



OpenKruise: 全方位的提升Pod生命周期管理能力

赵明山

阿里云 - 容器服务 技术专家



目 录

什么是OpenKruise?

01

如何提升Pod生命周期管理能力

02

社区发展与规划

03

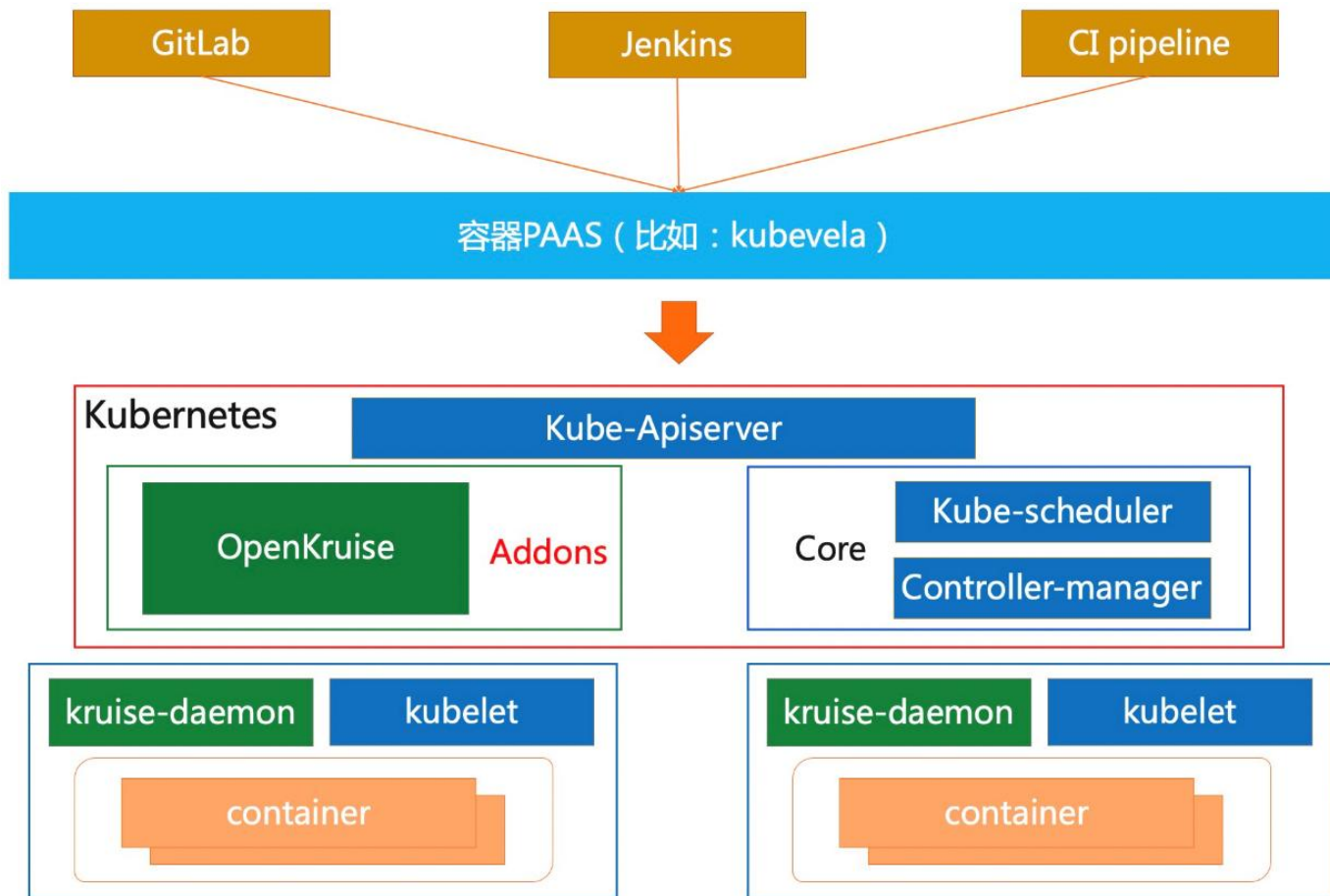
第一部分

什么是OpenKruise?

