咕泡学院
gupaoedu.com

专业互联网IT教育服务平台

[VIP课]

Kubernetes入门与实战

DO ONE THING AT A TIME AND DO WELL.

- 大白老师QQ号：1828627710

自我介绍

咕泡学院- 大白老师

前大众点评架构师

十余年Java经验，曾任职于1号店、大众点评、同程旅游、阿里系公司，担任过技术总监、首席架构师、team leader、系统架构师。有着多年的前后台大型分布式项目架构经验，在处理高并发、性能调优上有独到的方法论。精通Java、J2EE架构、Redis、MongoDB、Netty，消息组件如Kafka、RocketMQ。

课程目标

- 了解应用部署运行模式变迁
- 了解容器编排技术
- 了解Kubernetes是什么
- 深入理解Kubernetes架构和核心组件
- 了解Kubernetes基础概念

1

Kubernetes入门与实战

K8S总体架构

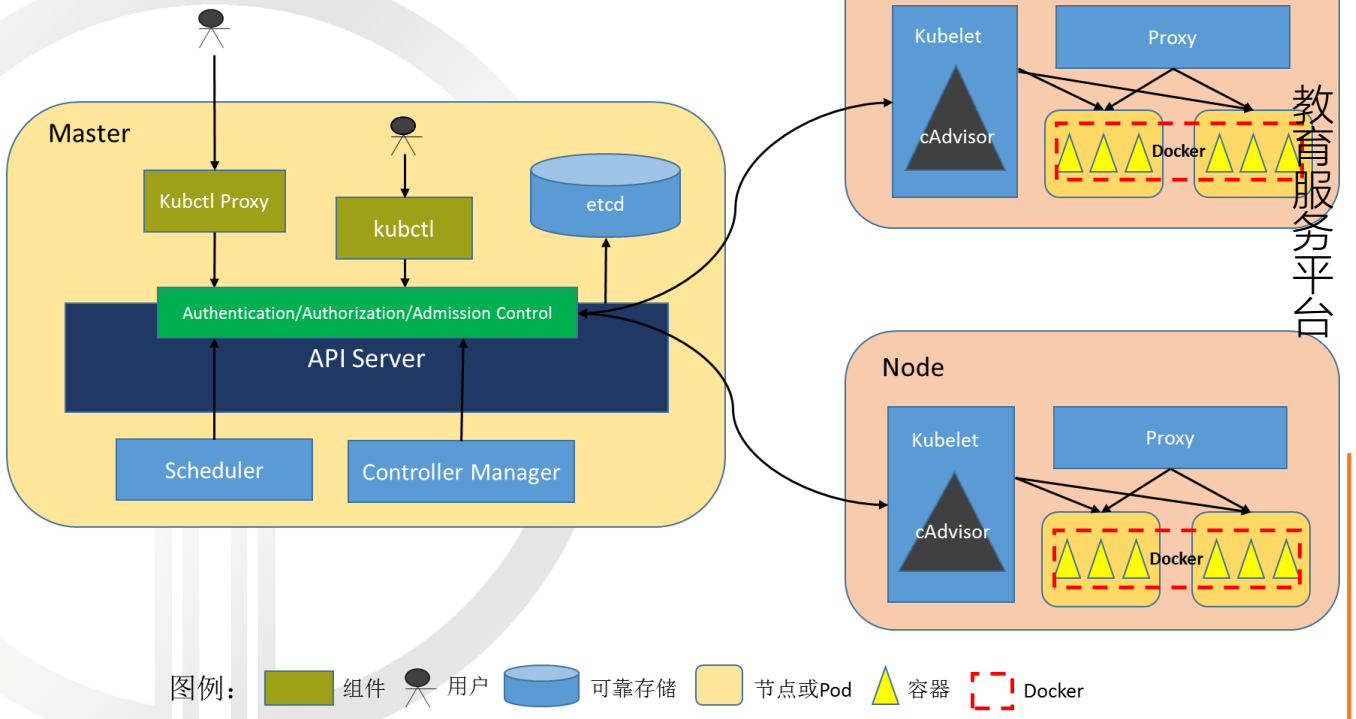
K8S是一个基于容器技术的分布式系统，

- 思想上的变革
  - ✓ 模块化思维（功能更加单纯，复用率高，高解耦）
  - ✓ 无需关心扩容、容错、负载均衡、通讯、安全、资源配额、服务发现等底层问题
  - ✓ 开发行为将由于“微服务”的理念而发生改变，

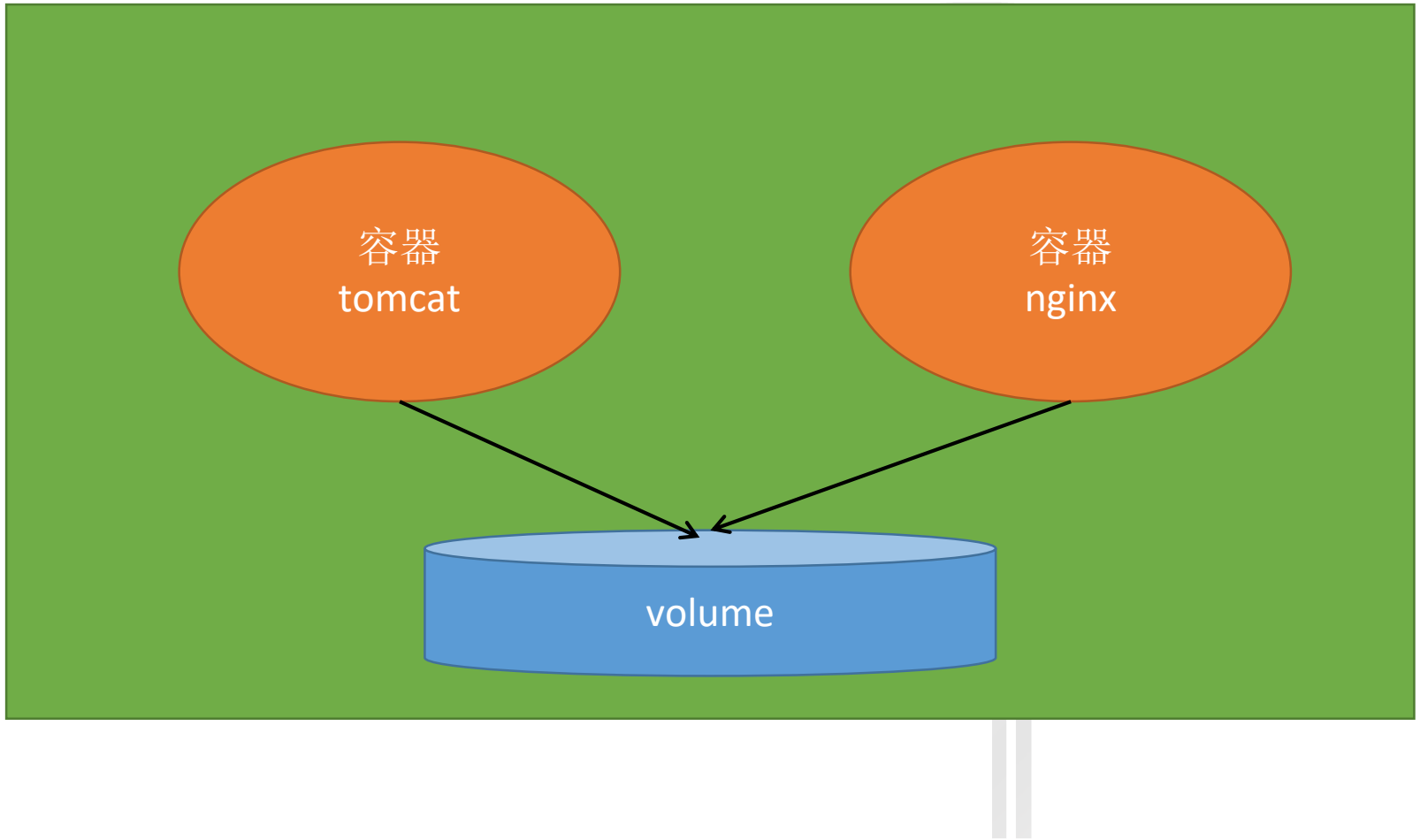
- 部署工作更加便捷和自动化
  - ✓ 将运行环境打包，它使得应用程序在开发、测试、生产系统中的运行环境没有差别
  - ✓ 具备自动部署能力

