

Kata Containers: when OCI meets virtualization

彭涛 bergwolf@hyper.sh

Outlines

- Docker 和 OCI 简介
- Kata Containers
- Kata Containers 和 OCI
- Kata Containers 和 CRI
- Q&A

关于 Hyper

- 来自中国的创业团队
- 关注于虚拟化容器技术及其应用
- 维护多个开源项目
 - hyperhq/hyperd
 - hyperhq/runv
 - hyperhq/hyperstart
 - kubernetes/frakti
- 提供容器云服务
 - <https://hyper.sh/hyper>
 - <https://hyper.sh/pi>
- runV -> Kata Containers




Docker

- 当前最热门的容器技术
- 从一开始的容器runtime,发展到集群,编排,各种网络/存储插件,乃至打包整个操作系统的复杂生态
- 引发行业变革,各种最佳实践,开源项目,创业公司,容器云如雨后春笋

从 Docker 到 OCI

- OCI: 容器的行业标准
- Image-spec: 如何传播
- Runtime-spec : 如何运行

Docker 的优点

 docker =  Container +  Docker Image

轻量级 ✓

快速 ✓

隔离性 ❌

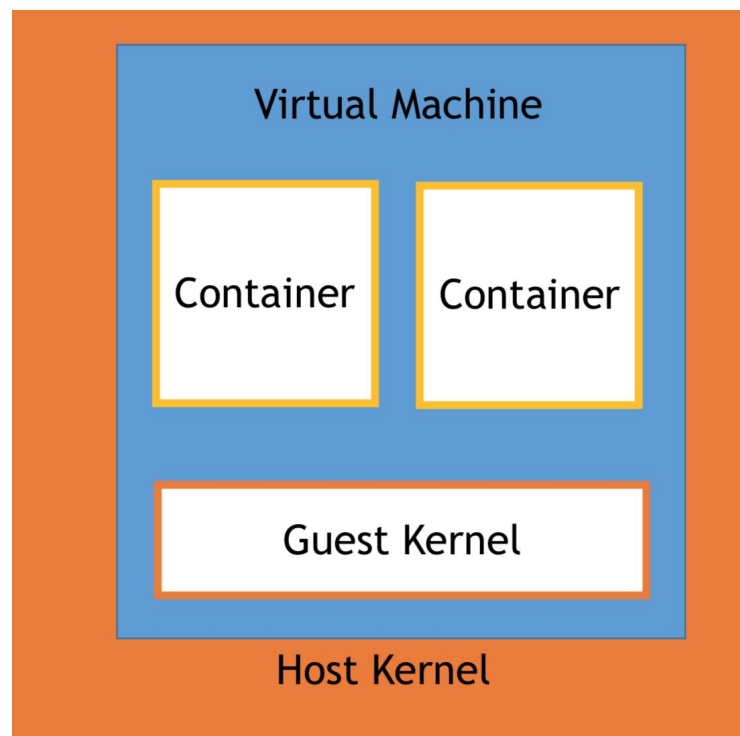
随时随地 ✓

便携 ✓

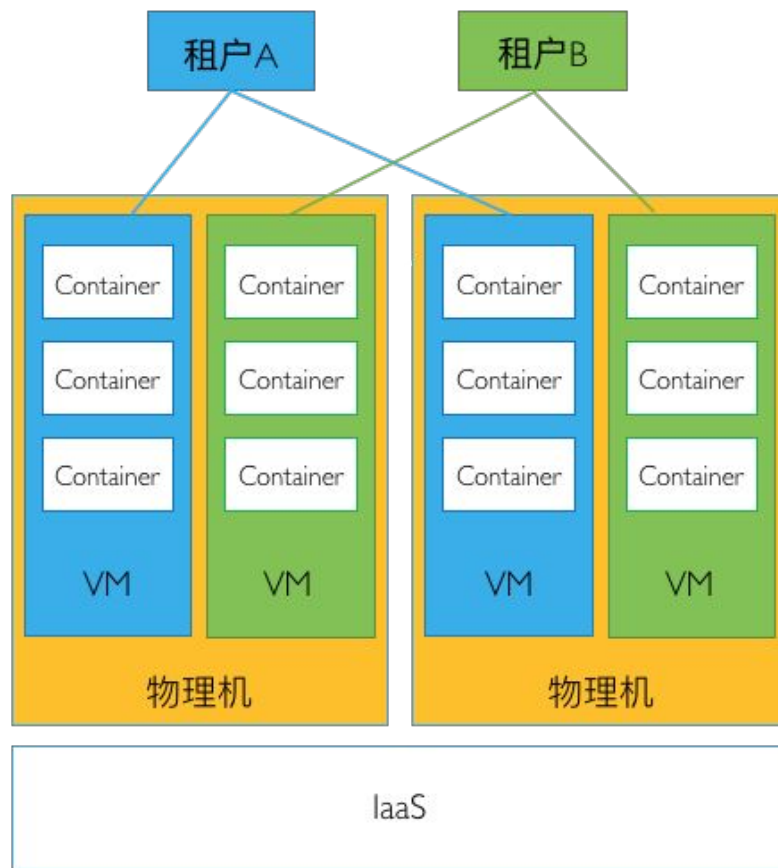
不可变更 ✓

提升 Docker 安全性

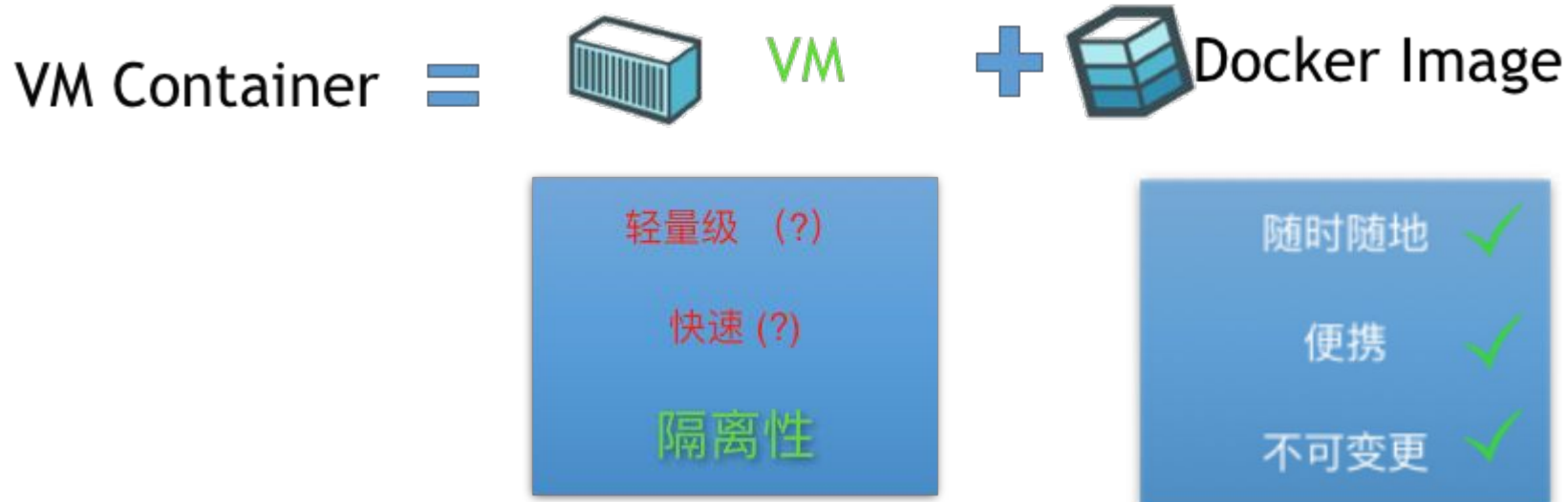
- 容器的隔离技术已经有很大进步了
- 仍然会是不是受到漏洞的影响
 - Dirty COW (CVE-2016-5195)
 - wait_pid (CVE-2007-5123)
- 总归不如虚机的隔离性好



