



云+容器 重新定义企业IT架构

阿里云高级技术专家 汤志敏

1 云原生计算与容器技术

云应用平台成熟度模型



阿里云容器服务

DevOps



微服务



企业应用



智能创新



容器服务

多集群管理

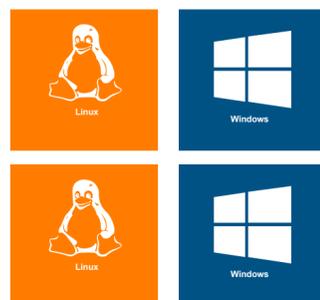
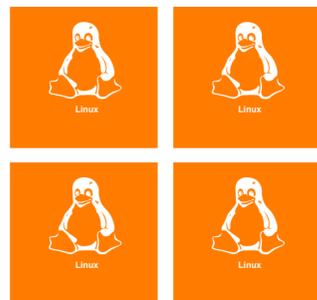
安全合规

混合云

日志、监控

应用仓库
镜像/解决方案

阿里云



专有云



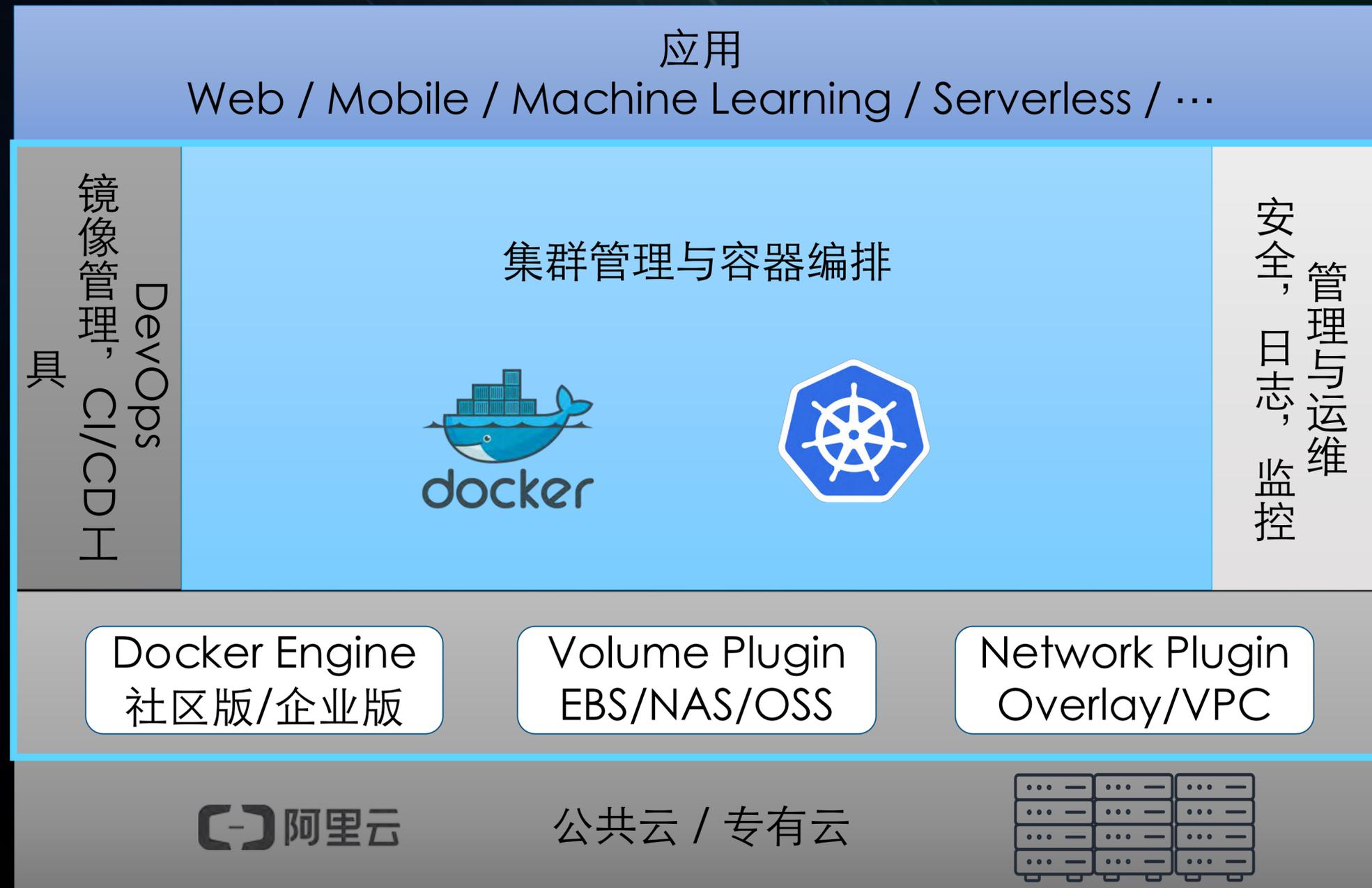
物理机



虚拟机



Kubernetes和Docker与云无缝集成



将容器技术和IT基础设施(公共云、私有云)集成到一起

内建混合云管理

内建GPU支持

容器技术发展



成熟



标准化



萌芽

发展



2011



2000

2006

2013

2015

2016



未来将去向何方？

2 应用为王

1

企业软件供应链

