网易蜂巢Docker研发实践

目录

- 网易蜂巢
- 蜂巢架构
- K8S 实践
- NODE
- 网络
- 负载均衡
- 容器迁移

网易蜂巢简介

- 不同于传统的IaaS服务，网易蜂巢完成了从以资源为中心的IaaS服务到以业务为中心的容器即服务（CaaS）新一代云计算的跨越，解决了用户在使用传统IaaS过程中面对的复杂的基础设施的规划、设计、安装、部署等痛点。
- 网易蜂巢提供了丰富的开发运维工具，解决了服务发现、可靠性、服务依赖、服务治理等问题，降低了运维的复杂度和难度。
- 网易蜂巢提供了一站式云计算服务，不仅包括以容器和容器编排的核心服务，还包括应用性能监控、日志管理等在内的通用运维工具链，满足了开发即运维的要求。
- 网易蜂巢构建于自研的云计算基础设施平台上，对基础设施具有完全的掌控力，研发团队据此做了大量的定制和优化。

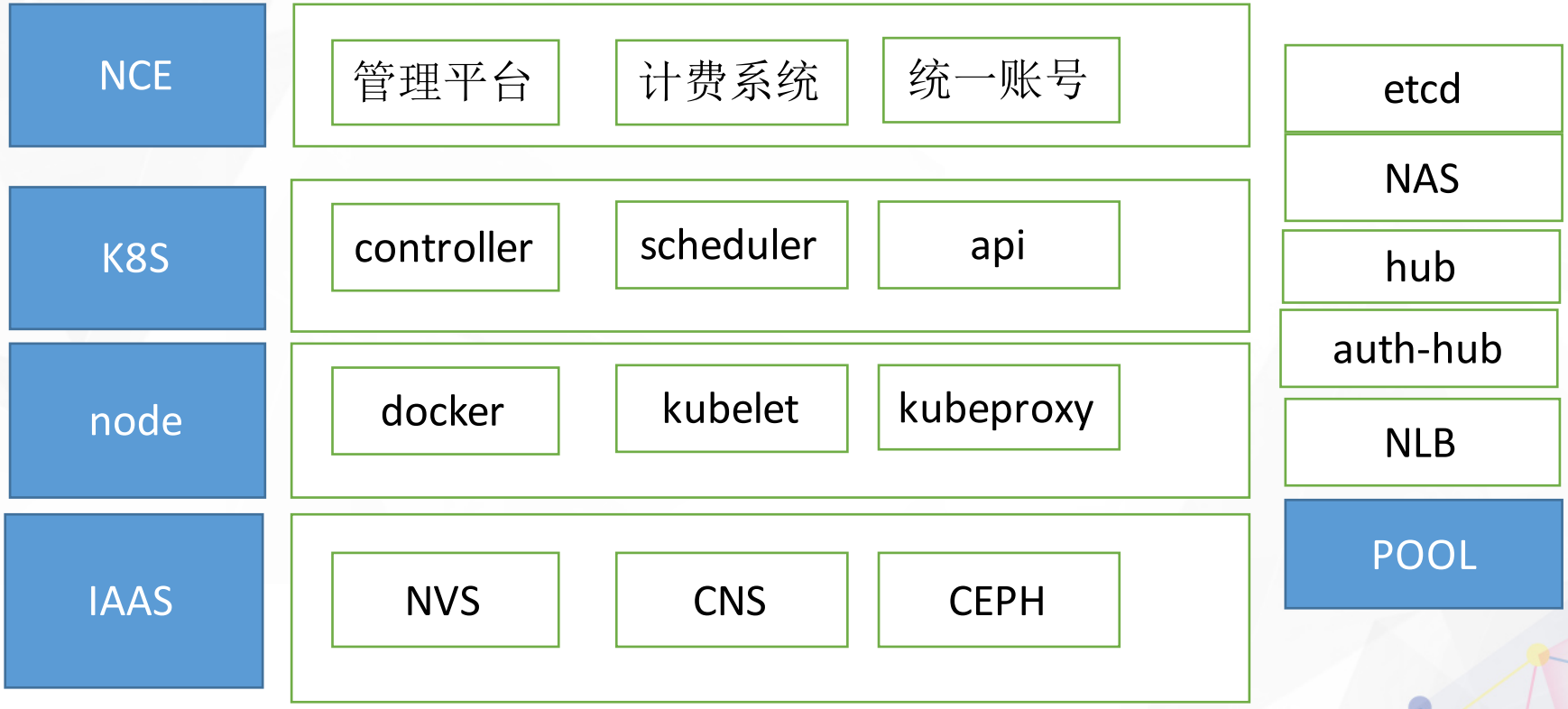
网易蜂巢解决了哪些问题，可以看两个数据：

- 网易考拉海购通过蜂巢的自动化以及微服务架构，整体运营效率提高了8倍，月发布每个月有10万次以上的发布，同时迭代效率提升了近20倍；
- 网易蜂巢的分布式架构能够实现200+的副本和每秒16000次的请求。

