极客时间

构建微服务云原生应用

架构师杨波

极客时间

扫码试看/订阅

《Spring Boot & Kubernetes 云原生微服务实践》

极客时间

CHAPTER 9

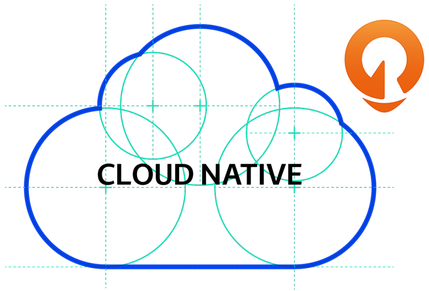
云原生架构和 Kubernetes 容器云部署

极客时间

第 1 部分

到底什么是云原生架构？

云原生（Cloud Native）应用定义



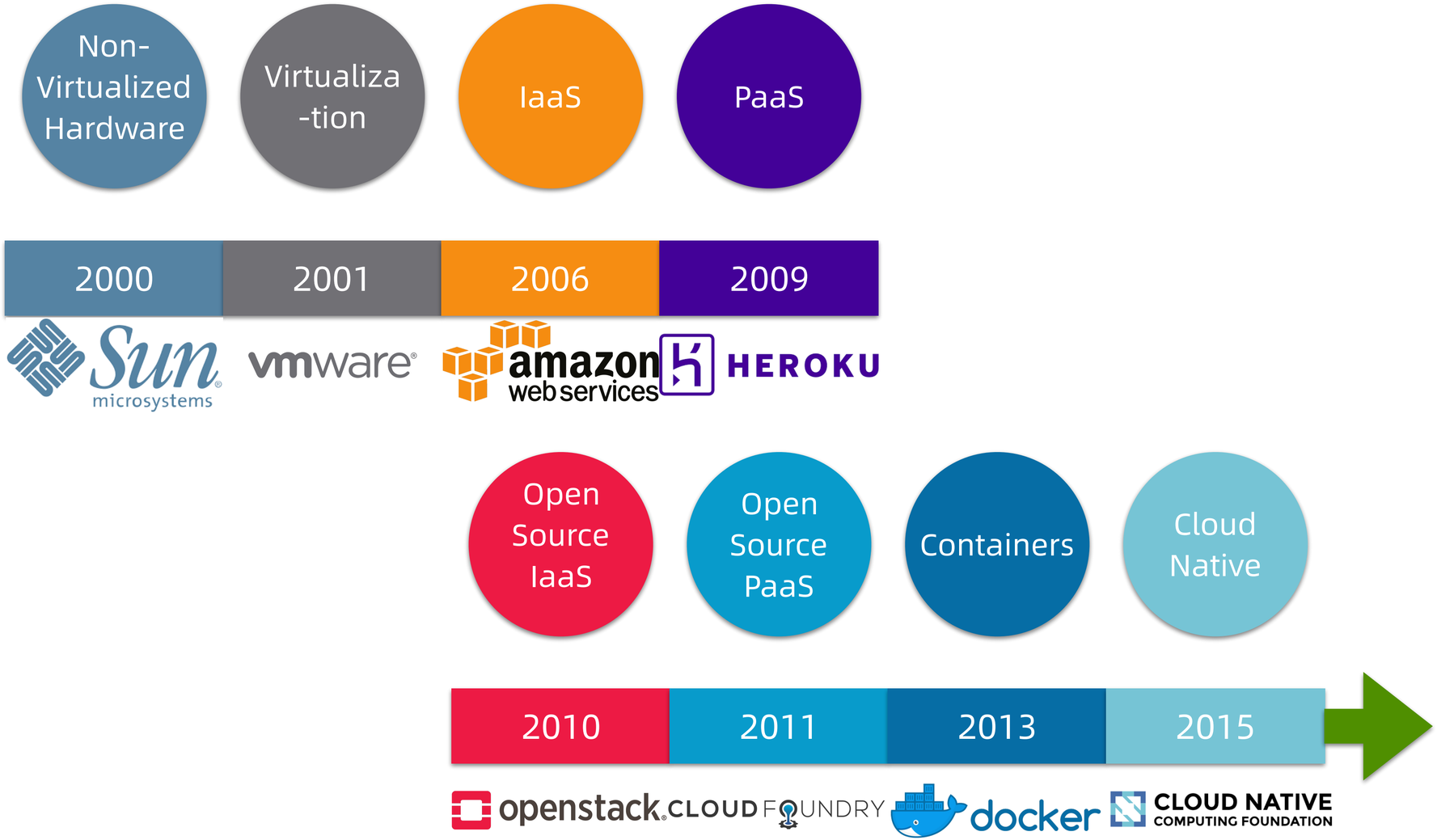
极客时间

- Applications adopting the principles of Microservices packaged as Containers orchestracted by Platforms running on top of Cloud infrastructure, developed using practices such as Continous Delivery and DevOps.

- 基于微服务原理而开发的应用，以容器方式打包。在运行时，容器由运行于云基础设施之上的平台进行调度。应用开发采用持续交付和 DevOps 实践。

https://www.slideshare.net/bibryam/designing-cloud-native-applications-with-kubernetes

云演进史


Non-Virtualized Hardware
Virtualiza-tion
IaaS
PaaS
2000
2001
2006
2009
Sun microsystems
vmware
amazon web services
HEROKU
Open Source IaaS
Open Source PaaS
Containers
Cloud Native
2010
2011
2013
2015
openstack
CLOUD FOUNDRY
docker
CLOUD NATIVE COMPUTING FOUNDATION

CNCF

cncf.io

landscape.cncf.io

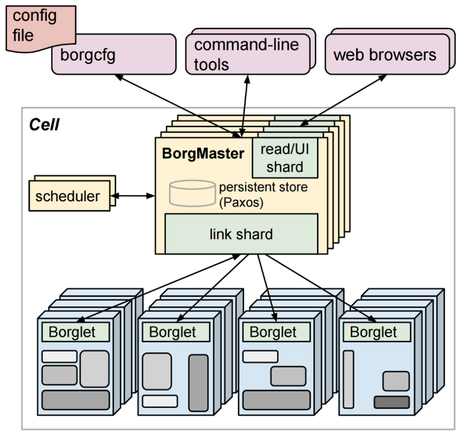
极客时间

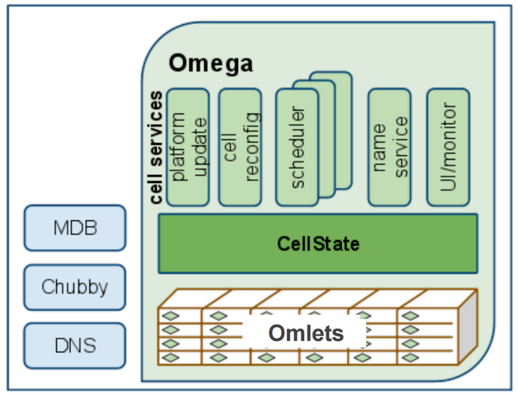
第 2 部分

Kubernetes 背景和架构

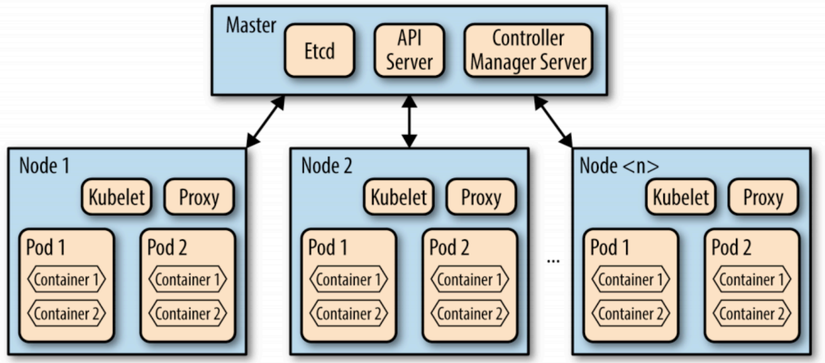
Kubernetes 历史

2003/2004





2014



master-slave架构

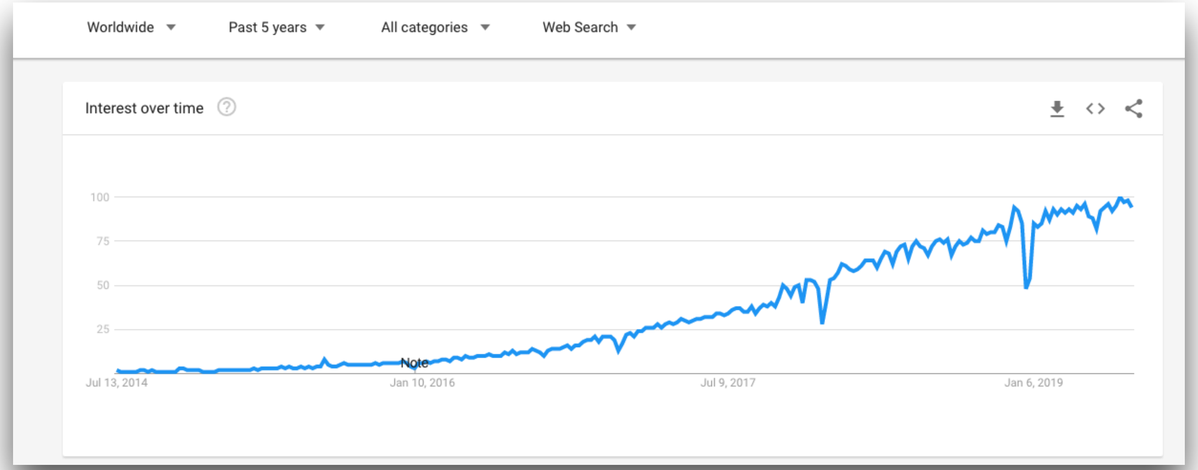
https://medium.com/containermind/a-new-era-of-container-cluster-management-with-kubernetes-cd0b804e1409

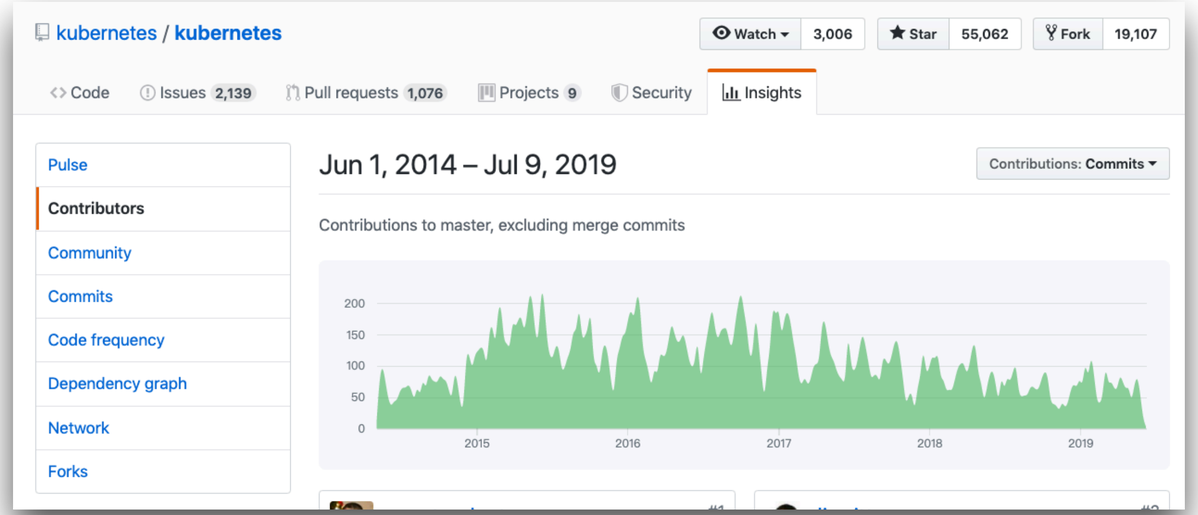
Google 特有的方式

从 Gmail 到 YouTube 和 Google 搜索，Google 的所有产品都是在容器中运行。容器化让我们的开发团队可以快速行动、高效部署软件，并以前所未有的规模运营。每个星期我们都要启动超过 20 亿个容器。这十年来，我们不但积累了有关如何在生产环境中运行容器化工作负载的丰富知识，而且一直与社区积极分享这些知识：从早期为 Linux 内核贡献 cgroups，到在 Kubernetes 项目中开放我们内部工具所采用的设计源代码，均是如此。我们将这些专业知识融入 Google Cloud Platform，使任何规模的开发者和企业都能轻松利用最新的容器技术创新成果。