- 运维更简单

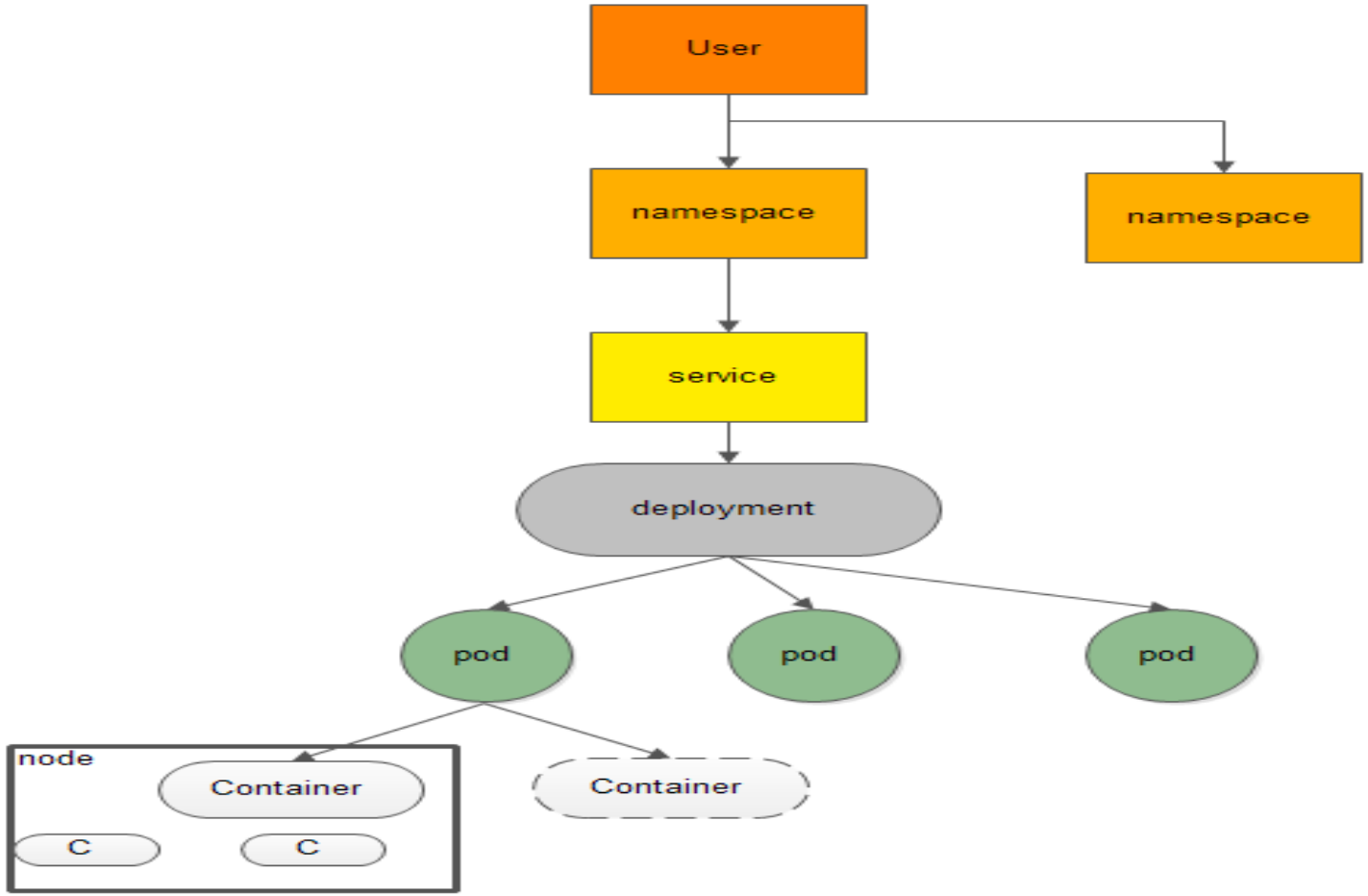
  监控、错误定位、安全、网络、容灾、扩容、资源管控等行为更加便捷



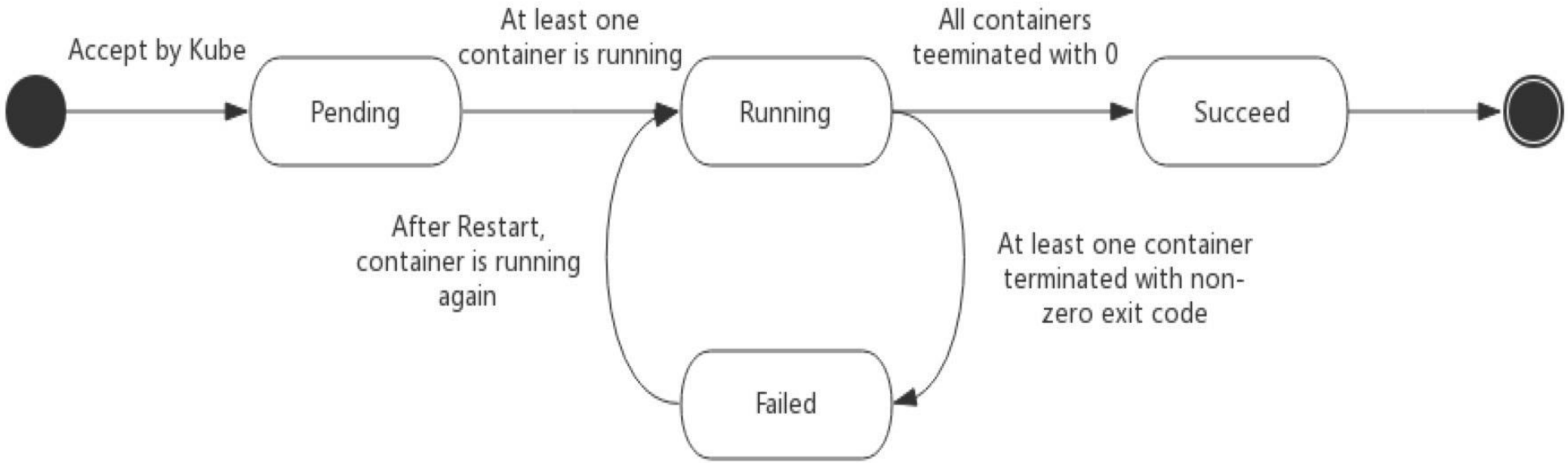
pod详解



pod详解(续)