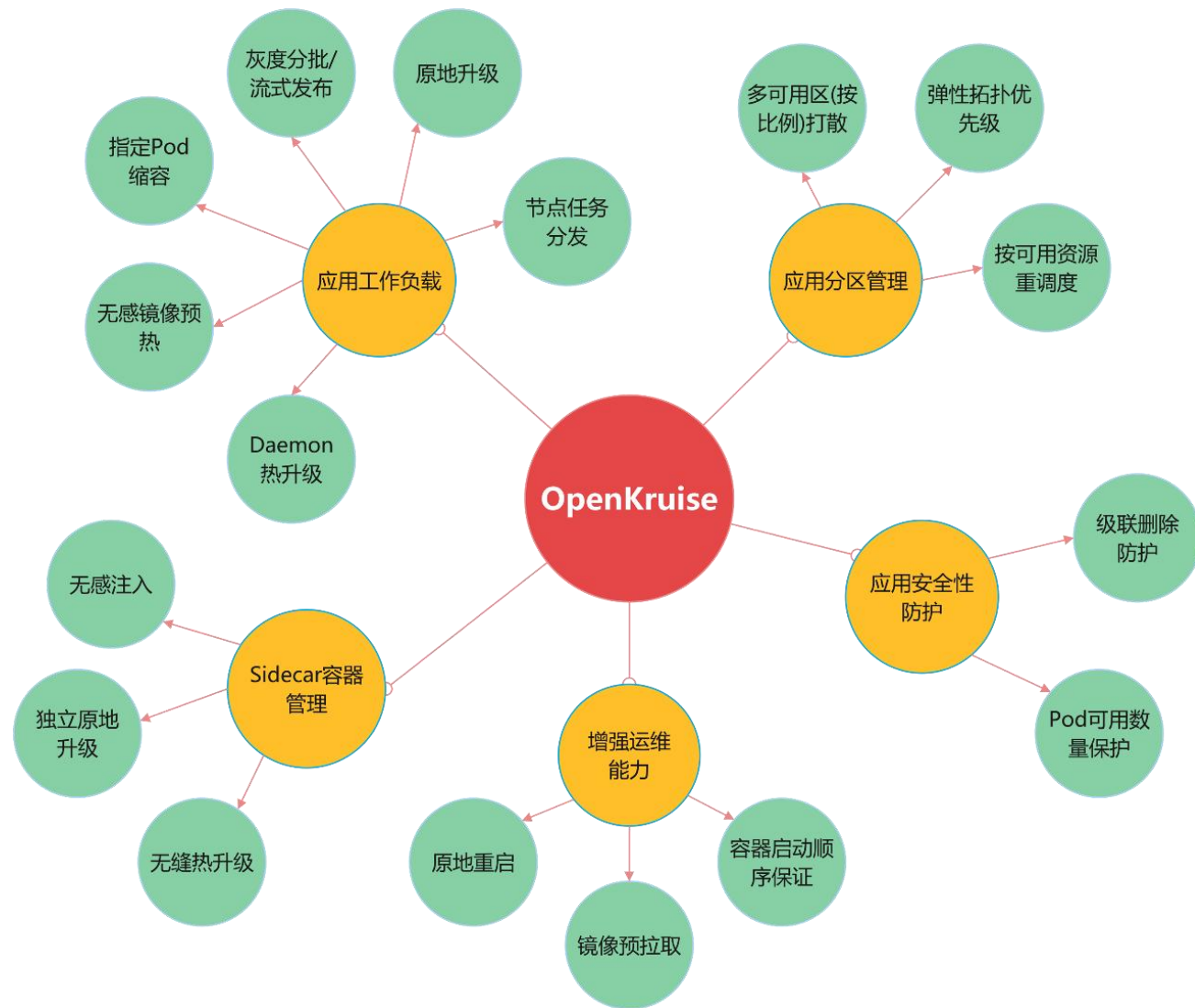


OpenKruise 是什么？

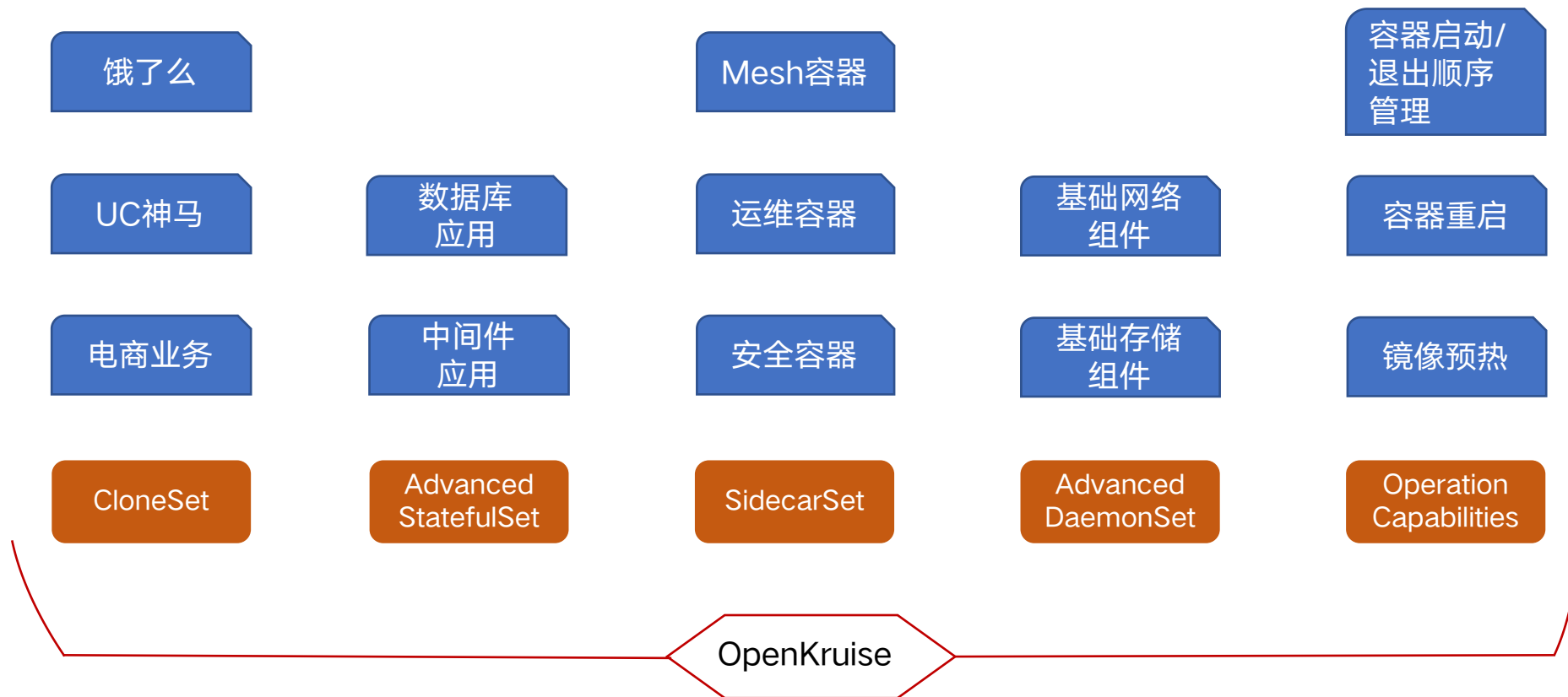
- 基于 Kubernetes 扩展的应用管理套件
 - 任意纯净的 Kubernetes 集群中
 - 通过CRD的方式扩展能力，与上层Paas对接
- 成本低