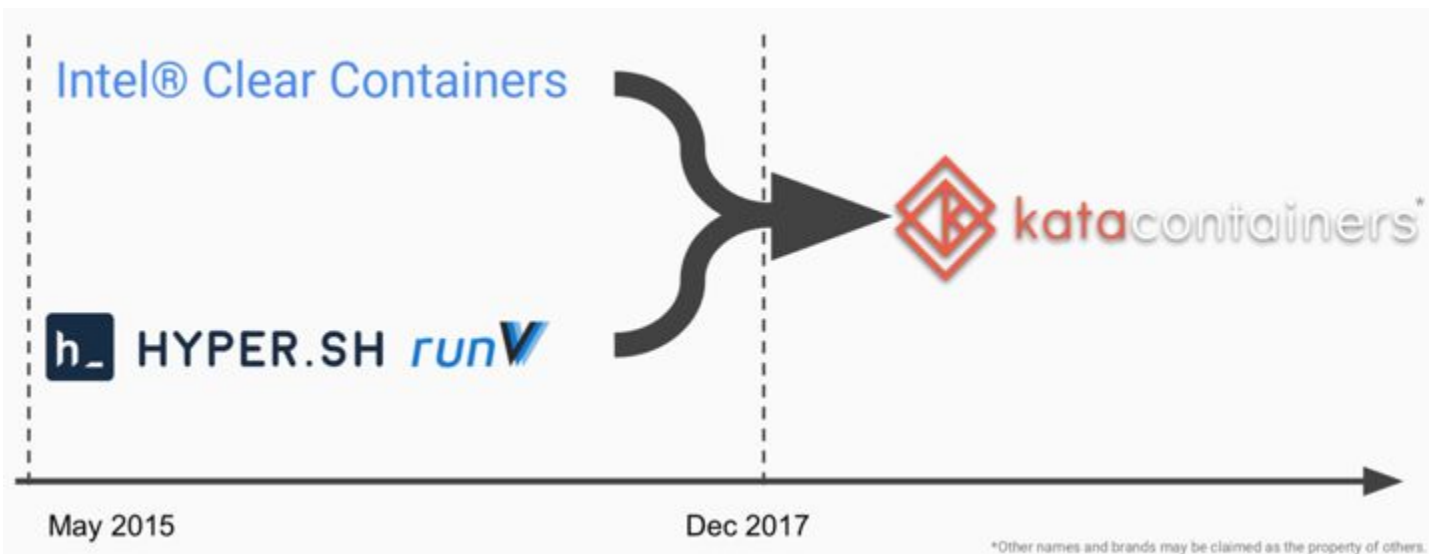
laaS 集群中的 docker



虚拟化容器



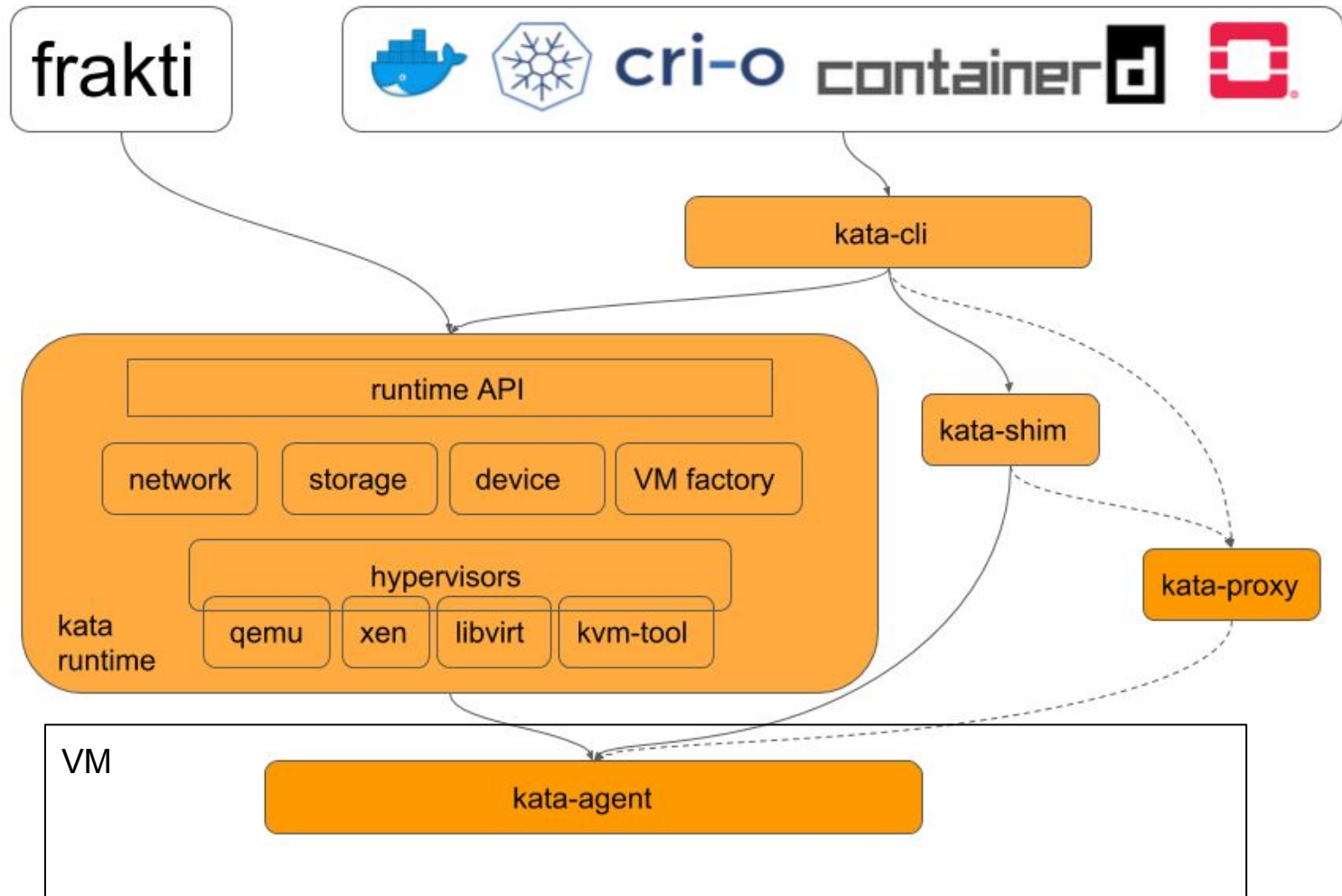
Kata Containers



Kata Containers

- 符合 OCI runtime-spec 的容器运行时
- 轻量快速的虚拟化容器实现
- 无缝集成 Kubernetes (CRI), docker, OpenStack
- 由 OpenStack 基金会管理