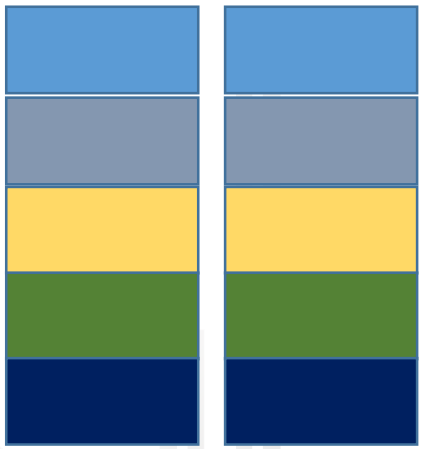
pod生命周期



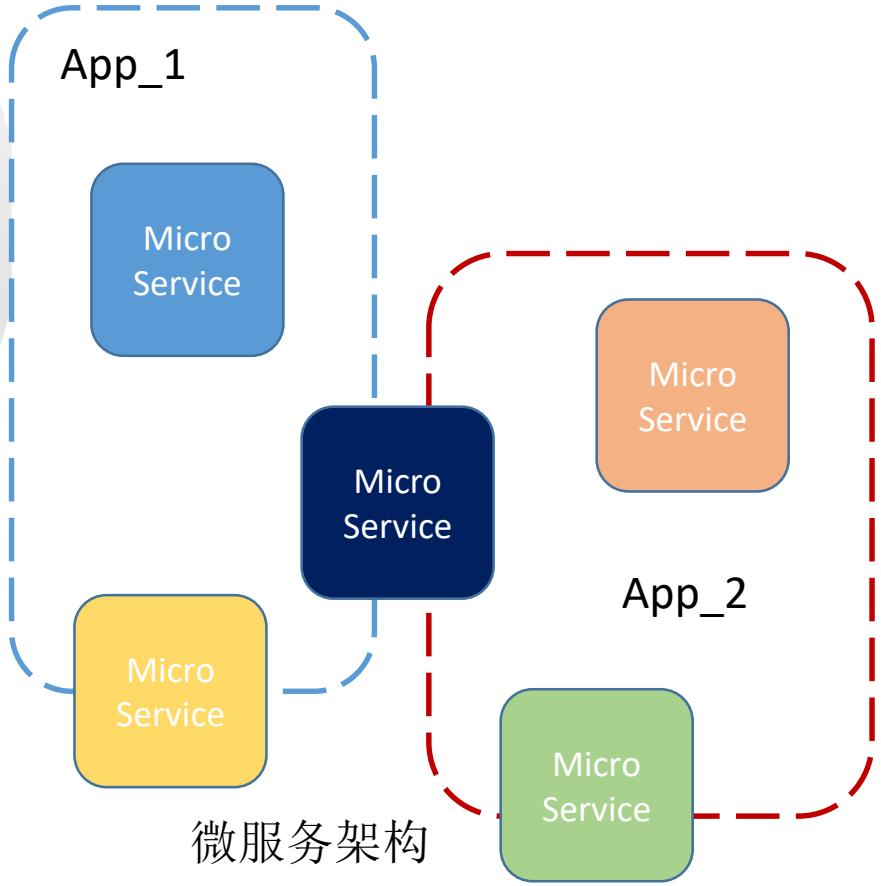
微服务--Service

微服务就是开发一个单纯的，小型的，有意义的功能作为一个单一服务。通过微服务，能够将主要作用是将功能分解到离散的各个服务当中，从而降低系统的耦合性，并提供更加灵活的服务支持。微服务的优点：

- 是一个体量小，用于实现一个特定功能或业务需求的系统
- 可以由一个小的开发组独立完成开发
- 松耦合，服务之间可以独立的开发和部署
- 可以用不同的语言开发
- 通过持续集成工具，可便捷地自动集成部署
- 易于理解， 容易修改和维护
- 易扩展



