CCTC 2017 中国云计算技术大会
Cloud Computing Technology Conference 2017
CSDN

Swarm的进化与大规模应用

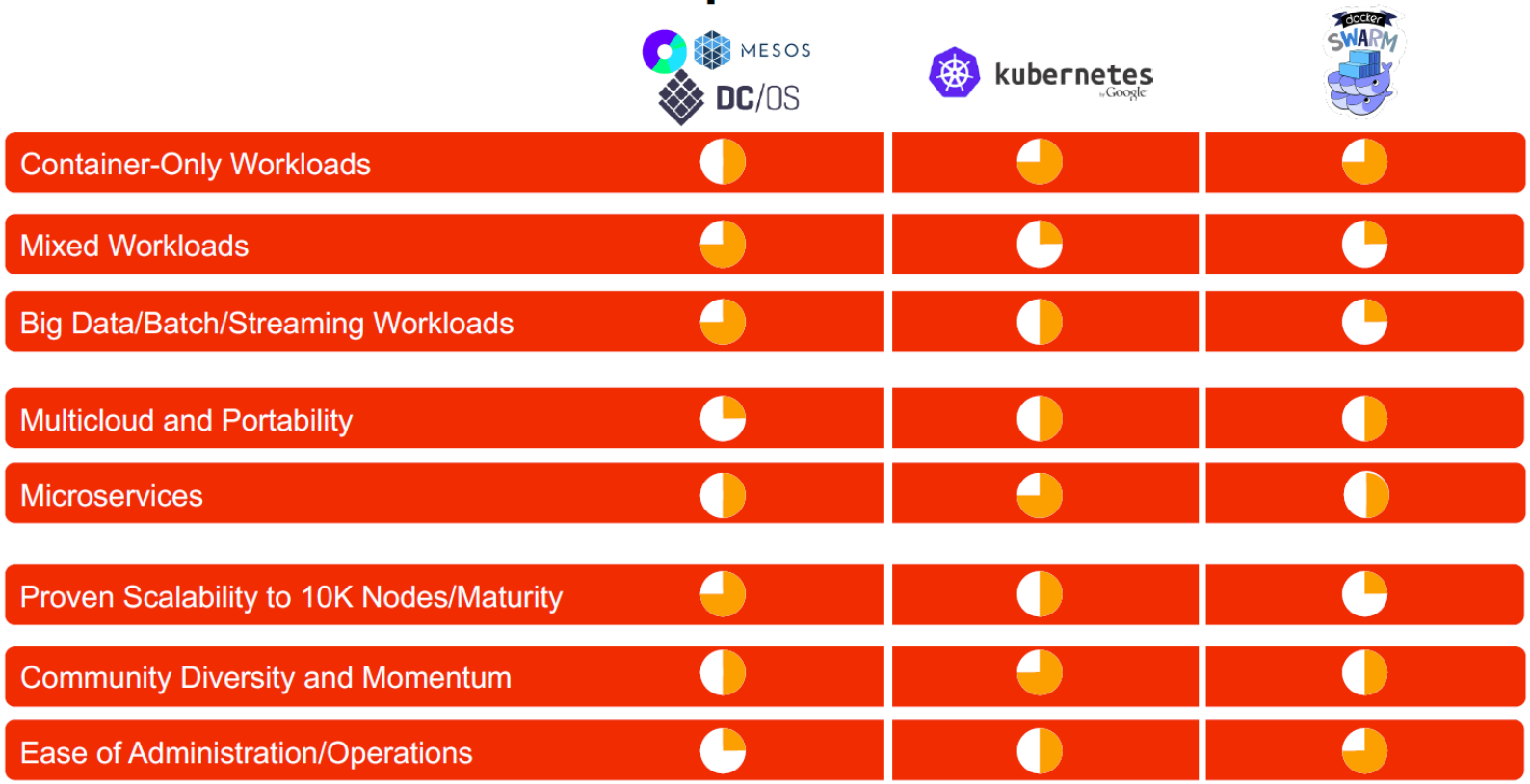
阿里云容器服务 陈萌辉

