腾讯GAIA平台DOCKER实践

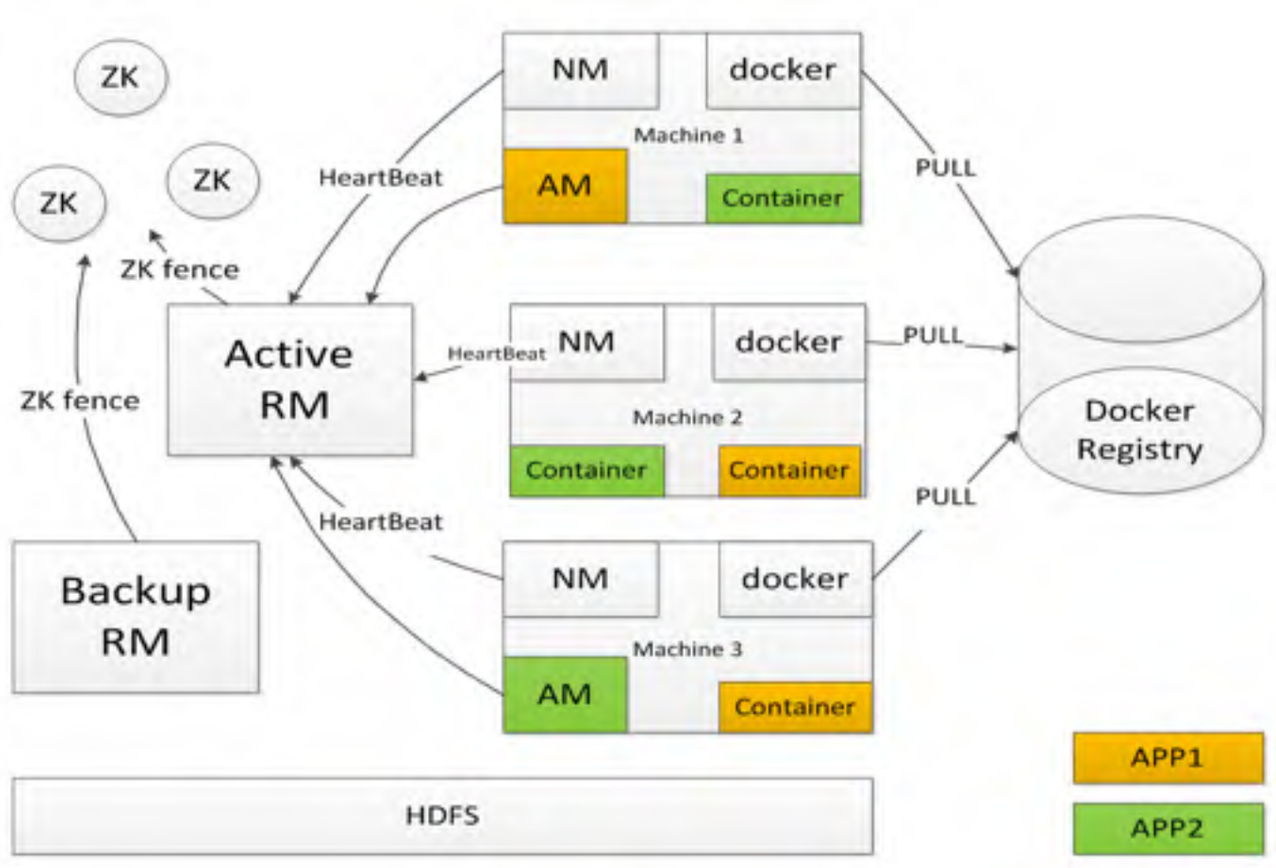
ramichen@tencent.com

GAIA平台

- Gaia平台是腾讯数据平台部的底层资源管理和调度系统，其上层业务包括离线、实时以及在线service服务，最大单集群规模达8800台、并发资源池个数达2500个，服务于腾讯所有事业群
- 2014年10月份正式上线对Docker类型作业的支持，通过Docker将Gaia云平台以更好用的方式呈献给各个业务。目前已经支持腾讯内部的游戏云、广点通、GPU深度学习等Docker业务