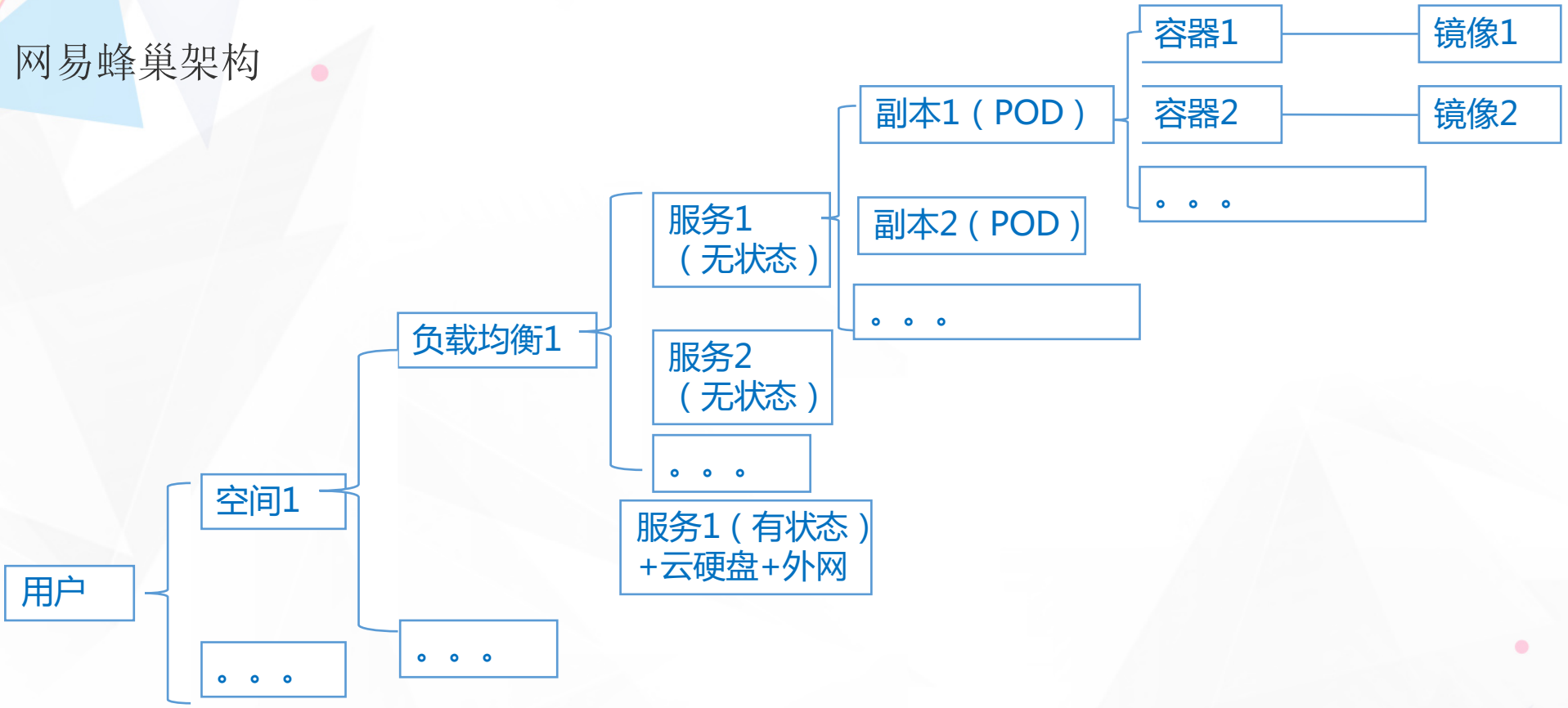
网易蜂巢对产品迭代速率的提升不言而喻。网易蜂巢目前已经支持了网易云音乐、网易考拉海购、网易云课堂等知名互联网产品。