传统应用架构



微服务架构

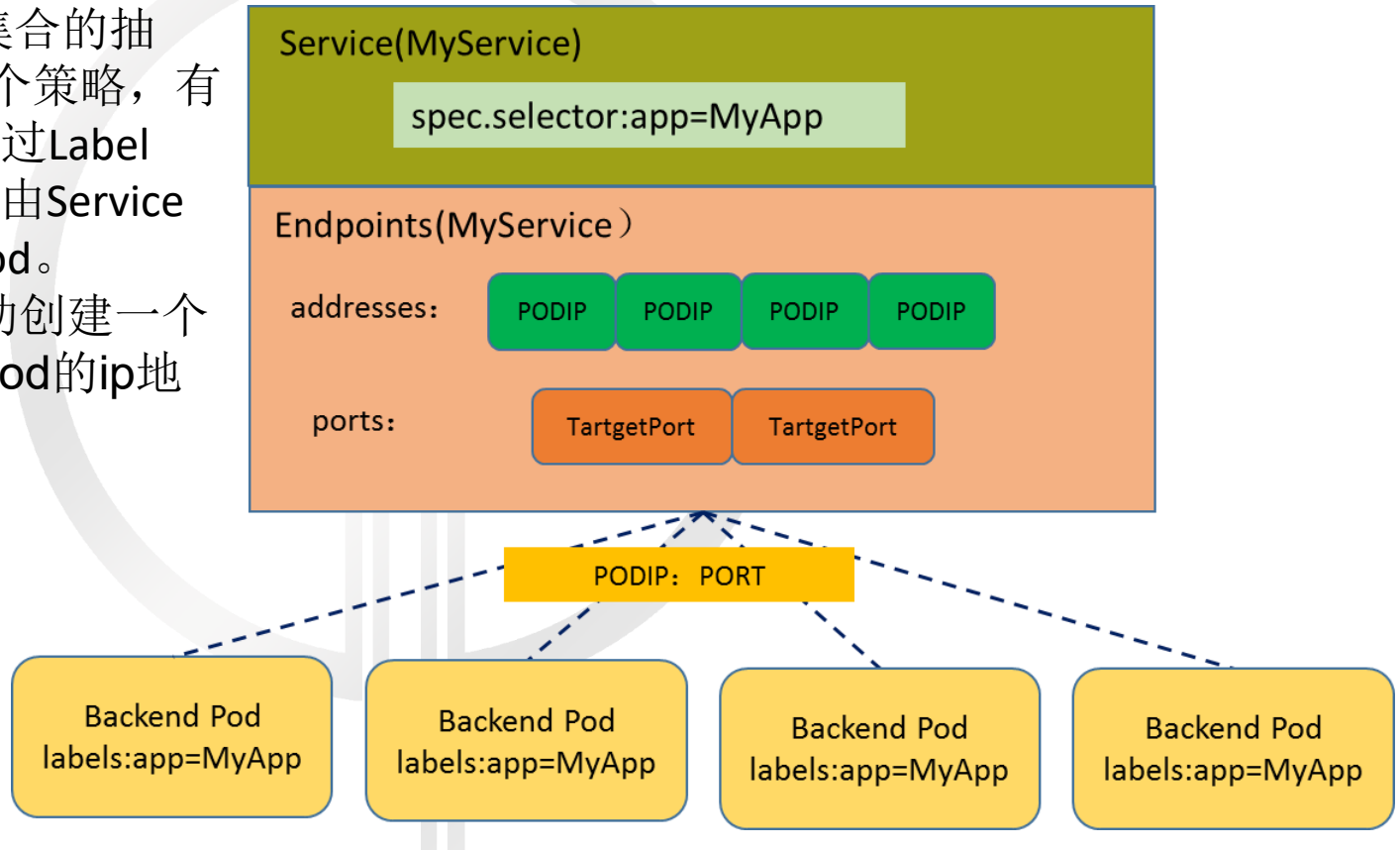
微服务--Service

K8S中的Service是对象资源之一，一个K8S Service是一系列pod的逻辑集合的抽象，同时它是访问这些pod的一个策略，有时候也被称为微服务。Service通过Label Selector和Pod建立关联关系，并由Service决定将访问转向到后端的哪个pod。

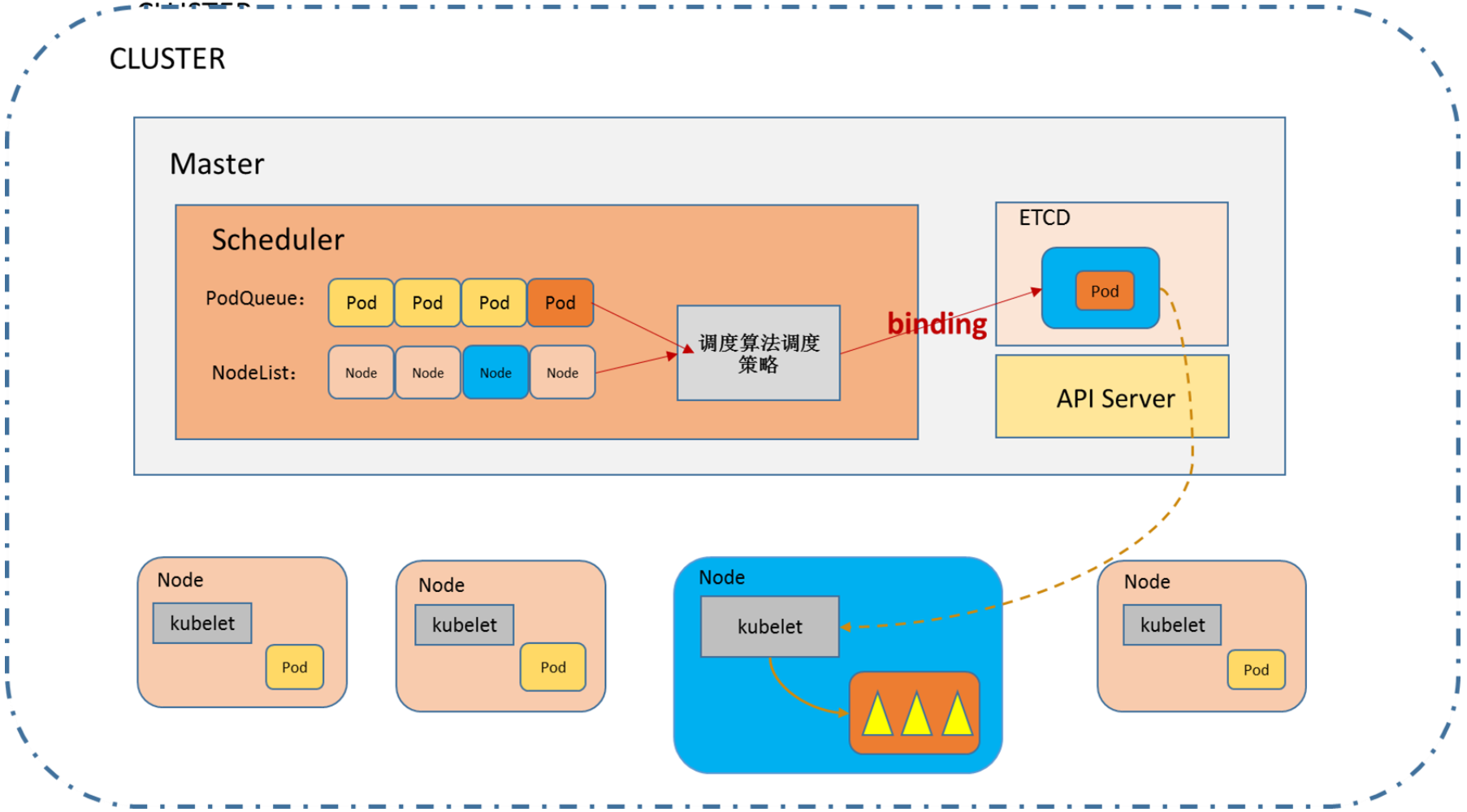
Service被创建后，系统自动创建一个同名的endpoints，该对象包含pod的ip地址和端口号集合

Service分为三类：

1. ClusterIP
2. NodePort
3. LoadBalancer



调度—Scheduler



调度—Scheduler

调度策略分两个步骤，第一个步骤为预选，第二个步骤为优选。Predicates预选调度策略的作用是过滤掉不符合资源要求的节点。Priorities优选调度策略的作用是，从Predicates 预选调度策选出的节点中，选出一个最优的节点。

Scheduler中Predicates 预选调度策略包含：NoDiskConflict、PodFitsResources、PodSelectorMatches、PodFitsHost 、CheckNodeLabelPresence、CheckServiceAffinity和PodFitsPorts策略。在其default算法中，预选调度策略Predicates包括："PodFitsPorts"（PodFitsPorts）、"PodFitsResources"（PodFitsResources）、"NoDiskConflict"（NoDiskConflict）、"MatchNodeSelector"（PodSelectorMatches）和"HostName"(PodFitsHost)，即每个节点只有通过前面提及的5个默认预选策略后，才能初步被选中，然后供Priorities调度策略挑选。

Scheduler中Priorities优选调度策略包含：LeastRequestedPriority、CalculateNodeLabelPriority和BalancedResourceAllocation。

调度—Kubelet

在Kubernetes集群中，在每个Node节点（又称Minion）上都会启动一个Kubelet服务进程。该进程用于处理Master节点下发到本节点的任务，管理Pod及Pod中的容器。

每个Kubelet进程会在API Server上注册节点自身信息，定期向Master节点汇报节点资源的使用情况，并通过cAdvise监控容器和节点资源。

它负责创建或销毁Pod，通过探针检测Pod的状态，并通过cAdvise监控Pod的状态。

Pod通过两类探针来检查容器的健康状态。一个是livenessProbe探针，用于判断容器是否健康，LivenessProbe探针告诉Kubelet一个容器什么时候处于不健康的状态。另一类是readinessProbe探针，用于判断容器是否启动完成，且准备接收请求。