GAIA架构

- Gaia系统Master节点RM/NM/AM/Docker均无单点故障，可进行热升级。Gaia也为用户的App提供了本地重试和跨机迁移两种容灾方式
- 机器资源CPU、内存、网络出带宽、GPU、磁盘空间的隔离都是弹性管理的
- 最大化的利用集群的所有资源，在保证用户最低资源使用量的同时，在集群有空闲资源时还能借集群的空闲资源使用
- 自研调度器SFair，解决了调度器效率和扩展性问题，每秒可调度4k个实例



内容

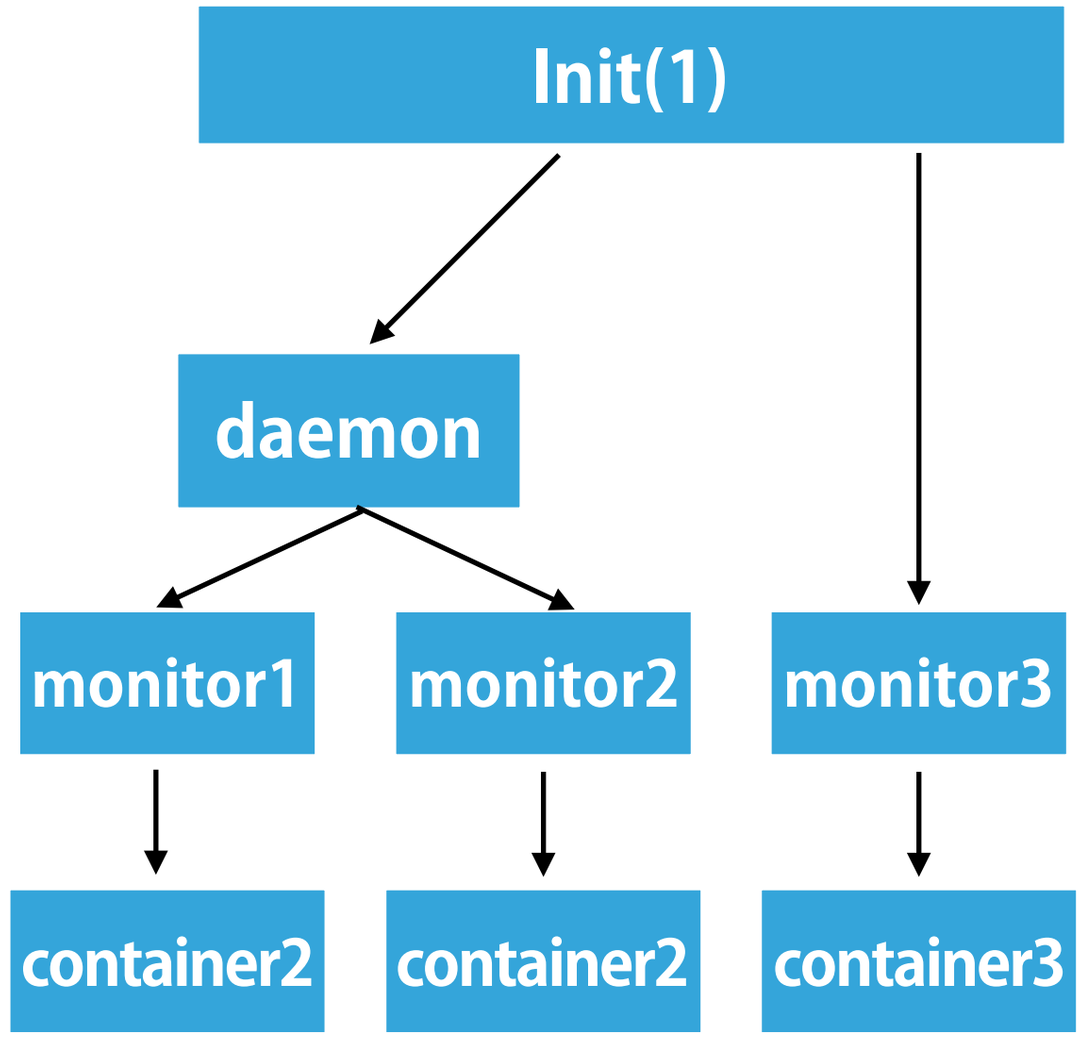
- Docker热升级功能
- Docker网络模式扩展
- Container数据存储问题
- Container资源隔离
- 其他