https://cloud.google.com/containers/?hl=zh-cn

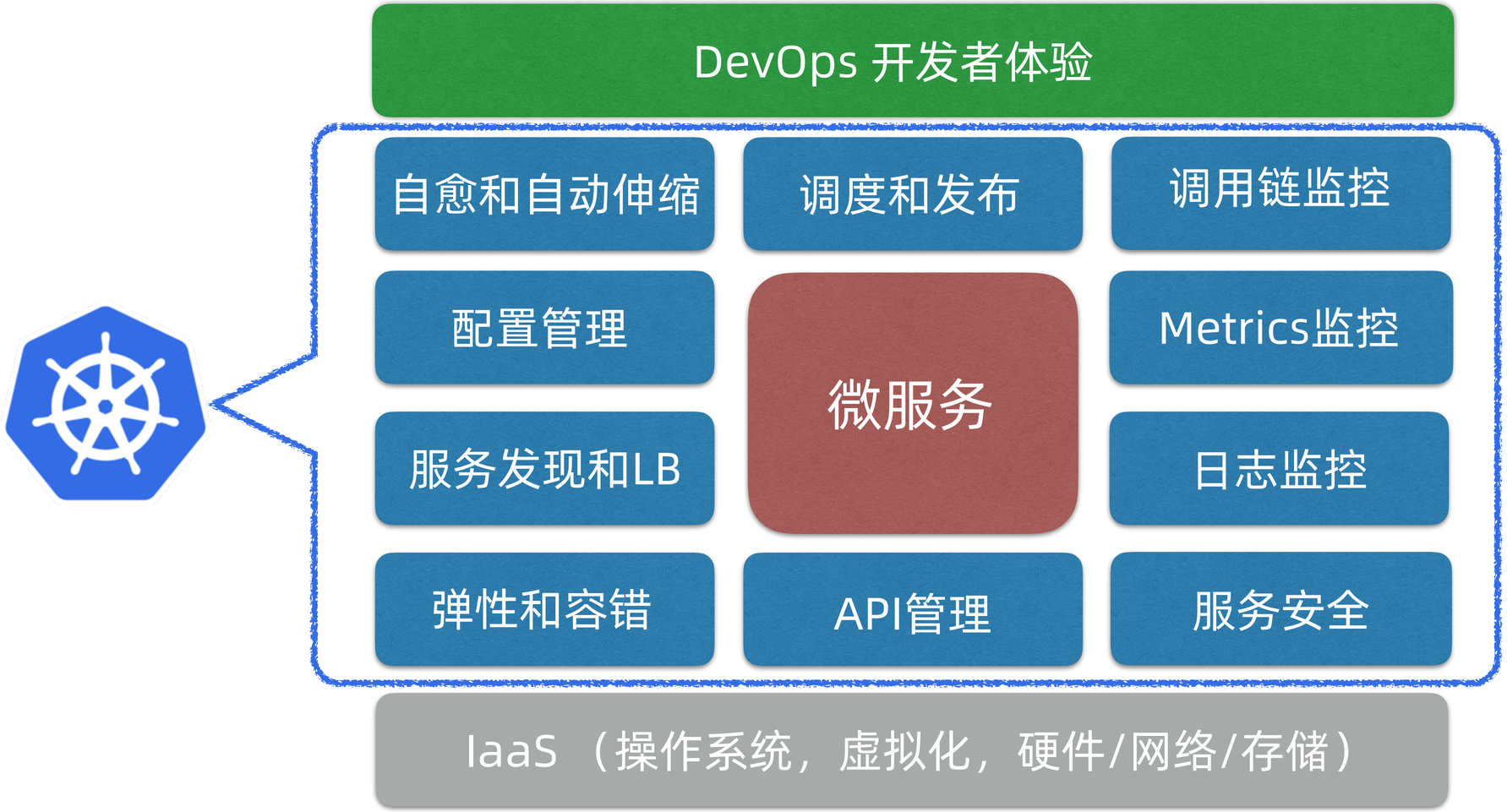
Kubernetes 趋势



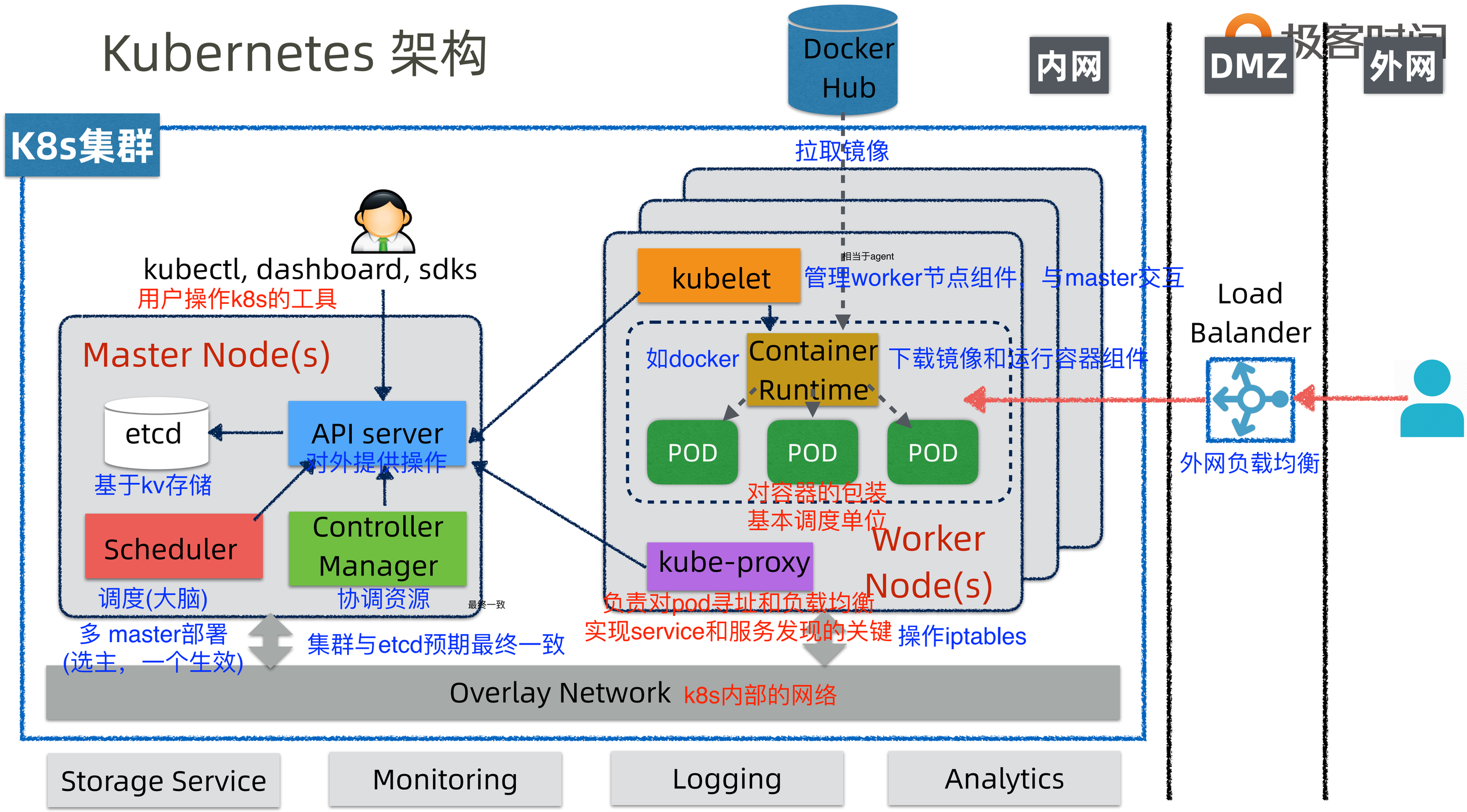


Kubenetes 解决什么问题？

简化微服务的管理和部署



Kubernetes 架构

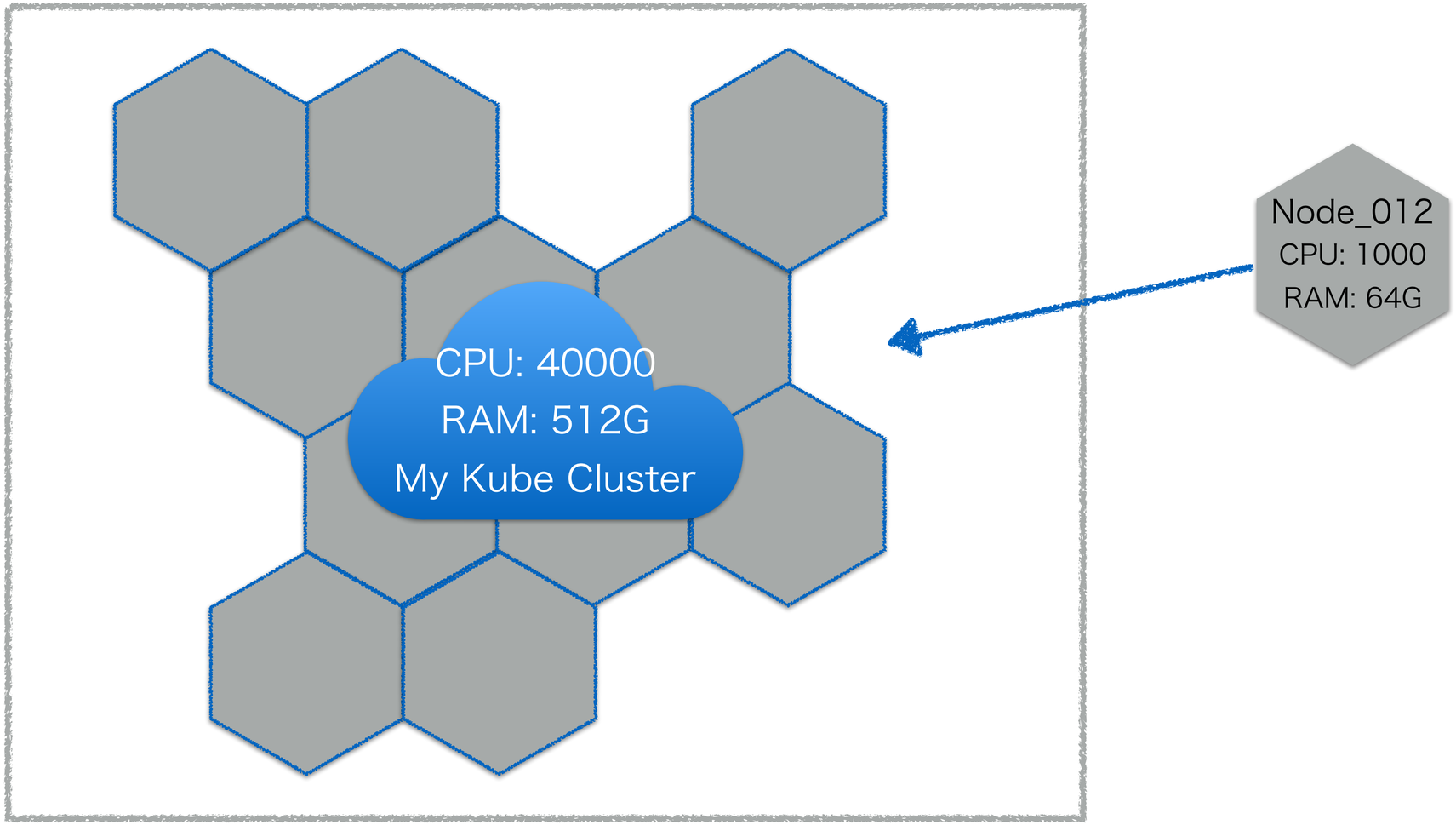


第 3 部分

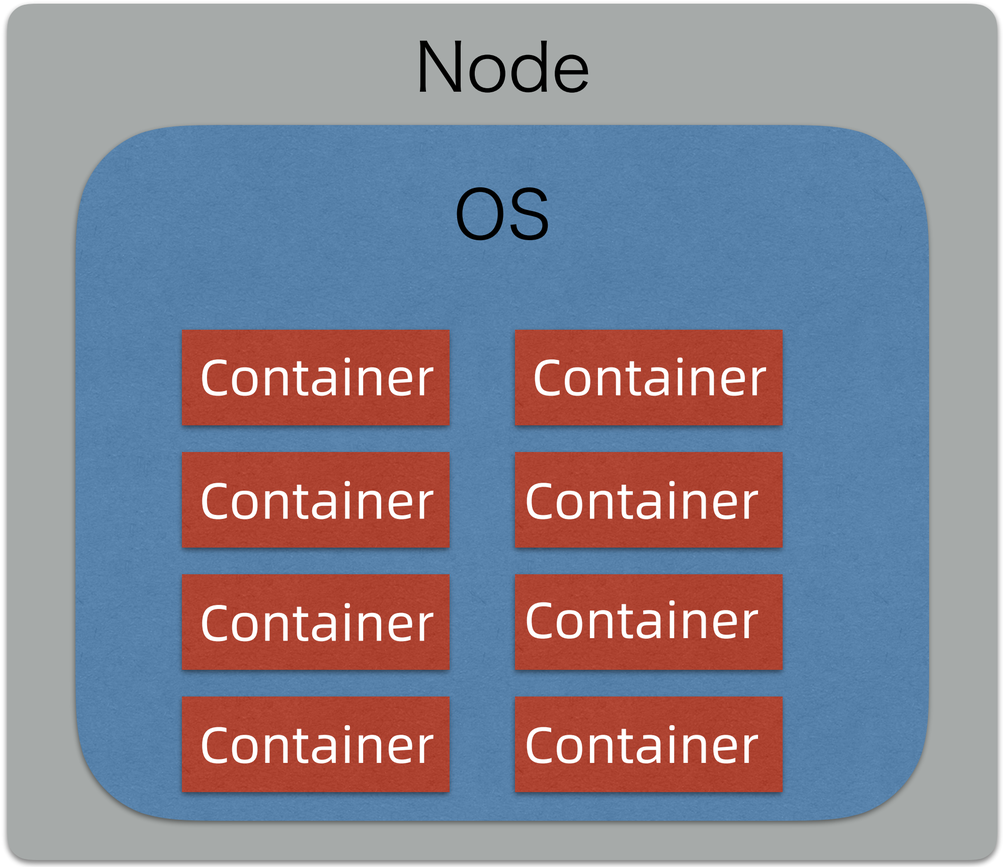
Kubernetes 有哪些基本概念？

Kubernetes 集群（Cluster）

多个节点，按需添加



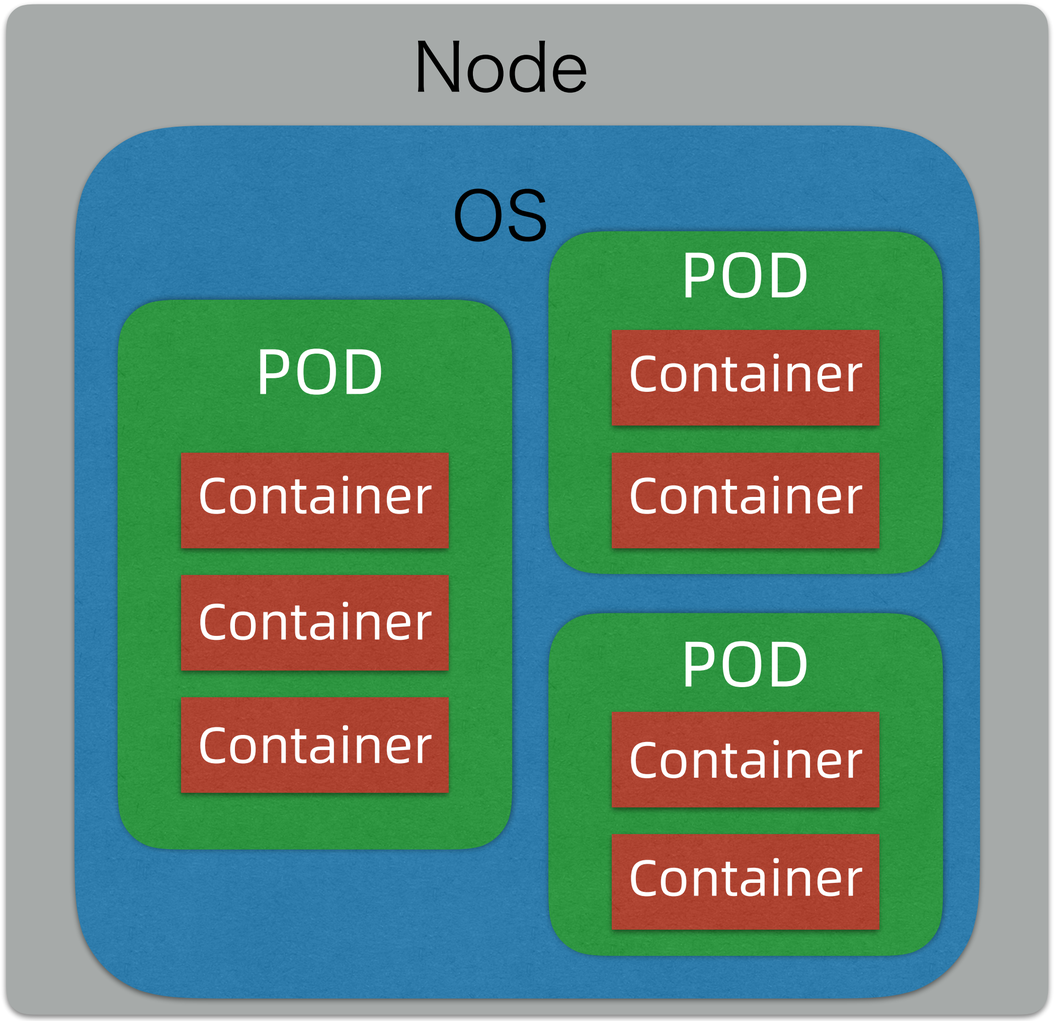
容器 Container



容器宿主机角度:一个个进程
自身角度:操作系统