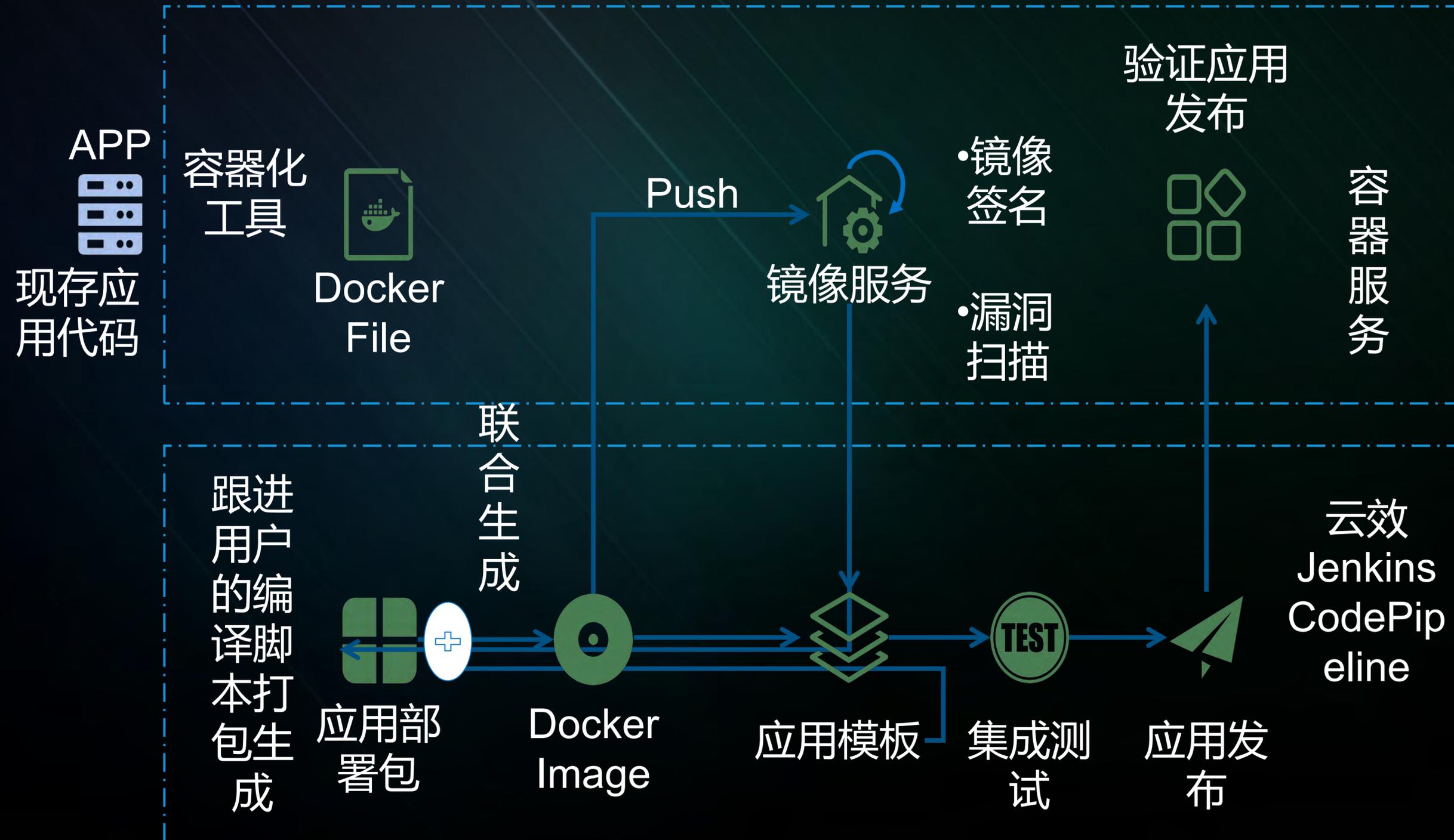
2

简化应用部署

3

应用容器化最佳实践

2.1 企业软件供应链



Derrick

让开发人员可以专注在应用的开发上，可快速完成应用Docker化

特点

代码探测，支持常见编程框架

自动生成容器配置

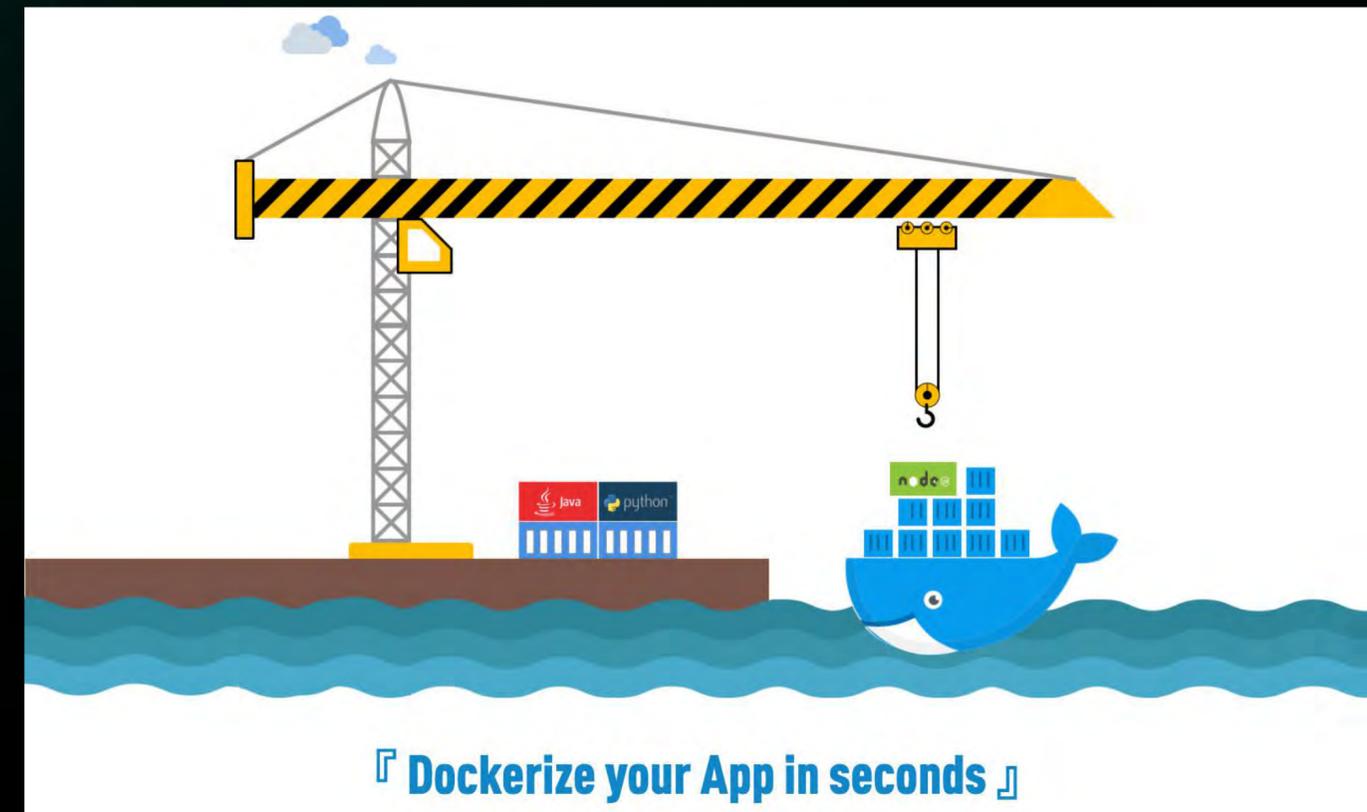
Dockerfile , docker-compose.yml, Kubernetes resource