DOCKER DAEMON单点问题

- Docker daemon本身是个单点，并且在退出时会杀掉所有container，对于在线服务完全不可接受
- Docker坑也太多，比如1.6.x版本
  - Docker stats crashed docker daemon https://github.com/docker/docker/pull/13906
  - Docker exec未同步导致docker crash https://github.com/docker/docker/pull/14899
  - Docker 缓存container的stdout/stderr过大导致OOM问题 https://github.com/docker/docker/issues/9139

DOCKER热升级功能设计

- Docker daemon停止时主动杀掉所有container，主要受限于
  - 用户进程是daemon的子进程
  - IO流经过daemon缓存
- 原来的两层进程父子关系变为三层，monitor由goroutine改为进程，由它等待container运行结束
- Docker重启时，monitor孤儿进程托管给init进程，container不受任何影响
- Docker重启后恢复所有Container状态



DOCKER CRASH不影响DOCKER WAIT

① 客户端向daemon发送wait请求 POST /containers/1b9ba1/wait

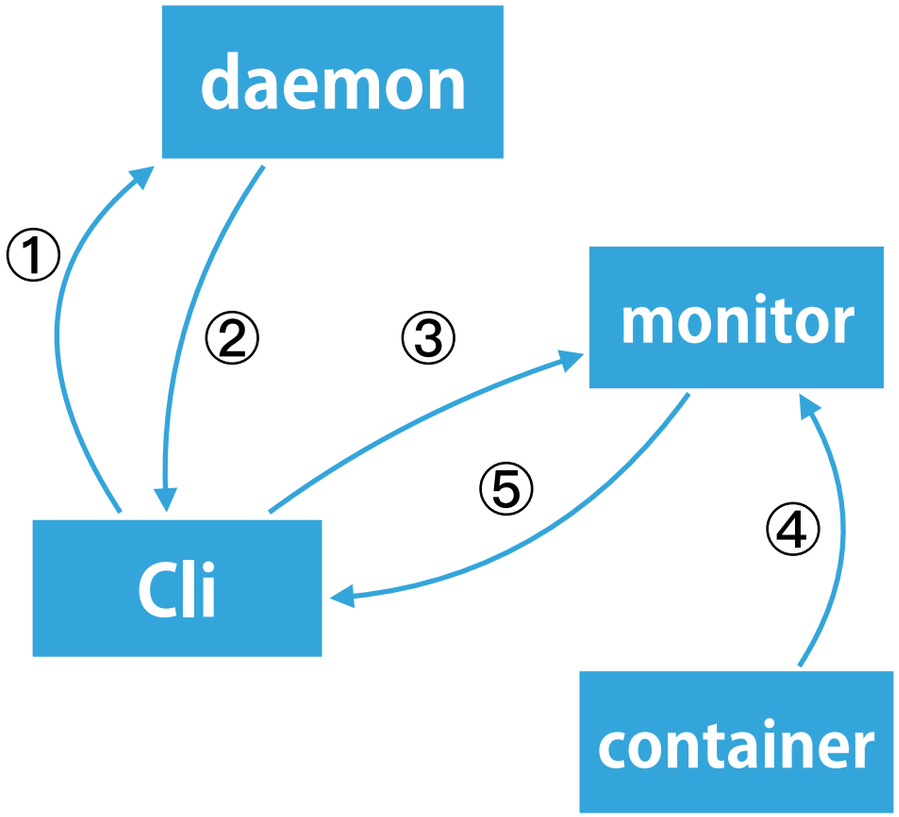
② daemon回复HTTP重定向code 305, 并返回monitor进程监听地址 Location: http://daemon_ip:port