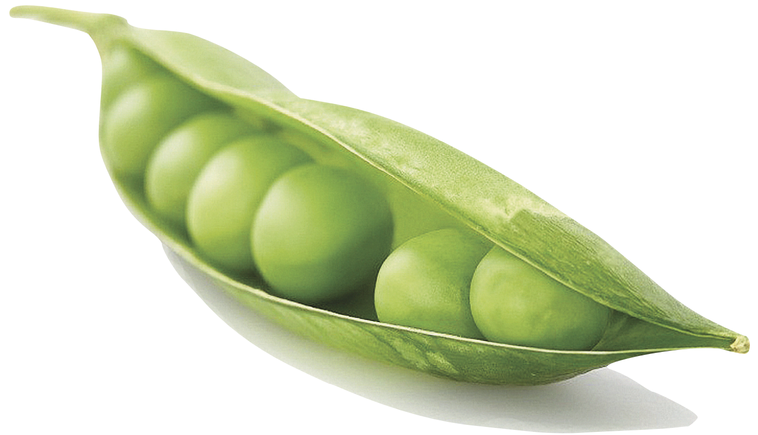
POD

k8s调度的基本单位:
同一个pod内的容器可以相互访问，共享文件



一般一个应用一个pod

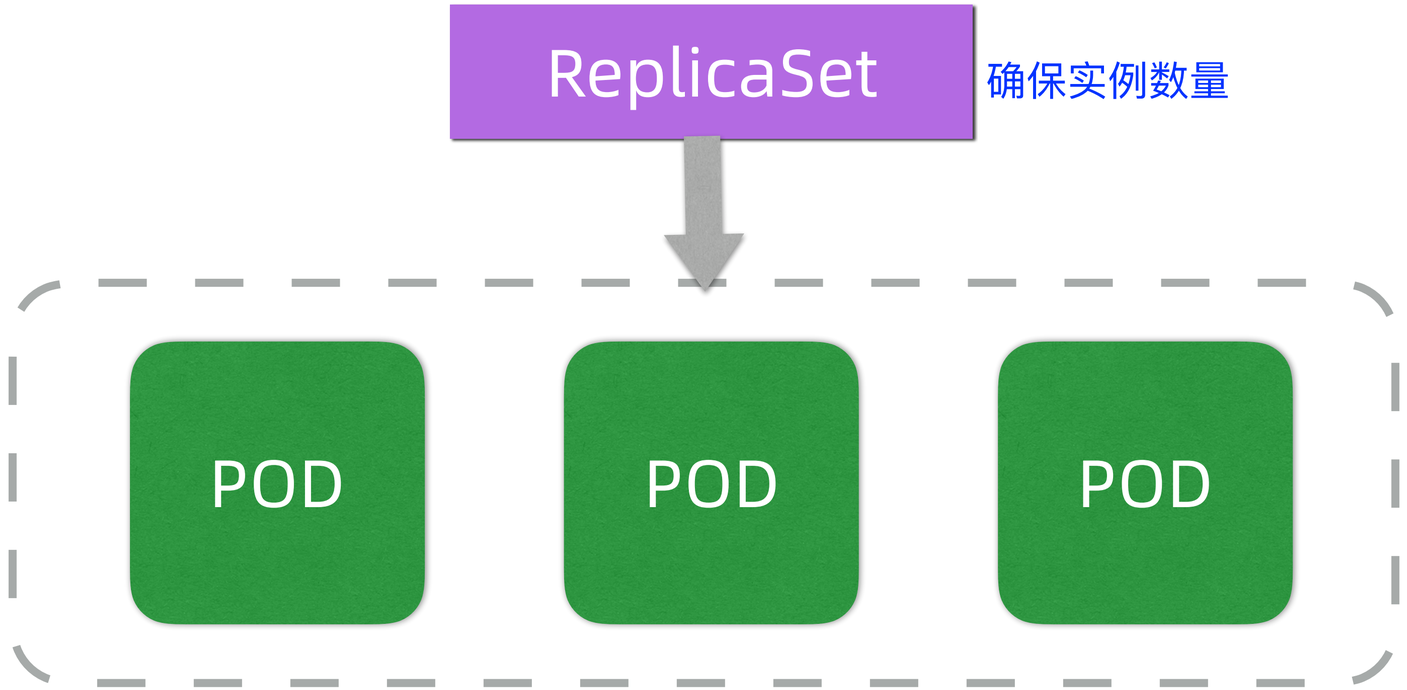
同一个pod共享ip,有独立ip



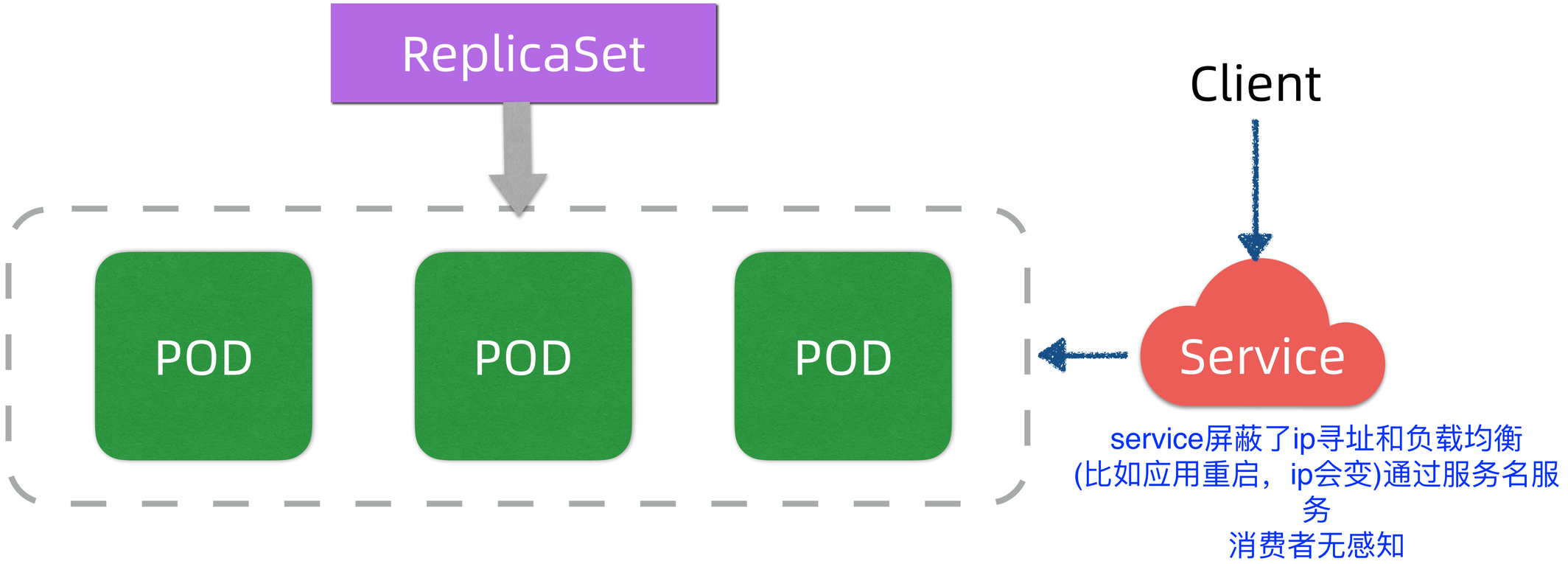
副本集 ReplicaSet

一个应用发布多个pod,实现高可用

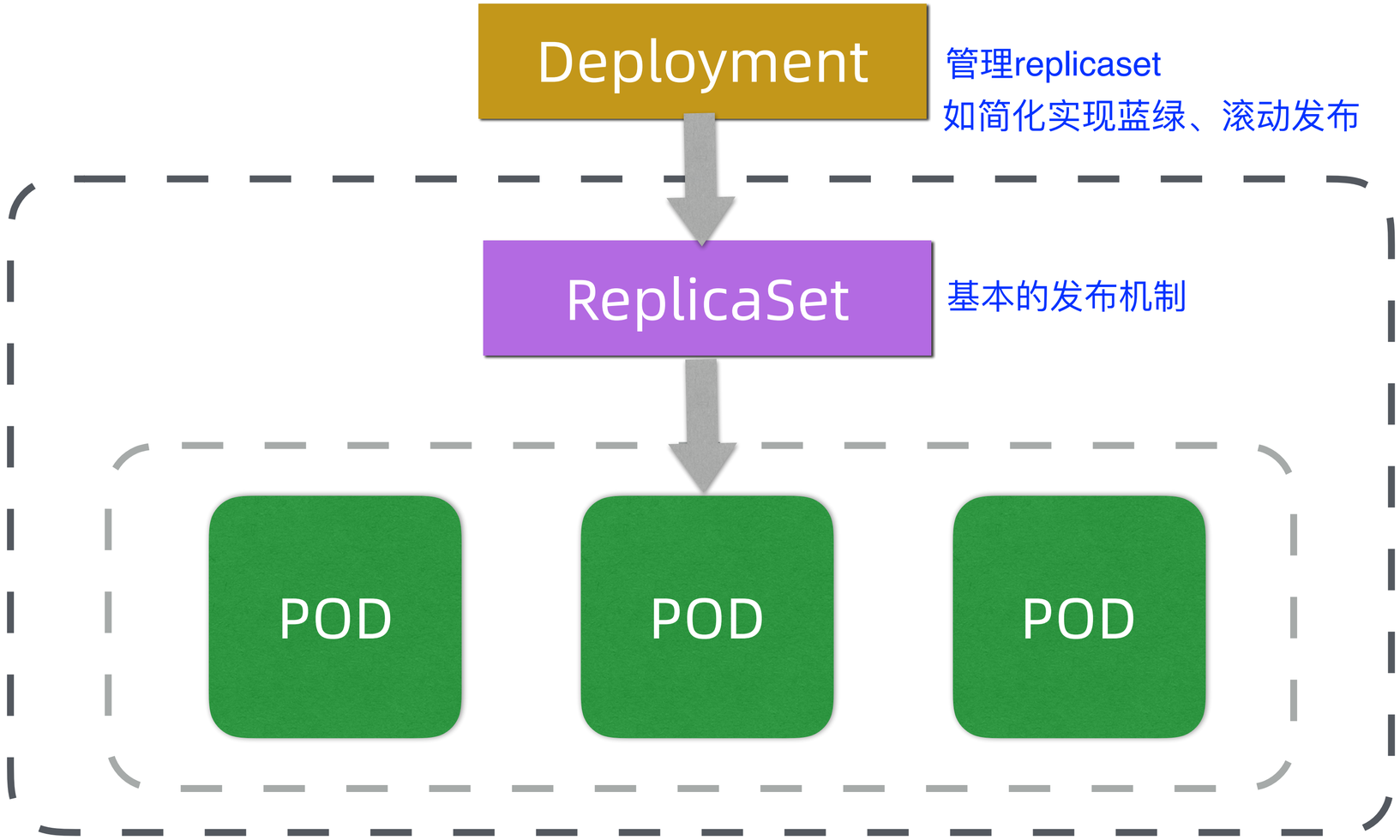
副本集就是和一个应用多个pod,相对应，是一个模版（yaml/json)



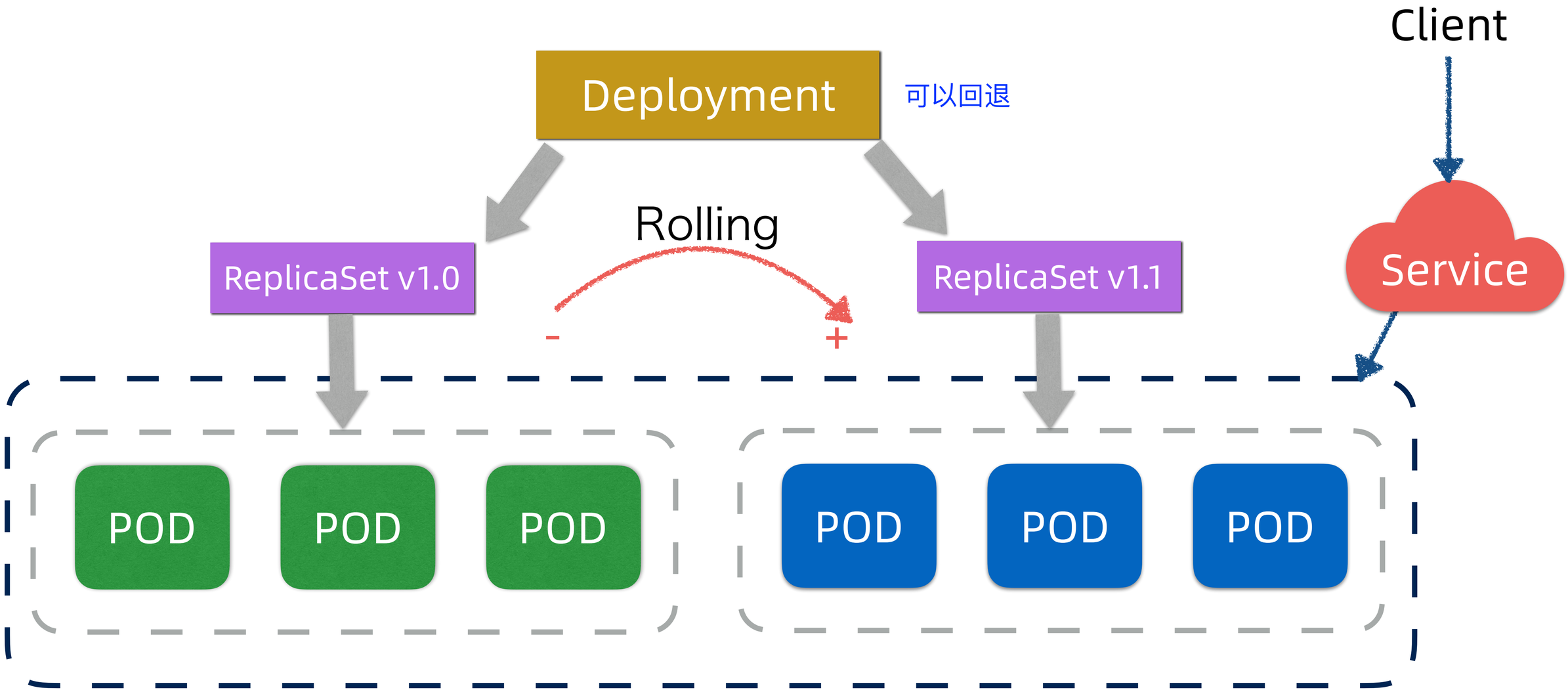
服务 Service


ReplicaSet
POD
POD
POD
Client
Service
service屏蔽了ip寻址和负载均衡
(比如应用重启，ip会变)通过服务名服务
消费者无感知

