安全沙箱容器在边缘计算场景的实践

高步双（花名：江博）

阿里云容器服务技术专家

自我介绍

高步双（花名：江博），阿里云容器服务安全容器负责人，2017 年加入阿里，主要从事 Kubernetes 相关产品的设计研发工作，负责或参与容器服务安全容器、边缘容器、SAE、资源调度系统稳定性分析引擎、Kubernetes&YARN 混部等相关产品、项目的设计和研发工作。

目录

安全容器带来的机遇与挑战

据 Gartner 预测 2019 年一半以上的企业会在其开发和生产环境中使用容器部署应用，容器技术日趋成熟稳定，然而 42% 以上的受访者表示**安全**成为其容器化的最大障碍之一，主要包括容器运行时安全、镜像安全和数据安全加密。

图片来源: https://i2.wp.com/foxutech.com/wp-content/uploads/2017/03/Docker-Security.png?resize=696%2C345&ssl=1

端到端云原生安全架构

基础架构安全

RAM认证，细粒度RAM授权，支持审计能力

专有云

公共云

安全软件供应链

镜像签名

镜像扫描

安全合规

静态加密 BYOK

DevSecOps

安全分发

容器运行时安全

KMS集成

内置安全设计

网络策略

安全沙箱容器

运行时安全

租户管理

安全容器运行时对比

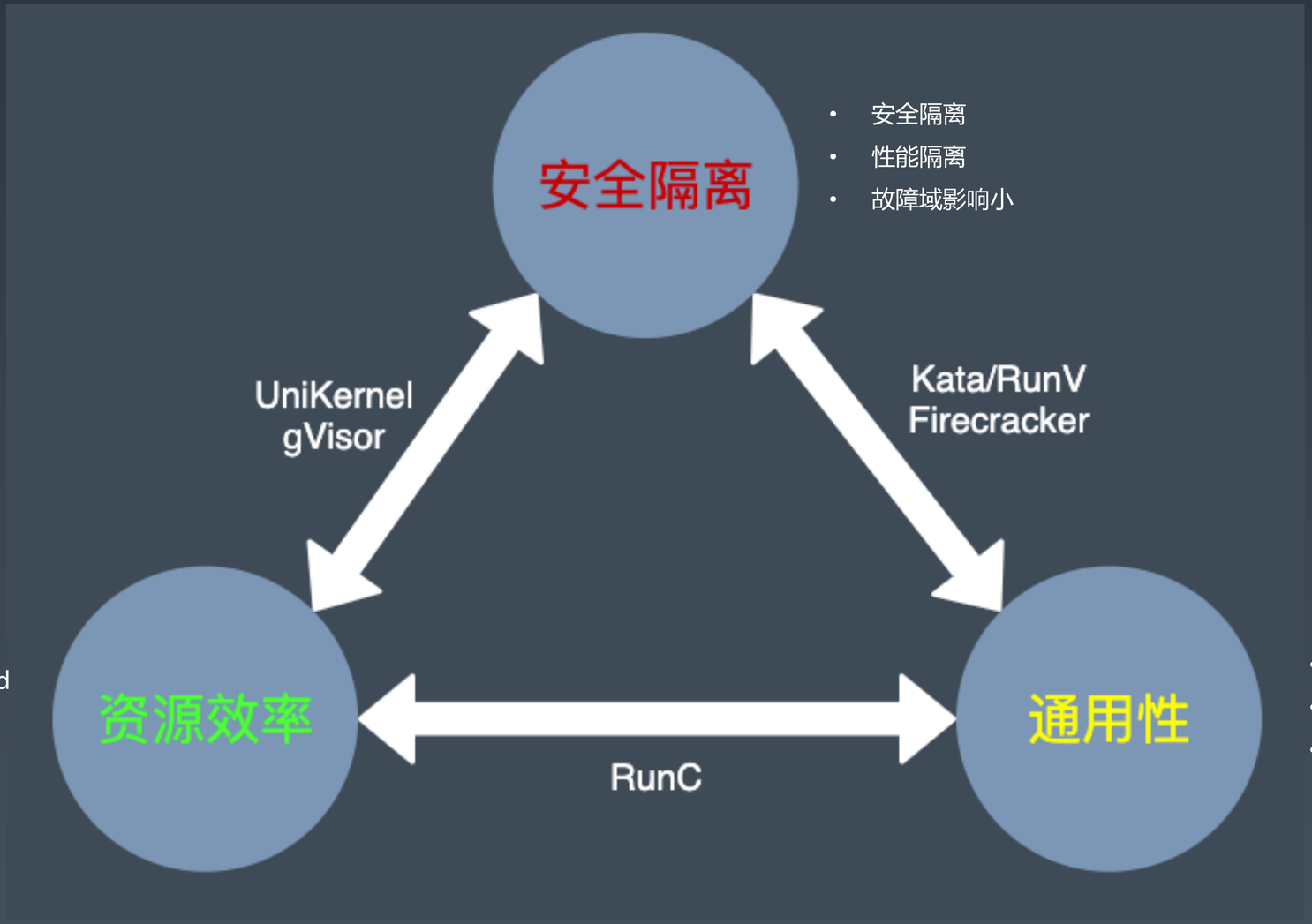
	安全容器运行时分类			
	OS容器+安全机制	用户态内核	Library OS	MicroVM
内核模式	共享内核	独立内核		
典型代表	Docker OS容器	gVisor	UniKernel Nabla Containers	Kata-Containers Firecracker
特点	• RunC 共享内核。 • 访问控制增强：SELinux、AppArmor、Seccomp等。 • Rootless Mode (docker 19.03+)	• 独立用户态内核，代理应用所有系统调用。 • 进程虚拟化增强。	• 基于 LibOS技术，内核深度裁剪，启动速度较快。 • 内核需与应用编译打包在一起。	• 基于轻量虚拟化技术，扩展能力优。 • 内核OS可定制。 • 应用兼容性优。 • 安全性最优。
缺点	• 无法有效防范内核漏洞问题。 • 安全访问控制工具对管理员认知和技能要求高。 • 安全性相对最差。	• 应用兼容性以及系统调用性能较差。 • 不支持 Virtio，扩展性较差。	• 应用兼容性差。 • 应用和LibOS的捆绑编译和部署为传统的 DevOPS 带来挑战。	• Overhead 较大，启动速度相对较慢。

彻底解决安全问题？不可能。

"The only real solution to security is to admit that bugs happen, and then mitigate them by having multiple layers.”

---Linus Torvalds (LinuxCon NA 2015, Seattle)