容错—RC 副本控制

在运维过程中，我们常常会发现这样的问题，在负载均衡管控下的多台后端服务器中，由于种种原因（硬件或软件的原因），总会出现某台服务器宕机或服务器上的应用失效的情况。运维人员一般的做法是：

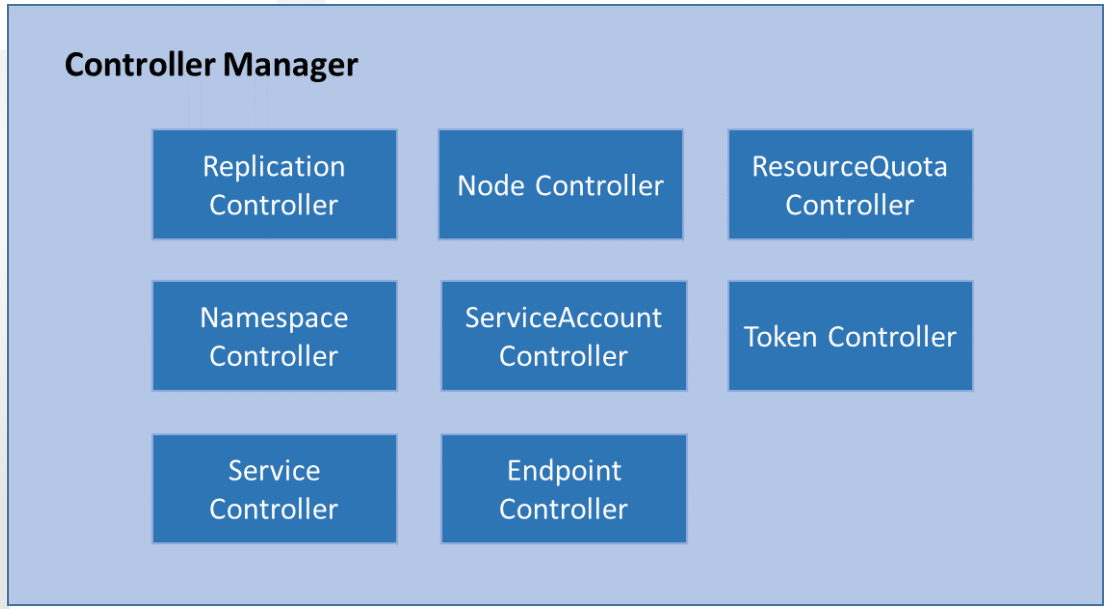
- 购买或另寻主机
- 安装操作系统
- 设置运行环境
- 调通网络
- 部署应用
- 将出故障的主机下架
- 配置负载均衡

K8S的Replication Controller负责管控的pod的副本数，一旦发现其中某个副本出现故障，Replication Controller自动补齐副本。

控制—Controller Manager

Controller Manager为集群内部的管理控制中心，负责集群内的副本控制（Replication Controller）、端点控制（Endpoint Controller）、命名空间控制（Namespace Controller）和服务账号控制（ServiceAccount）等，如图所示.

它包含Replication Controller、Node Controller、ResourceQuota Controller、Namespace Controller、ServiceAccount Controller、Token Controller、Service Controller以及Endpoint Controller等多个控制器。将来可能会把这些控制器拆分并且提供插件式的实现。Controller Manager是这些控制器的核心管理者。在智能和自动的应用中，通常情况下会通过一个操纵系统来不断修正系统的状态。在Kubernetes集群中，每个Controller就是一个操纵系统，它通过API Server监控系统的共享状态，并尝试着将状态从现有状态修正到期望的状态。



扩容–Scaling

随着应用系统用户的增加，往往需要通过扩容应用系统来承受更高的负载。传统的扩容方式大致需要下列步骤：

- 购买主机
- 安装操作系统
- 设置运行环境
- 调通网络
- 部署应用
- 配置负载均衡

K8S通过调整Replication Controller管控的pod的副本数，来实现应用系统扩容，它只需要一行命令就能完成，举例如下：

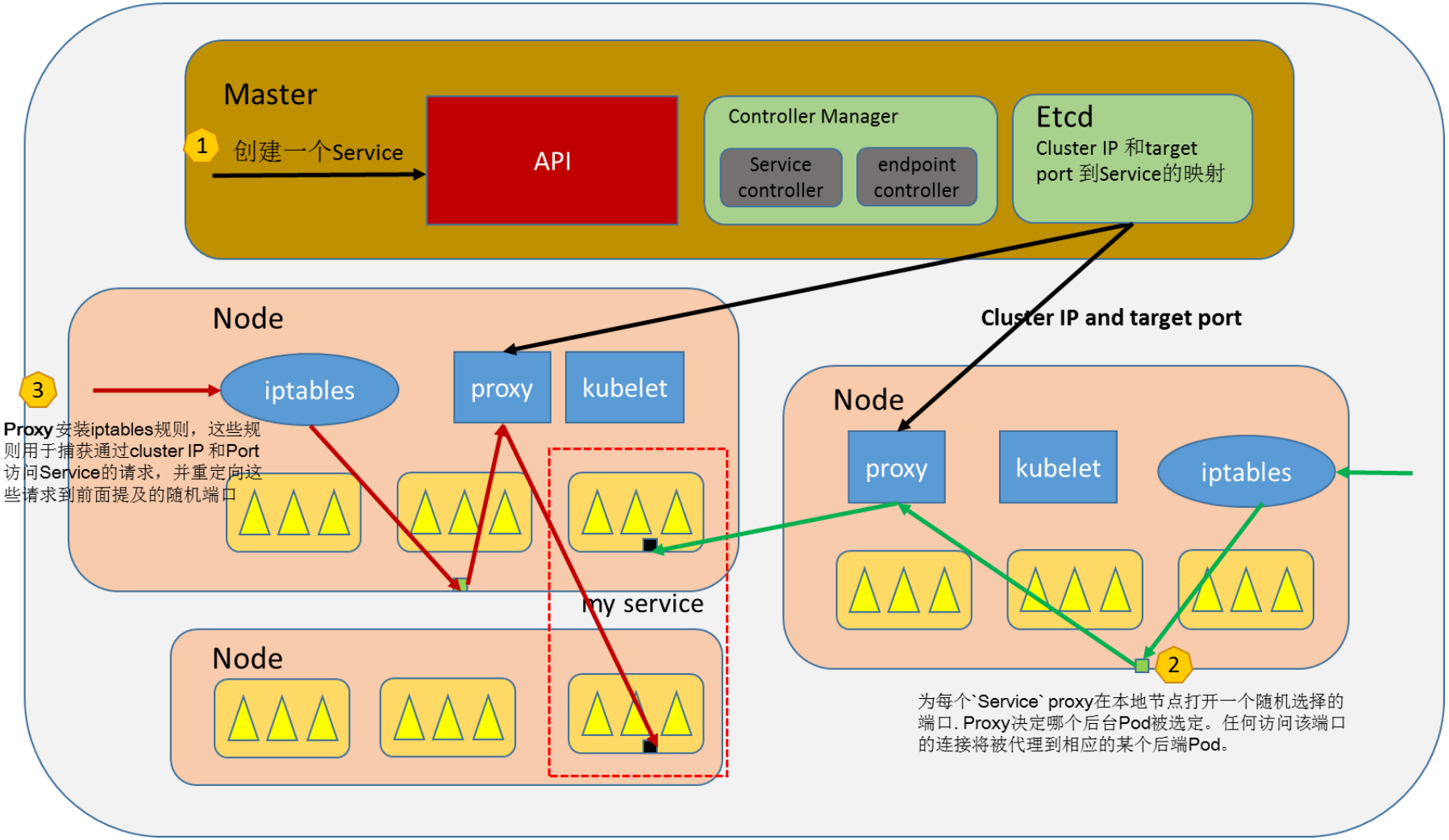
kubectl scale --replicas=3 replicationcontrollers foo