发布 Deployment

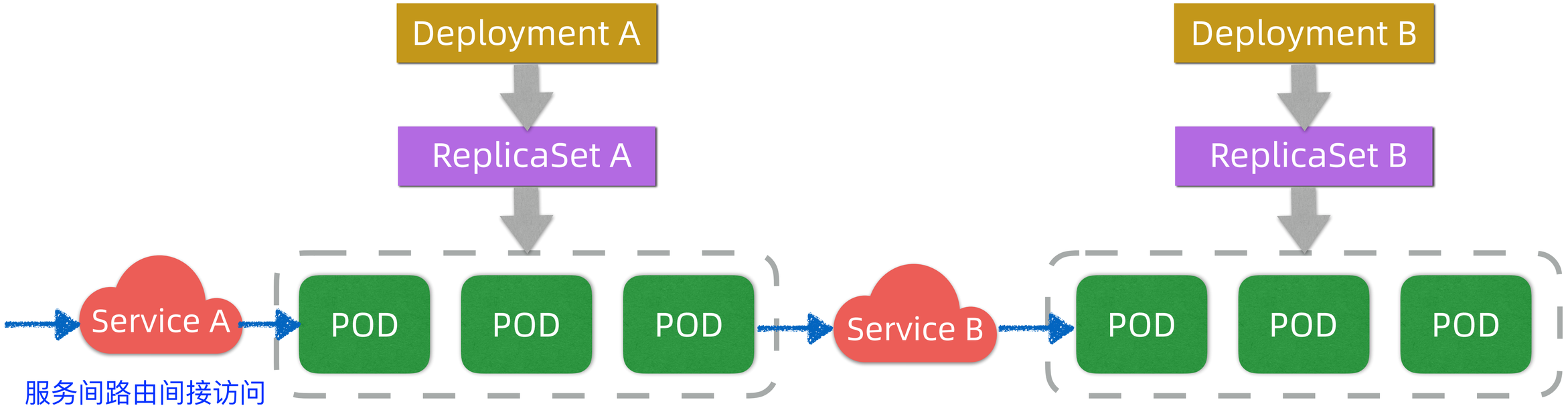


滚动发布 Rolling Update



发布和服务总结

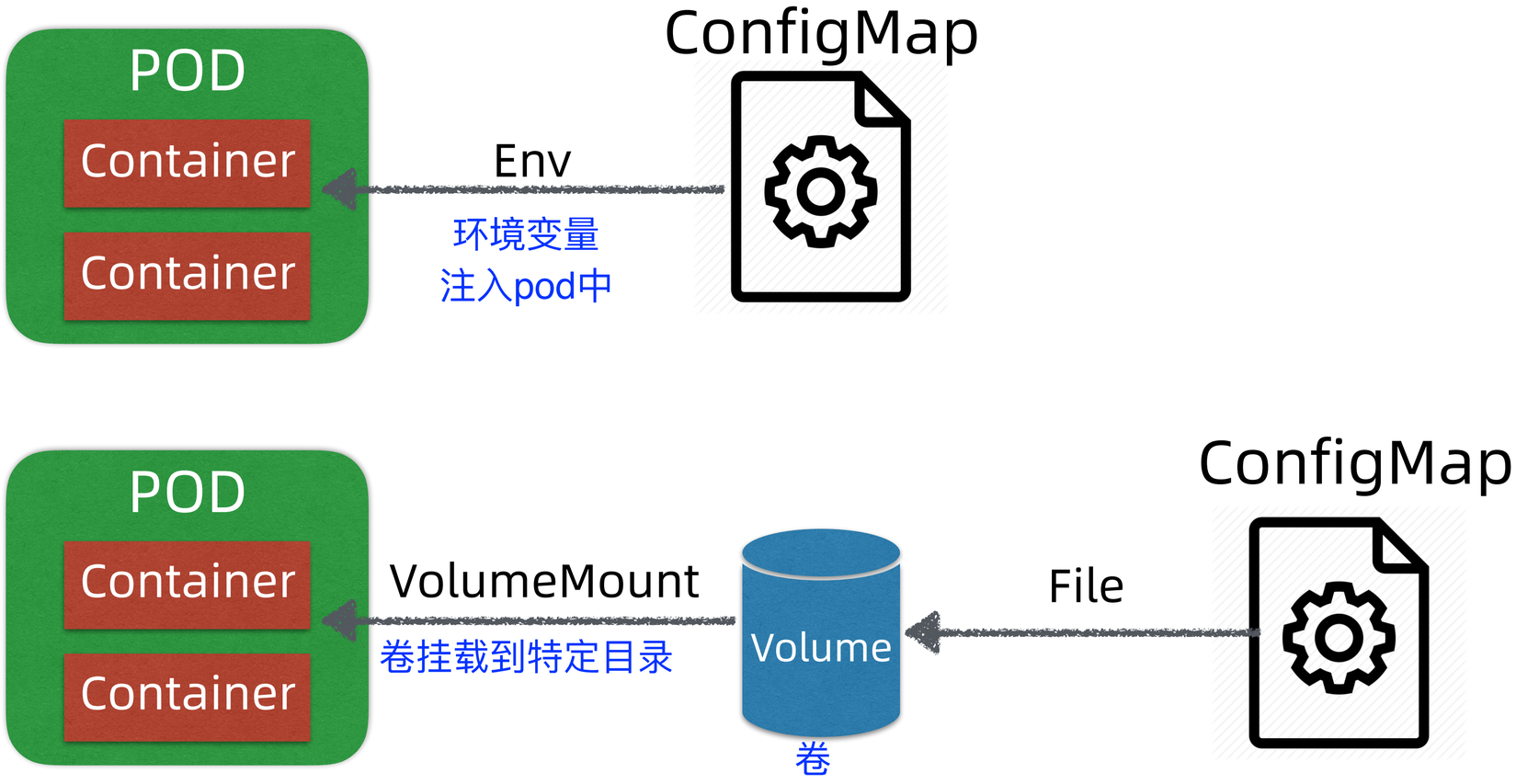
微服务配置主要配置 deployment和service规范



ConfigMap/Secret

配置中心

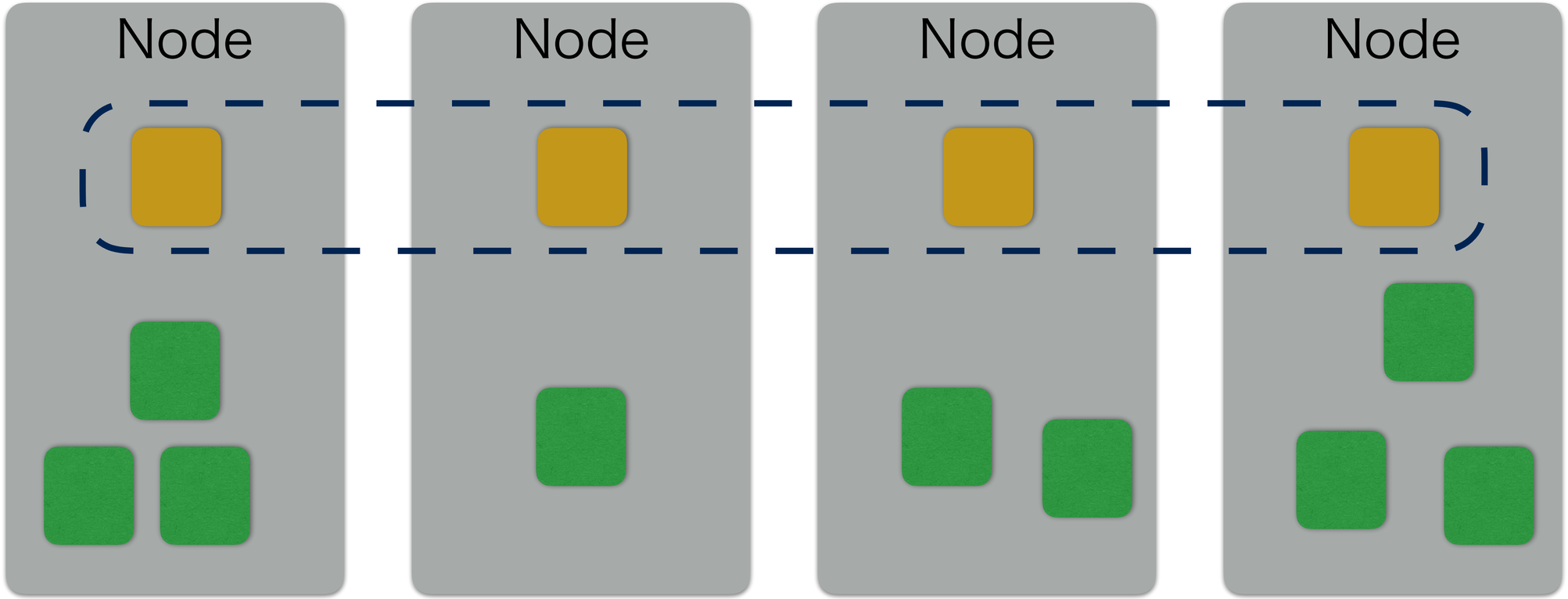
敏感数据(特殊配置)



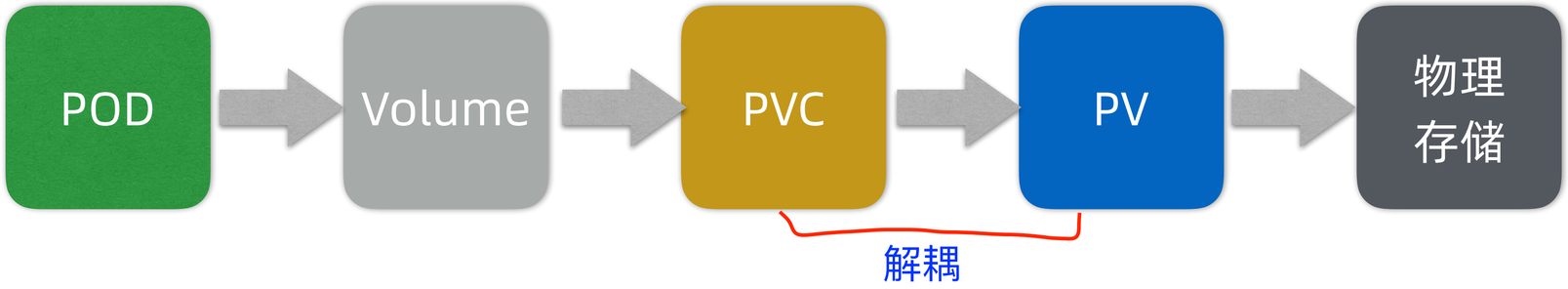
DaemonSet

在每个节点上部署守护进程 （每个worker节点有且仅一个）

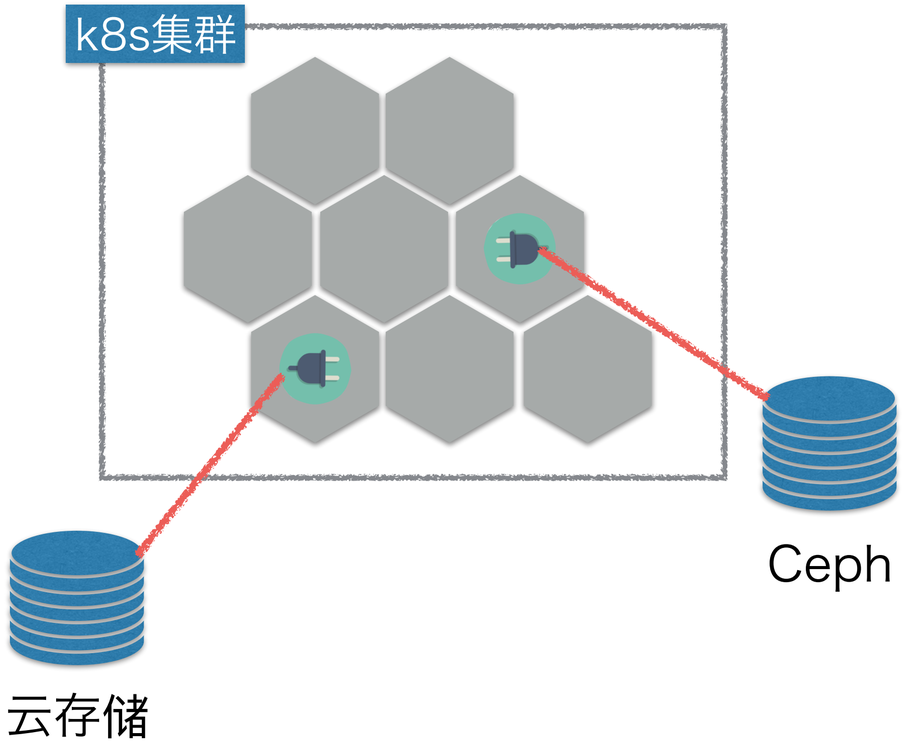
如 fluented日志采集、普罗米修斯等特殊守护进程



其它概念

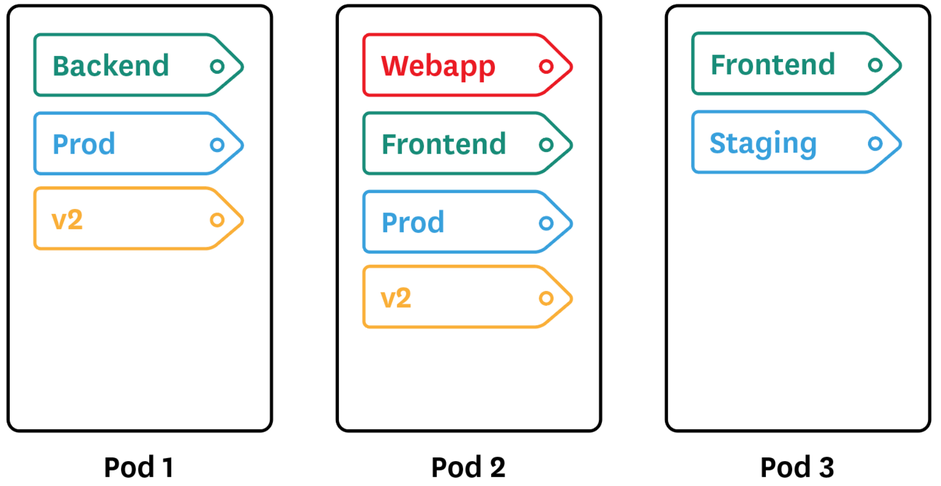


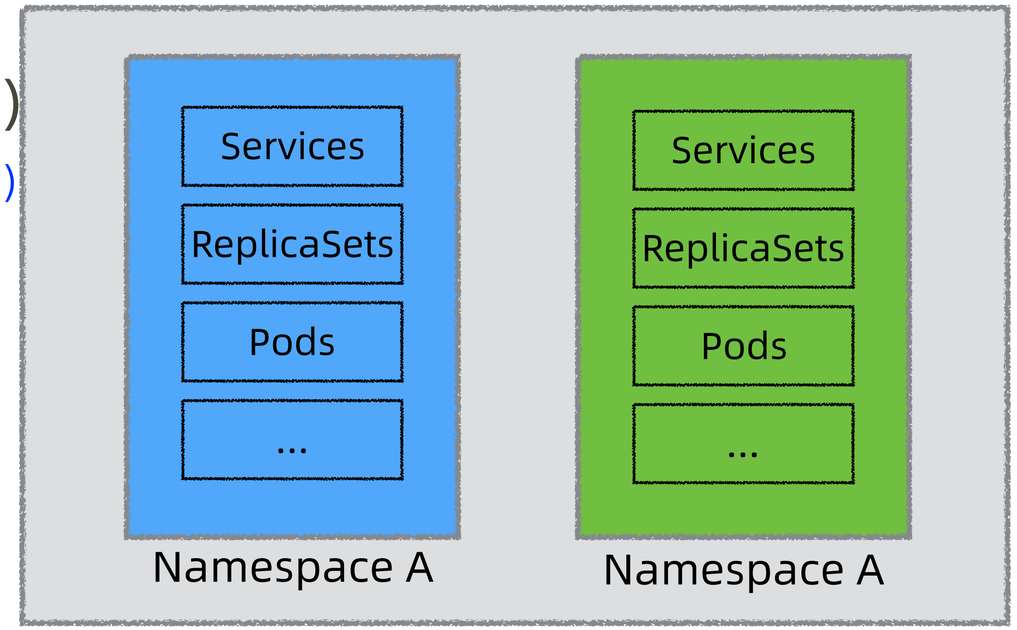
- Volume
  卷(存储) （本地或远程）
- PersistentVolume
  可以看作可插拔存储 对接各种云存储，重启不丢失
- PersistentVolumeClaims
  申请 pv 资源
- StatefulSet
  有状态应用发布 （如 mysql/redis）
- Job
  一次任务
- CronJob
  周期任务



概念补充

- Label/Selector
  打标签　　通过标签定位
  （如生产/版本/前端/后端）
- Namespace
  逻辑隔离　（多租户、应用隔离等）
- Readiness Probe（就绪探针）
  用于判断pod是否可以接入流量(如健康检查)
- Liveness Probe（活跃探针）
  用于判断pod是否存活





概念总结1

概念	作用
Cluster	超大计算机抽象，由节点组成
Container	应用居住和运行在容器中
Pod	Kubernetes 基本调度单位
ReplicaSet	创建和管理 Pod，支持无状态应用
Service	应用 Pods 的访问点，屏蔽 IP 寻址和负载均衡
Deployment	管理 ReplicaSet，支持滚动等高级发布机制
ConfigMap/Secrets	应用配置，secret 敏感数据配置
DaemonSet	保证每个节点有且仅有一个 Pod，常见于监控
StatefulSet	类似 ReplicaSet，但支持有状态应用