安全容器运行时选择



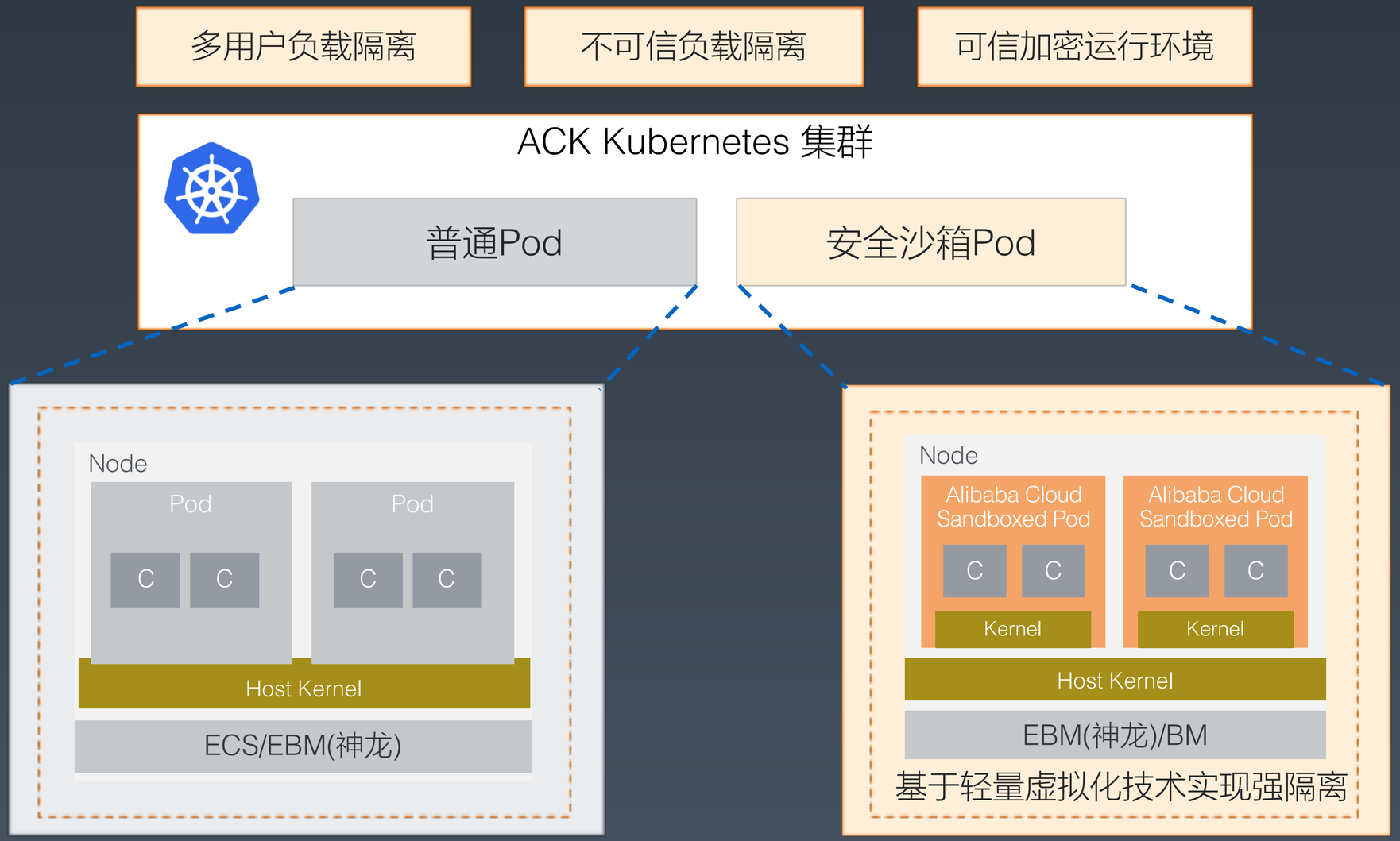
- 安全隔离
- 性能隔离
- 故障域影响小

- 低Overhead
- 启动速度
- 应用性能

- 应用兼容性
- OCI 兼容、K8sAPI 兼容等
- 生态

没有任何一种容器运行时技术可以满足所有场景的需求，需根据业务需求合理选择。

ACK安全沙箱容器



特点

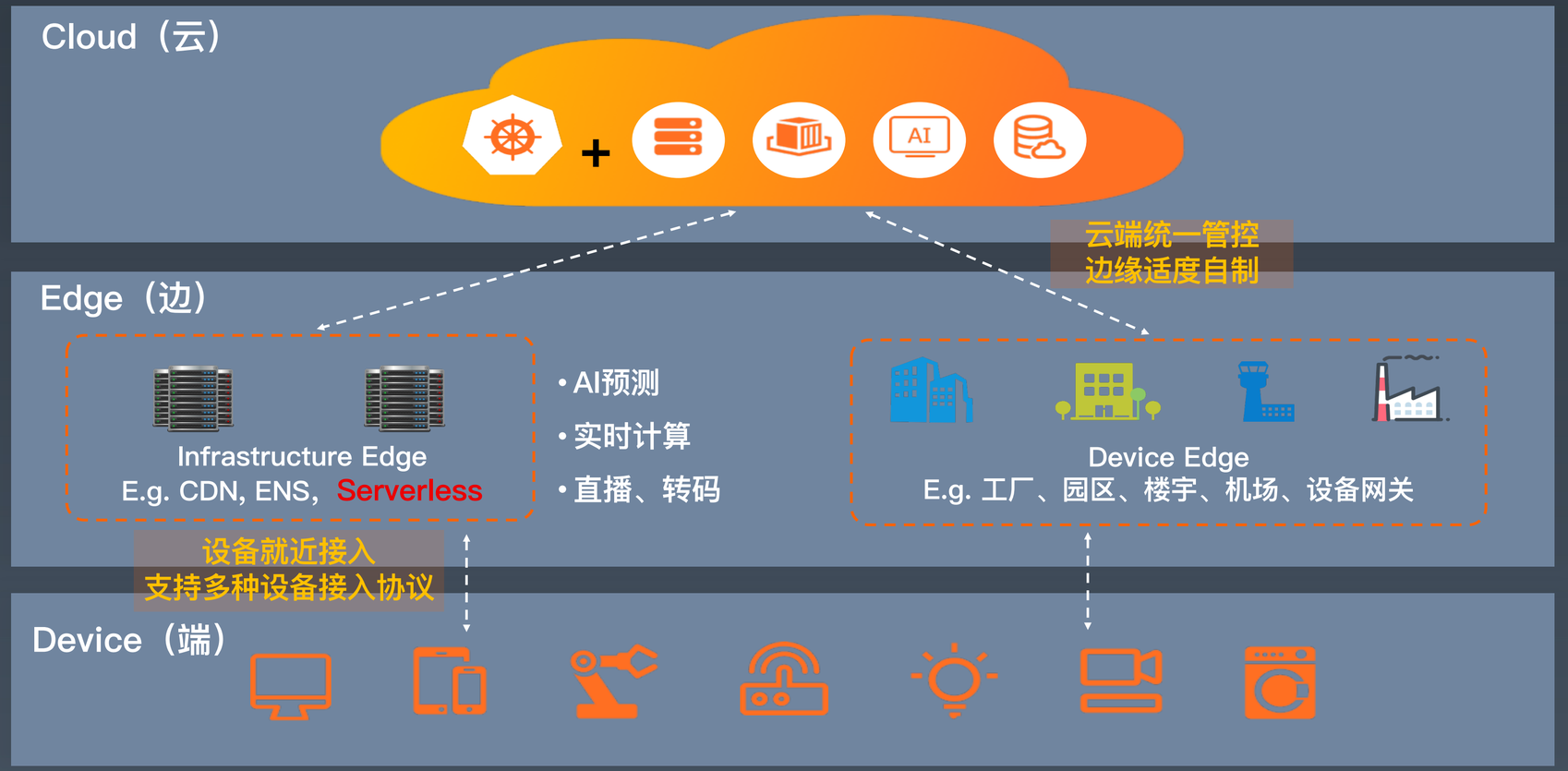
- 轻量虚拟机沙箱；
- 独立 kernel，强隔离; 安全影响域最小；
- 兼容OCI标准，兼容几乎所有 K8s API；
- 约25MB内存极低开销，500ms极速启动，原生 90%性能；
- 多租负载隔离、不可信应用隔离、Serverless等场景;

目录

1. 安全沙箱容器
2. Edge Kubernetes
3. 安全沙箱容器@edge方案
4. 新探索

ACK边缘容器

随着万物互联时代的到来，智慧城市、智能制造、智能交通、智能家居，5G时代、宽带提速、IPv6的不断普及，导致数百亿的设备接入网络，在网络边缘产生ZB级数据，传统云计算难以满足物联网时代大带宽、低时延、大连接的诉求，边缘云计算便应运而生。
边缘计算设施服务越来越难以满足边端日益膨胀的诉求，因而云上服务下沉，边缘 Serverless、边缘侧隔离不可信负载等日趋强烈...