内容提要

- Docker与容器编排
- Swarm简介
- SwarmMode简介
- Swarm在阿里的应用

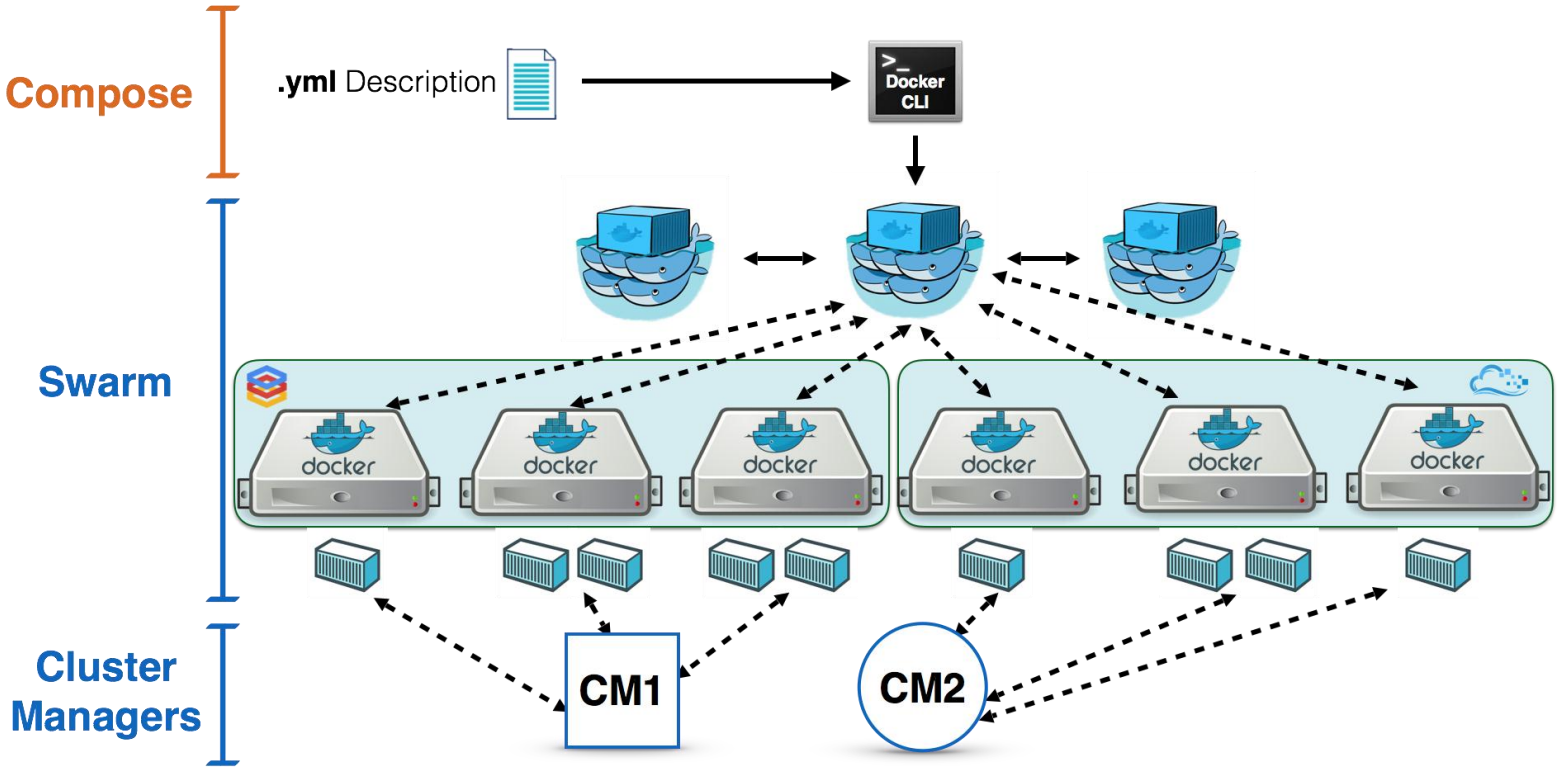
容器编排三分天下

How Do Orchestrators Compare?