升级—Rolling Update

应用系统无缝升级是一个高可用系统追求的目标，它对运维的要求甚高，而且工作量较大。K8S通过Replication Controller的Rolling Update功能，实现应用系统无缝升级。副本控制器被设计成通过逐个替换Pod的方式来辅助服务的滚动更新。推荐的方式是创建一个新的，只有一个副本的RC，随着新的RC副本数量加1，则旧的RC的副本数量减1，直到这个旧的RC的副本数量为零，然后删除该旧的RC。

通过上述模式，即使在滚动更新过程中发生不可预料的错误，Pod集合的更新也都在可控范围内。在理想情况下，滚动更新控制器需要将准备就绪的应用考虑在内，并保证在集群中任何时刻都有足够数量的可用Pod。

负载均衡—Service & Kube-proxy



服务发现—容器ENV&DNS

K8S支持两种服务发现方式，一种是容器环境变量，另一种是DNS。K8S创建Service后，会在每个容器中添加环境变量，例如：

“{SVCNAME}_SERVICE_HOST”和“{SVCNAME}_SERVICE_PORT”

用户通过这些环境变量访问服务。

但是使用环境变量是有限制条件的，即Service必须在Pod之前被创建出来，然后系统才能在新建的Pod中自动设置与Service相关的环境变量。DNS则没有这个限制，其通过提供全局的DNS服务器来完成服务的注册与发现。Kubernetes提供的DNS由以下三个组件组成。

- etcd：DNS存储。
- kube2sky：将Kubernetes Master中的Service（服务）注册到etcd。
- skyDNS：提供DNS域名解析服务。

这三个组件以Pod的方式启动和运行，所以在一个Kubernetes集群中，它们都可能被调度到任意一个Node节点上去。

Kubernetes常用命令

Pod相关

1.创建Pod

2.删除Pod

3.查看Pod

4.进入Pod管理的容器

5.描述资源

6.ReplicationController & Replica Set、

查看：kubectl get rs

7.横向扩展：kubectl scale

8.Deployment

查看：kubectl get deployment

升级：kubectl set image deployment

kubectl rollout history deployment nginx-deployment

kubectl rollout undo deployment nginx-deployment

kubectl rollout history deployment nginx-deployment

直接使用Pods有什么问题

- 当使用ReplicationController或者ReplicaSet做水平扩展，Pods有可能被Terminal
- 当使用Deployment去更新Image Version，Pods有可能被Terminal，再创建新的Pods

Kubernetes常用命令

Service相关

1.创建Service

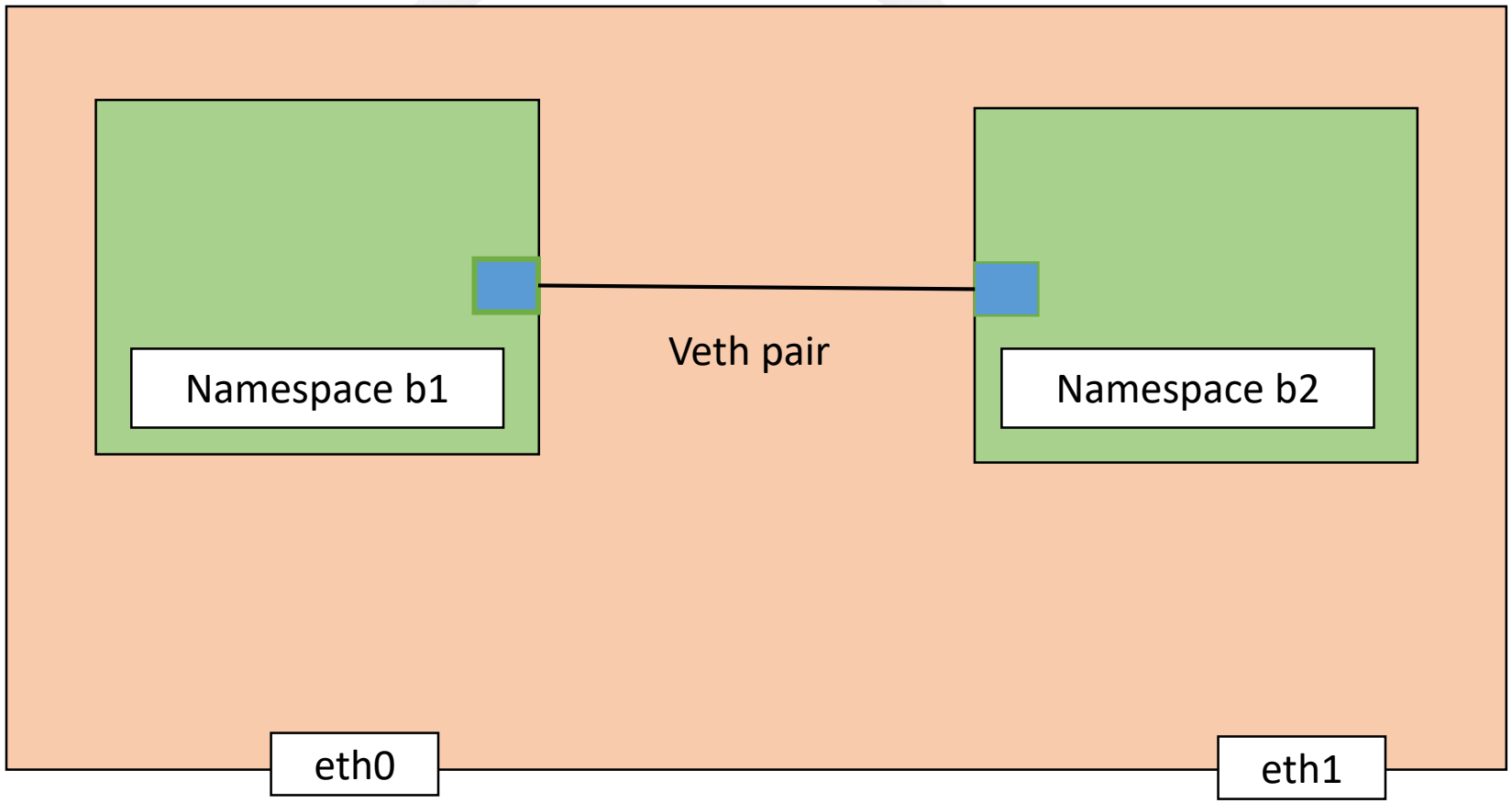
kubectl expose pods podname --type=NodePort