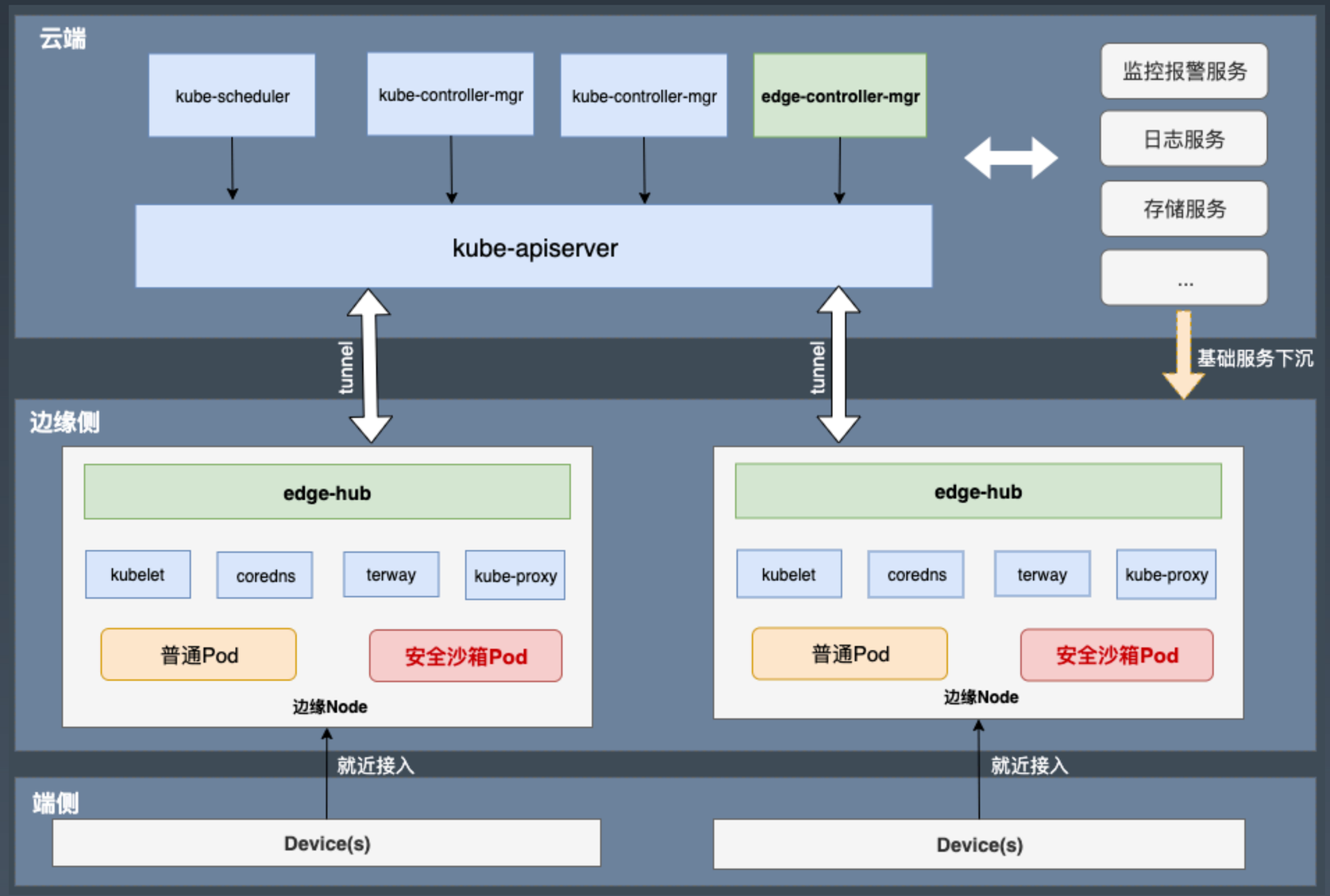


特点

- 云边端相互协同，提供统一的交付、运维、管控标准；
- 云端统一管控，节点自治、网络自治；
- 设备就近接入（边）；
- 支持安全沙箱容器。

目录

边缘安全沙箱容器整体架构



挑战

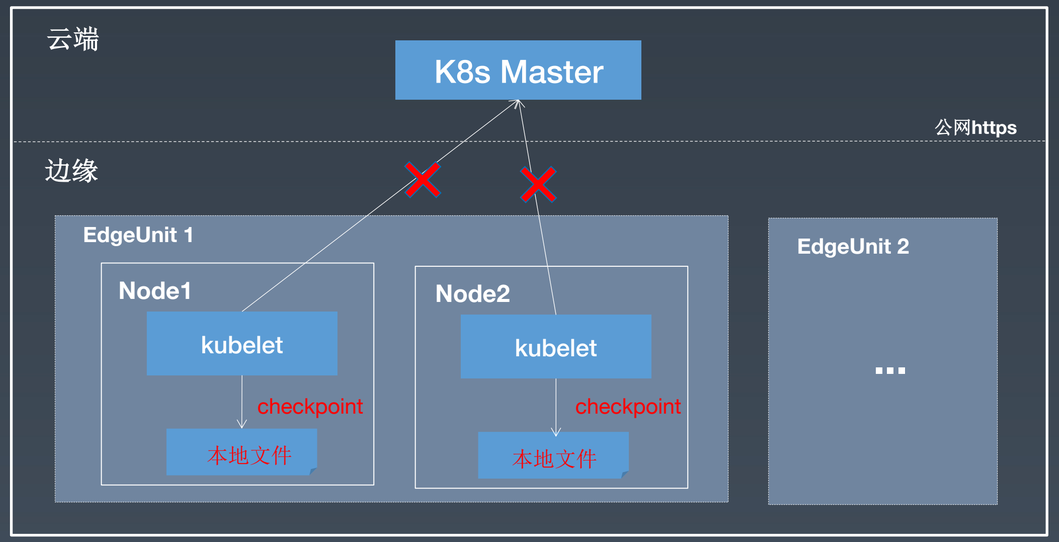
- 边缘弱网节点自治
  - 如何避免驱逐。
  - 断网节点数据/应用治理。
- 安全沙箱容器运行时兼容性
  - K8s API 兼容
  - 监控异常
  - 日志采集异常
  - 存储性能极差(Rootfs & Volume)
  - ...

边缘节点自治-社区

主要问题：

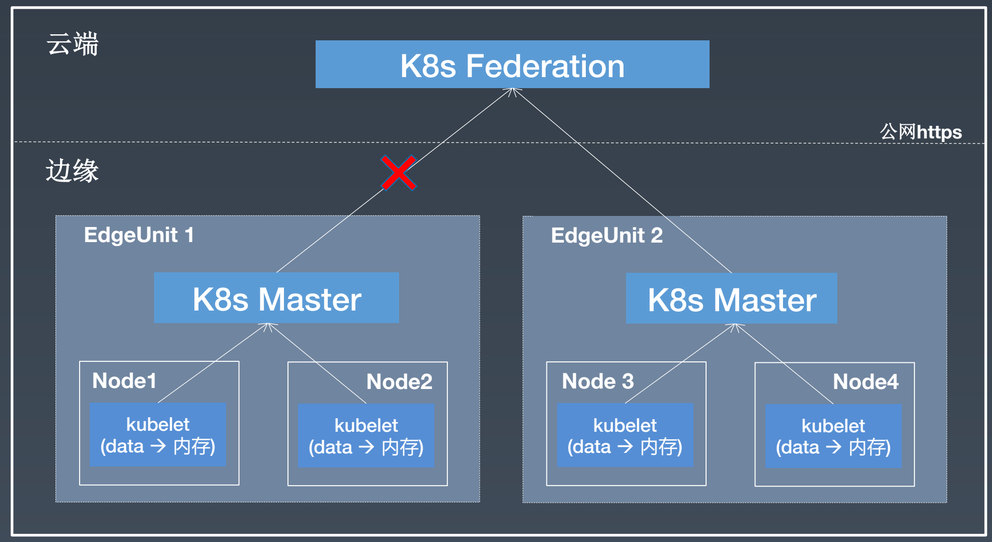
1. 节点失联超过pod容忍时间（默认300s）被驱逐问题。
2. 封（断）网期间，节点应用（如kubelet）重启无法从 apiserver 获取应用数据信息问题。

图1 社区方案(一)



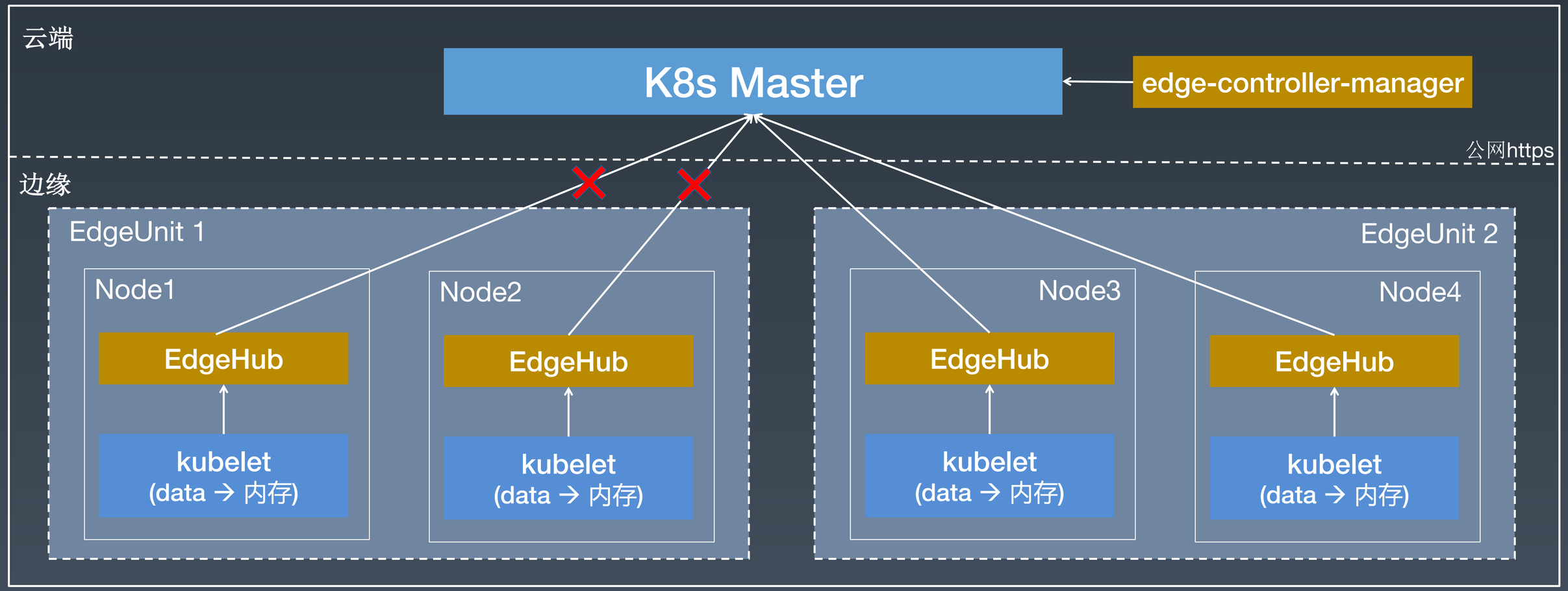
- 仅缓存 POD，对 ConfigMap/Secret/PV/PVC等暂不支持。
- 可定制修改kubelet，但面临升级难题。