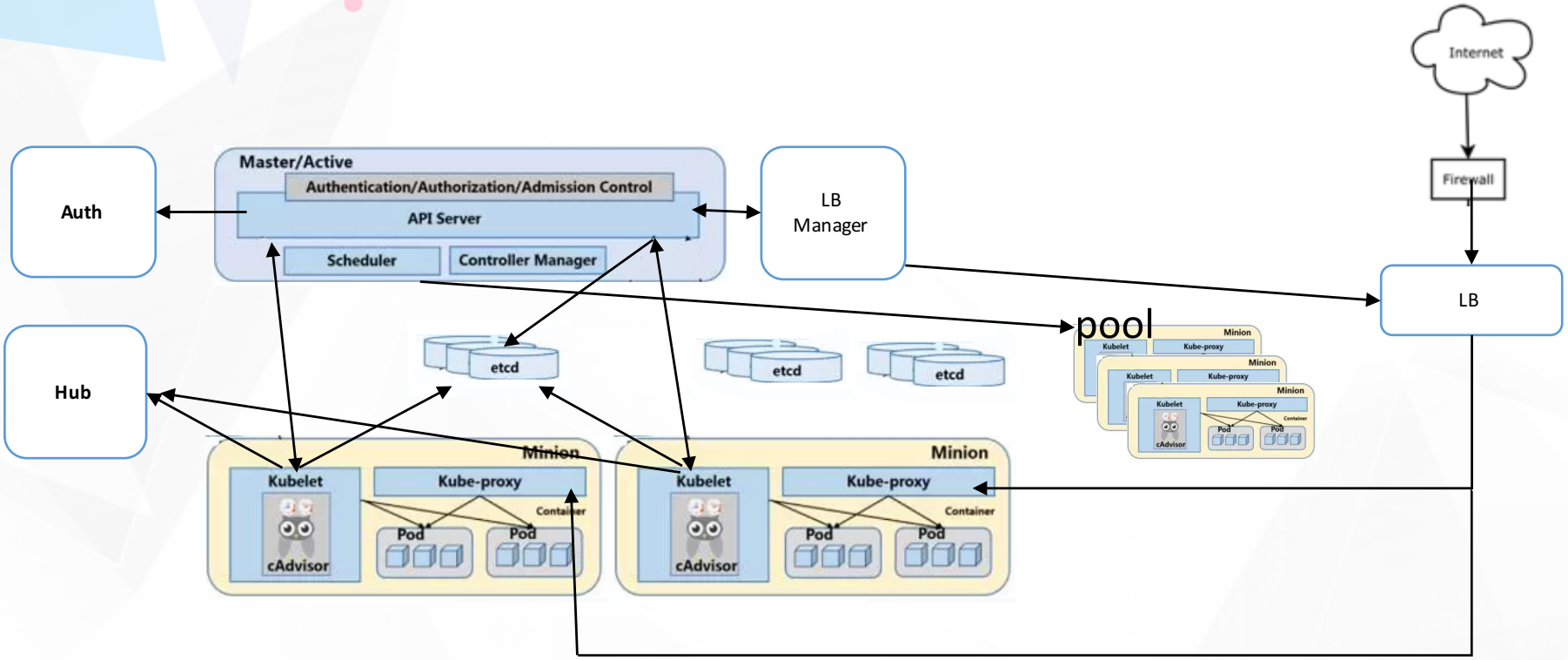

网易蜂巢架构


NCE
管理平台
计费系统
统一账号
K8S
controller
scheduler
api
node
docker
kubelet
kubeproxy
IAAS
NVS
CNS
CEPH
etcd
NAS
hub
auth-hub
NLB
POOL

网易蜂巢架构

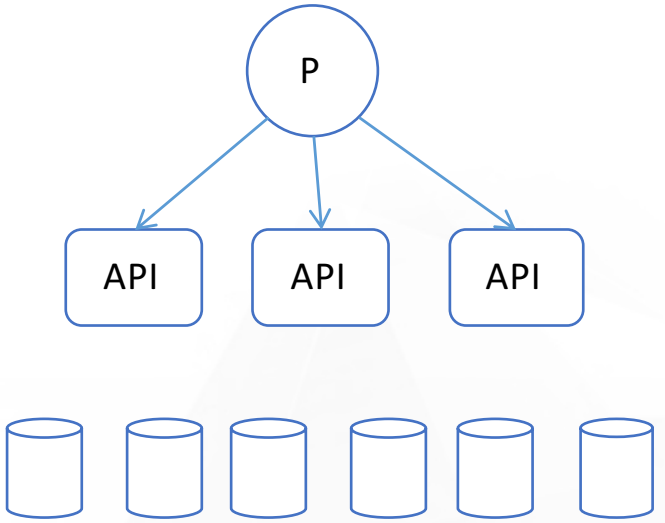

用户
空间1
。。。
负载均衡1
。。。
服务1（有状态）+云硬盘+外网
服务1（无状态）
服务2（无状态）
。。。
副本1（POD）
副本2（POD）
。。。
容器1
容器2
。。。
镜像1
镜像2