概念总结2

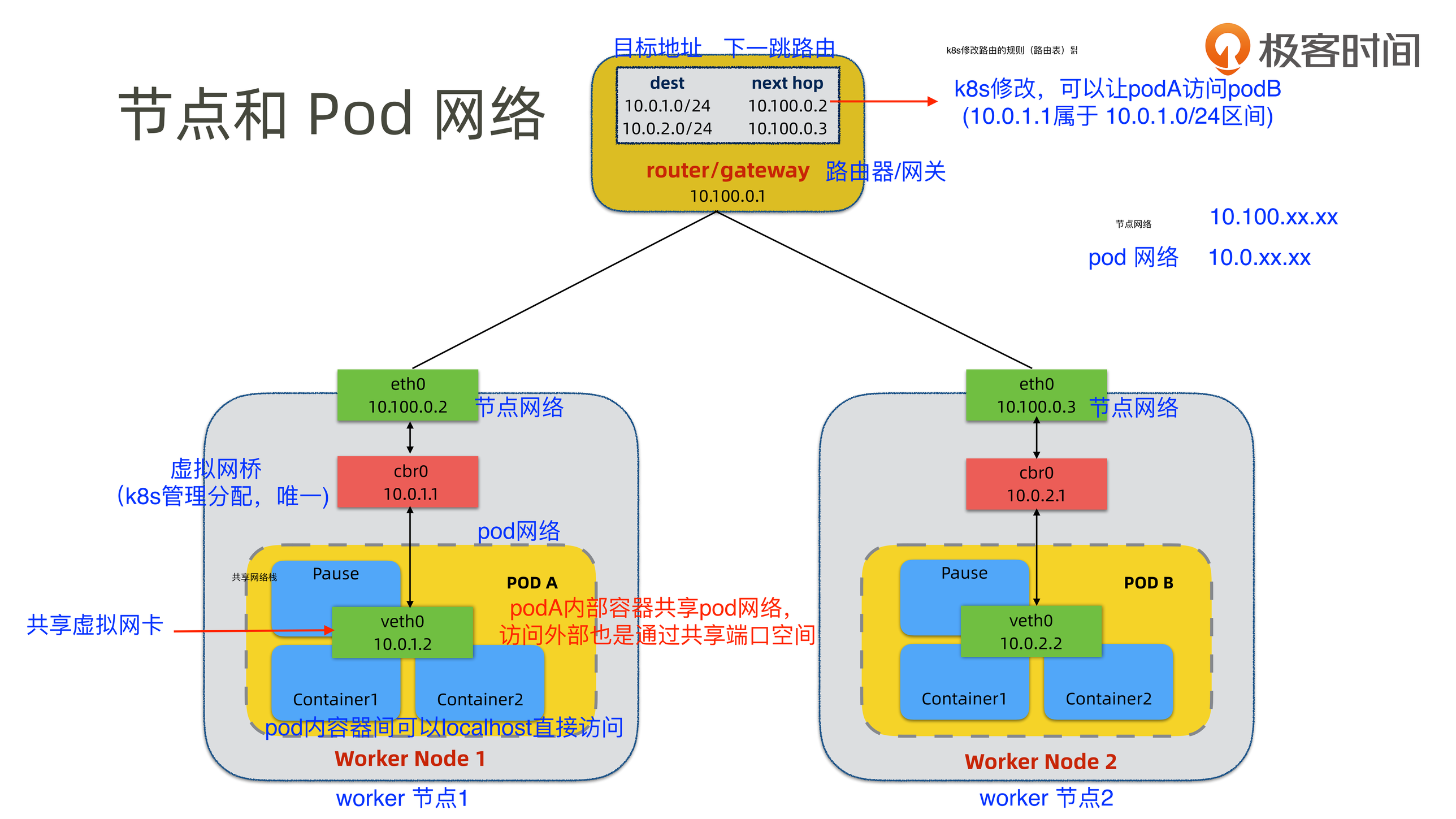
概念	作用
Job	运行一次就结束的任务
CronJob	周期性运行的任务
Volume	可装载磁盘文件存储
PersisentVolume/ PersistentVolumeClaims	超大磁盘存储抽象和分配机制
Label/Selector	资源打标签和定位机制
Namespace	资源逻辑隔离机制
Readiness Probe	就绪探针，流量接入 Pod 判断依据
Liveness Probe	存活探针，是否 kill Pod 的判断依据

极客时间

第 4 部分

理解 Kubernetes 节点网络和 Pod 网络

节点和 Pod 网络

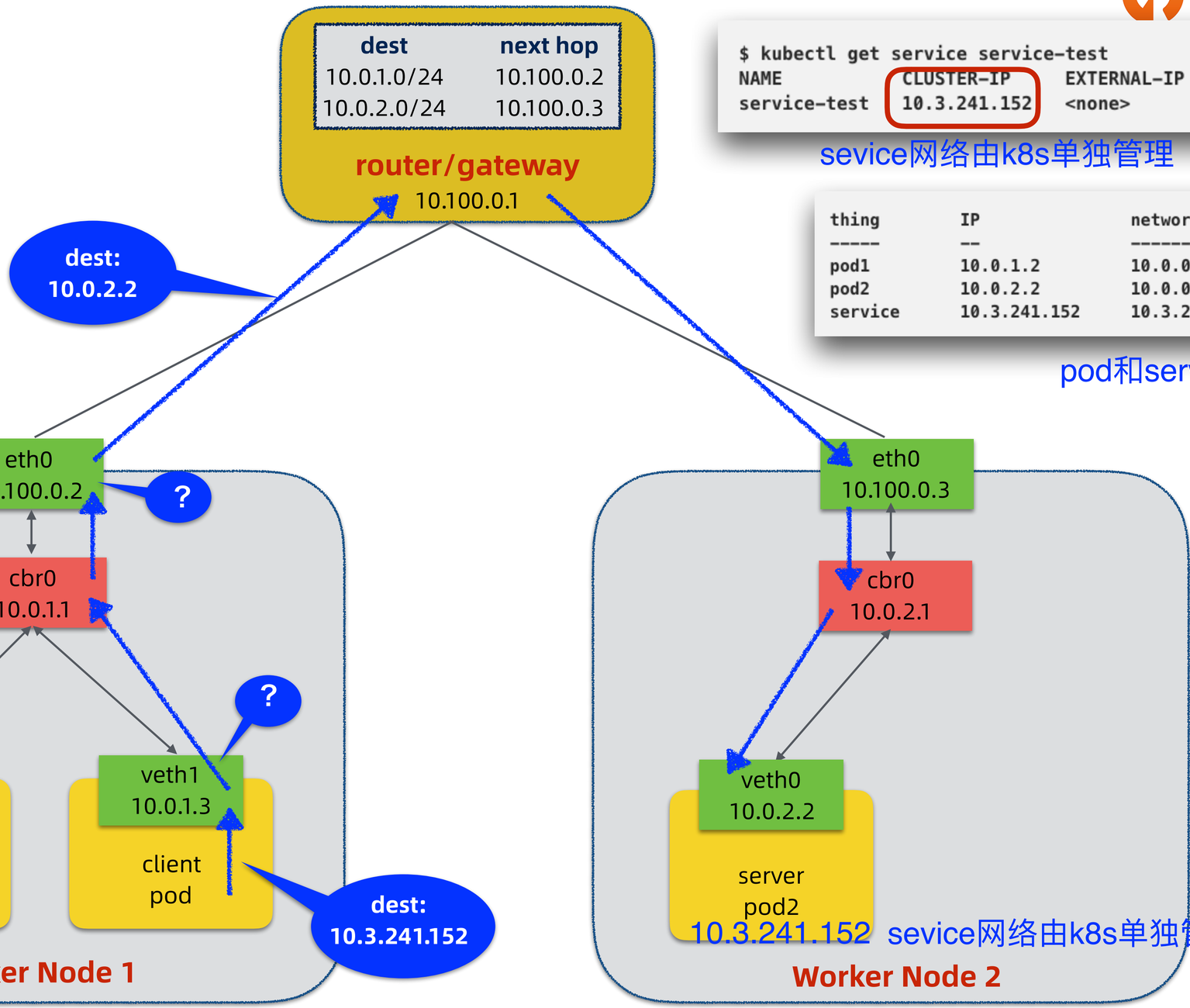

目标地址 下一跳路由
dest
next hop
10.0.1.0/24
10.100.0.2
10.0.2.0/24
10.100.0.3
router/gateway 路由器/网关
10.100.0.1
k8s修改路由的规则（路由表）
k8s修改，可以让podA访问podB
(10.0.1.1属于 10.0.1.0/24区间)
节点网络 10.100.xx.xx
pod 网络 10.0.xx.xx
eth0
10.100.0.2
节点网络
虚拟网桥
(k8s管理分配，唯一)
cbr0
10.0.1.1
pod网络
共享网络栈
Pause
POD A
共享虚拟网卡
veth0
10.0.1.2
podA内部容器共享pod网络，
访问外部也是通过共享端口空间
Container1
Container2
pod内容器间可以localhost直接访问
Worker Node 1
worker 节点1
eth0
10.100.0.3
节点网络
cbr0
10.0.2.1
Pause
POD B
veth0
10.0.2.2
Container1
Container2
Worker Node 2
worker 节点2

第 5 部分

深入理解 Service 和 Service Discovery

Service 如何寻址?

应用 Pods 的访问点,
解决问题: 屏蔽 IP 寻址和负载均衡




```
$ kubectl get service service-test
NAME           CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
service-test   10.3.241.152    <none>        80/TCP    11s
```

sevice网络由k8s单独管理

thing	IP	network
pod1	10.0.1.2	10.0.0.0/14
pod2	10.0.2.2	10.0.0.0/14
service	10.3.241.152	10.3.240.0/20