Kata Containers



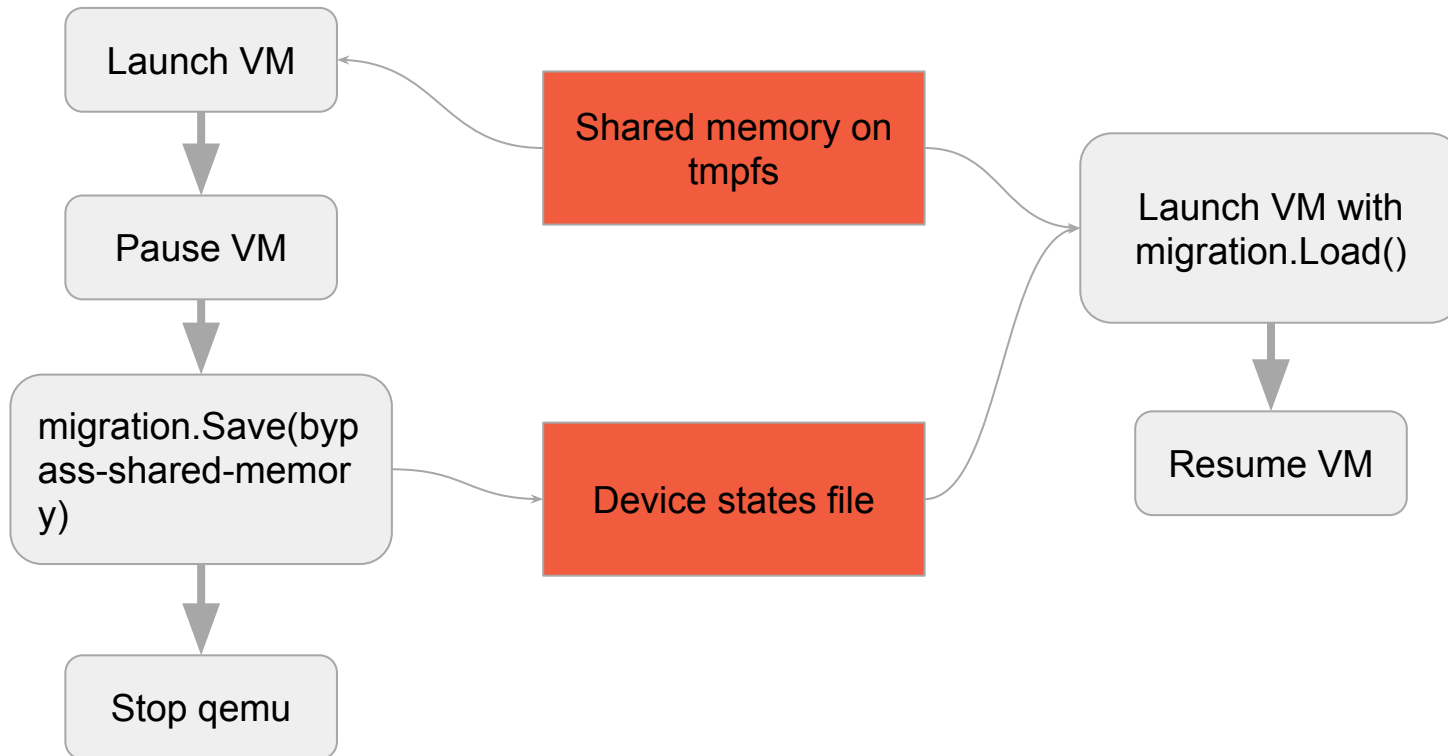
轻量快速的虚拟化容器

- 极度精简的 guest 硬件和内核
- 精简优化后的 qemu (qemu-lite)
- VM caches: 预启动一些stop状态的VM
- VM template: 所有虚拟机共享内核, initramfs 和早期初始内存 (已提交给 qemu 社区)
- DAX/nvdim: rootfs 镜像直接 map 到 guest 内存地址空间