Jenkinsfile

内建最佳实践

多阶段构建等

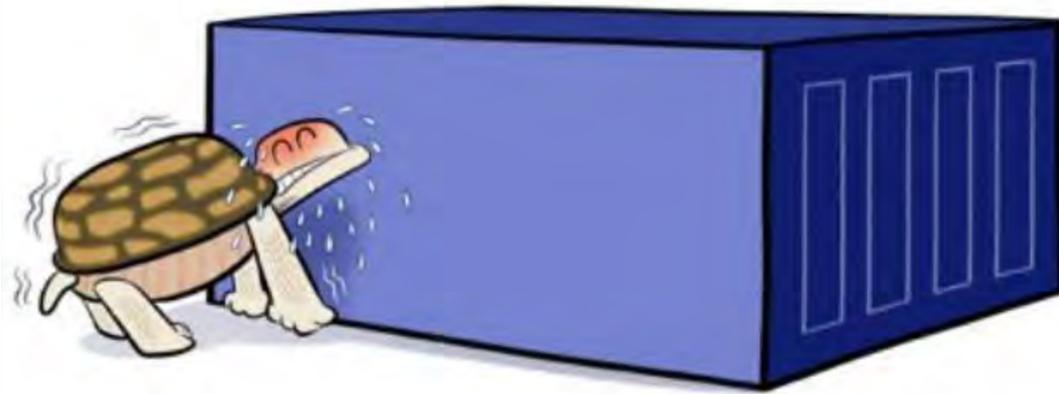
<https://github.com/alibaba/derrick>



Multi Stage

Build smaller images with Multi-stage builds

**First stage:
complete build
environment**



**Second stage:
minimal runtime
environment** ❤️



One Dockerfile, one build

2.2 简化应用部署

Cloud Provider

K8s的云资源适配



Helm/Charts
K8s应用管理

Kompose

Kompose
Compose转化为
k8s Definiton

CloudProvider

CloudProvider

定义了一组接口，不同的云厂商通过实现这组接口来将本厂提供的各种服务集成到kubernetes集群中。

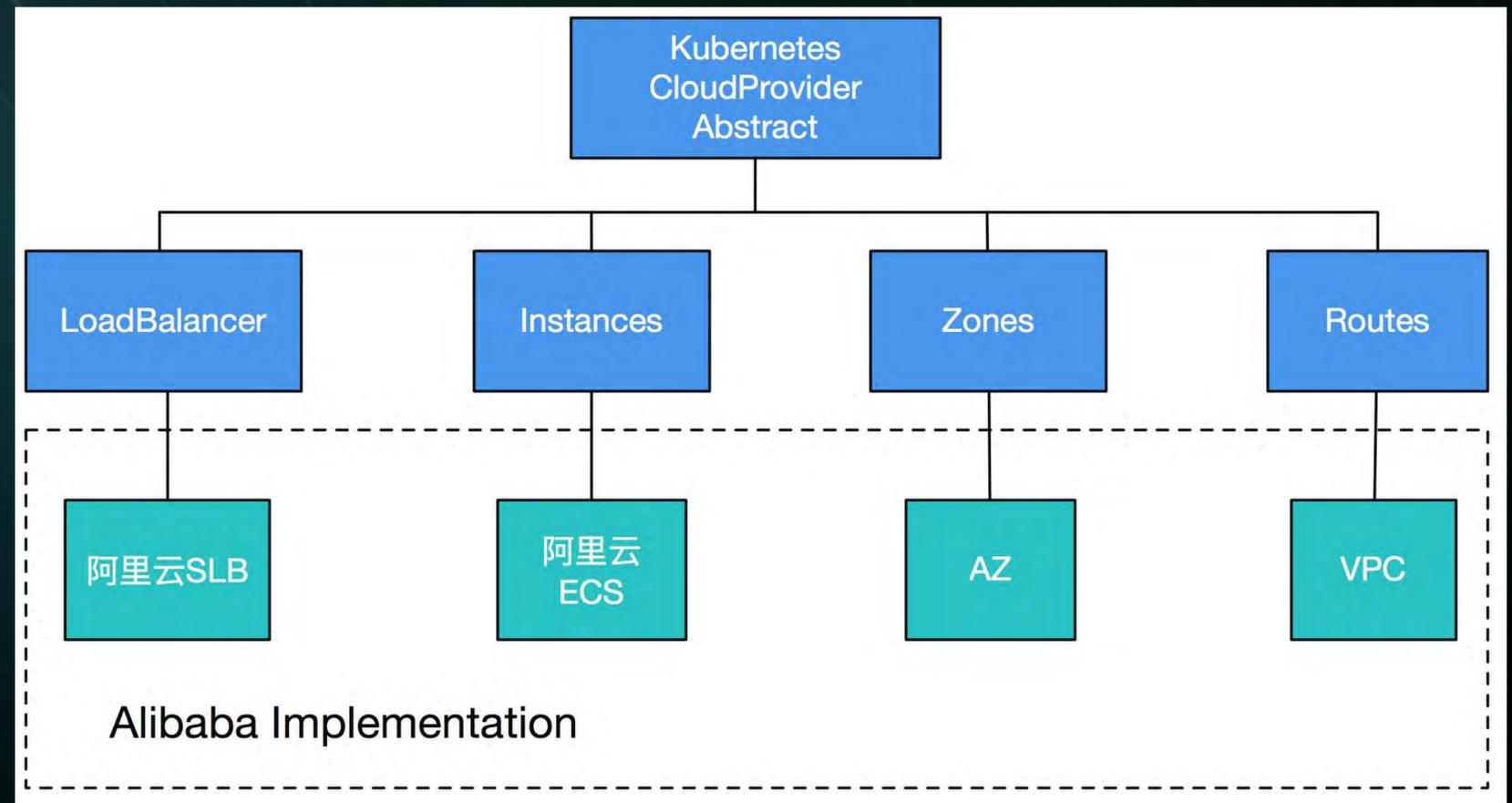
能力集成

网络路由配置、负载均衡动态配置、可用区管理、虚拟机管理

进化

In-tree-cloudprovider
cloud-out-of-tree cloud
controller

<https://github.com/AliyunContainerService/alibabacloud-controller-manager>



Helm/Charts

软件应用仓库

The screenshot displays the Helm Charts repository interface. On the left, a sidebar menu lists navigation options: 容器服务 (Container Service), Swarm, Kubernetes, 集群 (Cluster), 节点 (Node), 监控服务 (Monitoring Service), 日志服务 (Logging Service), 编排模板 (Orchestration Template), and 应用目录 (Application Directory). The main content area features a search bar labeled "Search charts...". Below the search bar, a grid of 12 application charts is displayed, each with its logo, name, version, and stability status.

| Chart Name | Version | Stability |
|-------------------|---------|-----------|
| consul | 0.8.3 | stable |
| grafana | | stable |
| hdfs-datanode-k8s | | stable |
| hdfs-namenode-k8s | | stable |
| jenkins | 2.67 | stable |
| monocular | v0.5.3 | stable |
| mysql | | stable |
| postgresql | | stable |
| redis | 3.2.9 | stable |
| spark | | stable |
| wordpress | 4.8.2 | stable |