- K8S 实践

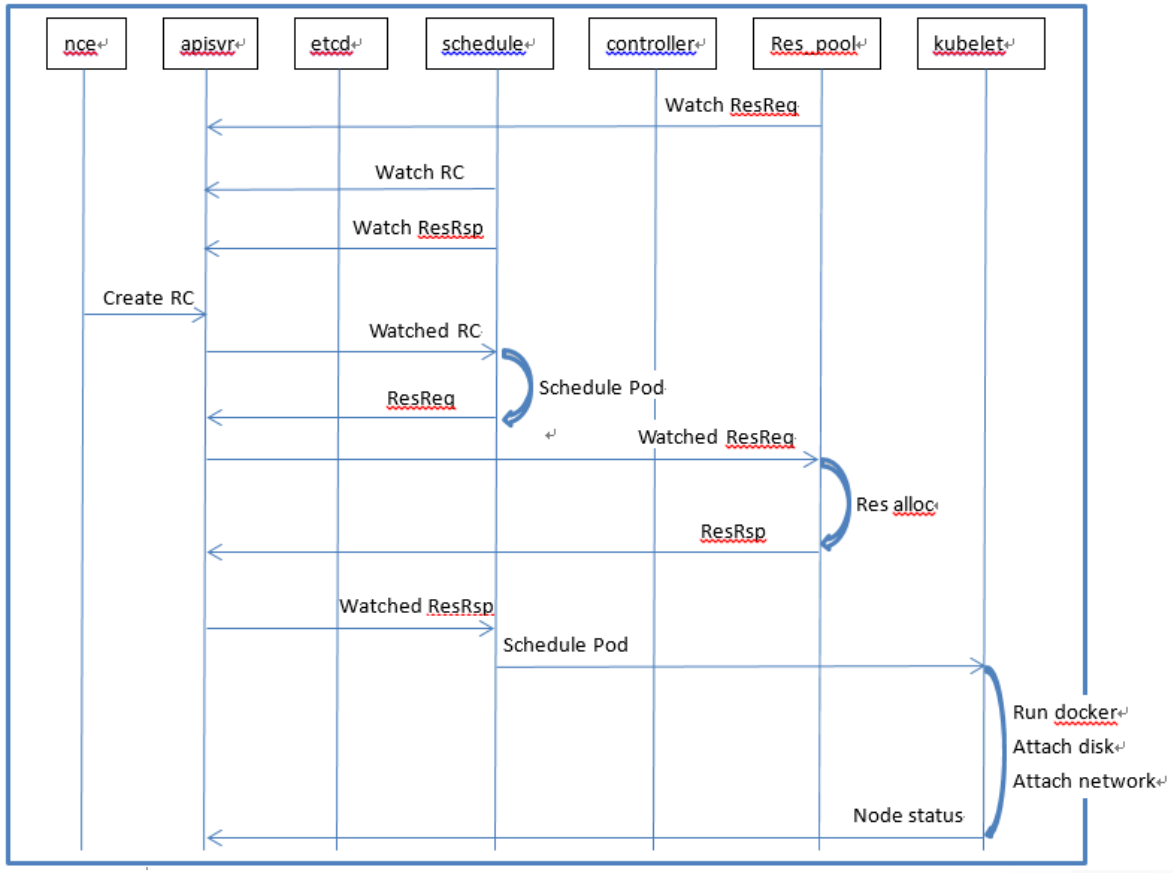

Internet
Firewall
LB
Auth
Master/Active
Authentication/Authorization/Admission Control
API Server
Scheduler
Controller Manager
LB
Manager
pool
Hub
etcd
etcd
etcd
Minion
Kubelet
Kube-proxy
Container
Pod
Pod
cAdvisor
Minion
Kubelet
Kube-proxy
Container
Pod
Pod
cAdvisor