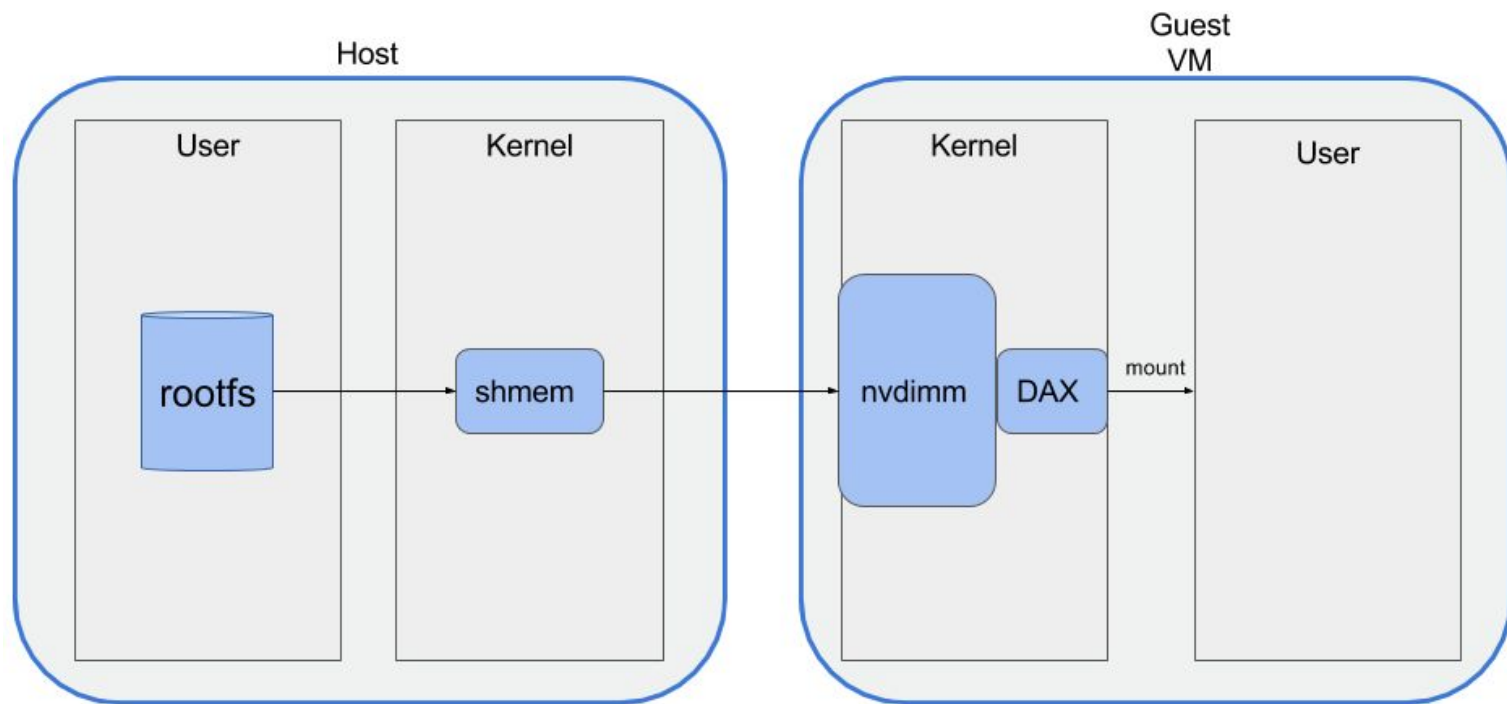
VM Template

Create VM Template

Create from VM Template



DAX/nvdim



轻量快速的 Kata Containers

- 亚秒级启动
 - <100ms (Xeon 1270 3.5GHz)
 - ~400ms (Pine64 ARM)
- 低开销
 - Qemu 进程额外开销小于 10MB
 - Guest 中 kernel/initramfs/rootfs/kata-agent 开销几乎可以忽略
- 高密度
 - 单机 5k vs 500 (传统虚机)

Kata Containers 和 OCI

- OCI runtime-spec 兼容的容器运行时
- Docker 中无缝替换 runc
 - runc commandline 的全兼容
 - 支持 docker 的 libnetwork 网络配置方式
 - `Docker run -d --runtime kata nginx`

Kata Containers 和 OCI

- runtime-spec 基于 linux container 的假设
- 缺少虚拟机相关参数设置
 - Hypervisor, kernel, initramfs, rootfs image etc.
 - 通过默认配置文件
 - 给 runtime-spec 增加 vm 描述 (已被上游合并)

Kata Containers 和 OCI

- runtime-spec 基于 linux container 的假设