图2 社区方案(二)



- Overhead 成本高：所有边缘区域均有 Master。
- 难运维：架构较复杂，集群规模数倍增加，监控、日志等复杂度。

边缘节点自治方案

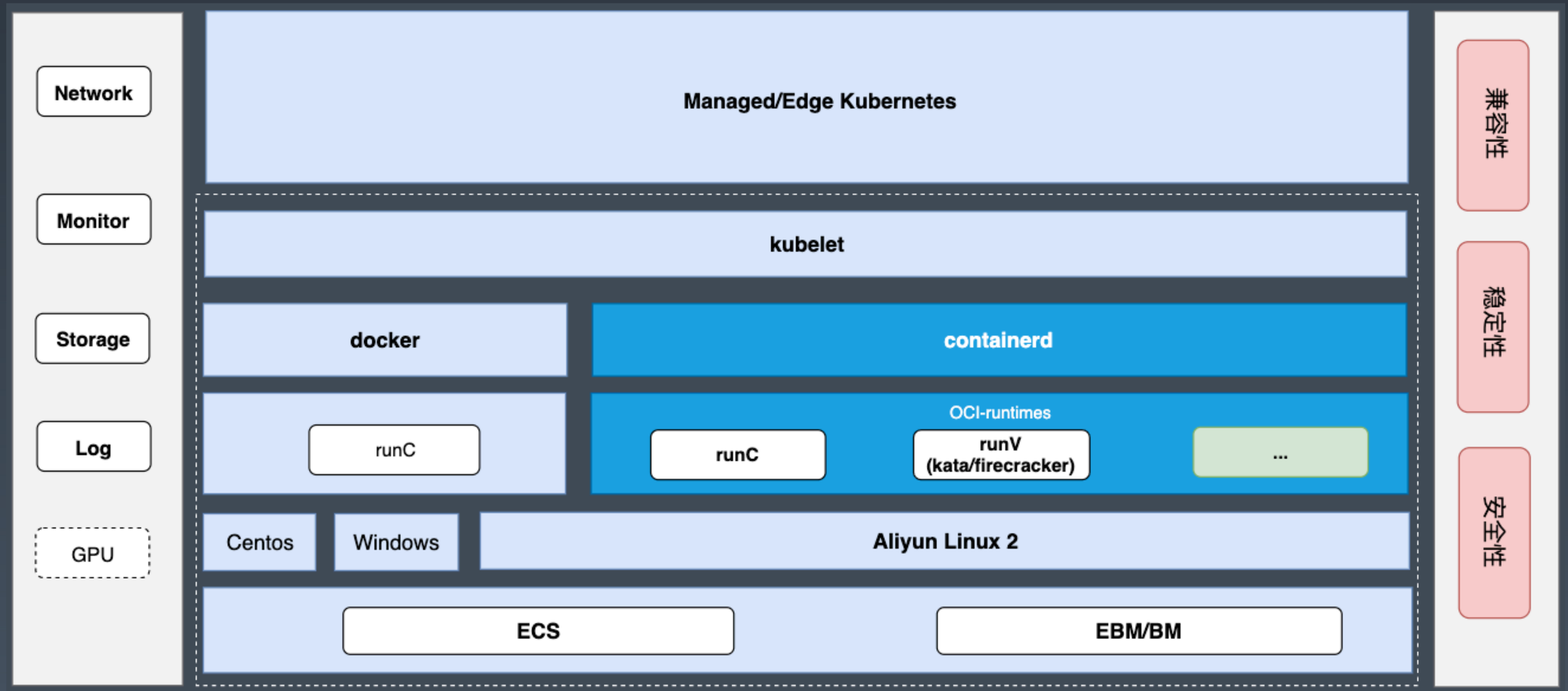


ECM(edge-controller-manager)：节点自治管理，忽略自治模式节点上的 Pod 驱逐等。

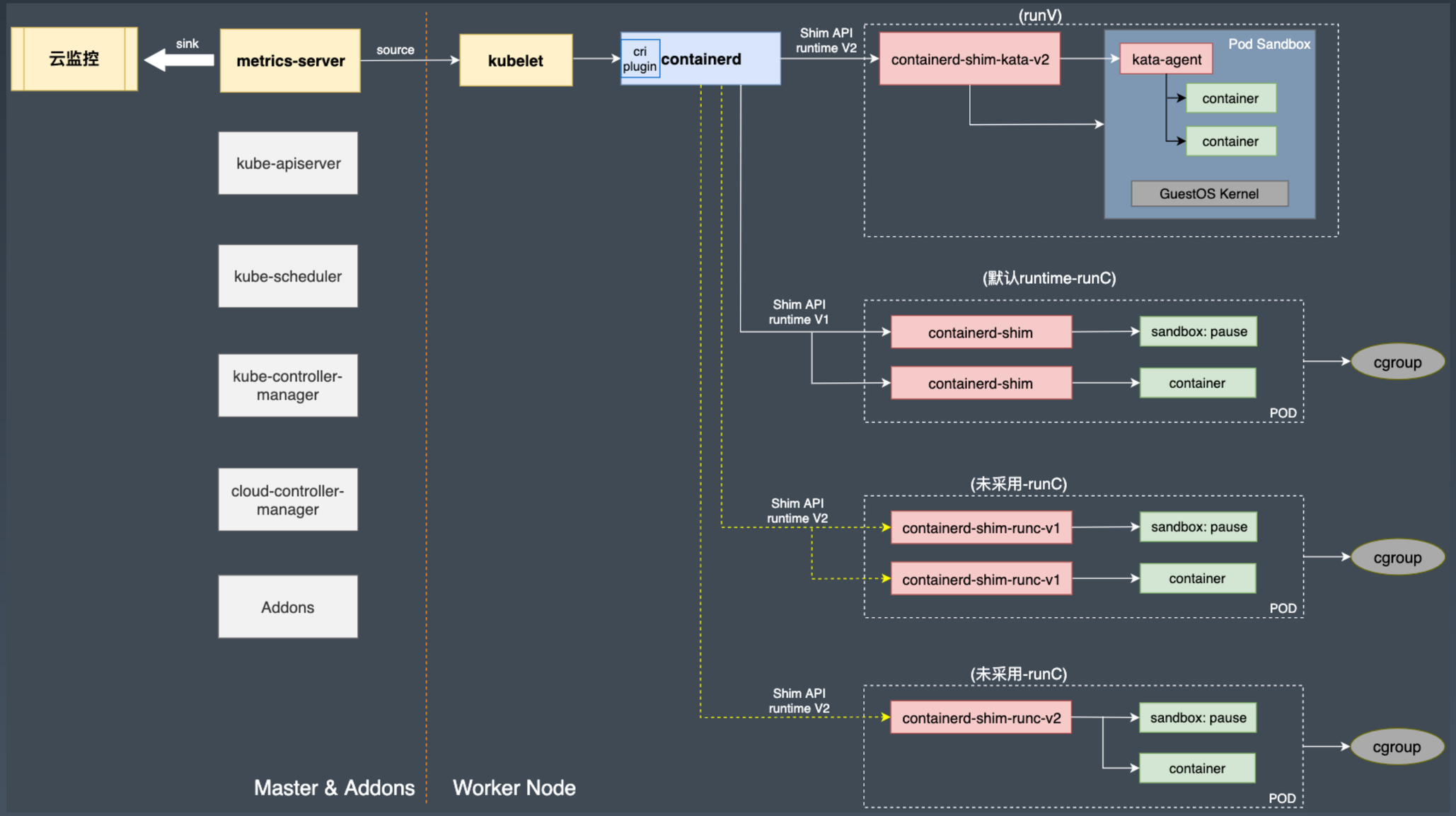
EdgeHub：边缘代理组件

- 作为 kubelet 和 apiserver 之间的缓存和代理。
- 在网络正常情况下，EdgeHub直接代理转发 Kubelet 请求到 apiserver，并缓存结果到本地。
- 在节点断网的情况下，EdgeHub 利用本地缓存充当 apiserver，并忽略所有写操作。

安全沙箱容器方案图


Network
Monitor
Storage
Log
GPU
Managed/Edge Kubernetes
kubelet
docker
containerd
runC
OCI-runtimes
runC
runV
(kata/firecracker)
...
Centos
Windows
Aliyun Linux 2
ECS
EBM/BM
兼容性
稳定性
安全性