OpenKruise 包含哪些能力？



OpenKruise 是阿里巴巴应用管理和部署的基座



第二部分

如何提升Pod生命周期管理能力？



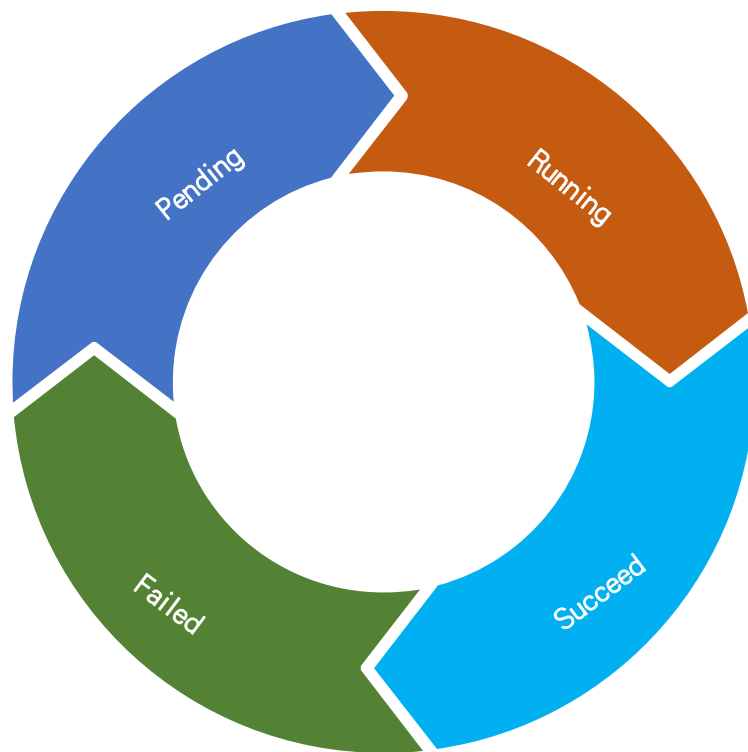
什么是 Pod 生命周期管理？

容器重启策略

- Always
- OnFailure
- Never

容器 Lifecycle

- PostStart
- PreStop



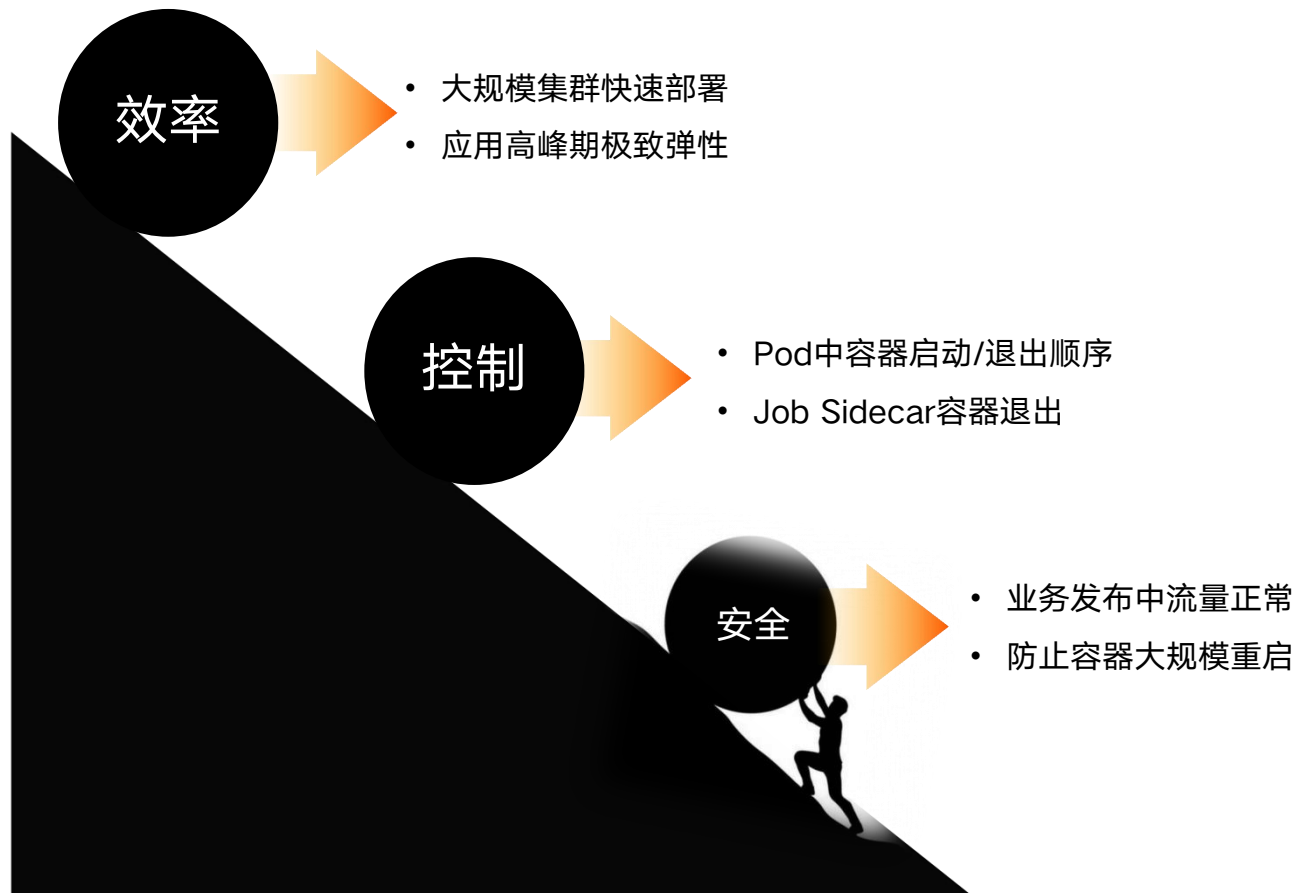
容器 Probe

- Startup Probe
- Liveness Probe
- Readiness Probe

Pod Condition

- ContainersReady
- Ready
- Initialized
- PodScheduled

大规模场景对Pod生命周期管理提出了新的要求？