- 缺少存储设备描述, rootfs 和 volume 都是本地目录
 - 通过共享文件系统 map 到 guest 中 9pfs
 - 自动检查基于裸设备的 rootfs/volume, 并 passthrough 到 guest 中

Kata Containers 和 OCI

- runtime-spec 基于 linux container 的假设
- 基于 major/minor 的设备描述
 - Host 上设备的 major/minor 对 guest 没有意义
 - 在 host 上寻找到对应的设备文件
 - 热插入给 guest

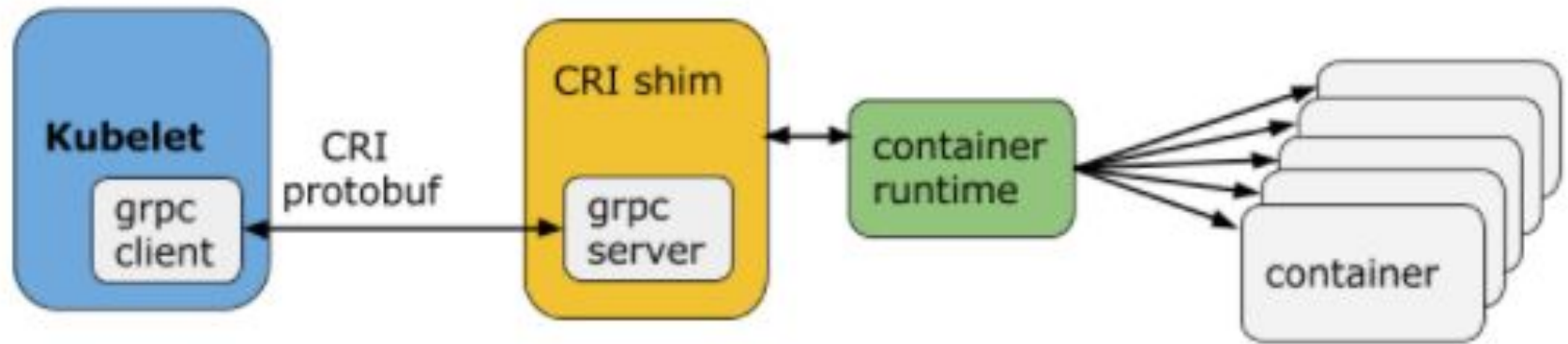
Kata Containers 和 OCI

- runtime-spec 基于 linux container 的假设
- 基于 linux cgroups 的 CPU和内存限制的描述
 - 使用近似转换获得 CPU 和内存设置
 - vCPU: $(\text{quota} + \text{period} - 1) / \text{period}$
 - 内存: `Memory.Limit` >> 20

Kubernetes CRI

- 为 kubernetes 扩展性而引入
- kubelet 和 container runtime 之间的插件接口
- 允许多种 container runtime 实现
 - docker-shim
 - cri-o
 - containerd-cri plugin
 - frakti