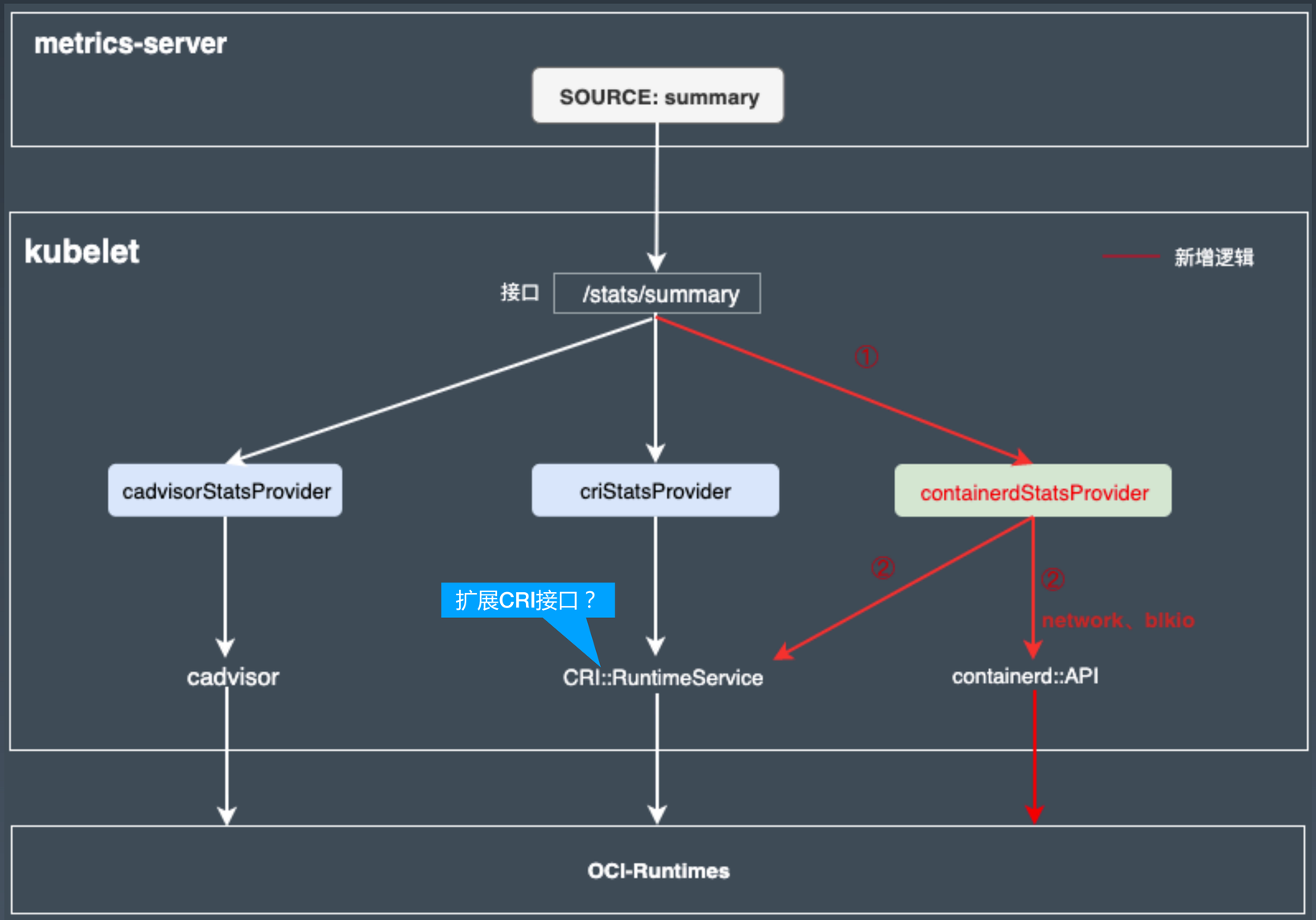
监控方案-拓扑图



主要问题:

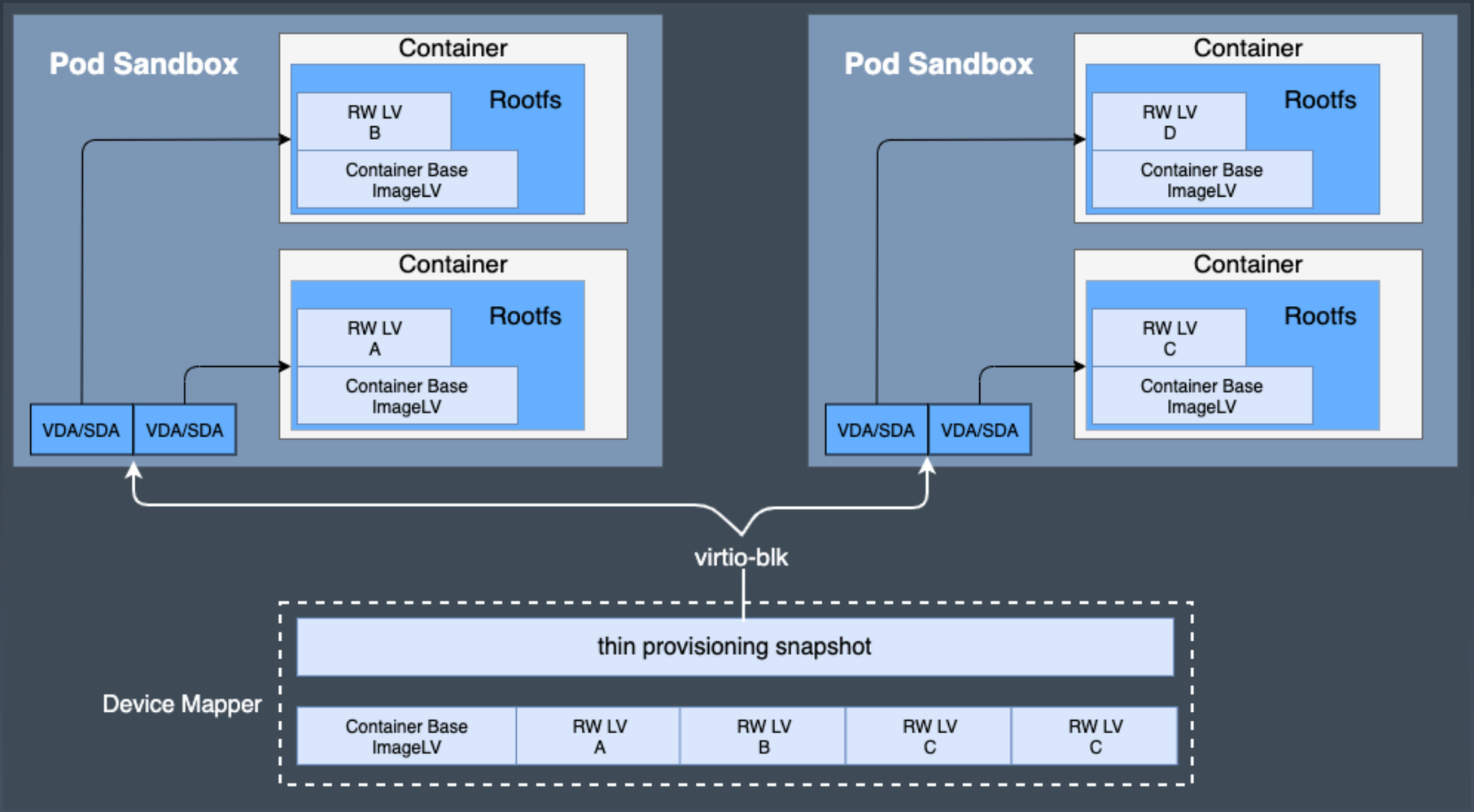
- runC、runV 容器缺失 network、blk io等。
- 如何架构改动/影响最小？

监控方案


metrics-server
SOURCE: summary
kubelet
新增逻辑
接口
/stats/summary
①
cadvisorStatsProvider
criStatsProvider
containerdStatsProvider
②
②
network、blkio
扩展CRI接口？
cadvisor
CRI::RuntimeService
containerd::API
OCI-Runtimes