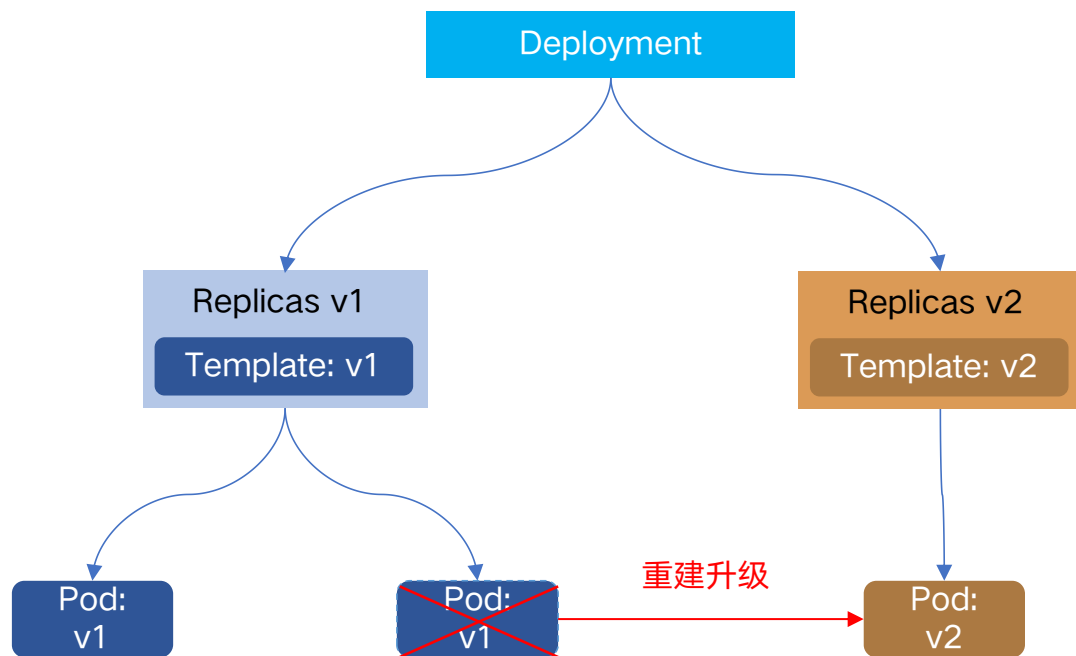


- **✘** 任何升级都会导致Pod重建
- **✘** 不能保证Pod中容器启动/退出顺序
- **✘** Job Sidecar容器无法退出
- **✘** 应用发布与LoadBalancer无法联动
- **✘** Liveness Probe 没有全局控制

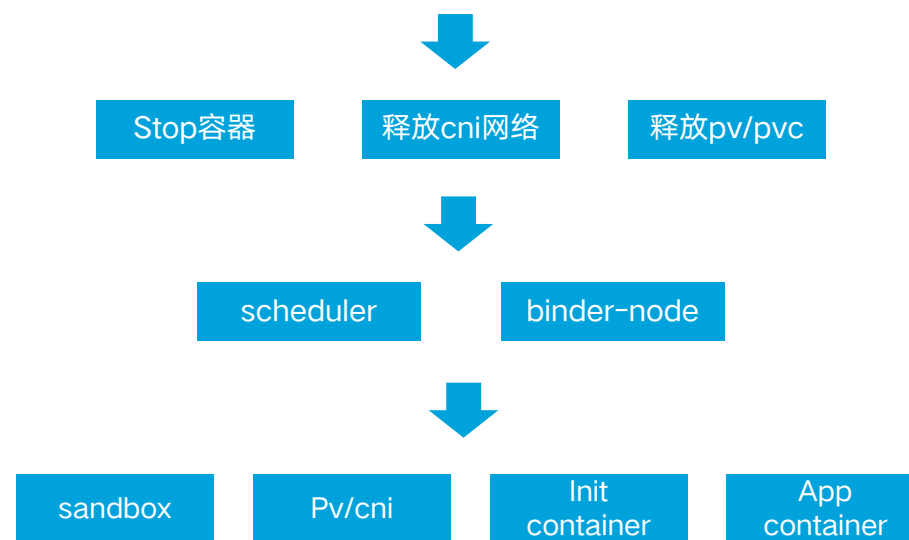
Pod生命周期管理能力

核心能力1 - 原地升级

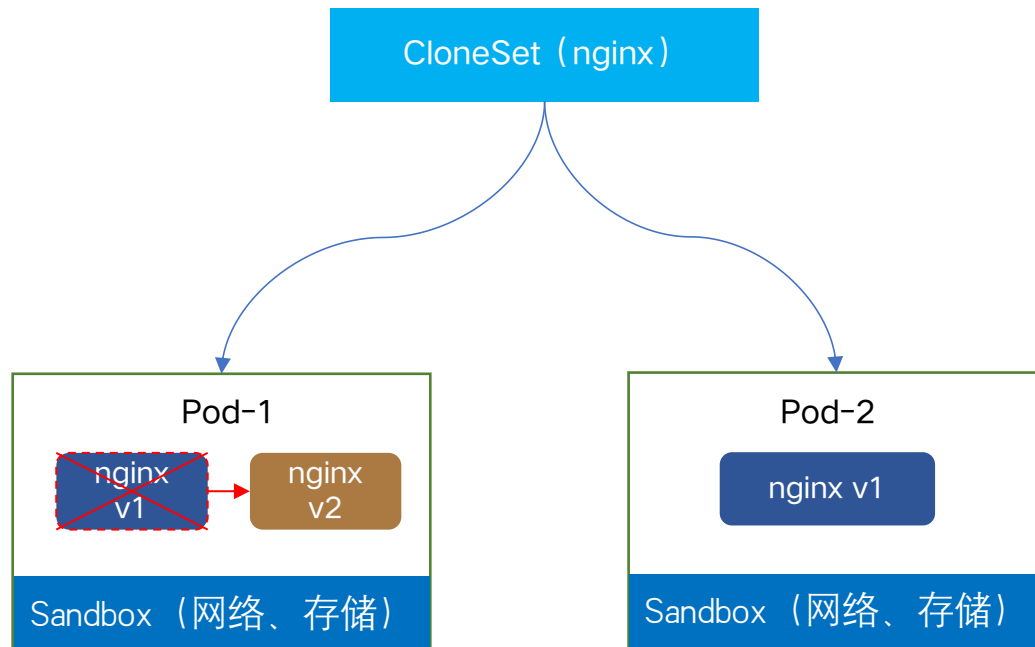
Kubernetes - Pod维度的不可变基础设施



Pod重建流程复杂，拉镜像耗时，无预热情况下新拉镜像需要几分钟；
Pod IP 变化、存储清空，新起 Pod 应用进程需要重新预热；



容器维度的管控能力

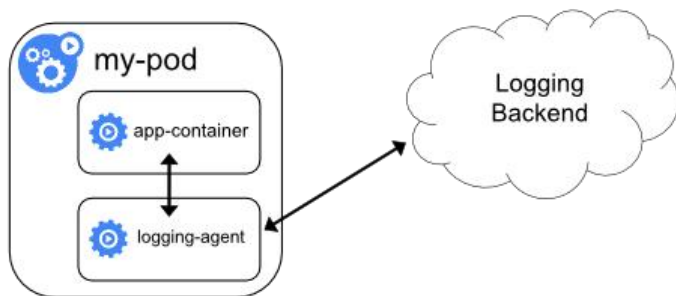


原地升级:

- 节省了**调度**的耗时，Pod 的位置、资源都不发生变化
- 节省了**分配网络**的耗时，Pod 还使用原有的 IP
- 节省了**分配、挂载PV**的耗时，Pod 还使用原有的 PV（且都是已经在 Node 上挂载好的）
- 节省了**大部分拉取镜像**的耗时，因为 Node 上已经存在了应用的旧镜像，当拉取新版本镜像时只需要下载少数的几层 layer
- 原地升级 Pod 中某个容器时，**其他容器保持正常运行**，网络、存储均不受影响

核心能力2 - Sidecar容器管理

容器设计模式：
Sidecar

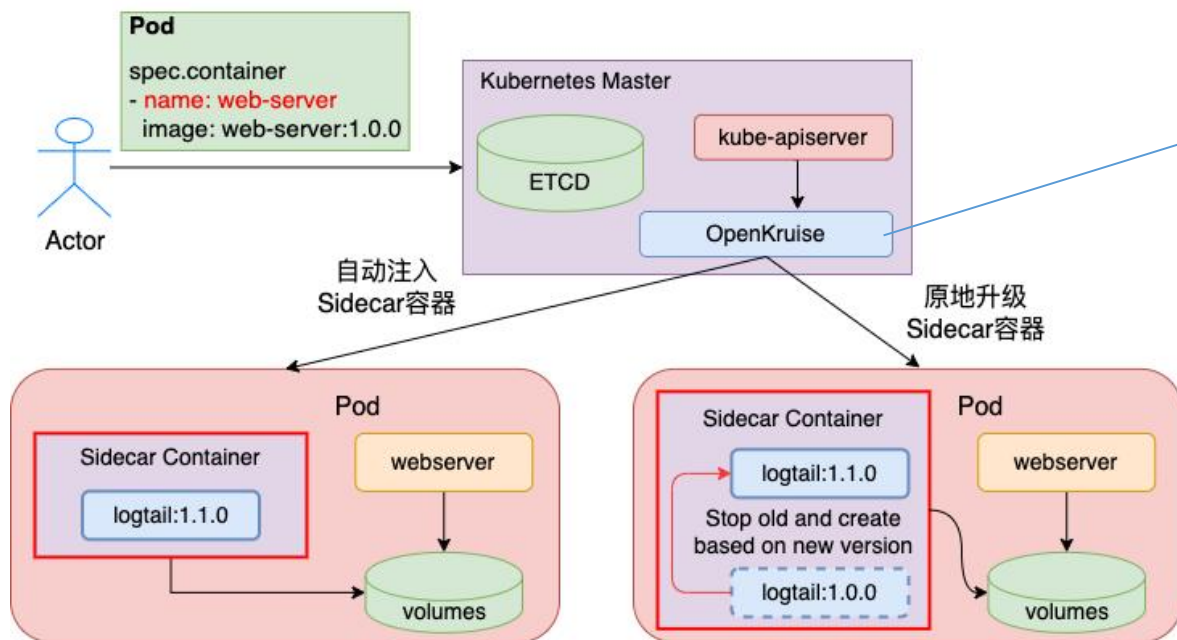


通过在Pod里定义专门容器，来执行主业务容器需要的辅助工作

- 比如：
 - 日志收集
 - Debug应用
 - Service Mesh Istio
- 优势：
 - 将辅助能力同业务容器解耦，实现独立发布和能力重用
- 缺点：
 - Sidecar升级将导致业务Pod重建
 - 业务Pod配置耦合（运维、安全、代理）多种sidecar，增加配置的复杂性，以及sidecar更新难度

```
apiVersion: apps/v1
kind: Deployment
spec:
  replicas: 5
  template:
    spec:
      containers:
      - name: nginx
        image: nginx:alpine
        volumeMounts:
        - name: log-volume
          mountPath: /var/log
      - name: logtail-sidecar
        image: logtail:v1
        volumeMounts:
        - name: log-volume
          mountPath: /var/log
      volumes:
      - name: log-volume
        emptyDir: {}
```

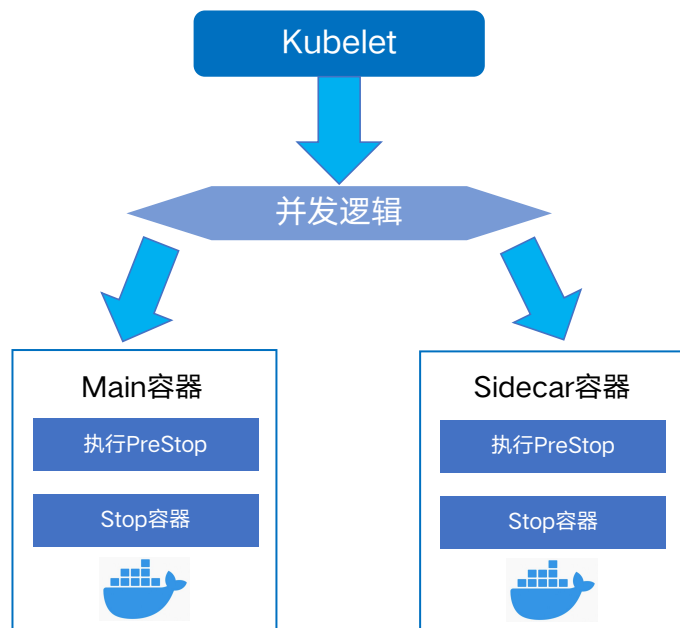
SidecarSet: 管理Sidecar容器的利器



```
kind: SidecarSet
metadata:
  name: logtail
spec:
  selector:
    app: webserver
  updateStrategy:
    type: RollingUpdate
    maxUnavailable: 1
  containers:
    - name: logtail-sidecar
      image: logtail:v1
      volumeMounts:
        - name: log-volume
          mountPath: /var/log
```