③ 客户端向monitor发送wait请求 POST /containers/1b9ba1/wait

④ monitor进程等待container进程结束

⑤ container结束时返回，monitor进程返回wait请求的HTTP response

即使daemon挂掉，客户端wait请求不受影响，消除对Gaia等上层调度系统的影响



DOCKER热升级功能实现

- Docker daemon增加–hot-restart参数，monitor代码接口化，支持goroutine和外部进程两种方式
- container启动和结束时，monitor先将启动和结束事件持久化到磁盘，再通知daemon更新container状态。daemon重启后从磁盘加载container状态迁移文件
- 部分Daemon API接口，如attach/wait等，重定向给monitor进程处理
- 网络状态，比如libnetwork CNM模型的network/endpoint/sandbox状态按网络模式类型在global和local KV store中分别存储，libkv提供统一接口，屏蔽global/local等KV存储的API差异
- 我们为libnetwork引入了localstore功能，https://github.com/docker/libnetwork/pull/466，存储bridge网络模式状态或者sandbox等其他本地状态数据

DOCKER网络模式

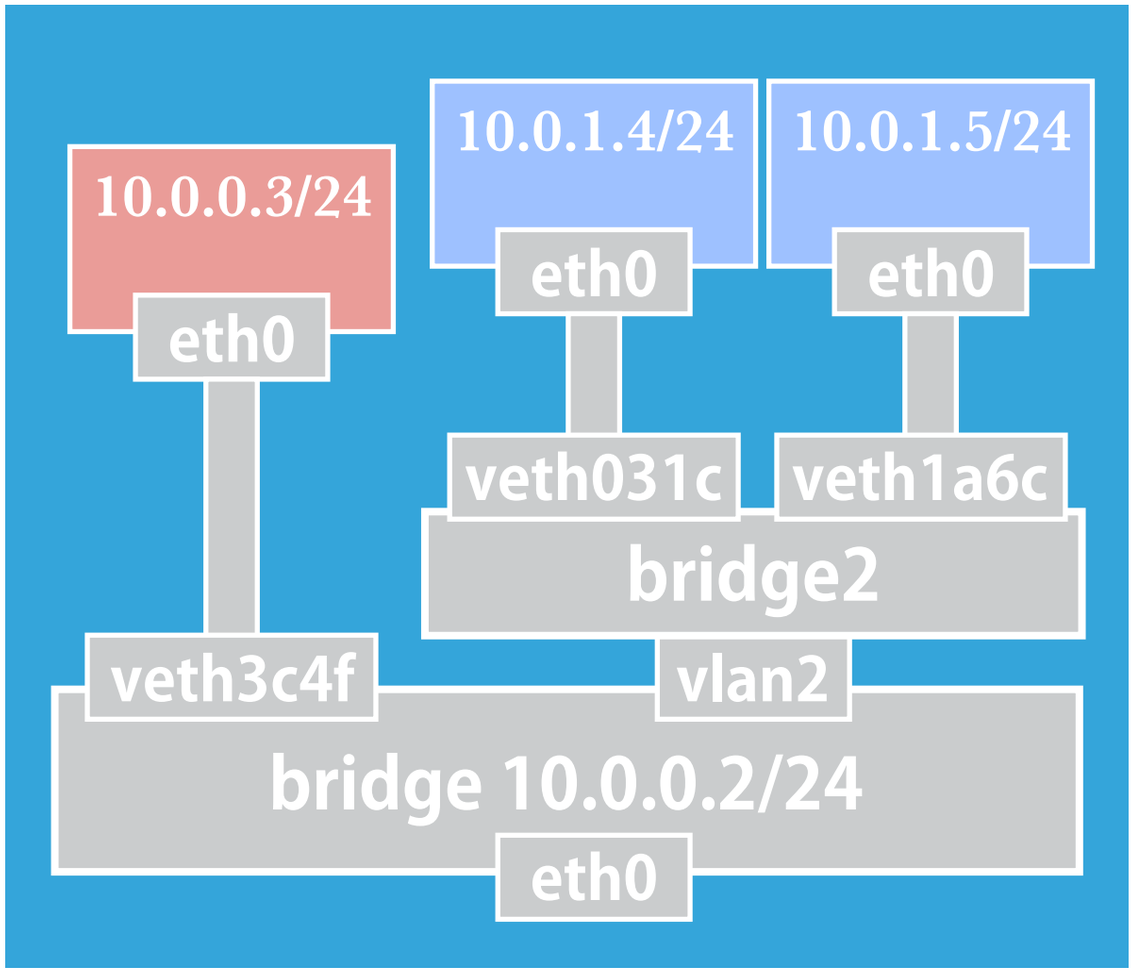
- Docker Host方式
  - 性能好
  - 没有网络隔离，有端口冲突问题
- Docker NAT网络模式
  - 提供了网络隔离功能，解决了端口冲突问题
  - Container IP对外不可见
  - 端口映射提高了业务的迁移成本
  - 网络IO性能比Host方式差接近10%