Kompose

支持Compose编排k8s部署：简化、复用

Kubernetes Deployment and Service

容器服务 | 创建应用 | 返回应用列表

Swarm | Kubernetes

常见问题: 限制容器的资源 | 高可用性调度 | 通过镜像创建Nginx | 通过编排模板创建Wordpress | 编排模板说明 | 标签说明

应用基本信息 | 应用配置 | 创建完成

```
1 version: '3'
2 services:
3   web:
4     image: wordpress:4
5     environment:
6       - WORDPRESS_DB_PASSWORD=password
7       - WORDPRESS_AUTH_KEY=changeme
8       - WORDPRESS_SECURE_AUTH_KEY=changeme
9       - WORDPRESS_LOGGED_IN_KEY=changeme
10      - WORDPRESS_NONCE_KEY=changeme
11      - WORDPRESS_AUTH_SALT=changeme
12      - WORDPRESS_SECURE_AUTH_SALT=changeme
13      - WORDPRESS_LOGGED_IN_SALT=changeme
14      - WORDPRESS_NONCE_SALT=changeme
15      - WORDPRESS_NONCE_AA=changeme
16     ports:
17       - 80
18     depends_on:
19       - mysql
20     deploy:
21       replicas: 3
22       restart_policy:
23         condition: on-failure
24     labels:
25       aliyun.routing.port_80: wordpress
26   mysql:
27     image: mysql:5.7
28     environment:
29       - MYSQL_ROOT_PASSWORD=password
30     deploy:
31       restart_policy:
```

包含服务

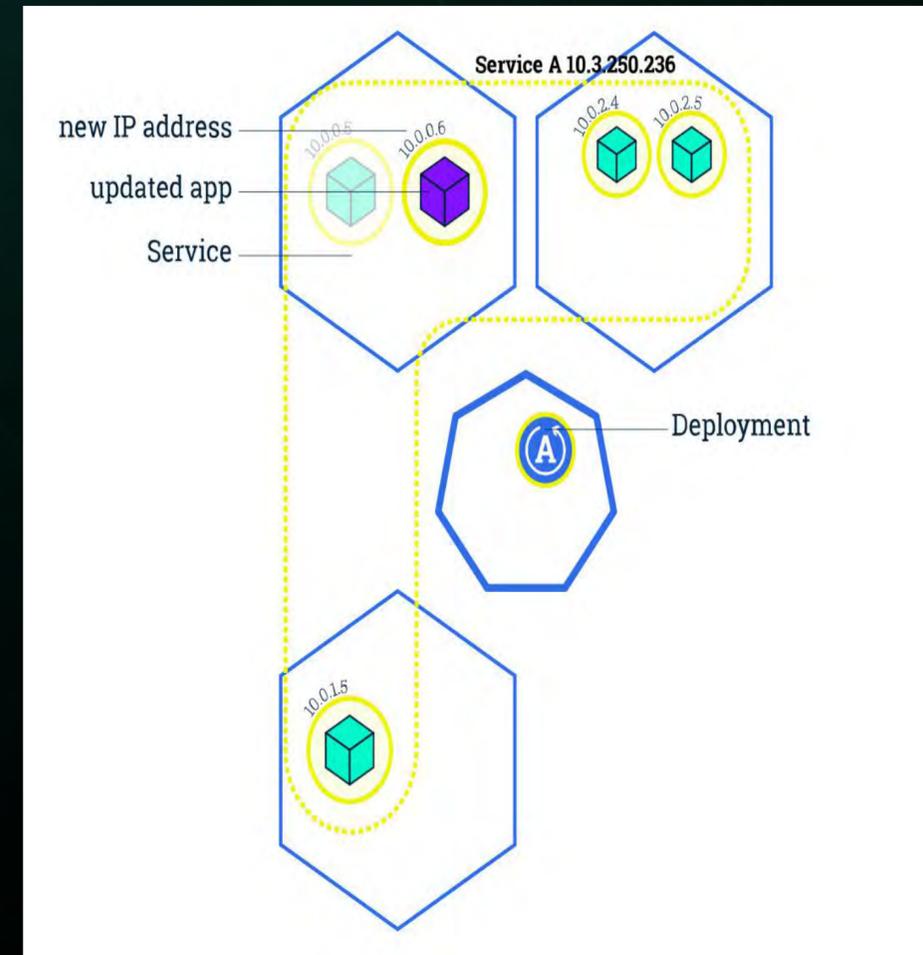
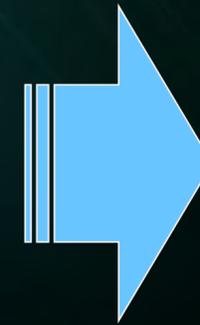
| | |
|-----------------------------|-------|
| 服务名: web 镜像: wordpress:4 | 编辑 删除 |
| 服务名: mysql 镜像: mysql:5.7 | 编辑 删除 |