Kubernetes CRI



Kata Containers 和 CRI

- Kubernetes Pod 抽象和 VM 天然符合
 - 运行多个容器的沙箱
 - 集群中的最小调度单位
 - 资源隔离和共享的边界

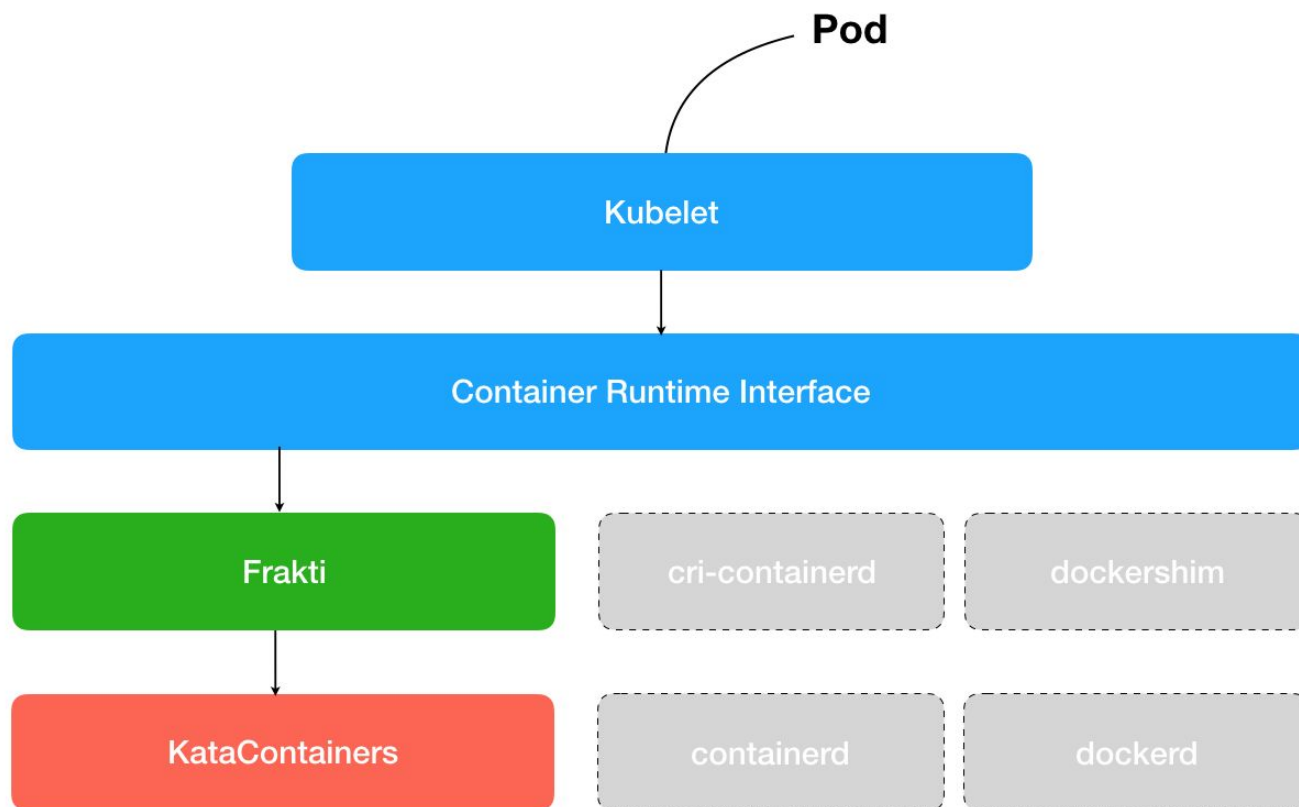
Kata Containers 和 CRI

- 通过 OCI 兼容的命令行对接
 - cri-o, containerd cri plugin, docker-shim
 - 缺少 sandbox 抽象
 - CRI CreateSandbox 被转义为创建 Pause container
 - 功能受到前述的 OCI spec 兼容性限制

Kata Containers 和 CRI

- 通过符合 CRI 语义的 API 对接
 - frakti
 - 不受 OCI 和 runc 兼容性约束
 - 不需要创建 pause container
 - 提供 API 支持描述不同的存储设备
 - 裸设备, Ceph rbd, NFS etc.

Kata Containers 和 CRI



虚拟化容器的未来

- 硬件 passthrough 和 GPGPU
- Edge
- Linuxd (“Run Linux Kernel as a Daemon”)

Q&A

谢谢!

我司招聘GO程序员
jobs@hyper.sh