用户对网络的需求

- 端口冲突问题不想改代码或者配置
- 有些业务会将本机IP注册到ZooKeeper上做服务发现
- 有些业务会对有权限访问服务的IP做限制，只有白名单IP可以访问服务
- 很多业务使用腾讯的TGW网关对外提供服务
- 某些业务（如GPU业务）要求很好的网络性能
- …

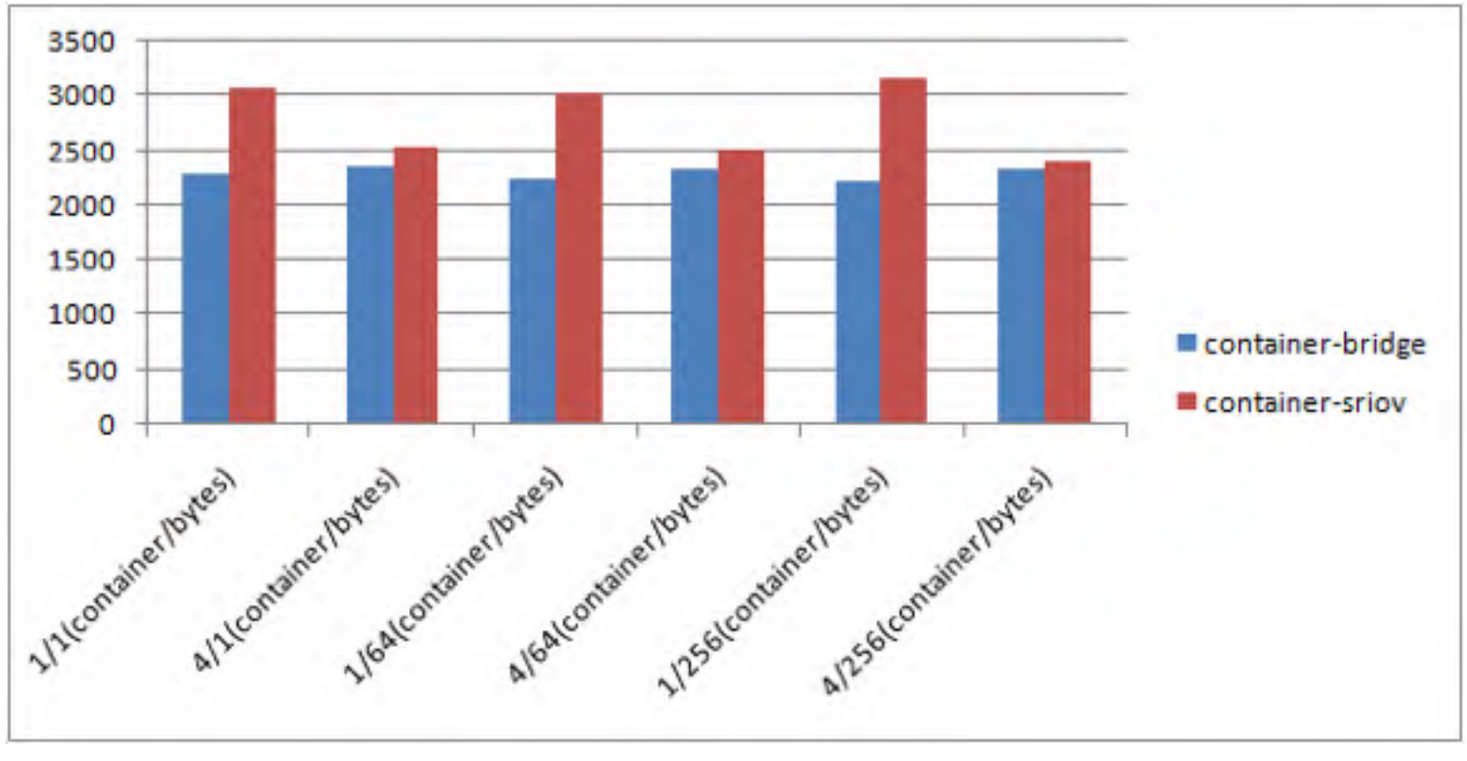
固定IP网络模式

- 为Container分配公司内网IP，使用bridge+vlan 实现docker网络插件
- 相比NAT方式，Container IP对外可见，没有端口映射带来的迁移成本
- 少了通过iptables或者用户态进程转发，性能略优于NAT方式
- 结合Gaia等上层调度系统实现了IP漂移功能，Container发生跨主机迁移时IP保持不变