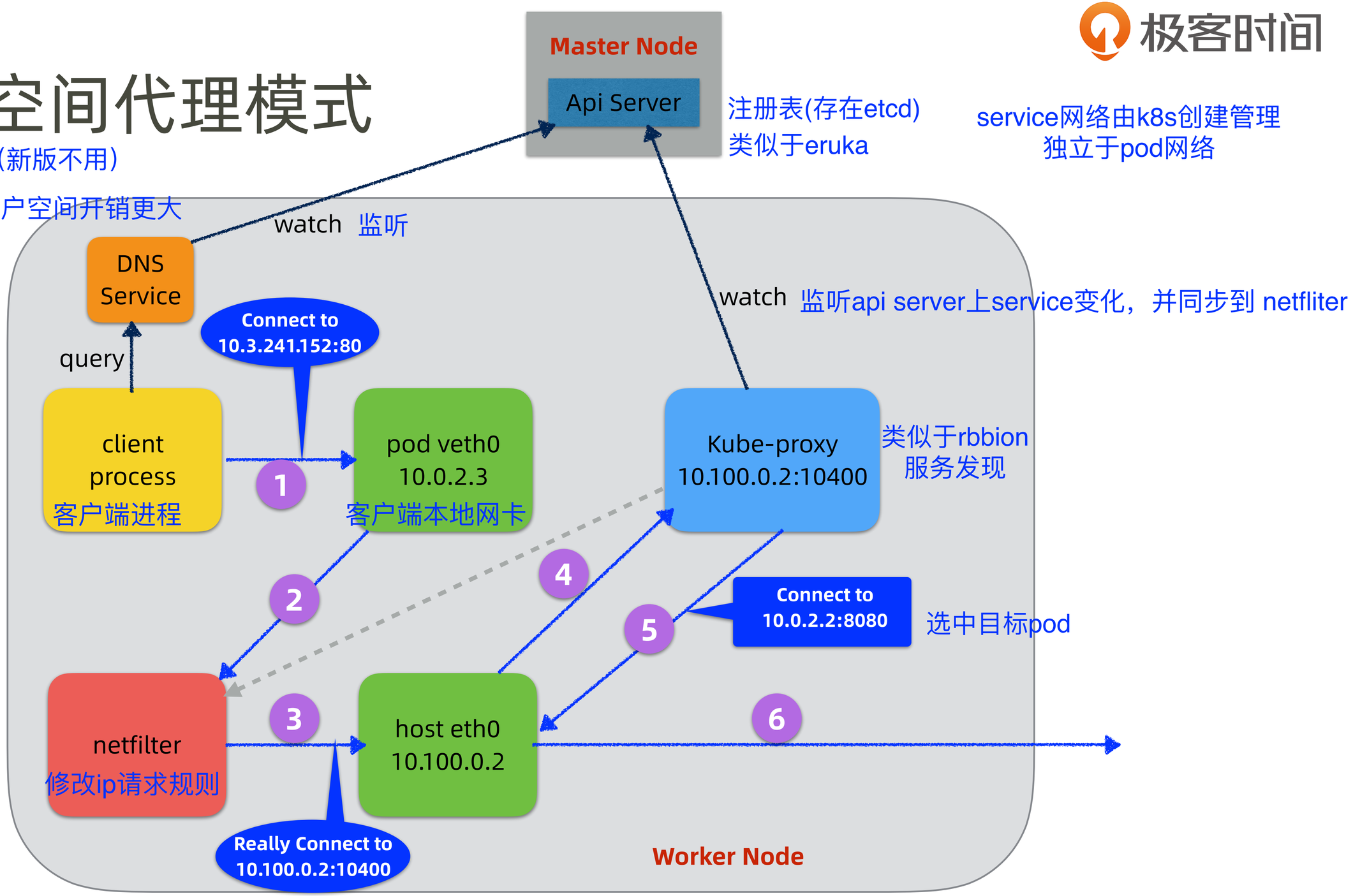
pod和service的网络ip不同

10.3.241.152 sevice网络由k8s单独管理

用户空间代理模式

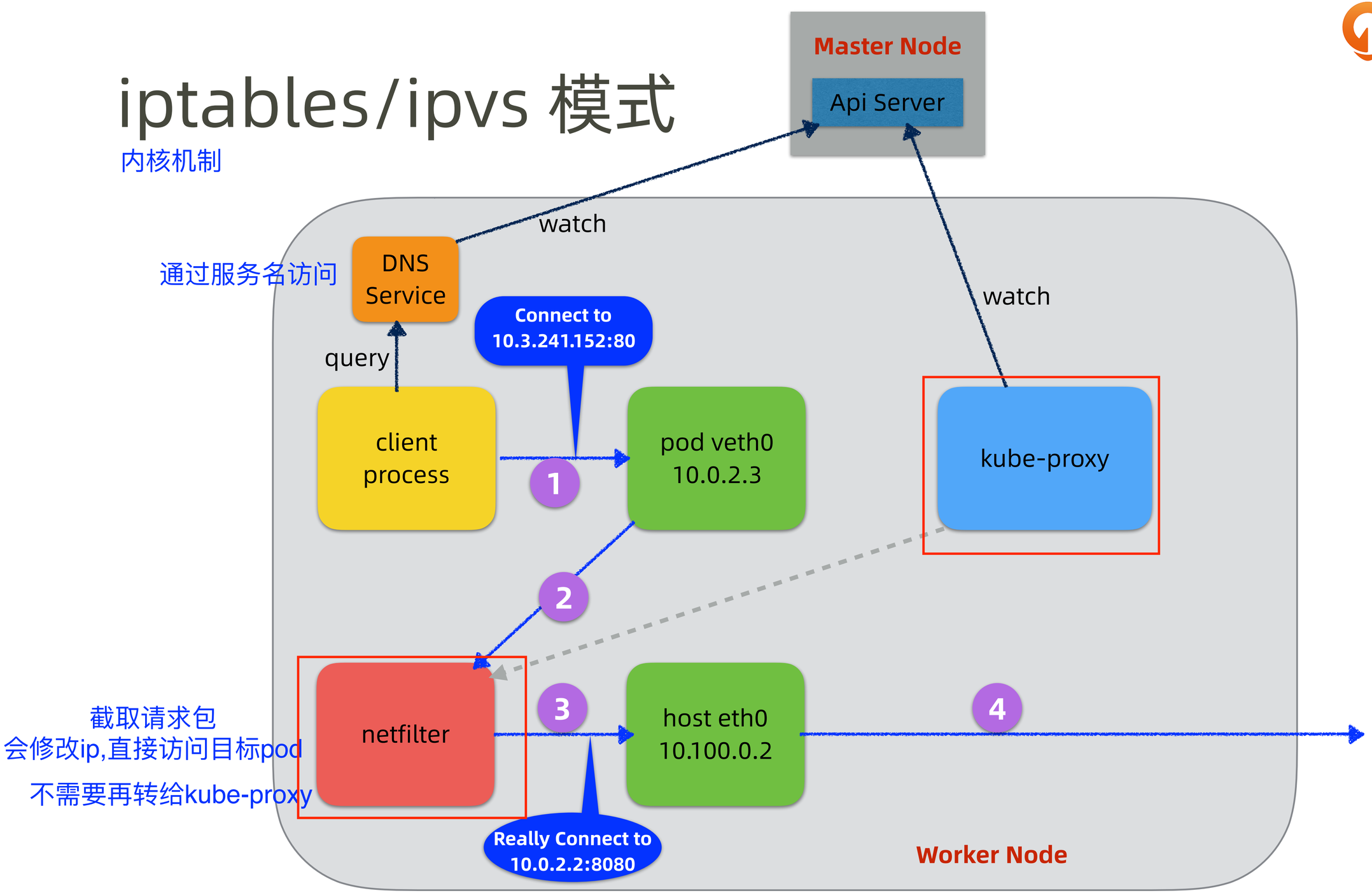
早期版本k8s（新版不用）

劣势:单点，用户空间开销更大



iptables/ipvs 模式

内核机制

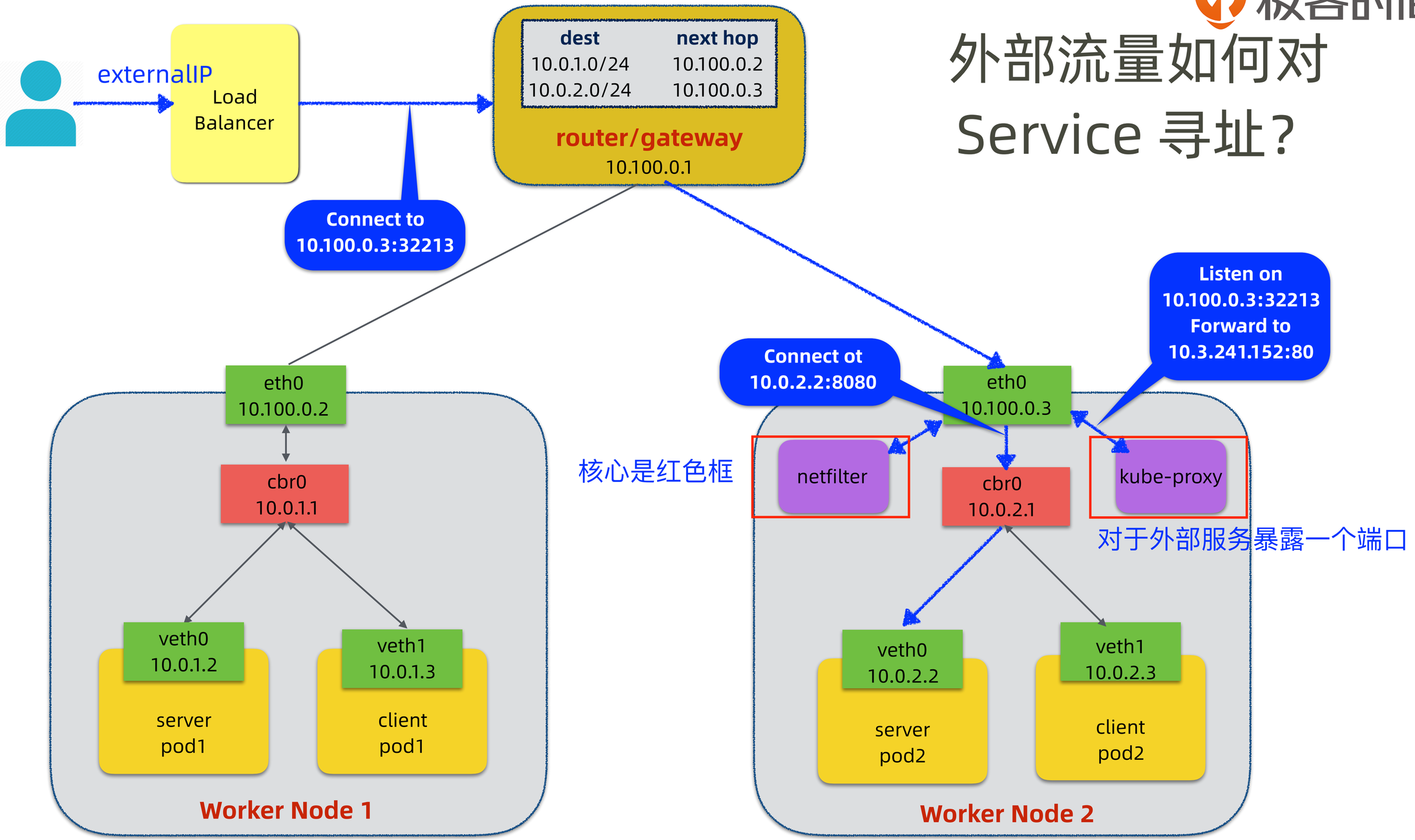


极客时间

第 6 部分

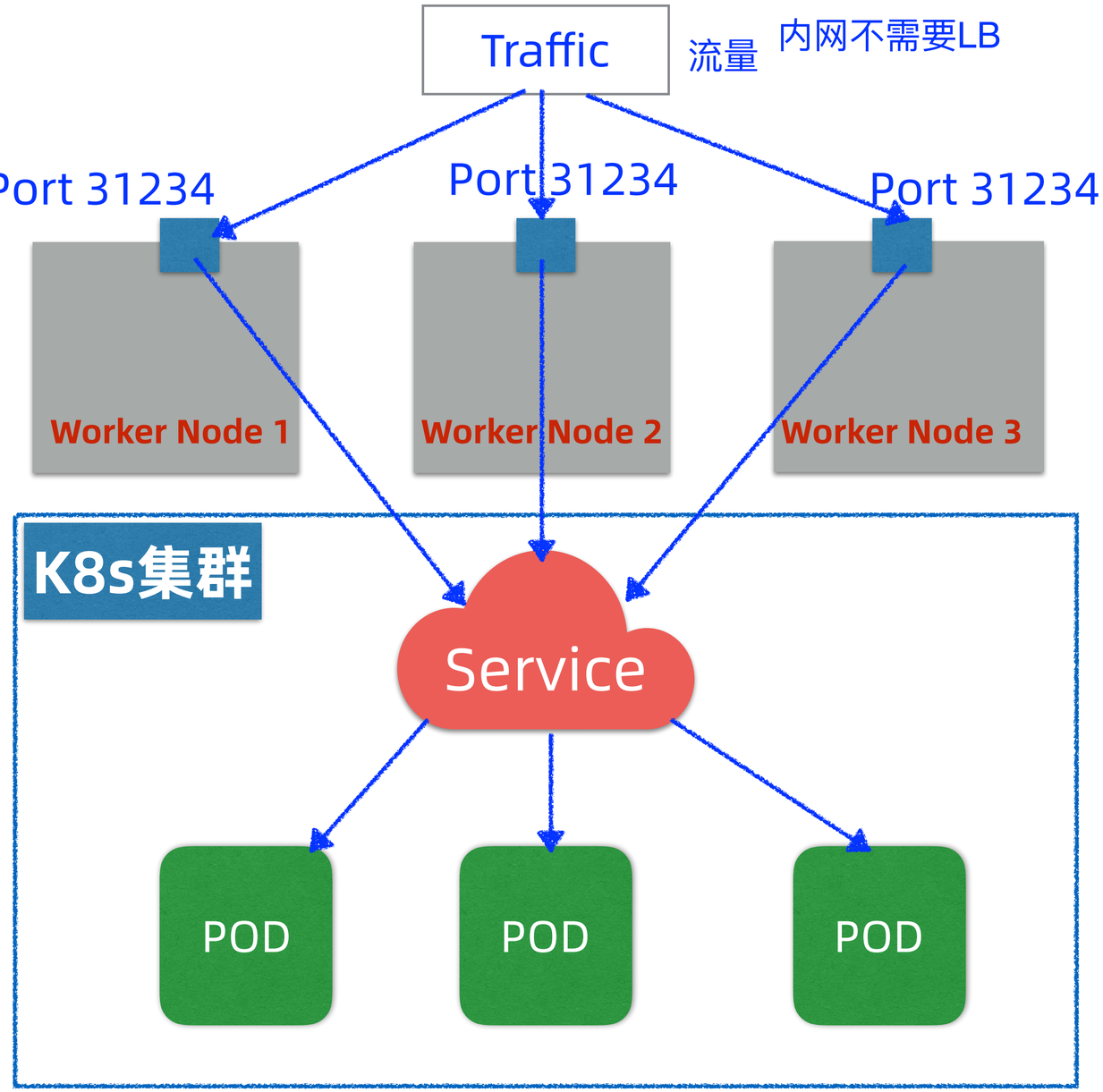
NodePort vs LoadBalancer vs Ingress

外部流量如何对 Service 寻址？

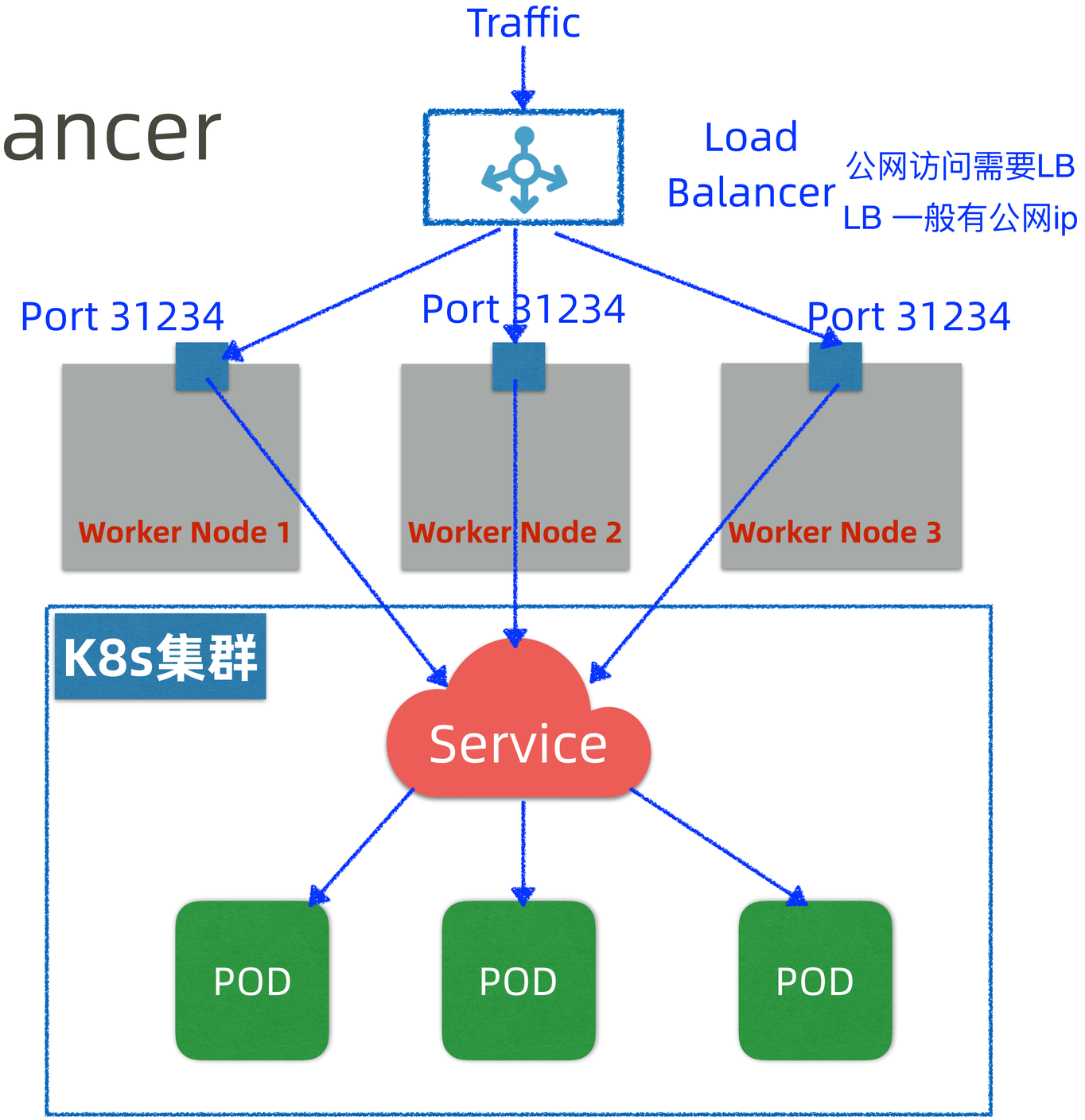


NodePort

一种特殊 service,
k8s在每个节点上暴露相同的端口



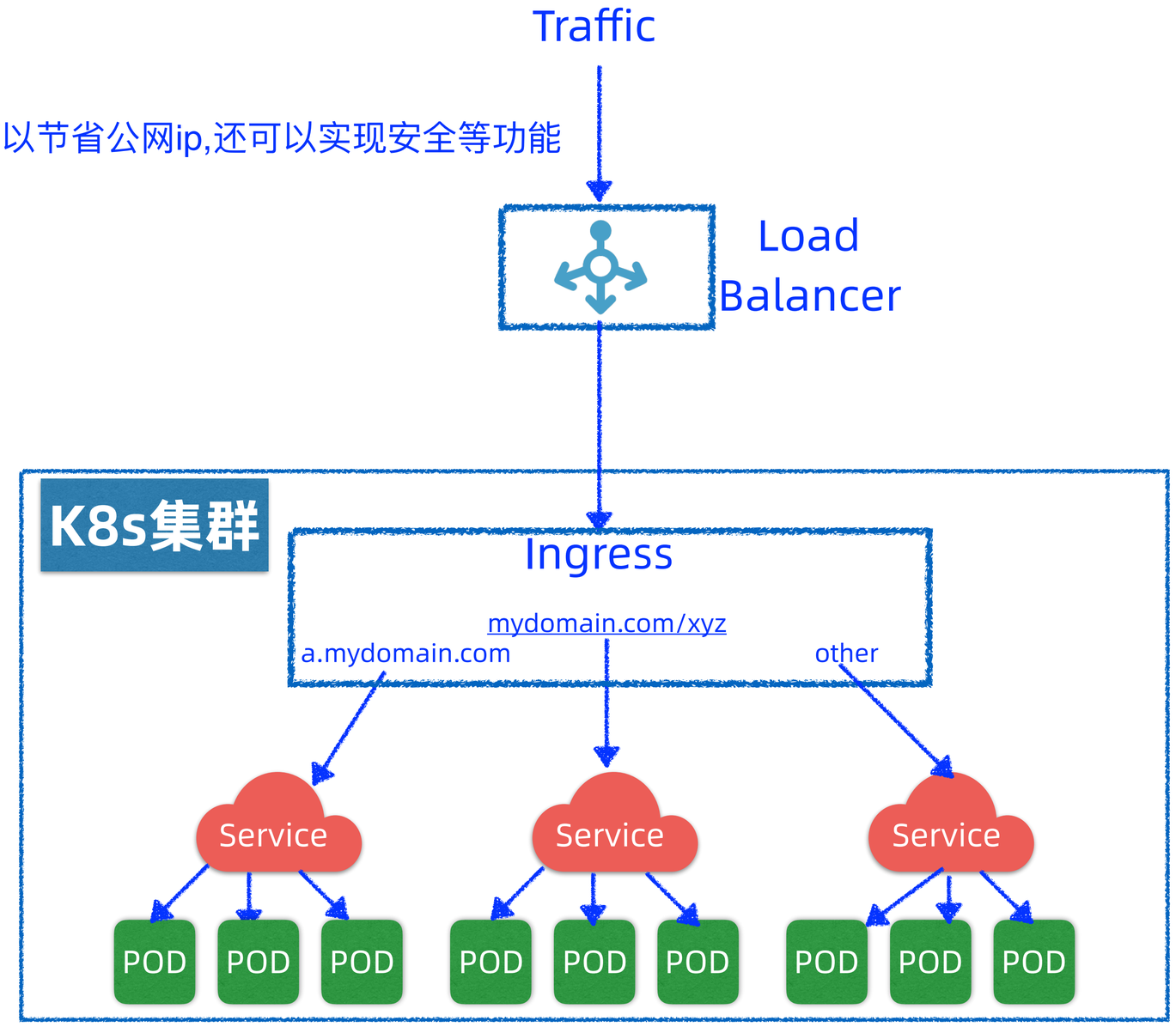
Load Balancer



Ingress

类似反向代理/网关，可以节省公网ip,还可以实现安全等功能

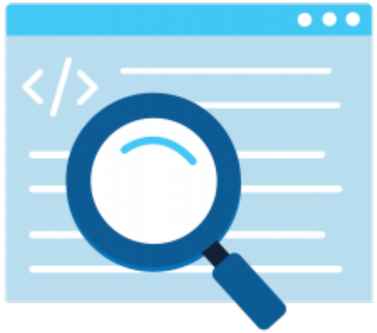
类似项目的faraday



总结

	作用	实现
节点网络	Master/Worker 节点之间网络互通	路由器，交换机，网卡
Pod 网络	Pod 之间互通	虚拟网卡，虚拟网桥，路由器
Service 网络	屏蔽 Pod 地址变化+负载均衡	Kube-proxy, Netfilter, Api-Server，DNS
NodePort	将 Service 暴露在节点网络上	Kube-proxy + Netfliter
LoadBalancer	将Service暴露在公网上+负载均衡	公有云 LB + NodePort
Ingress	反向路由，安全，日志监控 (类似反向代理 or 网关)	Nginx/Envoy/Traefik/Faraday

极客时间

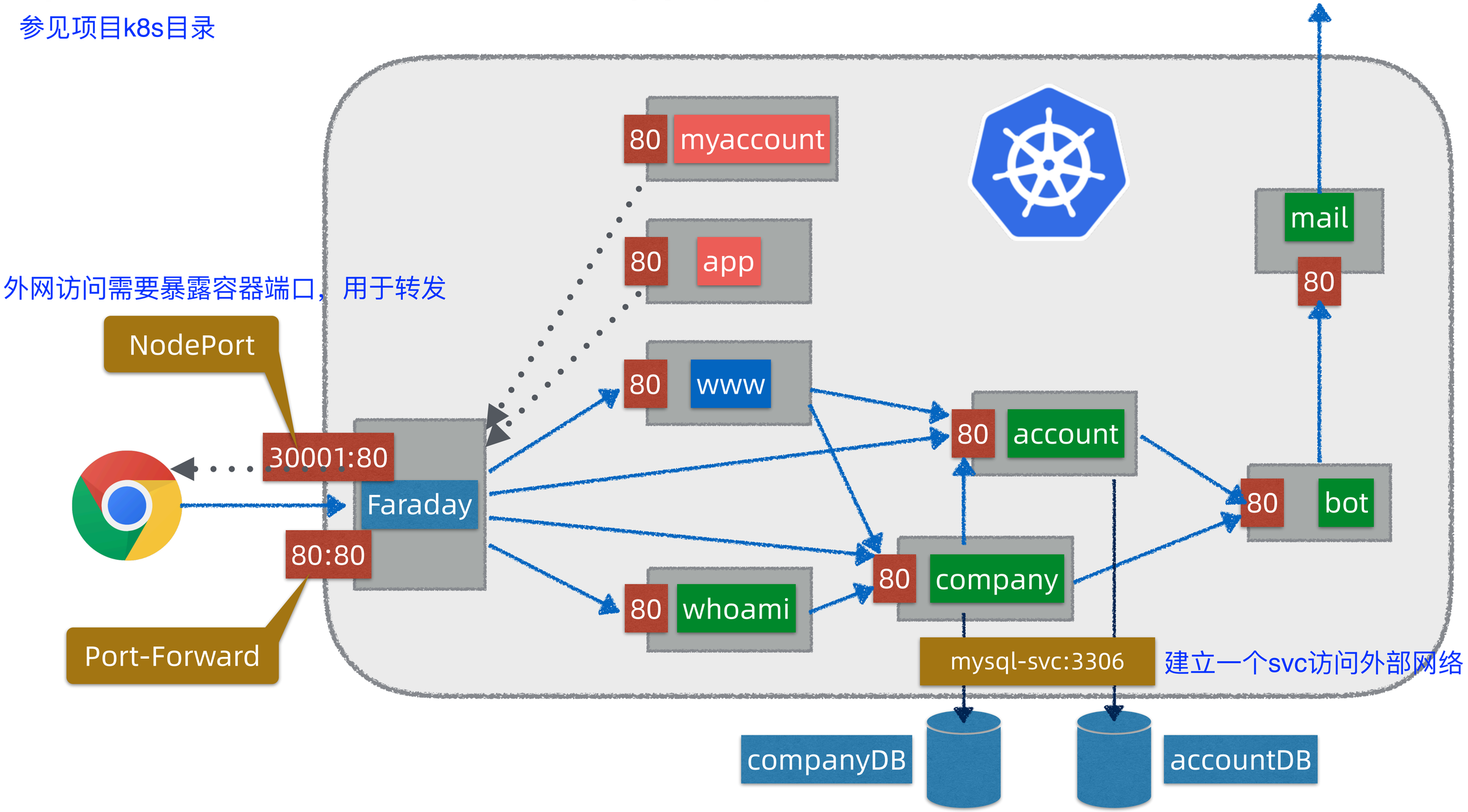


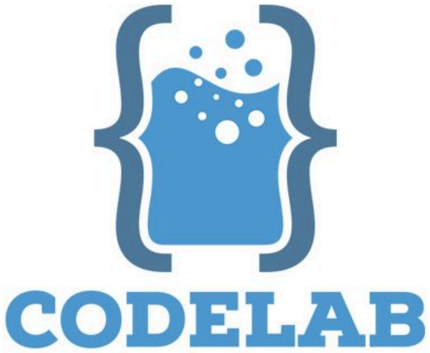
第 7 部分

本地测试 Kubernetes 部署文件剖析

本地 Kubernetes 部署架构

参见项目k8s目录

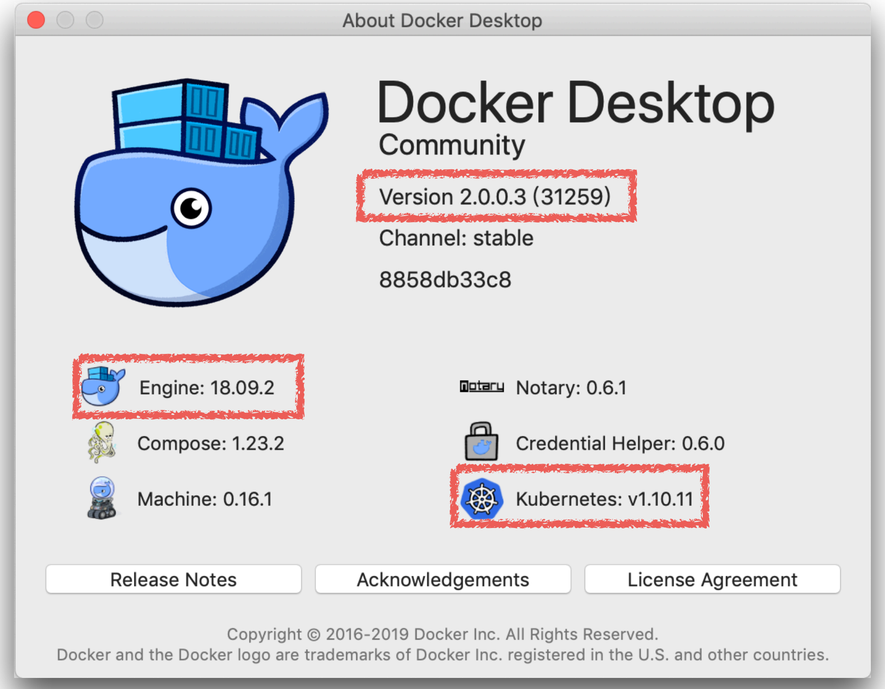


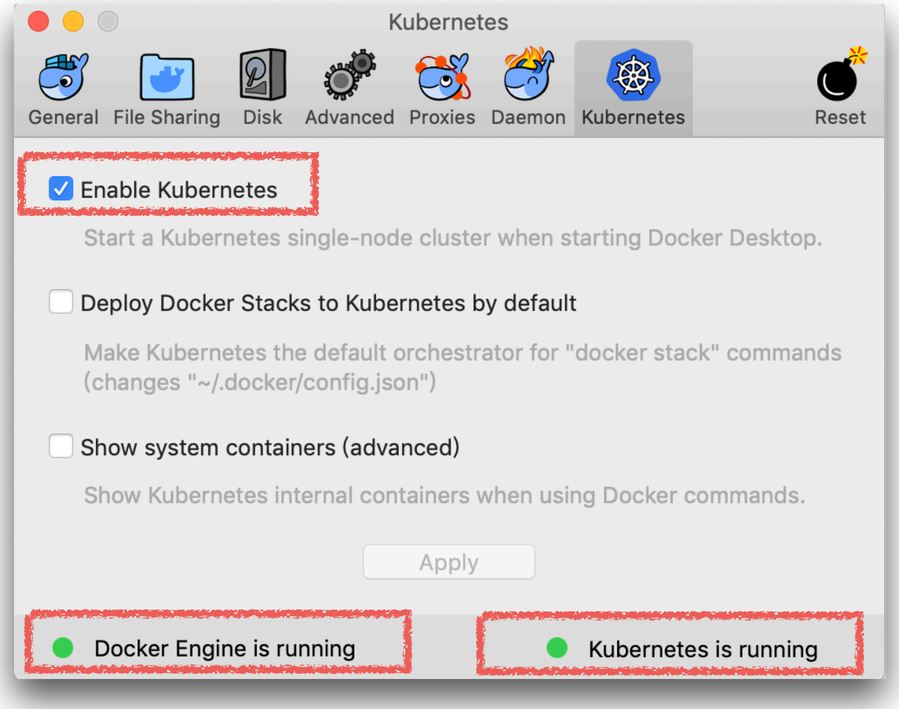


第 8 部分

本地测试 Kubernetes 环境搭建

Docker Desktop for Mac/Win





https://docs.docker.com/docker-for-mac/install/

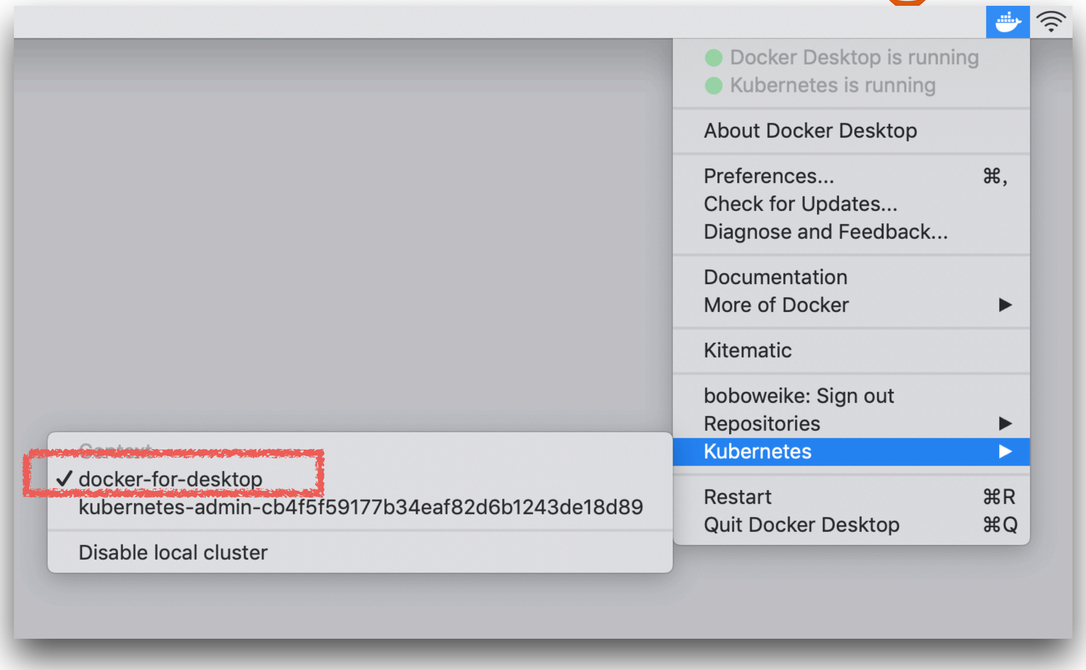
https://docs.docker.com/docker-for-windows/install/

https://github.com/AliyunContainerService/k8s-for-docker-desktop