安全沙箱容器存储方案 - RootFS

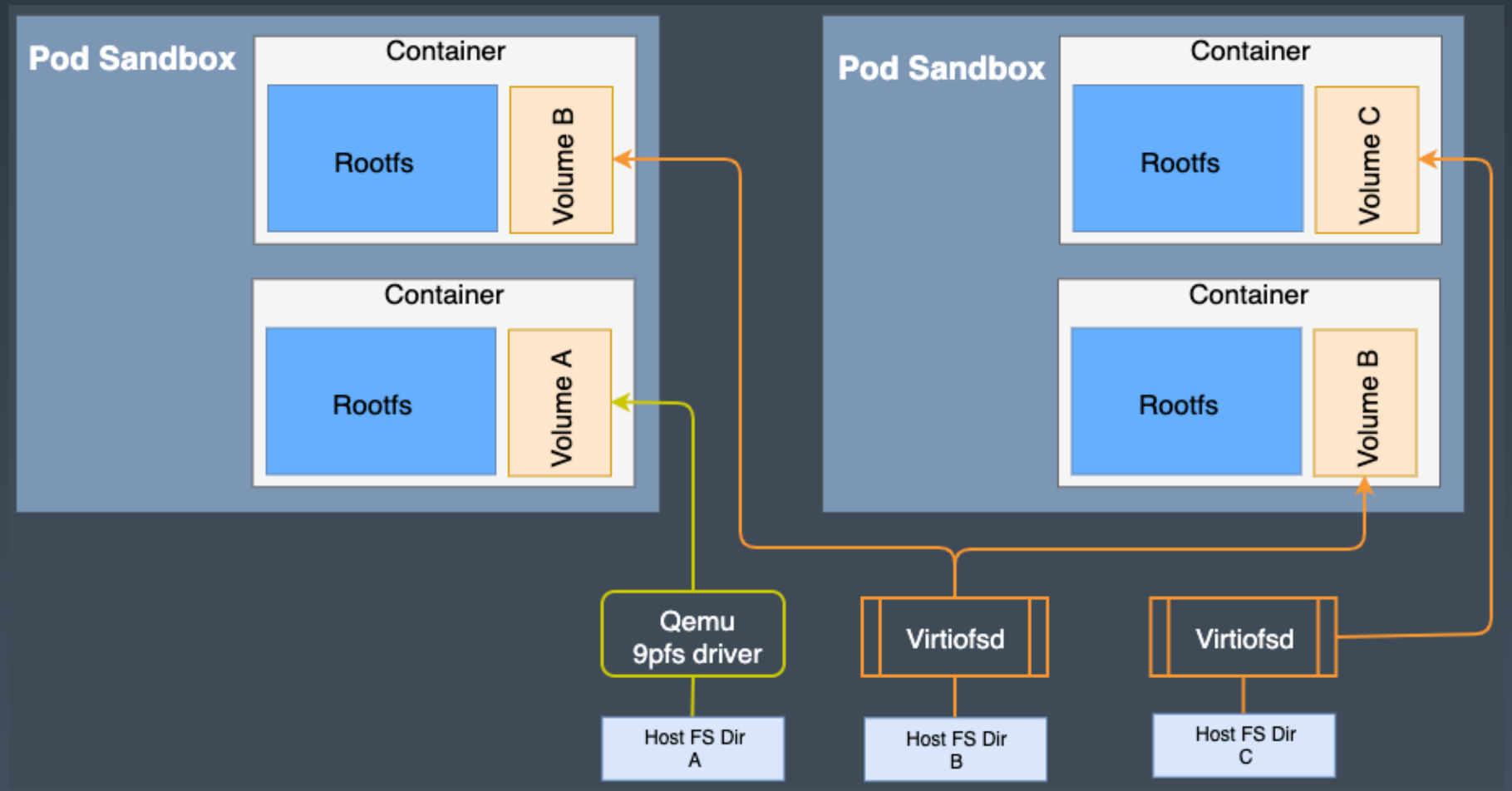
采用 devicemapper 方案构建高速、稳定的容器graph driver，实现功能、性能指标全面对齐 runC 场景。



GraphDriver:

- 采用 devicemapper snapshot 机制，实现镜像分层存储；
- IOPS、Bandwidth 与 RunC overlayfs + ext4 基本持平
- 基于snapshot增强开发，实现容器镜像计算、存储分离

安全沙箱容器存储方案 - Volume



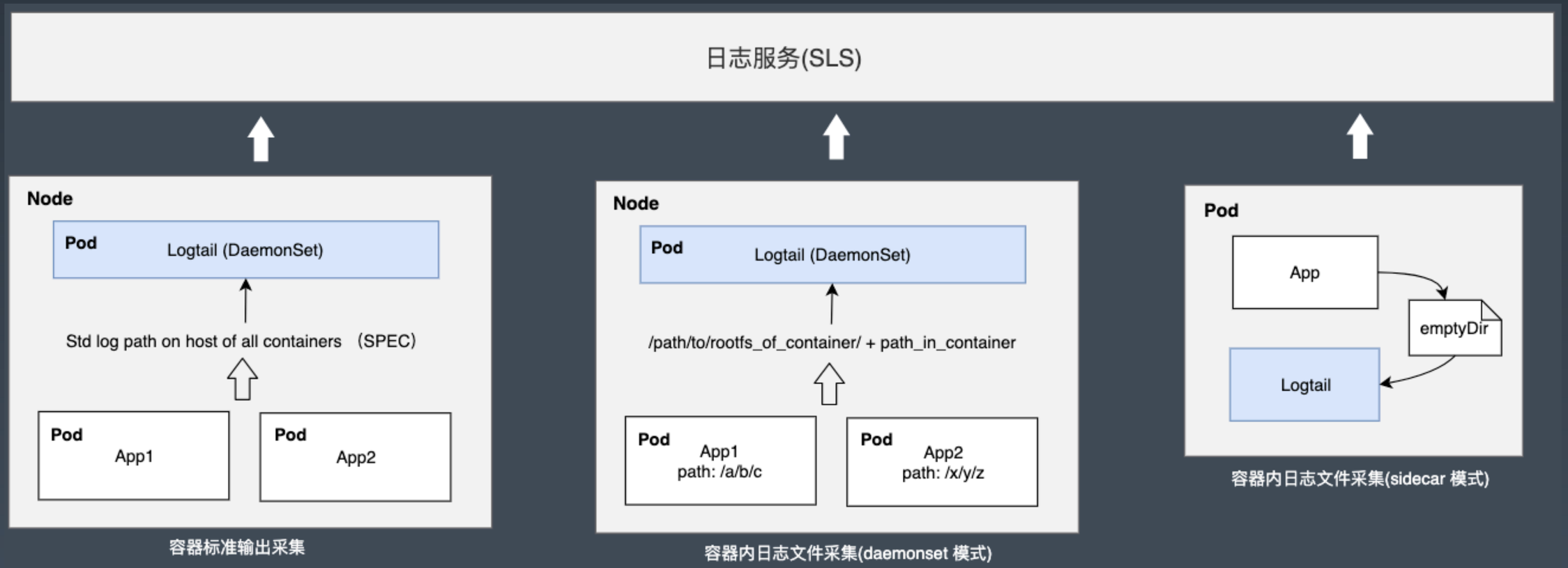
Volume:

- 9pfs（default，性能差）
- 兼容 CSI 和 FlexVolume 插件
  * 云盘
  * NAS
  * OSS
  * ...

