host'
NETWORK ID          NAME                DRIVER              SCOPE
6f3d8fb80ce2        host                host                local
[root@centos-linux ~]#
```

创建网络 – docker network create

- 作用：

  创建新的网络对象

- 命令格式：

  docker network create [OPTIONS] NETWORK

- 命令参数(OPTIONS)：

-d, --driver string	**指定网络的驱动(默认 "bridge")**
--subnet strings	**指定子网网段(如192.168.0.0/16、172.88.0.0/24)**
--ip-range strings	**执行容器的IP范围，格式同subnet参数**
--gateway strings	**子网的IPv4 or IPv6网关，如(192.168.0.1)**

- 注意：

  host和none模式网络只能存在一个

  docker自带的overlay 网络创建依赖于docker swarm(集群负载均衡)服务

  192.168.0.0/16 等于 192.168.0.0~192.168.255.255　192.168.8.0/24

  172.88.0.0/24 等于 172.88.0.0~172.88.0.255

创建网络 – docker network create

- 命令演示：

```
[root@centos-linux ~]# docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
62f268b886d5        bridge              bridge              local
6f3d8fb80ce2        host                host                local
ac7a20a911b6        none                null                local
[root@centos-linux ~]# docker network create -d bridge my-bridge
8de61becb4055d75c3d54b135334725258d9cb85f959a4f1e6d731f00e7e0b1a
[root@centos-linux ~]# docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
62f268b886d5        bridge              bridge              local
6f3d8fb80ce2        host                host                local
8de61becb405        my-bridge           bridge              local
ac7a20a911b6        none                null                local
```

网络删除 – docker network rm

- 作用：

  删除一个或多个网络

- 命令格式：

  docker network rm NETWORK [NETWORK...]

- 命令参数(OPTIONS)：

  无

查看网络详细信息 – docker network inspect

- 作用：

  查看一个或多个网络的详细信息

- 命令格式：

  docker network inspect [OPTIONS] NETWORK [NETWORK...]

  或者 **docker inspect [OPTIONS] NETWORK [NETWORK...]**

- 命令参数(OPTIONS)：