- K8S 实践

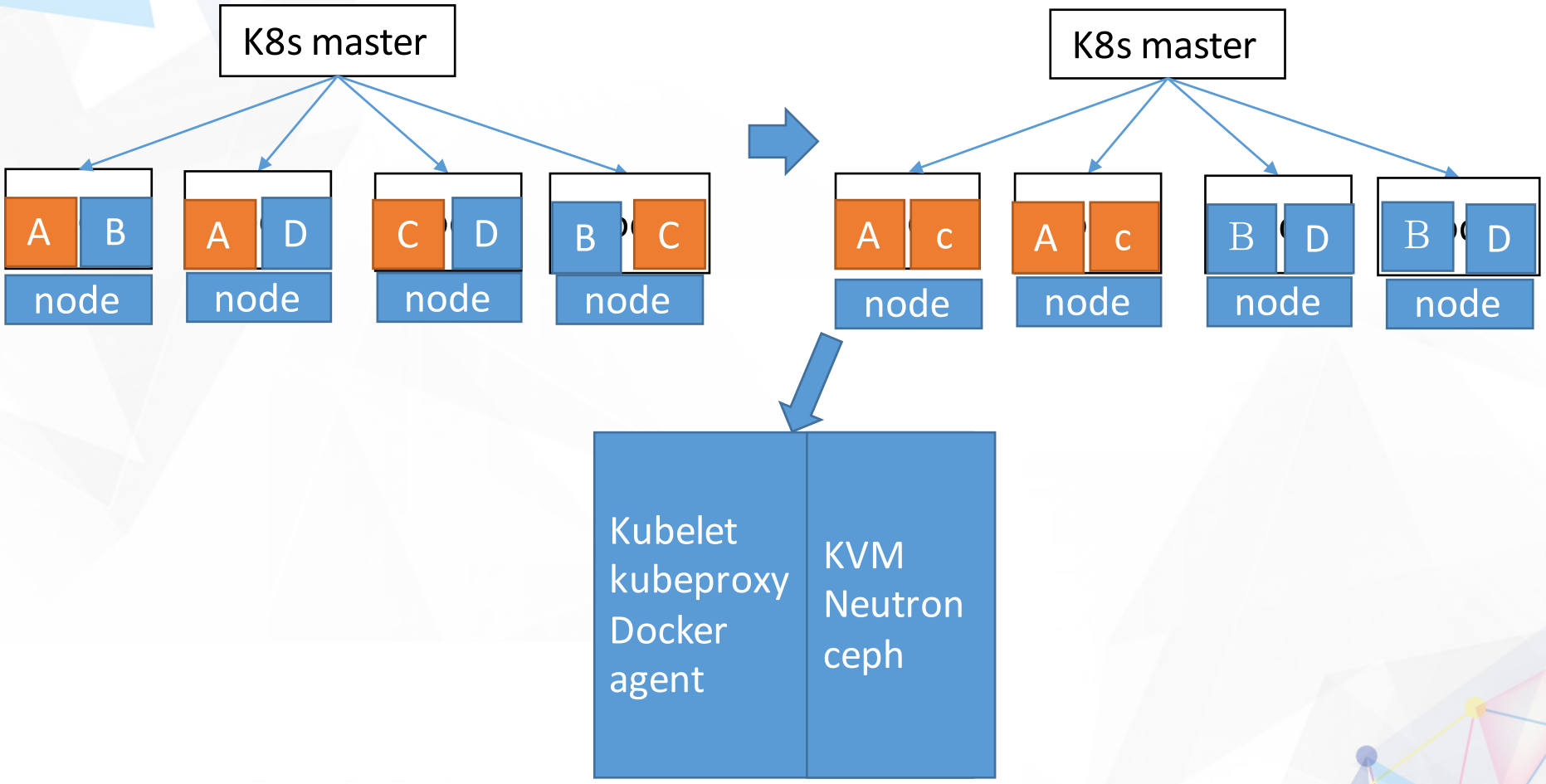
  - 完善多租户支持
    - node、存储、网络等资源租户天生隔离
    - 按租户实现独立的认证和授权
    - Kubeproxy 按租户隔离
  - 调度器/控制器并行处理优化
    - 将面向集群的串行调度优化为多租户并行调度
    - 将副本队列串行处理优化为按照多优先级队列并行处理
  - 实现资源按需分配
  - etcd集群
    - 根据Pod/Node/RC等资源到拆分不同的etcd集群