性能优化：

- Virtiofs vs 9pfs（通用）
- Blk device 直通
- Mount NAS on GuestOS

日志方案



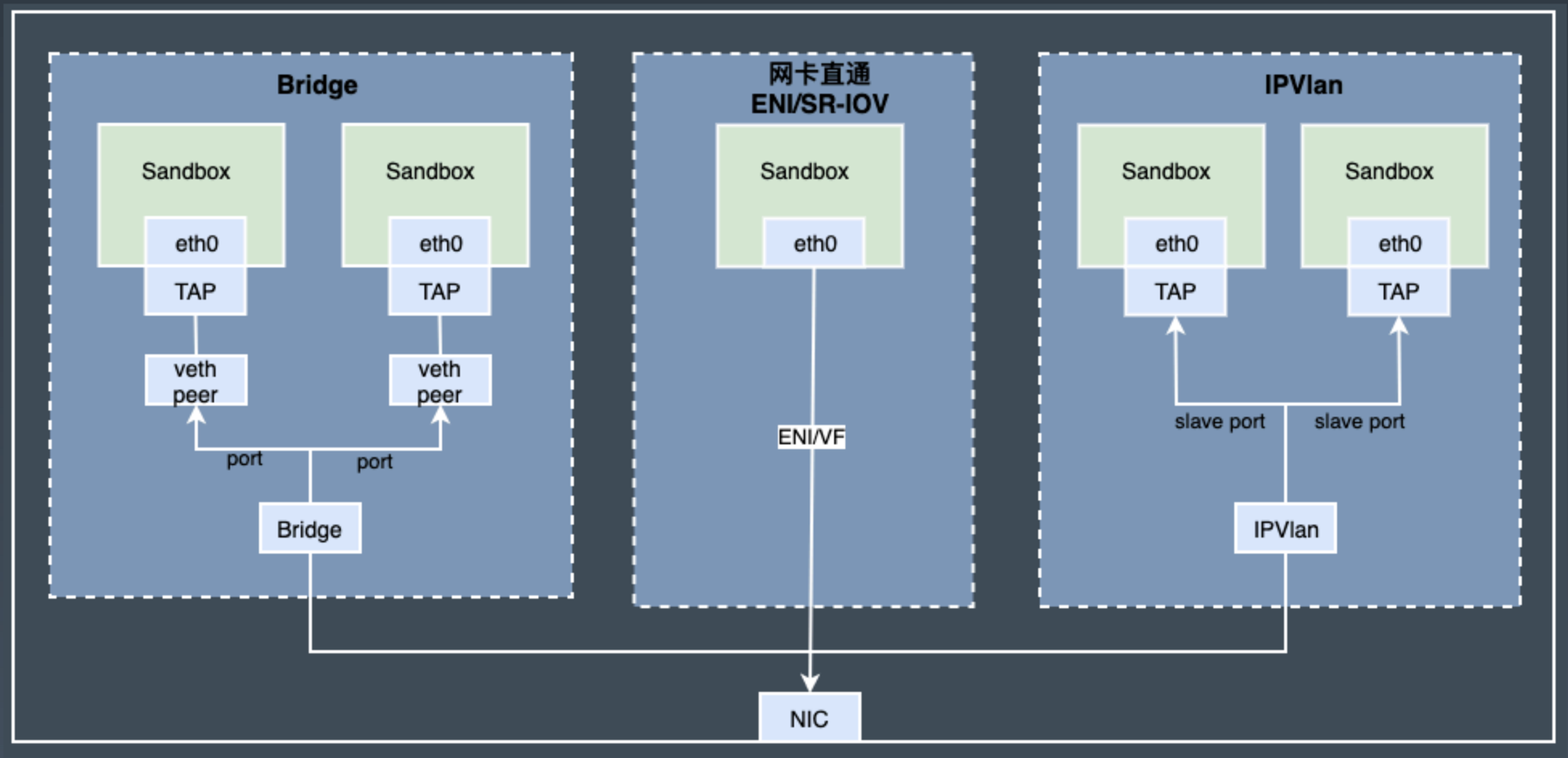
主要问题：

1. Logtail 只支持docker，不支持 containerd。
2. 所有 runC 和 runV 容器标准输出无法采集。
3. Logtail DaemonSet 模式无法直采 runC 和 runV 内。

解法：

1. 通用性：通过 CRI，而非 containerd SDK 连接 containerd。
2. 通过 Container Spec 获取容器标准输出日志路径。
3. 优先尝试查找 Upper Dir，否则查找 devicemapper 最佳匹配路径，由于 runV 有独立 kernel 无法在 Host 侧直采容器内日志文件。

网络方案



三种网络方案：

- Bridge桥接模式
  - 云上节点。
  - 通用性好。
  - 性能相对较差。
- 网卡直通模式
  - 云上场景：弹性网卡ENI。
  - 边缘场景：SR-IOV。
  - 性能最好，Host网卡数量限制。
- IPVlan模式
  - 下一代网络方案。

网络性能对比

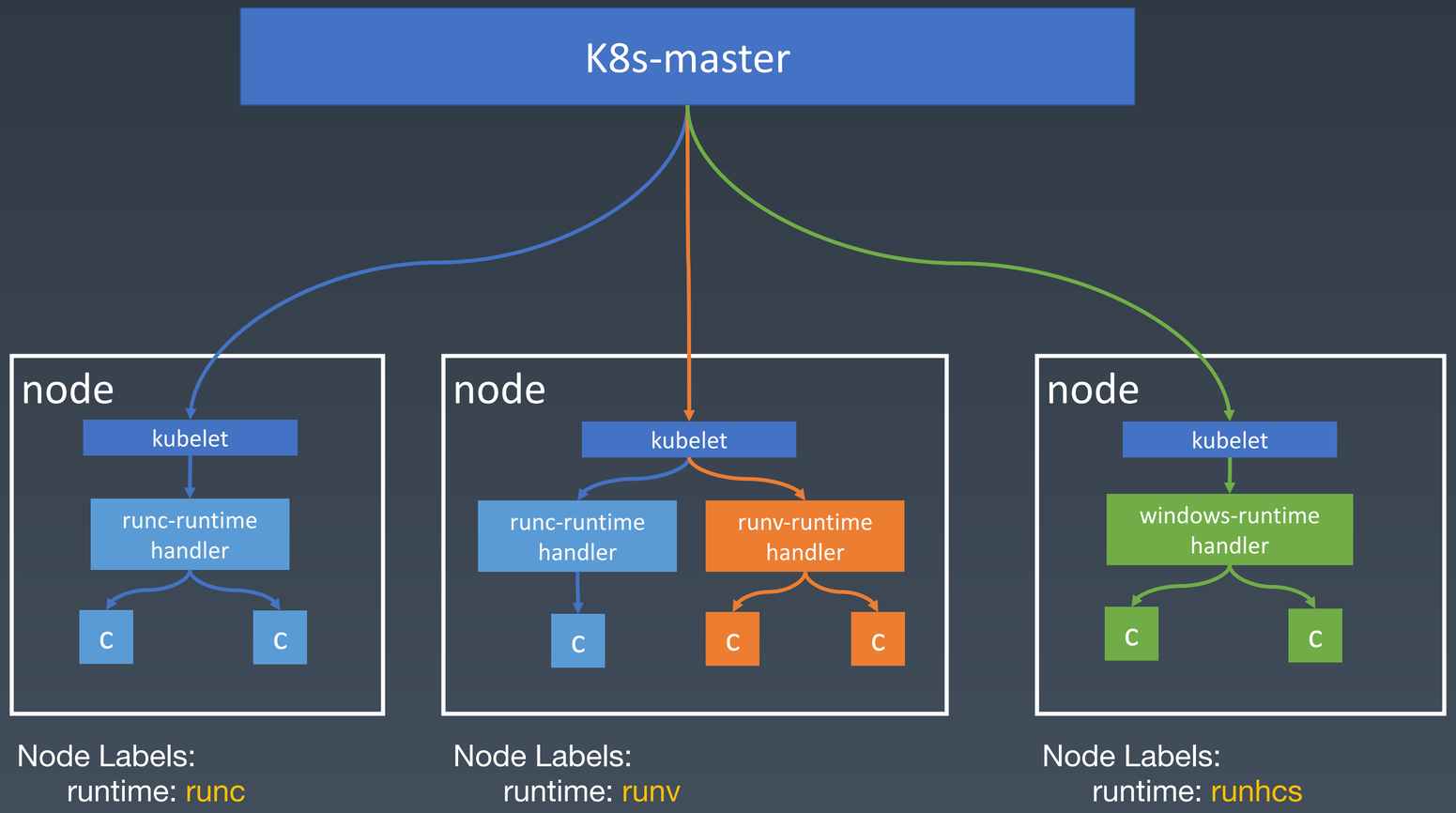
	Ping 时延	带宽(128B)	带宽(1024B)	TCP_RR	UDP_RR
Host	100%	100%	100%	100%	100%
网卡直通	100%	100%	100%	98%	92%
Bridge	140%	82%	80%	77%	75%
IPVlan	121%	81%	85%	80%	78%

结论：

直通网卡的性能可以做到接近host的性能，ipvlan和bridge与直通网卡方式有一定差距，但前两者通用性更好；总体来说 ipvlan 比 bridge 性能普遍更好。

注：以上为内部数据测试结果，仅供参考。

多运行时调度（ 1.14.0 <= kubernetes < 1.16.0 ）




```
apiVersion: node.k8s.io/v1beta1
handler: runv
kind: RuntimeClass
metadata:
name: runv
---
apiVersion: v1
kind: Pod
metadata:
    name: my-runv-ypod
spec:
    runtimeClassName: runv
    nodeSelector:
        runtime:  runv
    # ...
```

低于 1.16.0 版本的 K8s 调度器不支持 RuntimeClass，需通过 runtimeClassName 和 NodeSelector 或 Affinity 完成调度。

多运行时调度（ kubernetes >= 1.16.0 ）

```
apiVersion: node.k8s.io/v1beta1
handler: runv
kind: RuntimeClass
metadata:
name: runv
overhead:
    podFixed:
        cpu: 100m
        memory: 50Mi
    nodeSelector:
        runtime: runv
    tolerations:
        # ...
```

```
apiVersion: v1
kind: Pod
metadata:
    name: my-runv-ypod
spec:
    runtimeClassName: runv
    nodeSelector:
        runtime: runv
    # ...
```