  -f, --format string　　根据format输出结果

使用网络 – docker run --network

- 作用：

  为启动的容器指定网络模式

- 命令格式：

  docker run/create --network NETWORK

- 命令参数(OPTIONS)：

  无

- 注意：

  默认情况下，docker创建或启动容器时，会默认使用名为bridge的网络

网络连接与断开 – docker network connect/disconnect

- 作用：

  将指定容器与指定网络进行连接或者断开连接

- 命令格式：

  docker network connect [OPTIONS] NETWORK CONTAINER

  docker network disconnect [OPTIONS] NETWORK CONTAINER

- 命令参数(OPTIONS)：

  -f, --force　　强制断开连接(用于disconnect)

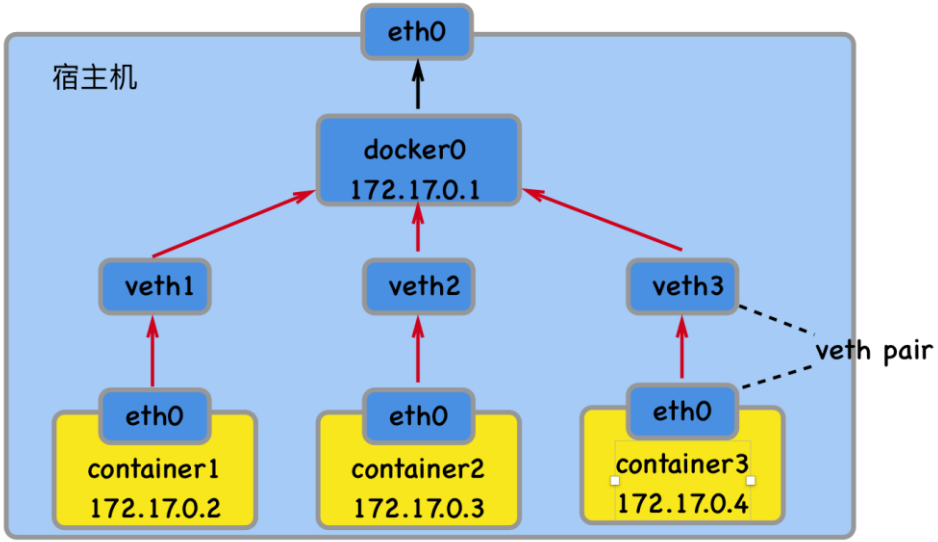
Docker核心技术之网络管理

Docker网络模式简介

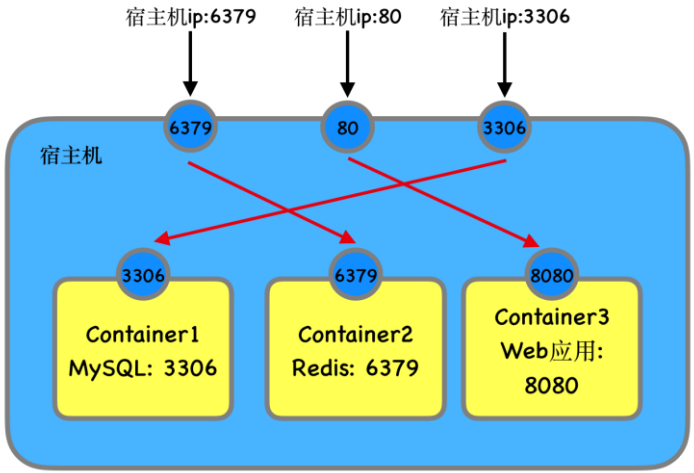
bridge 网络模式（一）

特点：

- 宿主机上需要单独的bridge网卡，如默认docker默认创建的docker0。
- 容器之间、容器与主机之间的网络通信，是借助为每一个容器生成的一对veth pair虚拟网络设备对，进行通信的。一个在容器上，另一个在宿主机上。
- 每创建一个基于bridge网络的容器，都会自动在宿主机上创建一个veth**虚拟网络设备。
- 外部无法直接访问容器。需要建立**端口映射**才能访问。
- 容器借由veth虚拟设备通过如docker0这种bridge网络设备进行通信。
- 每一容器具有单独的IP



bridge 网络模式（二） – 端口映射



- 作用：

  启动的容器时，为容器进行端口映射

