Brought by Geekbang. 极客邦科技 InfoQ

CNUT
Connect Container Community

全球容器技术大会

剖析容器企业实践 关注容器生态圈开源项目

深入Docker的资源管理

SpeedyCloud
李雨来

Docker

- 资源的隔离
  - Linux Namespaces
- 资源使用的限制（QoS）
  - cgroups
- 资源使用的监控
  - cgroups

Docker资源的隔离

Docker的资源隔离

- Linux Namespaces
  - CLONE_NEWNS (2.4.19)
  - CLONE_NEWUTS (2.6.19)
  - CLONE_NEWIPC (2.6.19)
  - CLONE_NEWPID (2.6.24)
  - CLONE_NEWNET (2.6.24)
  - CLONE_NEWUSER (3.8)

Docker的资源隔离

- /proc/$PID/ns/
  - ipc
  - mnt
  - net
  - pid
  - user
  - uts

```
lrwxrwxrwx 1 root root 0 Aug 16 10:11 ipc -> ipc:[4026531839]
lrwxrwxrwx 1 root root 0 Aug 16 10:11 mnt -> mnt:[4026531840]
lrwxrwxrwx 1 root root 0 Aug 16 10:11 net -> net:[4026531956]
lrwxrwxrwx 1 root root 0 Aug 16 10:11 pid -> pid:[4026531836]
lrwxrwxrwx 1 root root 0 Aug 16 10:11 user -> user:[4026531837]
lrwxrwxrwx 1 root root 0 Aug 16 10:11 uts -> uts:[4026531838]
```

Docker的资源隔离

- Namespaces相关系统调用
  - clone
  - unshare
  - setns
- Namespaces相关命令行工具
  - ip netns
  - unshare

Docker的资源隔离

- 通过对clone系统调用，在一组新的Namespace下运行的进程

通过Namespaces可以干点什么？

Docker网络的改造

- 为Docker容器添加网卡

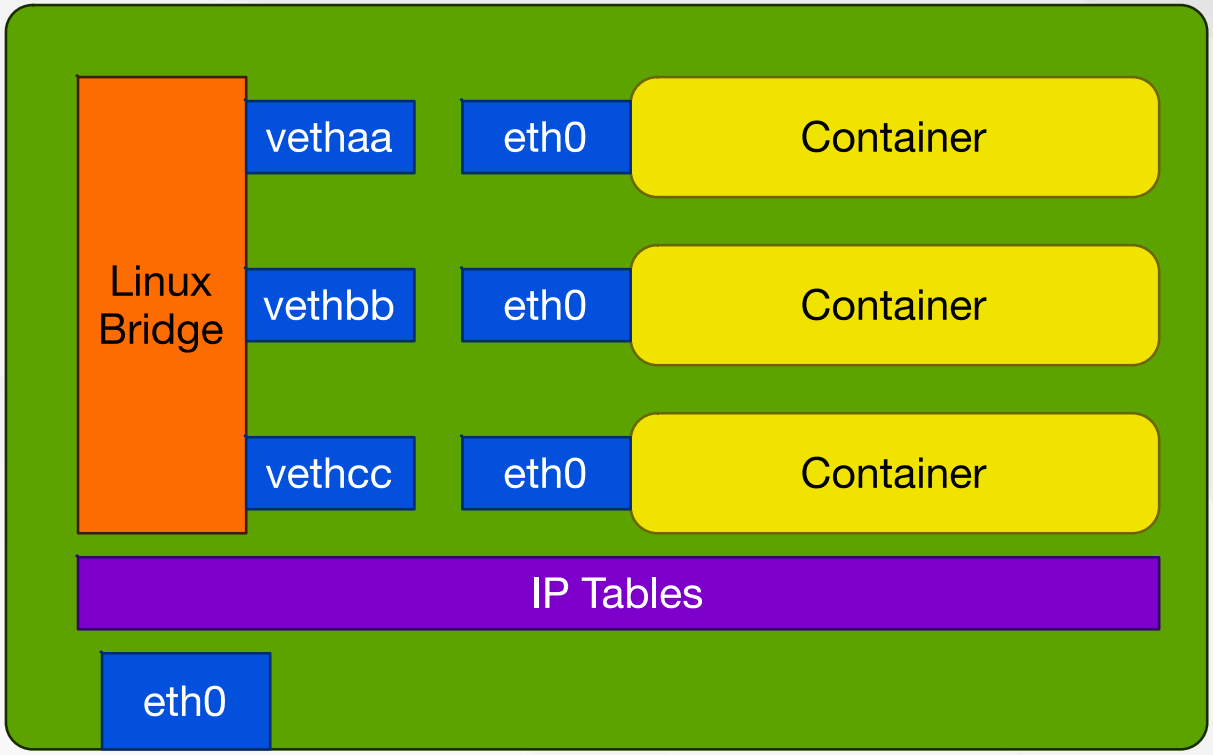
```
#!/bin/bash
PID=`docker inspect -f '{{.State.Pid}}' $1`
ID=`docker inspect -f '{{.Id}}' $1`
ETHNAME=$2
ln -s /proc/$PID/ns/net /var/run/netns/$ID
ip link add dev $ETHNAME.0 type veth peer name $ETHNAME.1
ip link set dev $ETHNAME.1 netns $ID
ip link set dev $ETHNAME.0 up
ip netns exec $ID ifconfig $ETHNAME.1 $3 up
rm -rf /var/run/netns/$ID
```