如何控制Sidecar容器的退出顺序？

Kubelet并发退出Pod中多个容器，Sidecar容器有可能早于Main容器退出，导致服务异常



触发场景（Pod）

- 滚动升级、驱逐、缩容

K8S官方方案（Kubelet代码改动过大，一直搁置）

- 2018年社区有提出相关Proposal，Sidecar先于Main容器启动，晚于Main容器退出

社区常用方案（定制化，不通用）

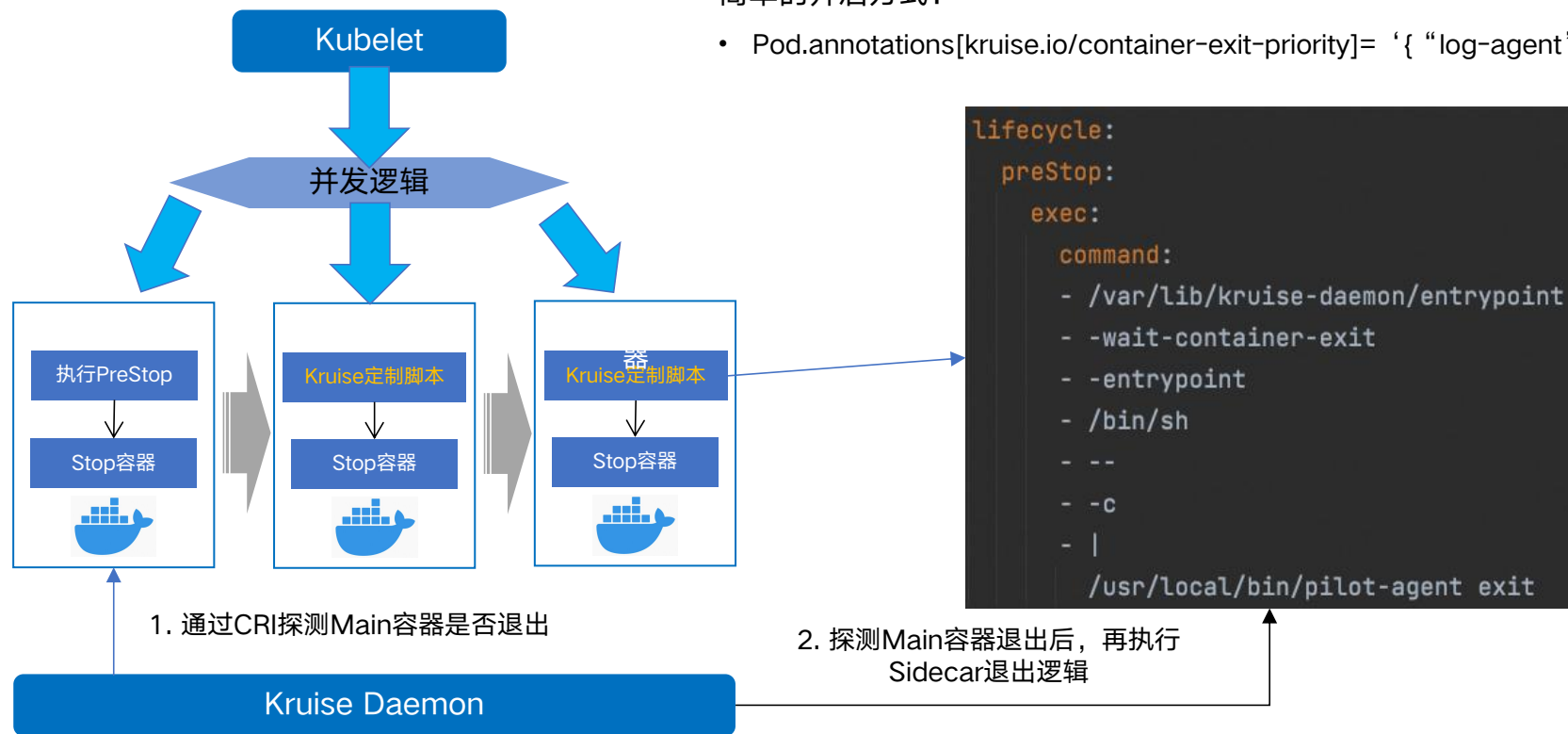
- Sidecar 容器配置等待时间
- Main容器与Sidecar容器共享目录，实现退出逻辑