- K8S 实践

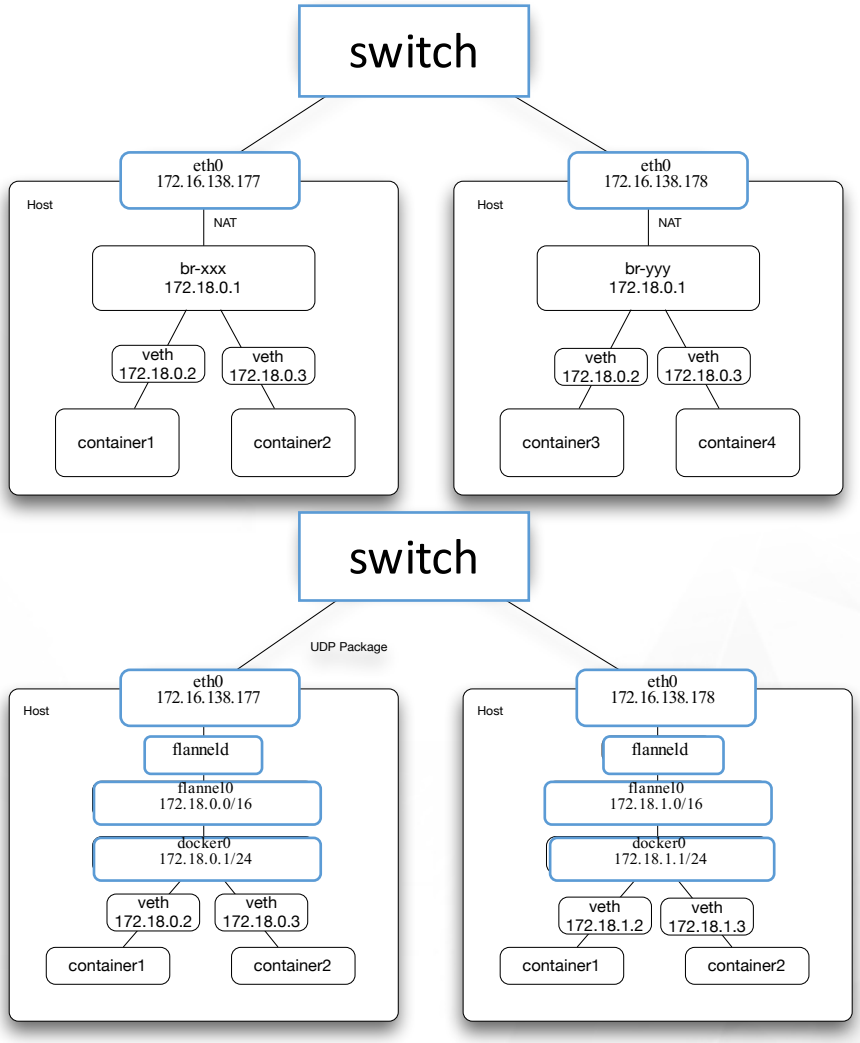

nce
apisvr
etcd
schedule
controller
Res_pool
kubelet
Watch ResReq
Watch RC
Watch ResRsp
Create RC
Watched RC
Schedule Pod
ResReq
Watched ResReq
Res alloc
ResRsp
Watched ResRsp
Schedule Pod
Run docker
Attach disk
Attach network
Node status

NODE


K8s master
A
B
node
A
D
node
C
D
node
B
C
node
K8s master
A
c
node
A
c
node
B
D
node
B
D
node
Kubelet
kubeproxy
Docker
agent
KVM
Neutron
ceph