新增服务



利用编排模板您可以定义和部署多容器应用，支持Docker Compose格式。详情请参见 <https://docs.docker.com/compose/>

使用已有编排模板



2.3 云上应用容器化的最佳实践

如何选择OS

应用容器化的最佳范式

JAVA应用的容器化

透明获取容器资源

应用高可用

OS 选择

■ Docker很多特性依赖于操作系统内核版本

■ 推荐

Ubuntu 16.04 LTS : 4.x内核, 支持丰富的存储驱动

CentOS/RHEL 7.4: 3.10内核, 向后移植了很多对容器的增强

Windows Server 2016

在Swarm集群中, Windows节点应该作为工作节点

应用容器化的最佳范式

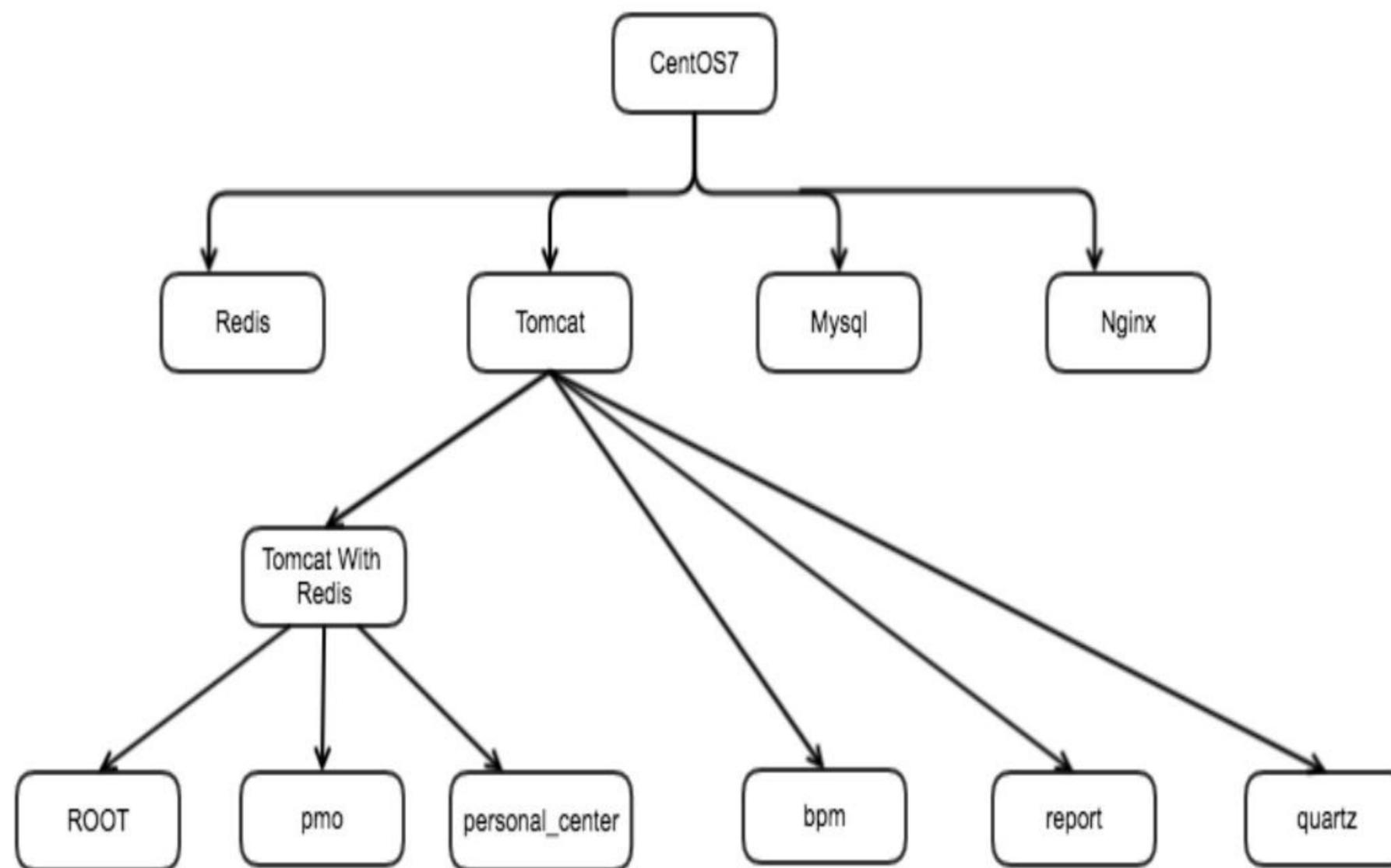
| | Virtual Machine | 容器 | 函数 |
|------|---------------------------------|--|-------------------------------------|
| 最佳场景 | 企业核心应用 (E.g. ERP, CRM, ...) | 互联网应用和企业应用 | 事件驱动计算 (E.g. 转码, ETL, ...) |
| 应用交付 | 虚拟机镜像/ 软件配置管理、自动化 脚本 | Docker 镜像 | 代码/lib |
| 对应方案 | 虚拟化 | ContainerService/ ContainerInstance | FunctionCompute OpenFaaS/Fission |