- 命令格式：

  docker run/create -P ...

  或者 **docker run/create –p ...**

- 命令参数(OPTIONS)：

  -P, --publish-all　　将容器内部所有暴露端口进行随机映射

  -p, --publish list　　手动指定端口映射

- 注意：

  -p [HOST_IP]:[HOST_PORT]:CONTAINER_PORT

  如：-p ::80　　将容器的80端口随机(端口)映射到宿主机任意IP

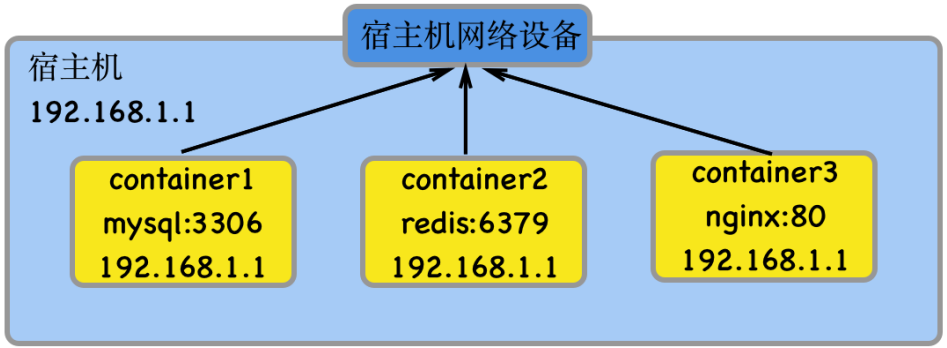
  -p :8000:6379　　将容器的6379端口映射到宿主机任意IP的8000端口

  -p 192.168.5.1::3306 将容器的3306端口随机(端口)映射到宿主机的192.168.5.1IP上

host 网络模式

特点：

- 容器完全共享宿主机的网络。网络没有隔离。宿主机的网络就是容器的网络。
- 容器、主机上的应用所使用的端口不能重复。例如：如果宿主机已经占用了8090端口，那么任何一个host模式的容器都不可以使用8090端口了；反之同理。
- 外部可以直接访问容器，不需要端口映射。
- 容器的IP就是宿主机的IP

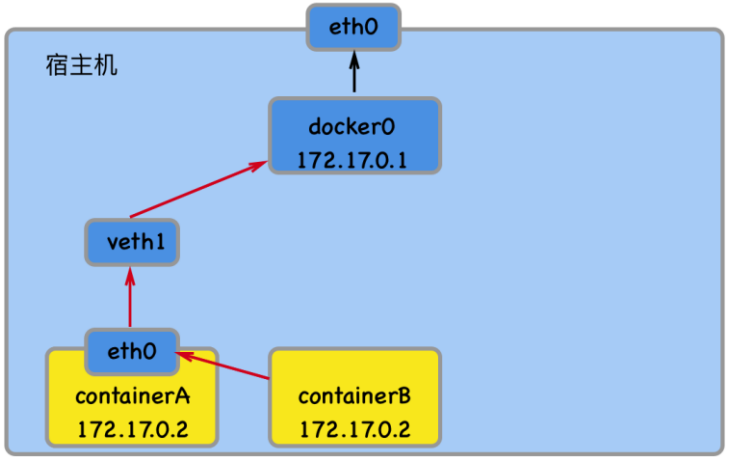


特殊host 网络模式（Container网络模式）

- Container网络模式，其实就是容器共享其他容器的网络。
- 相当于该容器，，在网络层面上，将其他容器作为“主机”。它们之间的网络没有隔离。
- 这些容器之间的特性同host模式。

使用方法：

Docker run/create --network container:CONTAINER ...



none 网络模式

特点：

- 容器上没有网络，也无任何网络设备。
- 如果需要使用网络，需要用户自行安装与配置。

应用场景

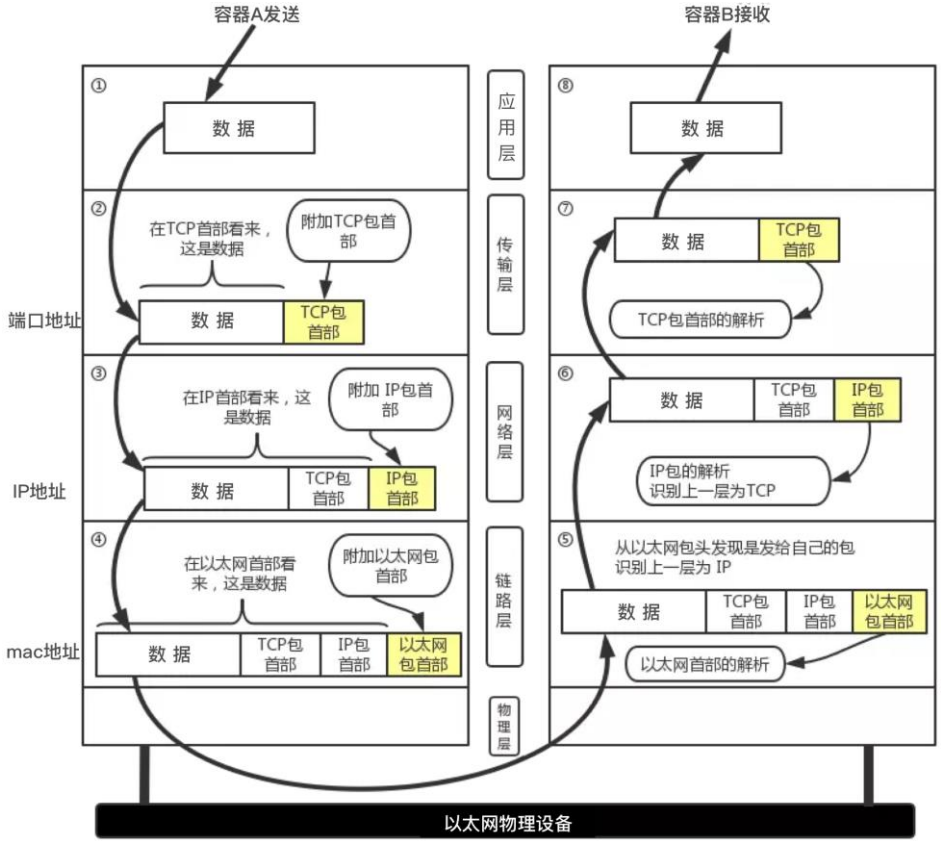