- 用法：network.sh docker-test veth0 192.168.1.10/24

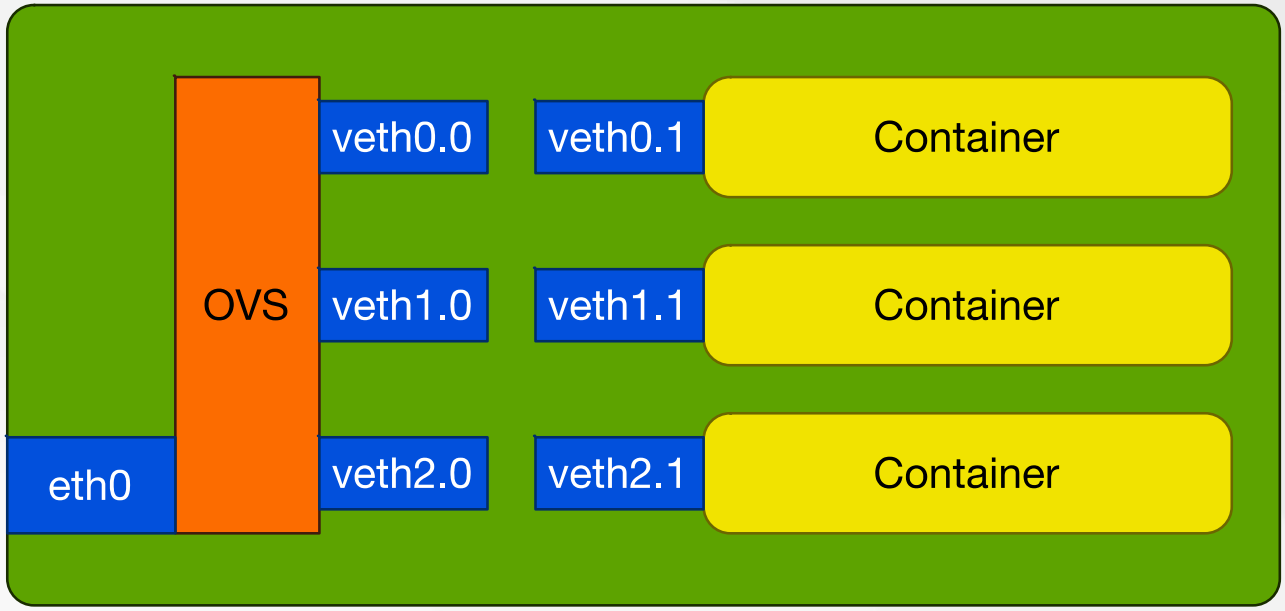
Docker网络的改造

- Linux Net Namespace: /proc/$PID/ns/net
- ip link add 命令创建 veth 网卡
- ip netns 命令通过检测 /var/run/netns 路径调整netns的配置
- 使用ovs-vsctl（或者brctl）来设置新网卡的网络

Docker网络的改造



Docker网络的改造



Docker资源的QoS

Docker的资源的QoS

- Linux cgroups
- Docker的cgroups模型
  - libcontainer：
    /sys/fs/cgroup/<subsystem>/docker/<containerID>
  - Linux Container:
    /sys/fs/cgroup/<subsystem>/lxc/<containerID>
- Kernel启动参数
  - cgroup_enable=memory swapaccount=1

Docker的资源QoS

- CPU的QoS
  - cpuset.cpus
  - cpu.shares
- 内存的QoS
  - memory.memsw.limit_in_bytes
  - memory.limit_in_bytes

swap = memory.memsw.limit_in_bytes – memory.limit_in_bytes

内存，Swap和OOM的纠葛

内存，Swap和OOM的纠葛

- 内存，Swap和OOM的触发
  - 当内存不够用时，Kernel会换出内存的page到swap
  - 如果swap和内存都满了，触发OOM
- OOM怎么工作
  - 寻找占用内存最大的进程（子进程的内存占用量会累计到父进程的内存占用量）
  - 杀掉内存占用最多的进程

内存，Swap和OOM的纠葛

- 想想看，如果你跑得是php-fpm如果OOM了...

内存，Swap和OOM的纠葛

- 如果你不想死得很快，请保证你的Container中给swap留足空间！

Docker的资源QoS

- 磁盘IO的QoS
  - blkio.throttle.read_bps_device
  - blkio.throttle.write_bps_device
  - blkio.throttle.read_iops_device
  - blkio.throttle.write_iops_device

- echo "<major>:<minor> <limit>" > throttle.write_iops_device

Docker的资源QoS

- 关于磁盘的major和minor号
  - cat /proc/partitions
  - ls –l /dev/
- 使用设备的major和minor，不是分区的major和minor

Docker的资源QoS

- 网络带宽的QoS
  - OpenVSwitch
  - Linux Bridge
- OpenVSwitch
  - ovs-vsctl set interface veth1 ingress_policing_rate=1000
- ebtables + tc
  - ebtables -A FORWARD -i veth1 -j mark --mark-set 0x1 --mark-target ACCEPT
  - qdics, class, filter
  - tc filter add dev eth0 parent 1:0 protocol ip handle 1 fw flowid 1:1

Docker的资源QoS

- 磁盘容量的限制
  - LVM创建卷，挂载之后通过--volume参数让Container访问
  - 使用btrfs，并设置quota
    - btrfs qgroup limit –e 100G /var/lib/docker/btrfs/subvolumes/CONTAINER_ID

Docker资源的监控

Docker的监控

- CPU： cpuacct.usage
- 内存：
  - memory.usage_in_bytes
  - memory.memsw.usage_in_bytes
- 磁盘IO：
  - blkio.throttle.io_serviced
  - blkio.throttle.io_service_bytes
- 带宽：
  - /sys/class/net/<ethname>/statistics/

Out of Docker

Out of Docker

- Bocker
  - https://github.com/p8952/bocker
- Dockerlite
  - https://github.com/docker/dockerlite

THANKS

全球容器技术大会

剖析容器企业实践　关注容器生态圈开源项目