2.删除Service

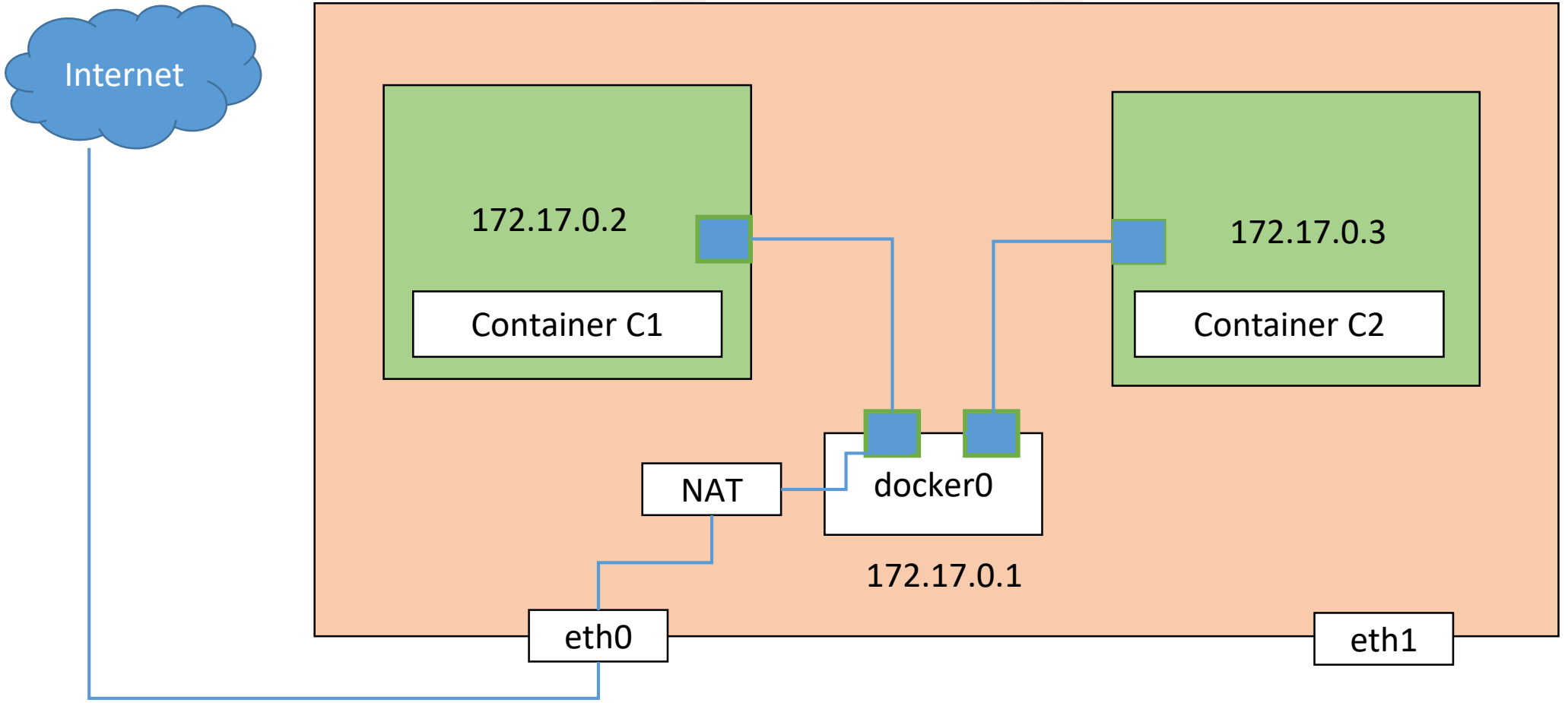
3.查看Service

kubectl get svc

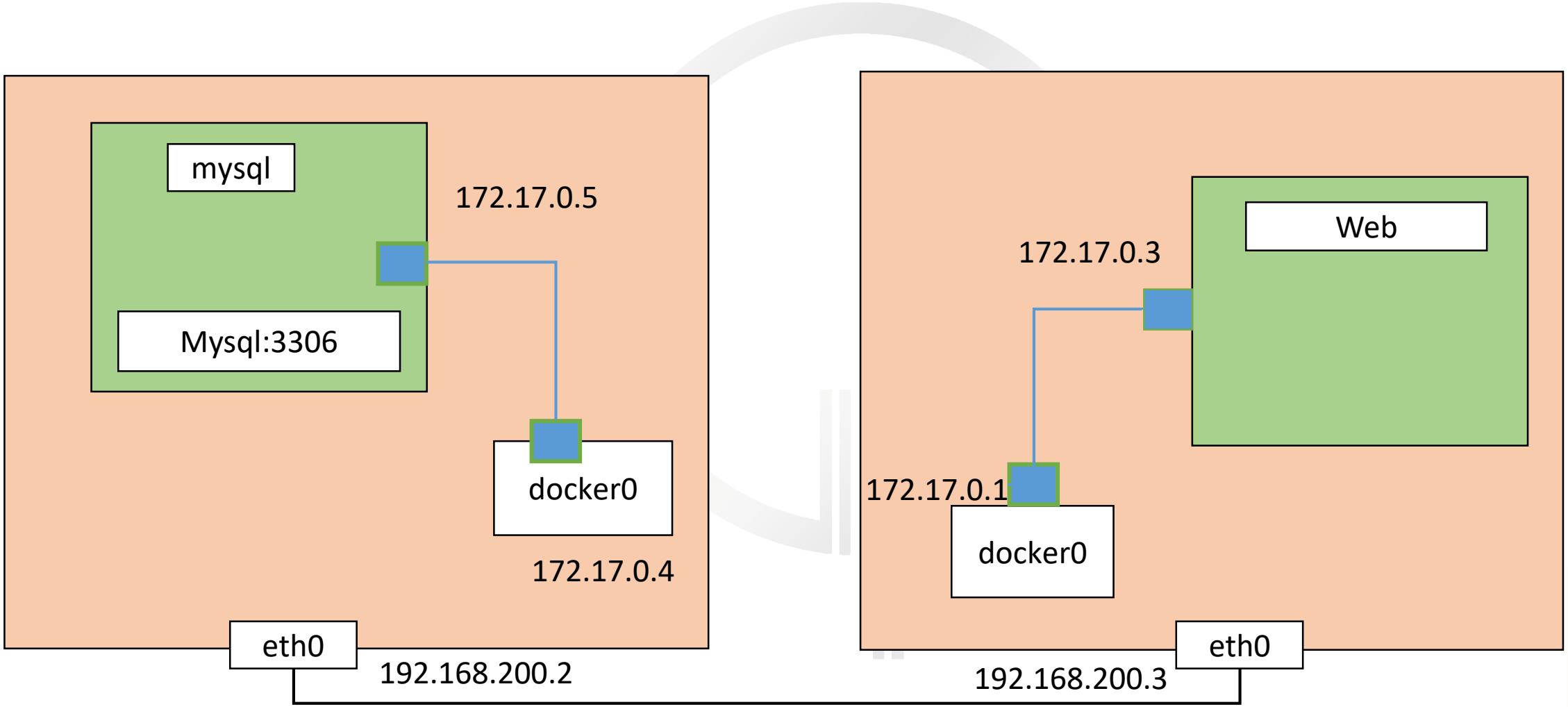
Docker & Kubernetes网络介绍



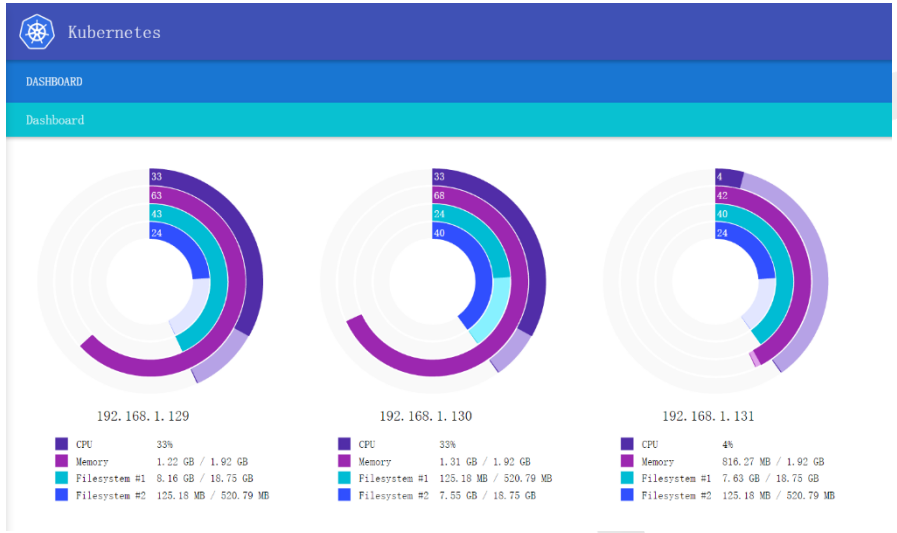
Docker & Kubernetes网络介绍