https://github.com/AliyunContainerService/minikube

校验 Kubernetes 安装

1. kubectl version
2. kubectl config current-context
3. kubectl cluster-info
4. kubectl get nodes



安装和访问 Kubernetes Dashboard

1. 安装Kubernetes Dashboard

- https://github.com/kubernetes/dashboard
- kubectl get pods —namespace=kube-system

2. 启动 kube proxy

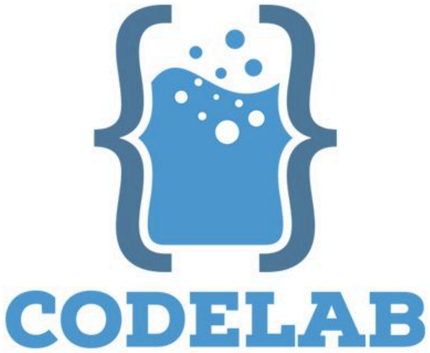
- kubectl proxy

3. 生成访问令牌

- `kubectl -n kube-system describe secret $(kubectl -n kube-system get secret | grep admin-user | awk '{print $1}')`

4. 访问 Dashboard

- http://localhost:8001/api/v1/namespaces/kube-system/services/https:kubernetes-dashboard:/proxy/



第 9 部分

将 Staffjoy 部署到本地 Kubernetes 环境

构建和部署

极客时间

Step By Step

1. 镜像构建(可选)
   - mvn clean package -DskipTests
   - docker-compose build
   - docker images
2. 部署mysql数据库(可选)
   - staffjoy_account
   - staffjoy_company
   - 授予ip访问权限
3. 部署Staffjoy(k8s/test)
   - kubectl apply -f config.yml　　kubectl get configmaps
   - kubectl apply -f test　　在项目根目录执行
4. 端口转发
   - 查询faraday pod名：Kubectl get pods
   - kubectl port-forward faraday-svc-deployment-8584d9c74d-v92wt 80:80
5. 启用SwitchHosts
6. 命令行校验　　校验发布的状况
   - kubectl get pods -o wide
   - Kubectl get services
   - Kubectl get deployments
7. K8s Dashboard校验
8. Staffjoy校验　注册一个管理员校验
9. 清理
   - kubectl delete deployments —all
   - kubectl delete services —all
   - kubectl delete configmaps —all