应用容器化的最佳范式-镜像分层

操作系统

中间件

应用

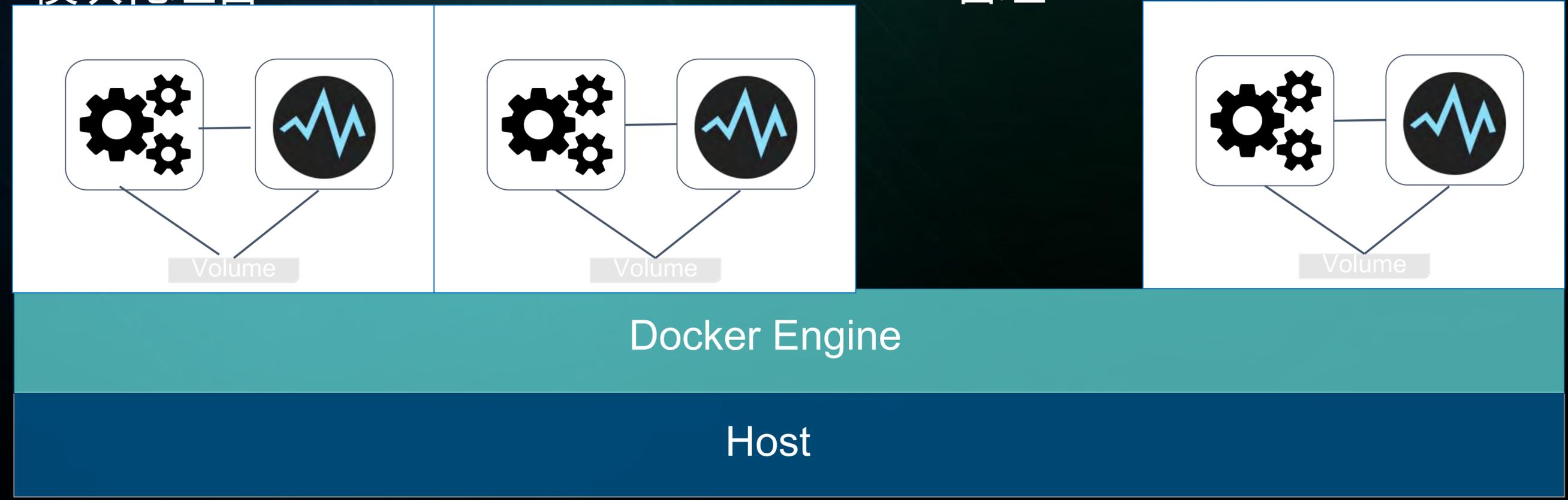


应用容器化的最佳范式-SideCar

镜像化应用交付
模块化组合

容器动态发现注册，如DNS，LB或定制IPAM分配IP

- 安全 - 减少攻击面
- 解耦 - 应用运维，独立更新
- 高效 - 宿主机内部共享监控、日志
- 可控 - 不可变基础架构，支持版本管理



容器管理命令
API
控制台

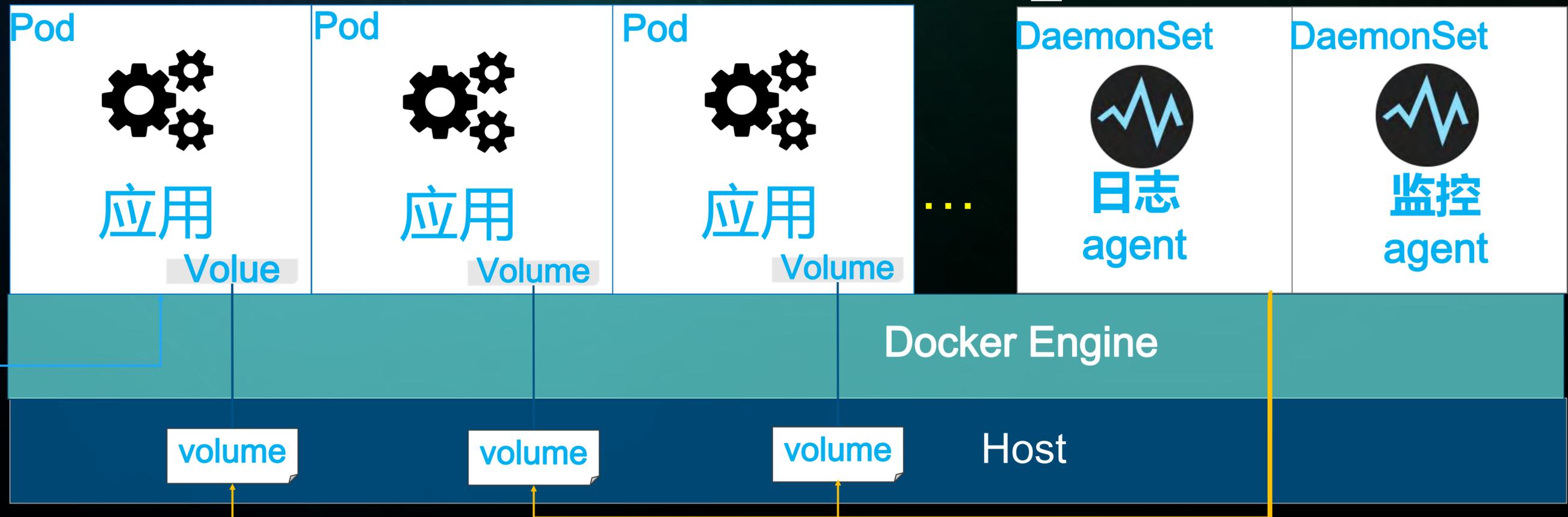
应用容器化的最佳范式-DaemonSet

镜像化
应用交付

容器动态发现注册，如DNS, LB
或定制IPAM保证Ip不变

安全 - 减少攻击面
解耦 - 应用运维，独立更新
高效 - 宿主机内部共享监控、日志
可控 - 不可变基础架构，支持版本管理

容器管理命令
API
控制台



Java应用的容器化迷思

Java应用容器化后相互影响、莫名OOM：早期JVM

- 缺省堆栈大小取决于宿主机内存大小
- 缺省GC, JIT编译线程取决于宿主机CPU核数

解决方案

- Java SE 8u131+ 和 JDK 9 已经支持对容器环境资源限制的自动感知
 - `java -XX:+UnlockExperimentalVMOptions -XX:+UseCGroupMemoryLimitForHeap ...`