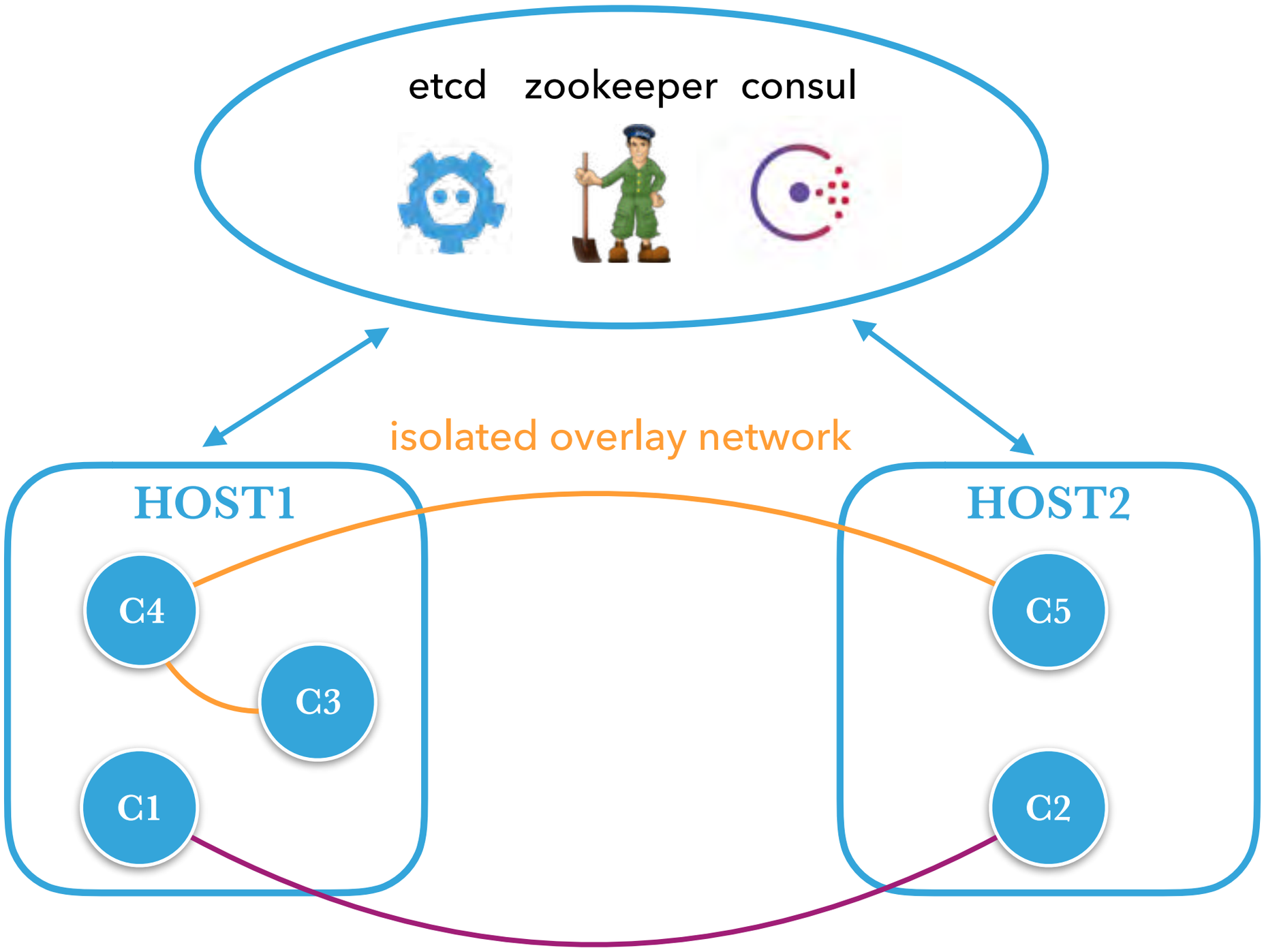
SR-IOV技术

- 硬件虚拟化技术，一个网卡硬件虚拟出多个功能接口，每个功能接口可以作为一个网卡使用。硬件取代内核的虚拟网络设备，极大的提高网络性能，并且减少了物理机的CPU消耗
- Docker 1.9.0+版本支持网络插件方式，我们实现了SR-IOV的网络插件



DOCKER OVERLAY 网络使用

- Overlay网络提供了
  - 多租户网络的隔离，独立虚拟IP段
  - 不需要分配内网IP，不依赖NAT
- libnetwork的overlay driver为接入的container默认连接了一个default_gateway的NAT网络提供外网的访问，很多container并不一定需要访问内网或者外网
- 我们给network对象增加internal开关，创建完全与外界隔离的网络，https://github.com/docker/libnetwork/pull/831


etcd
zookeeper
consul
isolated overlay network
HOST1
HOST2
C4
C3
C1
C5
C2
isolated overlay network

CONTAINER数据存储

- Container迁移后不需要保留的数据，使用host volume存储
- Container迁移后需要保留的数据，使用Ceph RBD存储
  - 使用Ceph volume plugin为每个container分配一个RBD存储目录

弹性内存控制

- 为所有的container设置一个总的内存上限hard_limit
- 单个container内存使用超出申请值时，若没有超出设置的总hard_limit值，不会触发kill
- 总内存使用接近hard_limit一定的百分比上限后，kill超出内存申请值最多的container
- 大大降低总集群container被kill的几率

网络出带宽控制

- 直接使用net_cls进行标记？数据包经过bridge后pid丢失
- 在iptables mangle表中对container的出流量按IP进行标记
- 使用TC工具对打上标记的流量进行限制

容器中资源显示问题

- 原先跑在物理机或者虚拟机中的业务使用腾讯网管系统记录机器资源使用情况
- 通过FUSE实现用户态的文件系统，使用cgroup的数据统计container的实际资源使用，生成仿真的meminfo, stats, diskstats, cpuinfo文件，bind mount到container中
- 使用go语言实现，更方便嵌入到docker代码中，整个过程对docker用户透明 https://github.com/chenchun/cgroupfs

THANKS ALL