MySQL 访问权限

1. 登录 MySQL

- mysql -u root -p

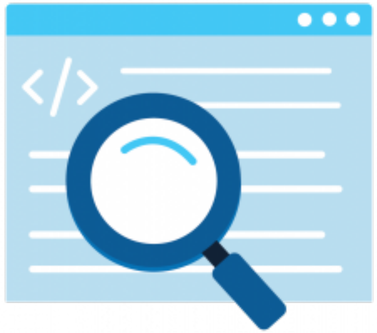
2. 查询 user 表

- use mysql;
- Select user, host from user;

```
mysql> select user,host from user;
+---------------+---------------+
| user          | host          |
+---------------+---------------+
| root          | 192.168.1.100 |
| mysql.session | localhost     |
| mysql.sys     | localhost     |
| root          | localhost     |
| springmall    | localhost     |
+---------------+---------------+
5 rows in set (0.00 sec)
```

3. 授权

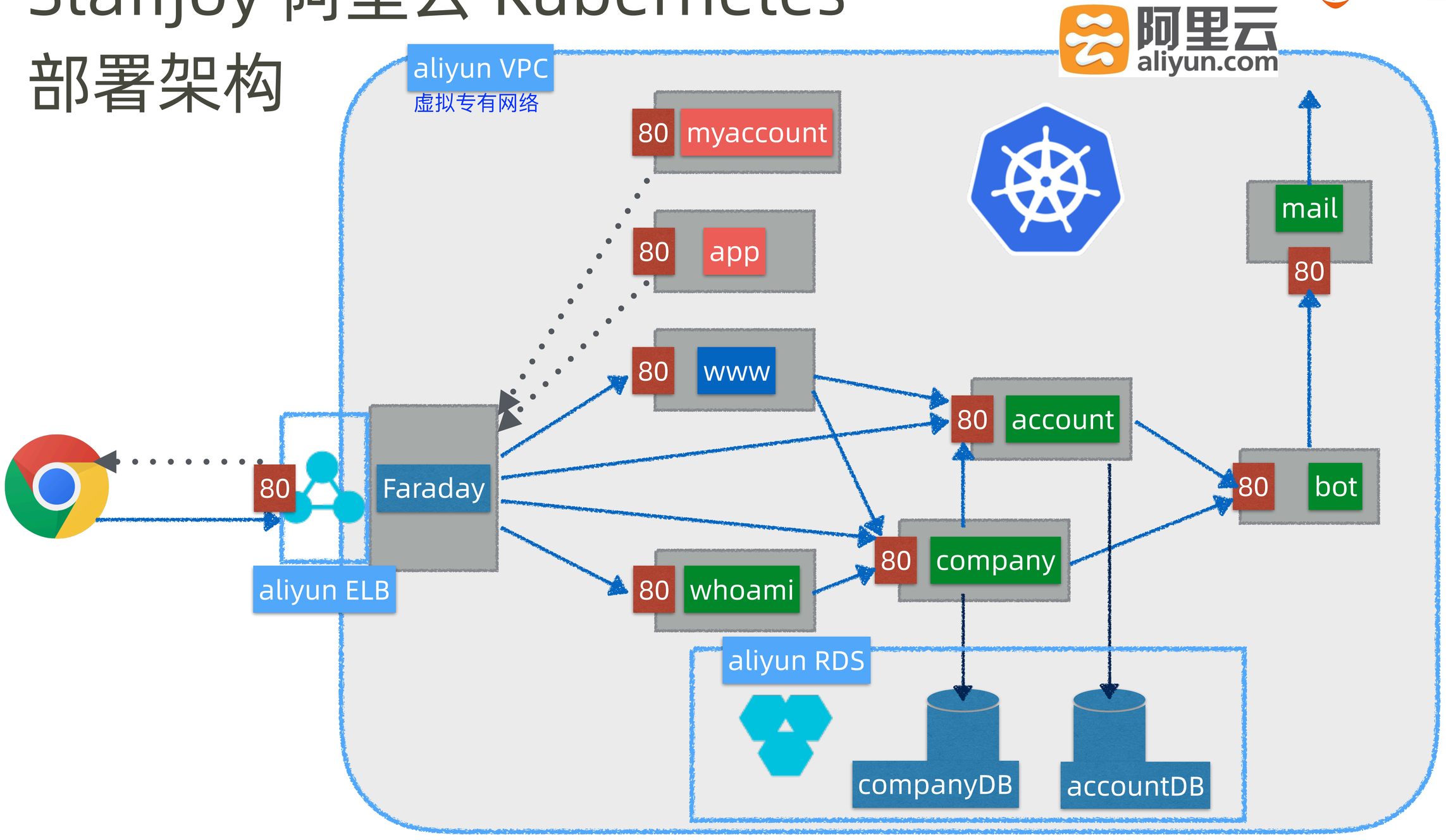
- grant all privileges on *.* to root@your_ip identified by ‘root_password’ with grant option;



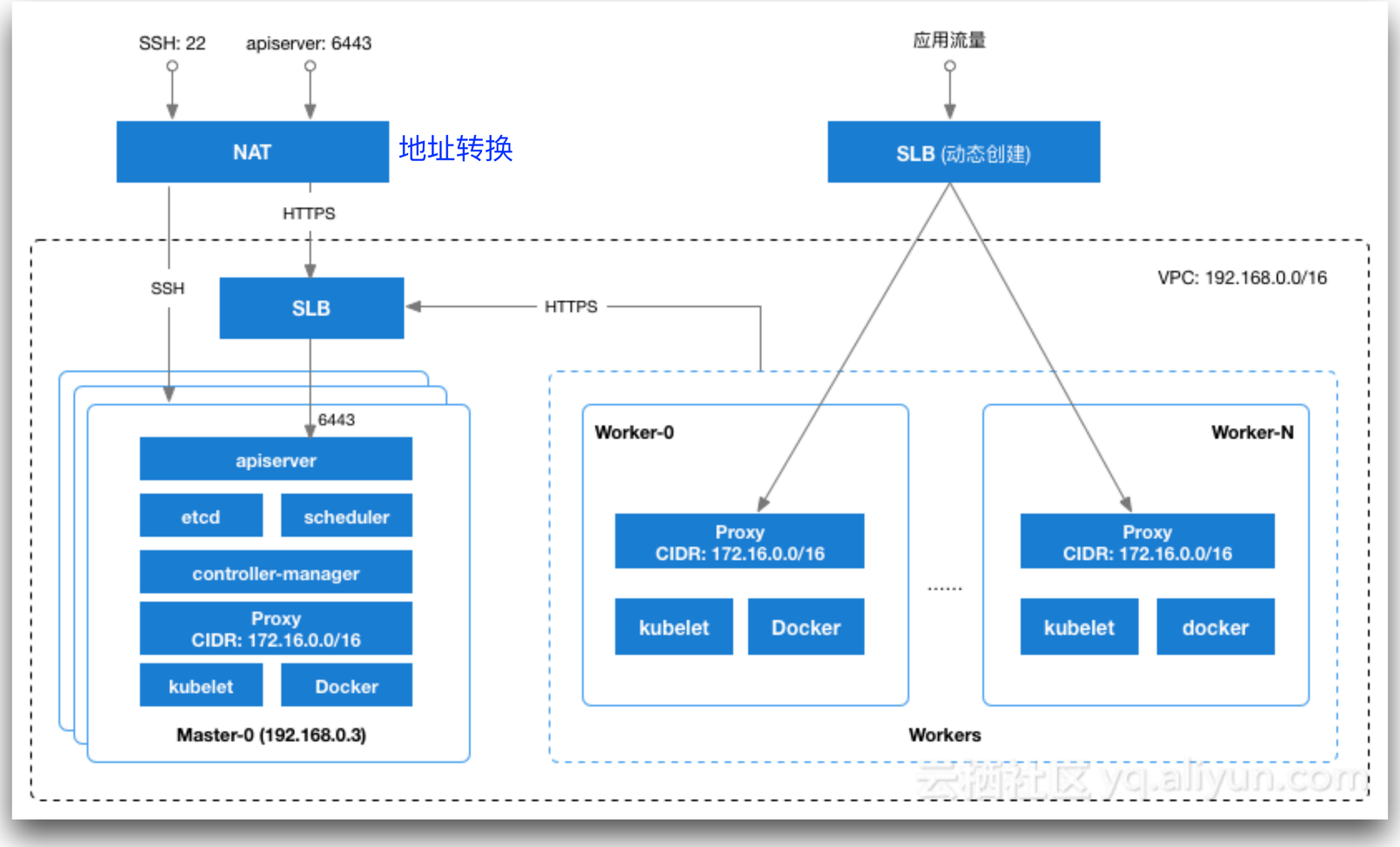
第 10 部分

生产环境 Kubernetes 部署文件剖析

Staffjoy 阿里云 Kubernetes 部署架构

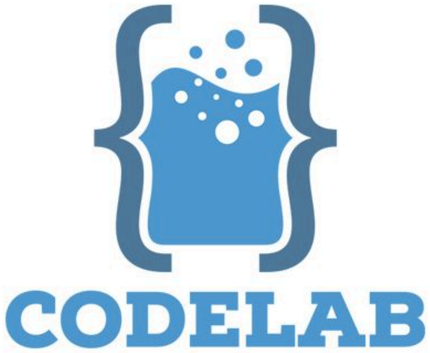


阿里云 Kubernetes 拓扑结构



https://yq.aliyun.com/articles/88526

极客时间



第 11 部分

阿里云 Kubernetes 环境创建

Kubernetes 环境创建

1. 创建 VPC

2. 创建 RDS 数据库

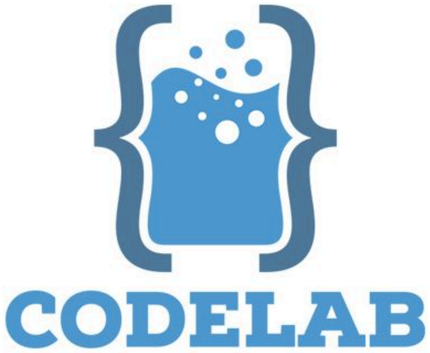
- 加 IP 白名单
- 更新 JDBC 连接字符串
- 创建 root 账户
- 创建数据库和表

3. 创建共享版 Kubernetes 集群

- 更新 .kube/config

4. 校验

- kubectl config current-context
- kubectl cluster-info
- kubectl get nodes



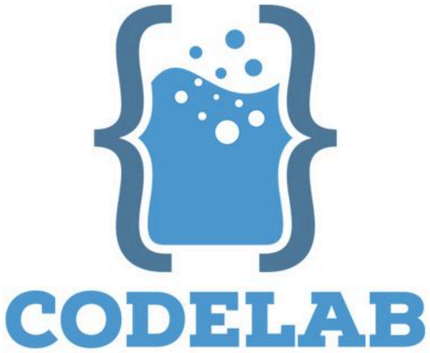
第 12 部分

将 Staffjoy 部署到阿里云 Kubernetes 环境

Staffjoy 部署

1. 部署 Staffjoy(Kubernetes/UAT)
   - kubectl apply -f config
   - kubectl apply -f uat
2. 阿里云 Dashboard 校验
3. 命令行校验
   - kubectl get pods -o wide
   - Kubectl get services
   - Kubectl get deployments
4. 启用 SwitchHosts
   - 更新 ELB IP 地址
5. Staffjoy 校验
6. 删除阿里云 Kubernetes 和 RDS

极客时间



第 13 部分

Kubernetes 应用动态配置实验

Staffjoy 动态配置实验步骤

1. Kubernetes Dashboard 校验

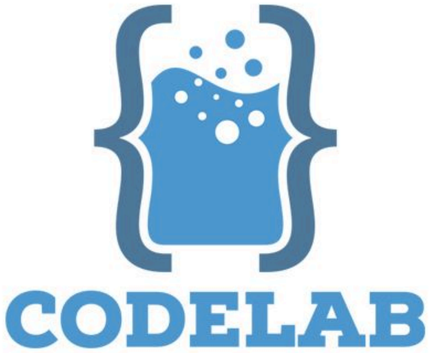
- account-svc pod 命令控制台校验 logback-config.yaml 文件内容

2. 更新 logback 配置（Kubernetes/UAT/config）

- 更新 logback 为 debug 输出
- kubectl apply -f logback-config.yaml

3. Kubernetes Dashboard 校验

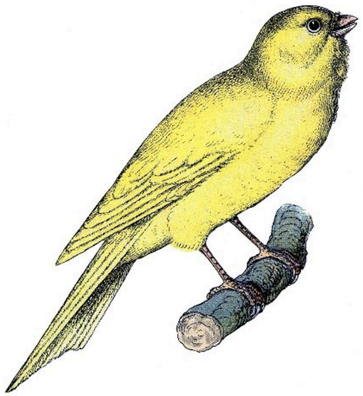
- account-svc pod 命令控制台校验 logback-config.yaml 文件更新
- 等待 30 秒
- account-svc pod 日志控制台输出 debug 日志

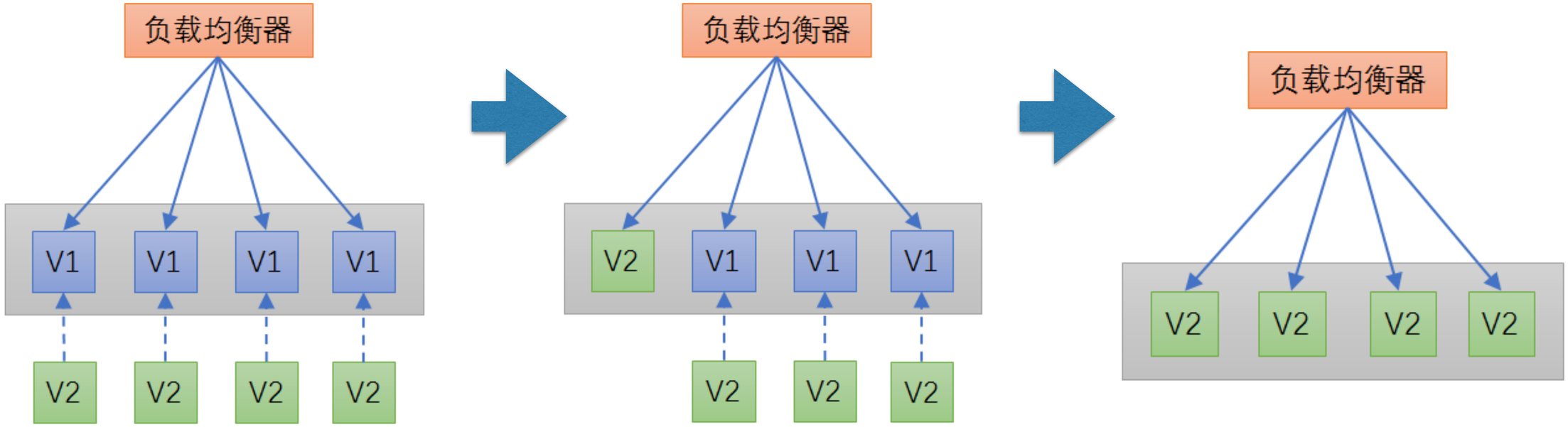


第 14 部分

Kubernetes 应用金丝雀发布实验

金丝雀发布




负载均衡器
V1
V1
V1
V1
V2
V2
V2
V2
负载均衡器
V2
V1
V1
V1
V2
V2
V2
负载均衡器
V2
V2
V2
V2

金丝雀发布步骤

1. www-web-deployment 扩容到 3 个实例
2. 发布金丝雀 www-web-deployment-canary
   - kubectl apply -f canary
3. 校验
4. 扩容金丝雀到 3 个实例
5. 删除 www-web-deployment

极客时间

扫码试看/订阅

《Spring Boot & Kubernetes 云原生微服务实践》