社区首个通用的容器退出优先级能力

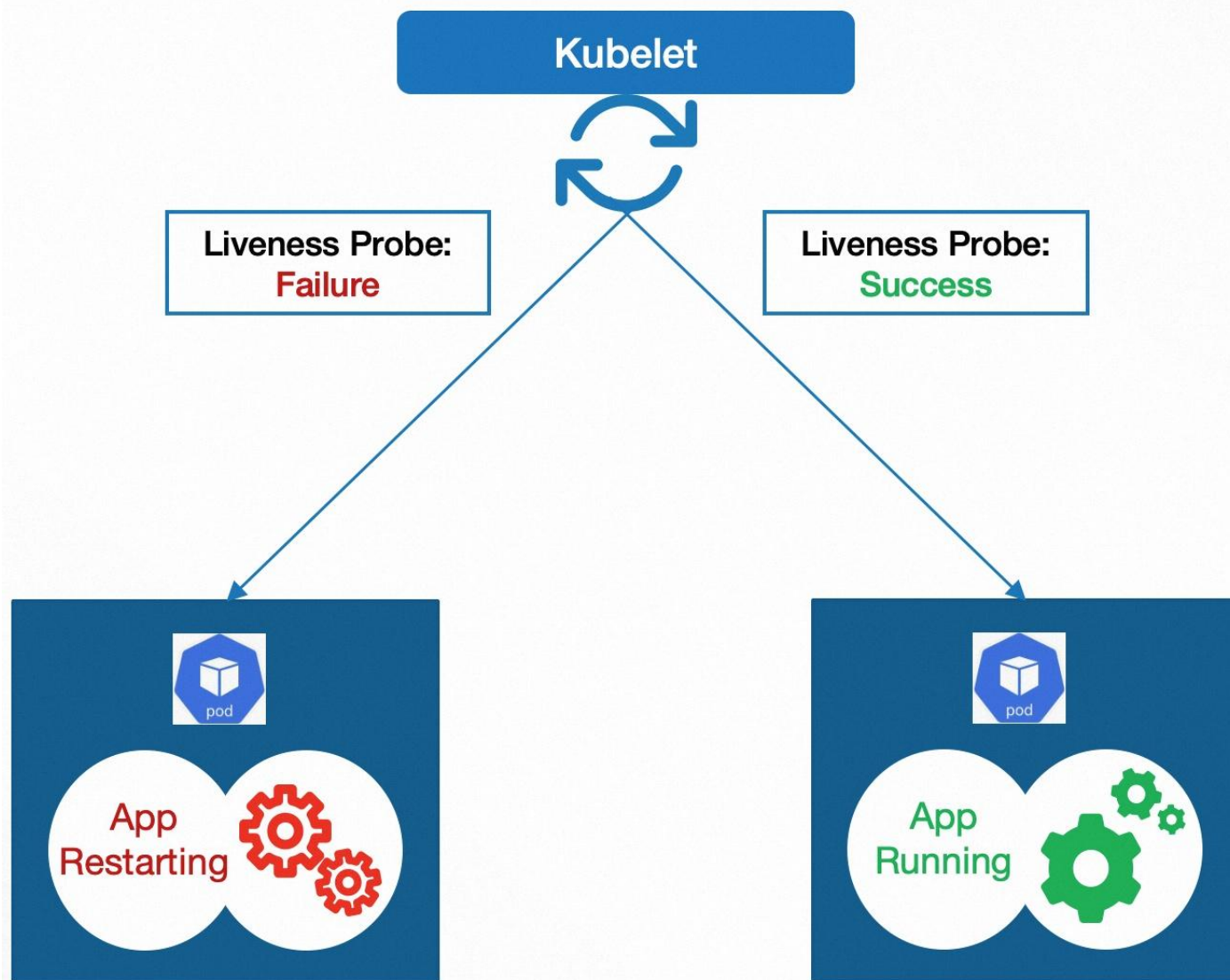
容器按照顺序退出: Main -> LogAgent -> Envoy

简单的开启方式:

- Pod.annotations[kruise.io/container-exit-priority]= '{ "log-agent" :1, "envoy" :2}'



核心能力3 – 增强的LivenessProbe能力



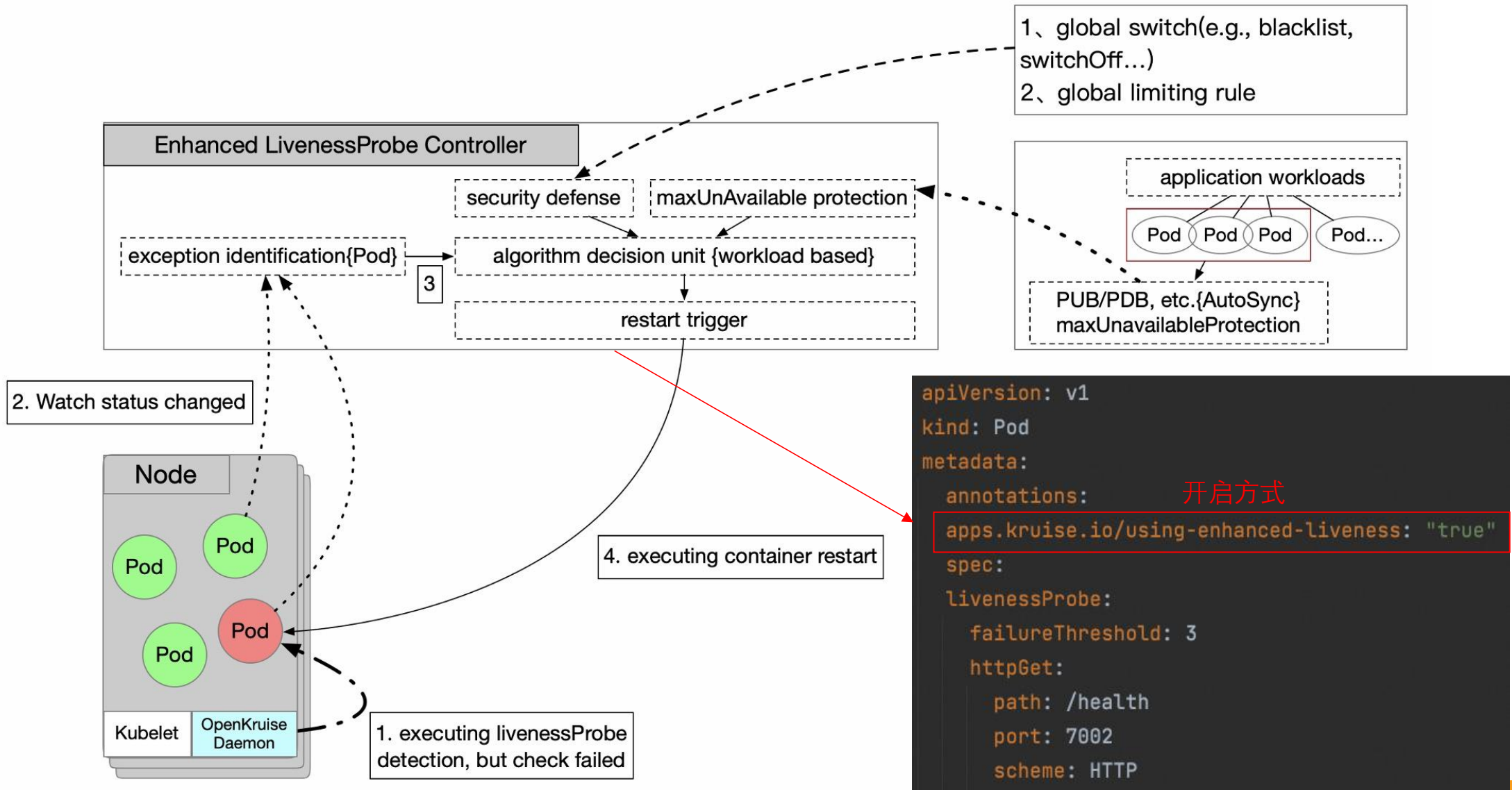
社区原生Liveness Probe

- 不间断的探测容器健康状态
- 异常，立即重启

不足:

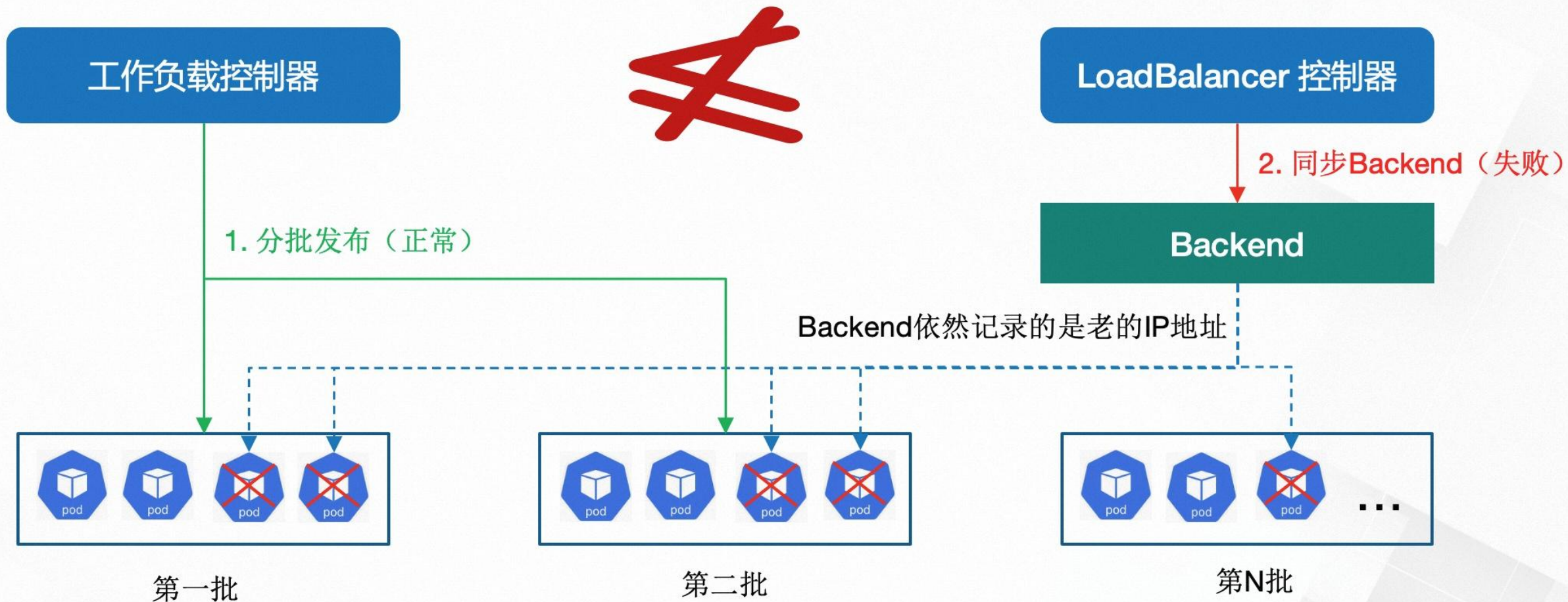
- 没有全局视角，极端情况会一次性重启所有的容器，导致业务异常

OpenKruise 提供全局的Liveness Probe能力



核心能力4 - Pod Lifecycle (优雅上下线)

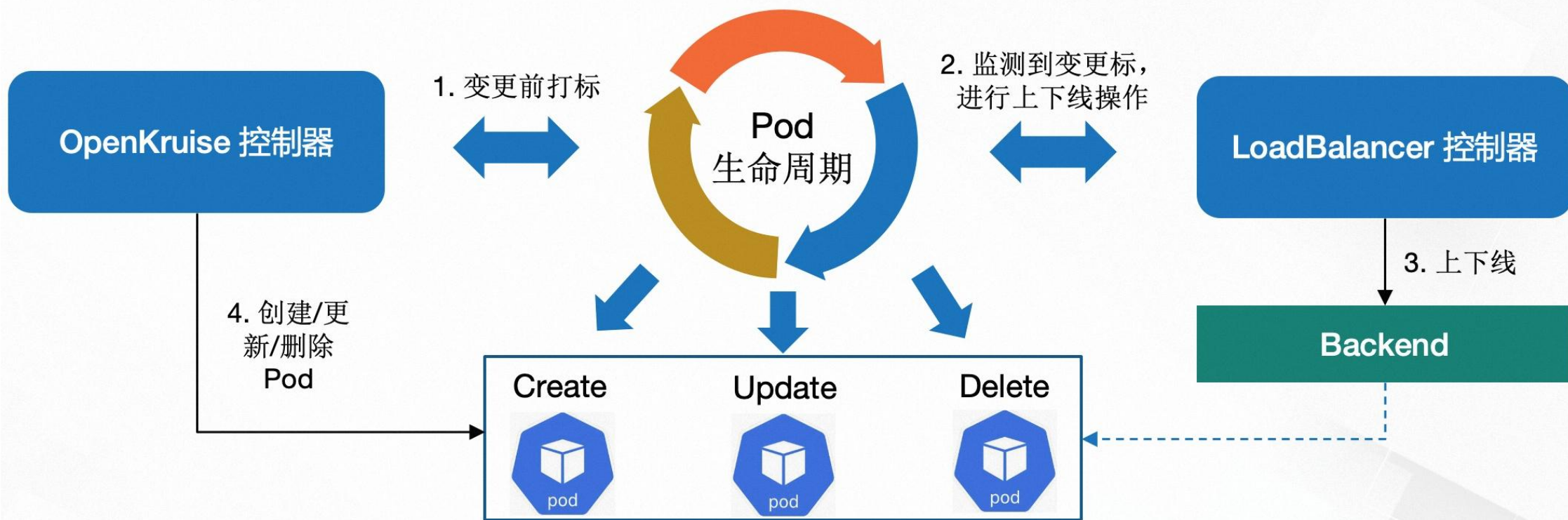
LoadBalancer同步Backend失败，并不能阻塞或通知控制器停止发布



OpenKruise 支持 Pod 优雅上下线

工作负载: CloneSet、Advanced StatefulSet、Advanced DaemonSet

- PreNormal
- Updating
- PreDelete



第三部分

社区发展与规划



逐渐扩展 OpenKruise 能力边界

Kruise Rollout

- 无侵入的渐进式发布系统

OpenKruiseGame

- 云原生游戏最佳实践

ControllerMesh

- Operator灰度和隔离系统

社区运营

<https://github.com/openkruise/kruise>

- Star: 3.9k
- Contributor: 150+
- 2023 年 CNCF Sandbox -> Incubation
- 双周周会（周四晚19点30）

业界用户：

- 国内：阿里巴巴、蚂蚁、携程、苏宁、OPPO、小米、斗鱼TV、有赞、Boss直聘、小红书等 45+ 登记企业用户
- 国外：LinkedIn、Lyft、Bringg、Arkane Systems、Spectro Cloud 等 5+ 登记：


OpenKruise 社区交...

服务



1531人



 扫一扫群二维码，立刻加入该群。



Q&A