- Docker 1.7 开始将容器cgroup信息挂载到容器中
 - 如可以从 `/sys/fs/cgroup/memory/memory.limit_in_bytes` 获取内存设置
 - 在容器的应用启动命令中根据Cgroup配置的正确设置 `-Xmx, XX:ParallelGCThreads`等参数

透明获取容器资源

lxcfs

透明挂载lxcfs



apiVersion: apps/v1beta1

kind: Deployment

metadata:

annotations:

"initializer.kubernetes.io/lxcfs": "true"

labels:

app: helloworld

name: helloworld

spec:

replicas: 1

template:

metadata:

labels:

app: helloworld

name: helloworld

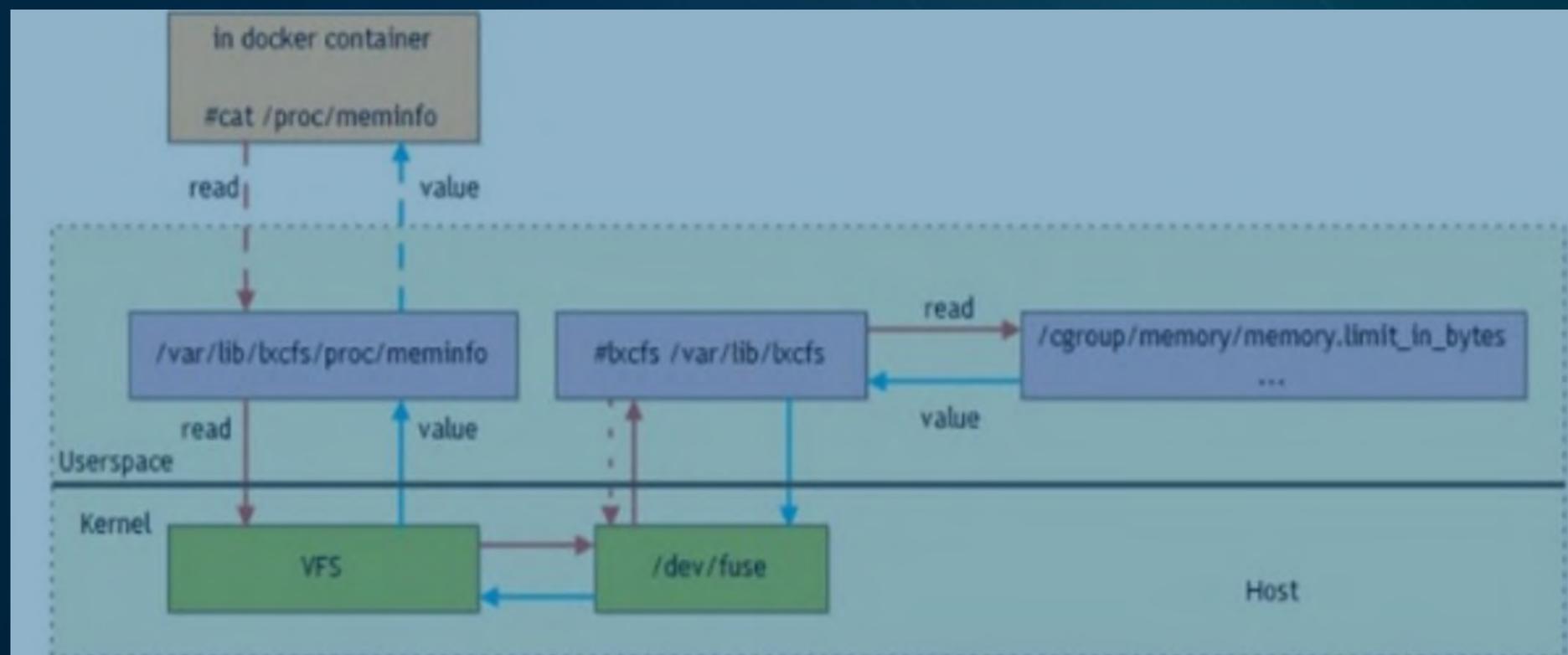
spec:

containers:

- name: helloworld

image: myapp

imagePullPolicy: Always

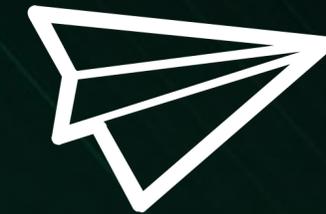


应用高可用

跨AZ (可用区)

跨Region(地域)





感谢聆听

Thanks !