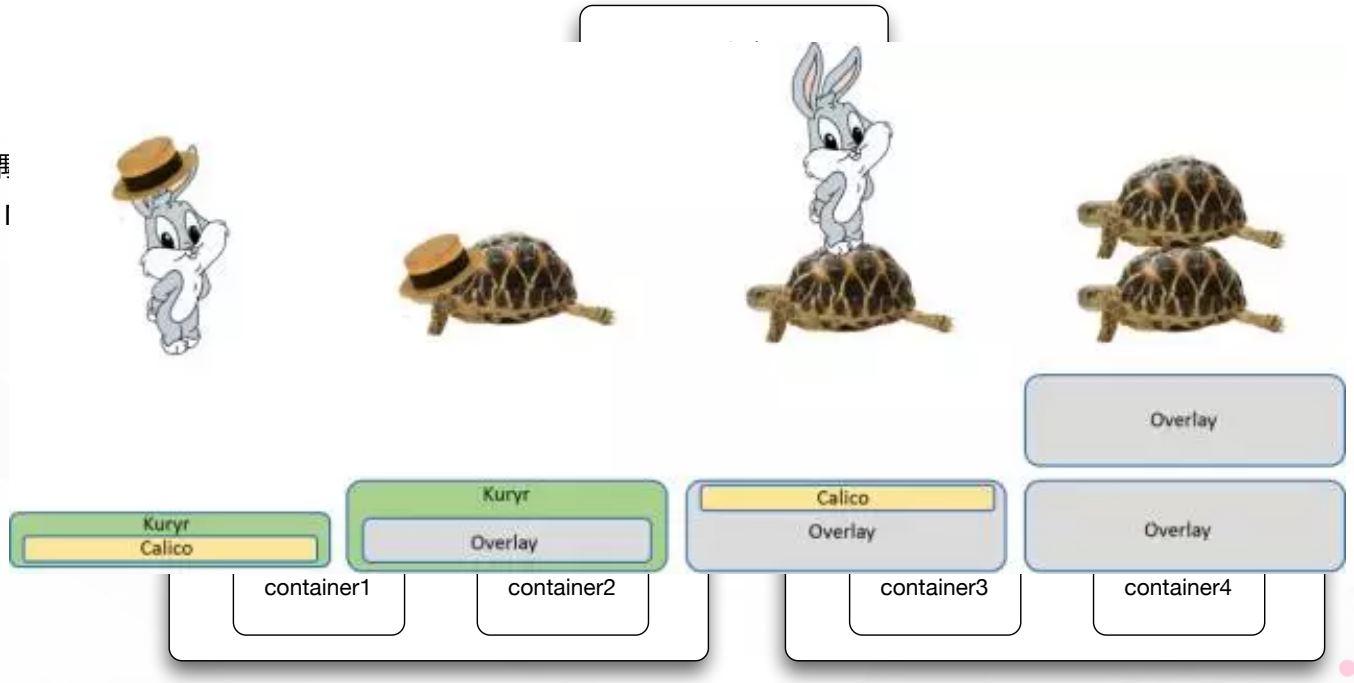
网络

- 容器间互连
  - 复杂：NAT、端口映射、层级网络
    - 遗留系统迁移的影响
      - 长连接状态问题
      - 基于 IP 注册的服务发现
    - 运维复杂度增加
      - 端口冲突
      - 内外IP 不一致
    - 不利于故障恢复
      - IP 变化

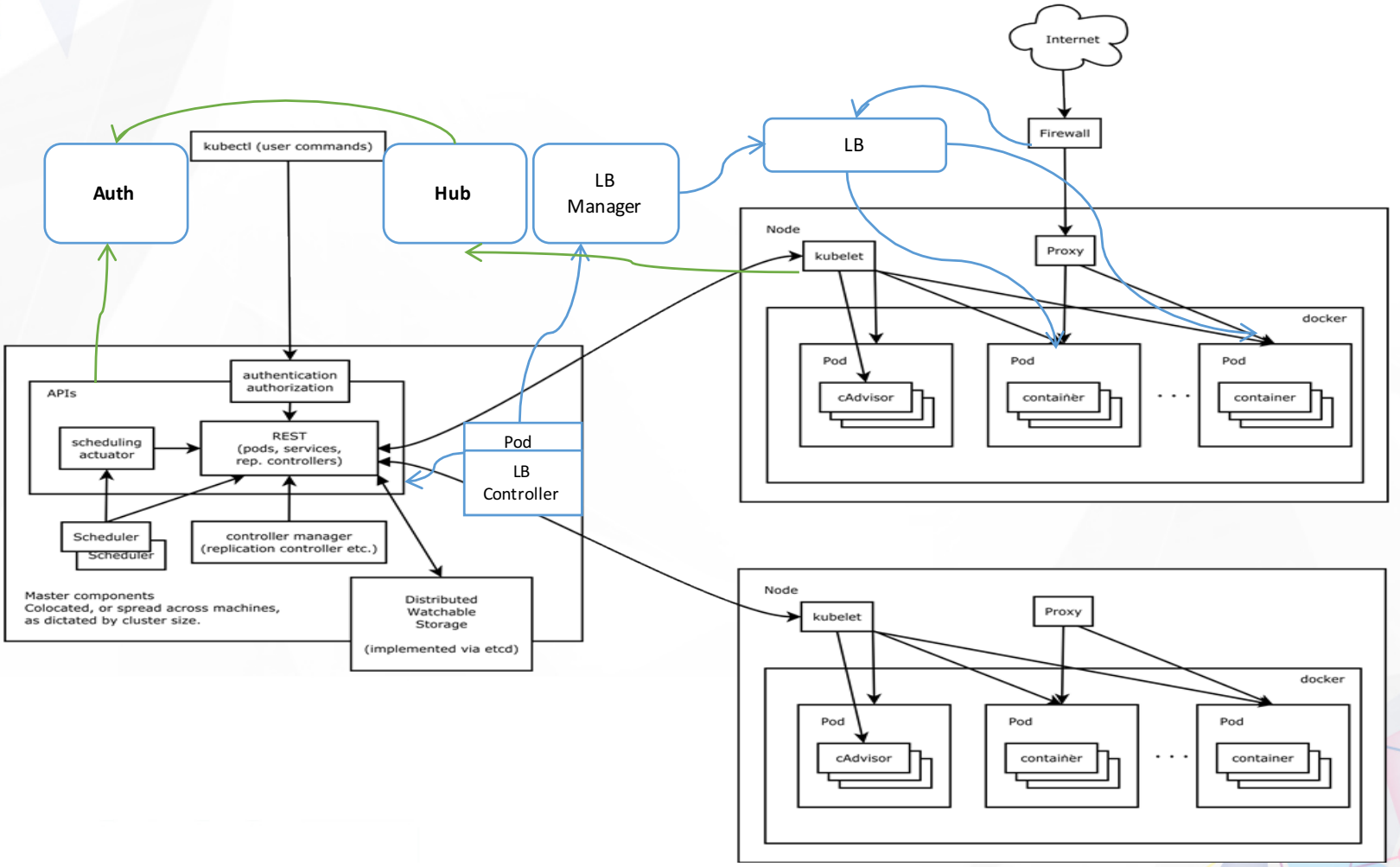


网络

- 简单：虚拟化扁平二层网络
  - 遗留系统兼容性好
  - 控制 IP 分配
    - 利于故障恢复
    - 与物理网络拓扑解耦
- VxLAN 网络，基于 Openstack I
- 每租户一张独立私有二层网络
- 外网网卡直接挂载
- 私有网络 > 容器网络