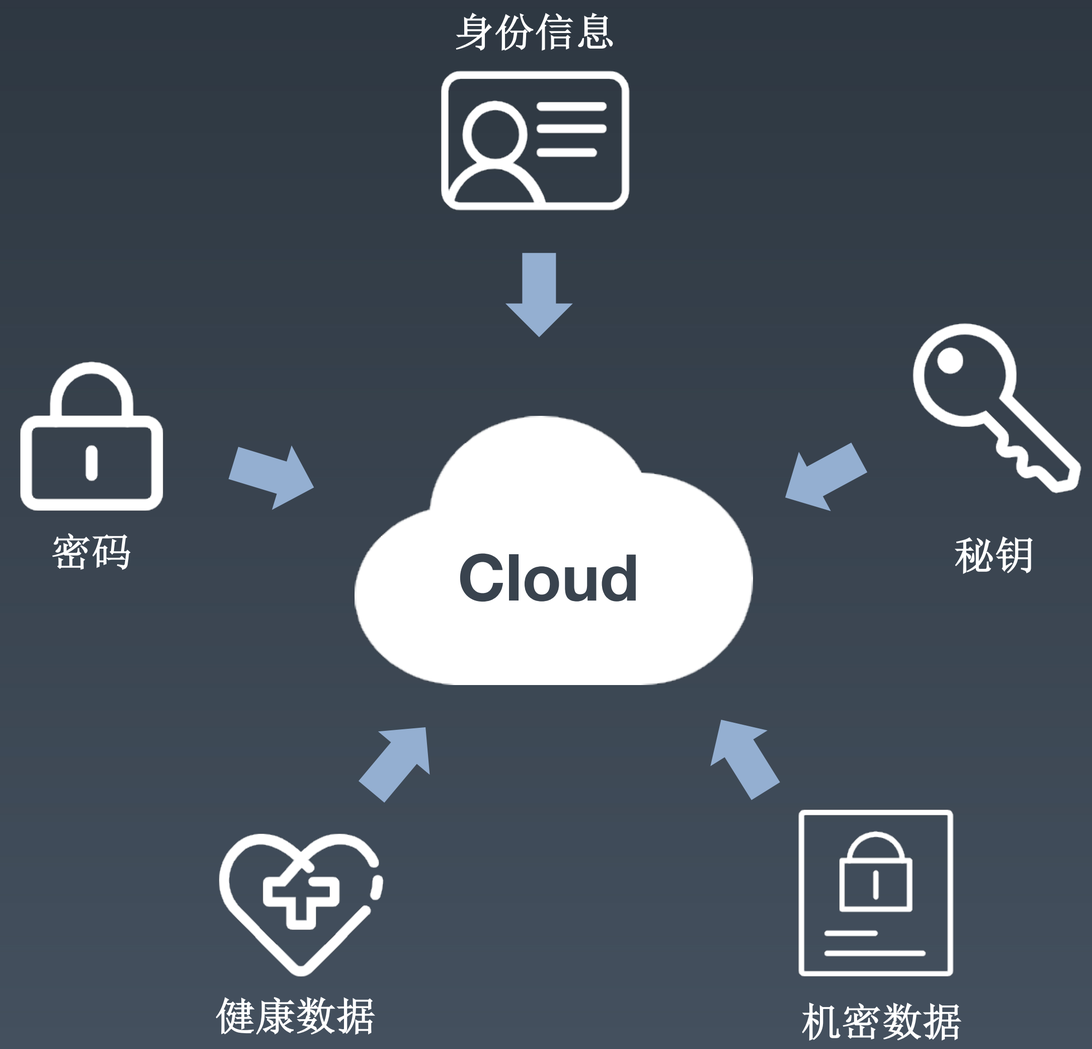
Kubernetes 1.16.0+ 支持 RuntimeClass 调度，同时支持 pod overhead 设置，参考：

- runtimeclass issue：https://github.com/kubernetes/enhancements/pull/909
- runtimeclass kep：https://github.com/kubernetes/enhancements/blob/master/keps/sig-node/runtime-class-scheduling.md （已加入1.16.0）
- pod-overhead: https://github.com/kubernetes/enhancements/blob/master/keps/sig-node/20190226-pod-overhead.md

目录

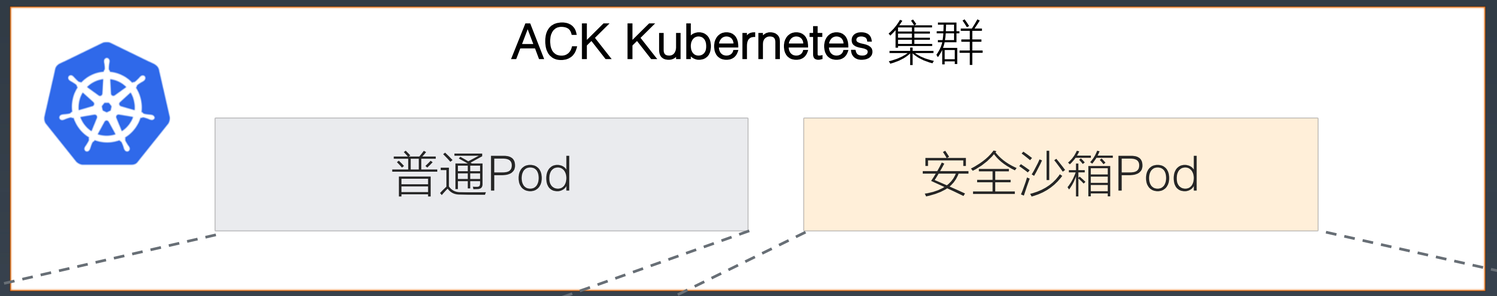
1. 安全沙箱容器
2. Edge Kubernetes
3. 安全沙箱容器@edge方案
4. 新探索

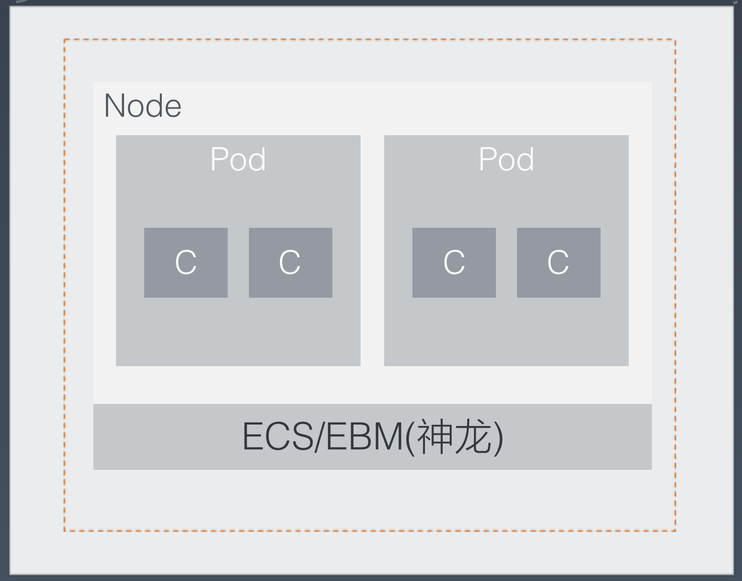
探索 – We Can Do More !

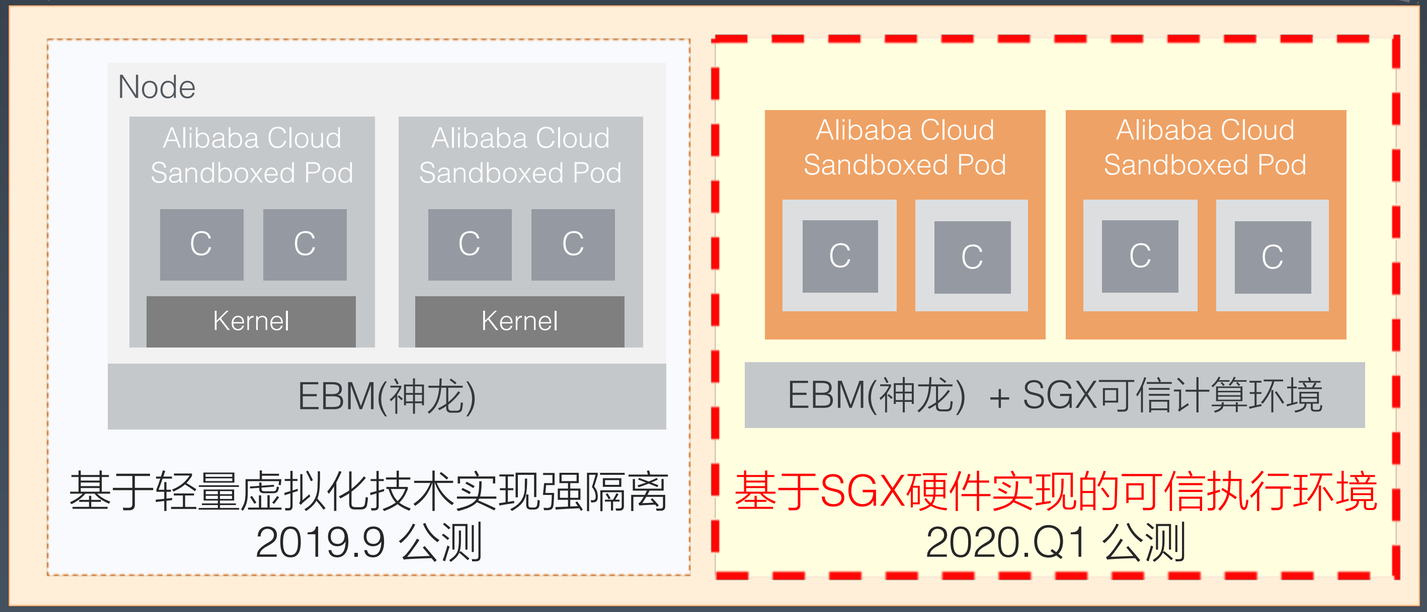


探索：机密计算 – 隐私/机密数据保护

多用户负载隔离 | 不可信负载隔离 | 可信加密运行环境







Geekbang 极客邦科技 | InfoQ 极客传媒

InfoQ官网 全新改版上线

促进软件开发领域知识与创新的传播

关注InfoQ网站
第一时间浏览原创IT新闻资讯

免费下载迷你书
阅读一线开发者的技术干货

THANKS!

QCon 10th