- 该模式适合需要高度定制网络的用户使用。

overlay 网络模式（一）

- Overlay 网络，也称为覆盖网络。
- Overlay 网络的实现方式和方案有多种。Docker自身集成了一种，基于VXLAN隧道技术实现。
- Overlay 网络主要用于实现跨主机容器之间的通信。

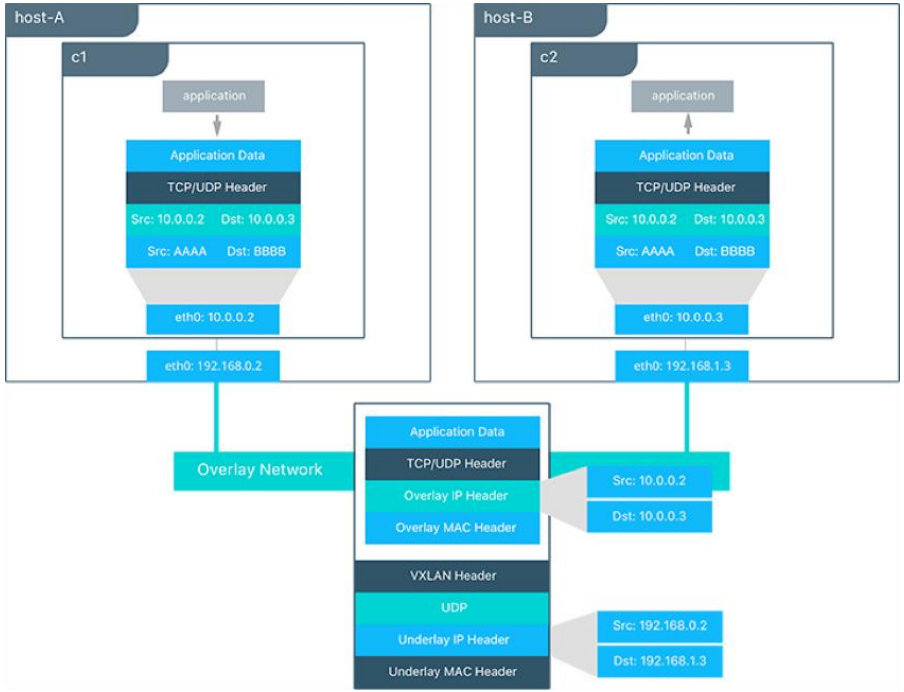
应用场景：需要管理成百上千个跨主机的容器集群的网络时。

overlay 网络模式（二）- 了解TCP/IP协议栈



overlay 网络模式（三）- 实现原理

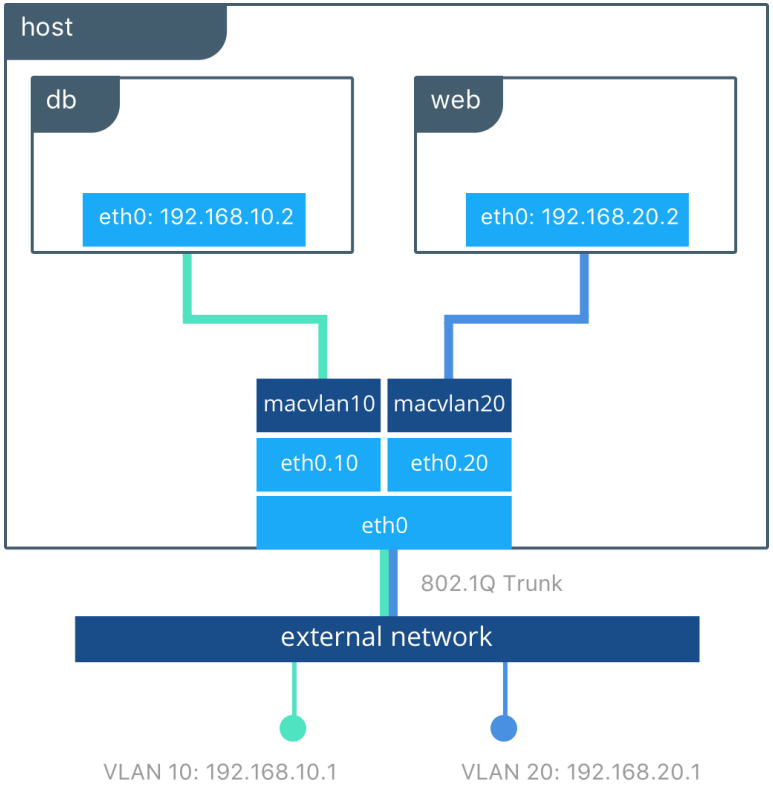
IP隧道网络原理



macvlan 网络模式

- macvlan网络模式，最主要的特征就是他们的通信会直接基于mac地址进行转发。
- 这时宿主机其实充当一个二层交换机。Docker会维护着一个MAC地址表，当宿主机网络收到一个数据包后，直接根据mac地址找到对应的容器，再把数据交给对应的容器。
- 容器之间可以直接通过IP互通，通过宿主机上内建的虚拟网络设备（创建macvlan网络时自动创建），但与主机无法直接利用IP互通。

应用场景：由于每个外来的数据包的目的mac地址就是容器的mac地址，这时每个容器对于外面网络来说就相当于一个真实的物理网络设备。因此当需要让容器来的网络看起来是一个真实的物理机时，使用macvlan模式



Docker核心技术之网络管理

总结

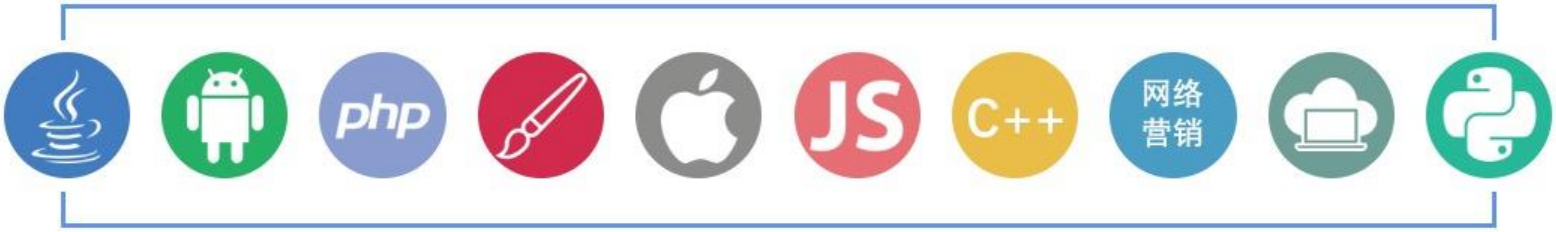
Docker核心技术之网络管理 - 总结

重点掌握：

- bridge网络、host网络、Container网络模式的原理和使用（应用得较多，且host网络性能最优）。
- docker network命令的使用

了解：

- none网络的效果
- overlay网络、macvlan网络的原理。（使用起来难度较大）



Thank you!

改变中国IT教育，我们正在行动