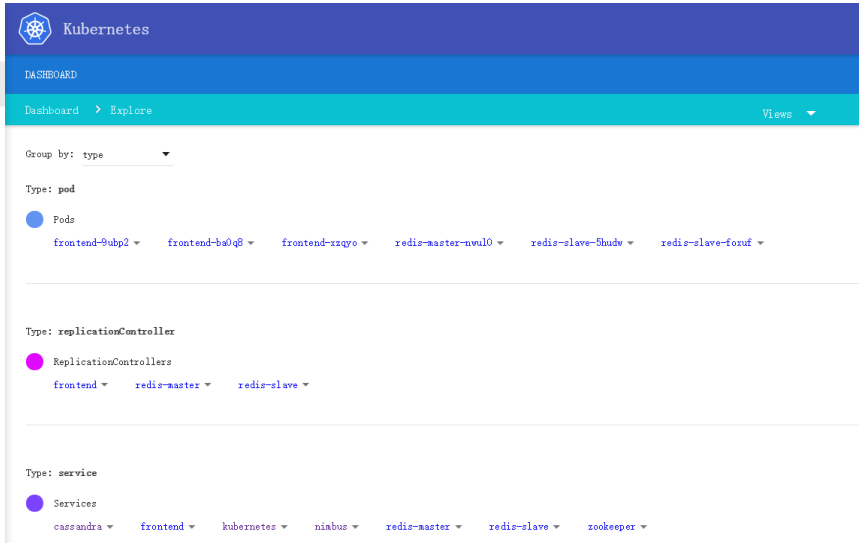


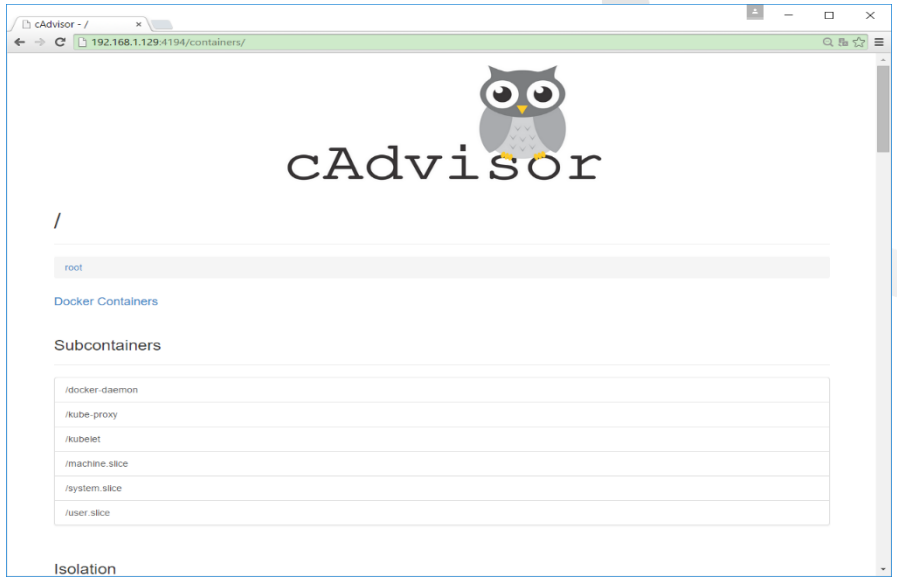
Docker & Kubernetes 多机器通信

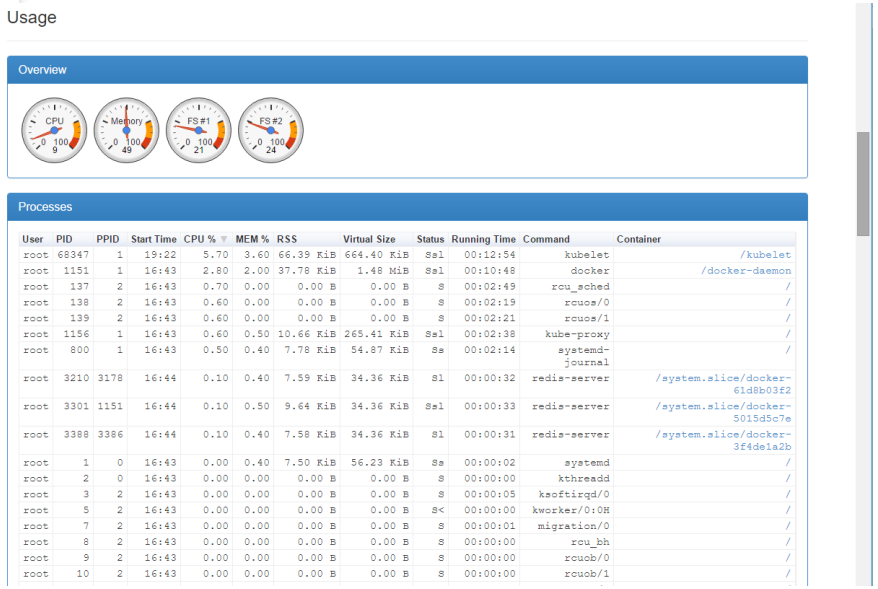


监控—Kube-UI&Cadvise&InfluxDB&Heapster









专业互联网IT教育服务平台

谢谢观看

- 大白老师QQ号：1828627710