负载均衡



容器迁移

Docker daemon 在启动时，通过–graph=指定docker运行时根目录。
Docker根目录结构：

```
├── containers   配置相关文件
├── graph        镜像相关文件
├── overlay      文件系统相关数据
└── volumes      卷相关目录
  ...
```

Docker daemon 启动时，会轮询containers 目录，加载容器配置信息，当启动完成后，通过docker ps -a 就可以看到该node上的所有容器：

```
03a09a384cecf8921f1b0171776991c340555494f80627a395e0492b4189c5c9
257c33a74a64d696759ba371aa646a5e7037c18690fbe8f67cc1a62d8983120a
...
```

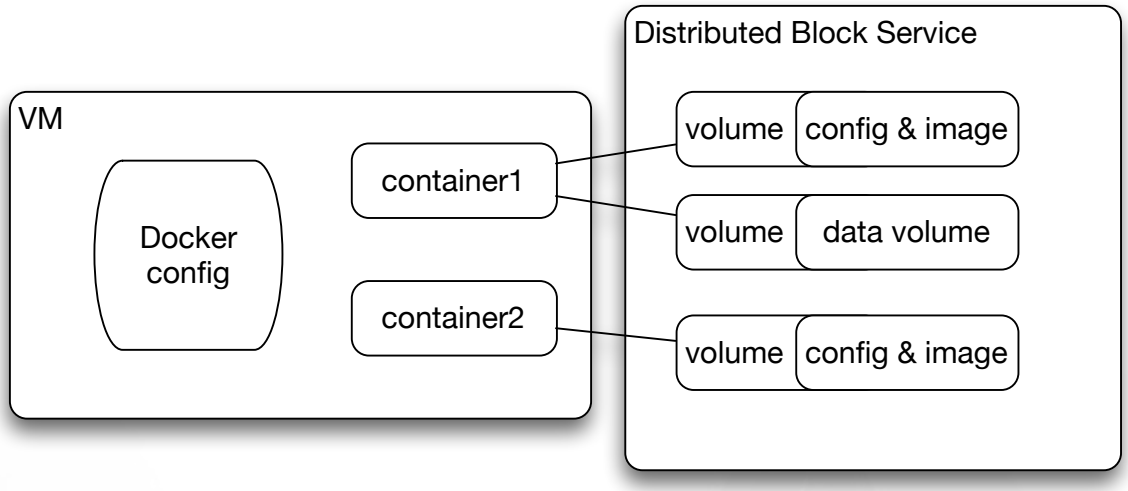
每个容器都有一个目录，目录结构如下：

```
├── 03a09a384cecf8921f1b0171776991c340555494f80627a395e0492b4189c5c9-json.log
├── config.json
├── hostconfig.json
├── hostname
├── hosts
├── resolv.conf
└── resolv.conf.hash
```

容器迁移

- docker run 增加—container-home=$dir 参数，指定用户独立的rootfs路径
- 将Daemom中全局的变量移到container结构中。

```
type Daemon struct {
        ID                  string
        repository          string
        containers          container.Store
        ...
        volumes             *store.VolumeStore
        ...
        shutdown            bool
        ...
        layerStore          layer.Store
}
```



功能：

指定该参数后，容器文件系统相关数据及数据卷都存放在指定的dir目录下。在实践中，kubelet 中的netease插件将云盘挂载到node上某一mountpoint，并指定container-home为该mountpoint。从而实现每个容器的rootfs保存在自己独立的云盘上。指定container-home参数后，对于用户通过–v /dir指定的匿名卷，也会保存在这个云盘上。

容器迁移

- 增加reload 指令
  通过模仿daemon.restore 方法，轮询containers 目录，加载未加载过得容器配置信息

功能：
执行docker reload 指令，无需重启daemon、dockerd，就可以看到新增加容器。

迁移流程：
1. 在一个node上挂载云硬盘到指定dir目录
2. 启动容器，设置container-home=dir将容器数据保存在dir目录
3. 当node宕机或需要迁移该容器时
4. kubelet将该容器配置信息copy到新的node containers目录
5. kubelet挂载云硬盘到新node dir目录，将attach容器网络。
6. 执行docker reload后docker ps –a可以看到容器
7. 执行docker start 就可以启动容器，迁移成功

THANKS

SequeMedia 盛拓传媒 | IT168.com 专注导购16年 | ChinaUnix.net | ITPUB www.itpub.net