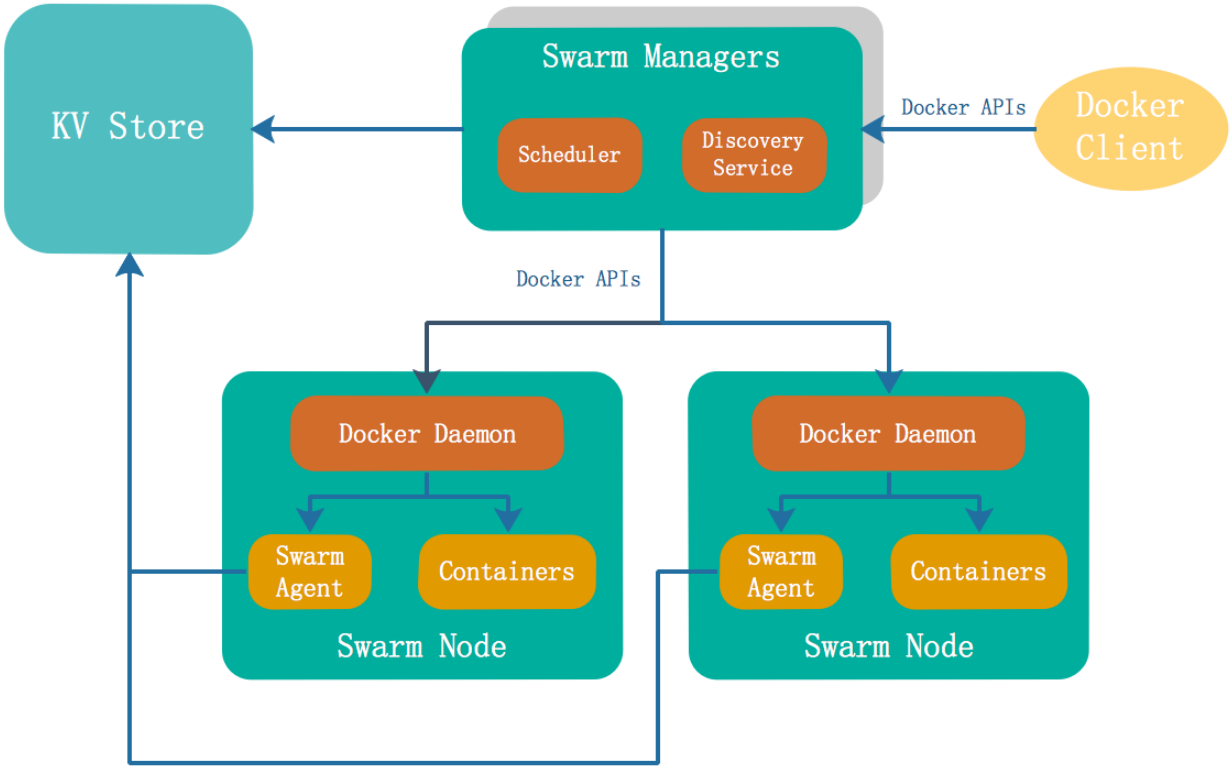


Swarm是什么

- Docker公司继Docker Engine之后的重要产品
- 集群管理系统
- 容器编排与调度系统



- 依赖外部存储来完成节点发现并保证一致性
- Manager只跟Daemon通信，不跟Agent通信
- Manager可以有多副本



- Swarm Manager的高可用设计
  - 一主多热备
    - 所有manager都同时连接所有Daemon
    - 备转发请求至主
  - 依赖外部KV选主
    - 抢锁 -〉保活

- 集群类：info events
- 容器类：get/list、create、start/stop等
- 镜像类：get/list、push、pull、tag等
- 数据卷类：get/list、create、delete
- 网络类：get/list、create、delete等

- 高度兼容Docker Engine API
  - 集群级汇总
  - 转发到相应节点的Docker Daemon
  - 在集群中广播
- 单个容器级别的API

- 资源调度
  - 资源维度： CPU / Memory / 端口
  - CPU / Memory支持超卖
  - 调度策略：spread / binpack
  - 不支持优先级、抢占

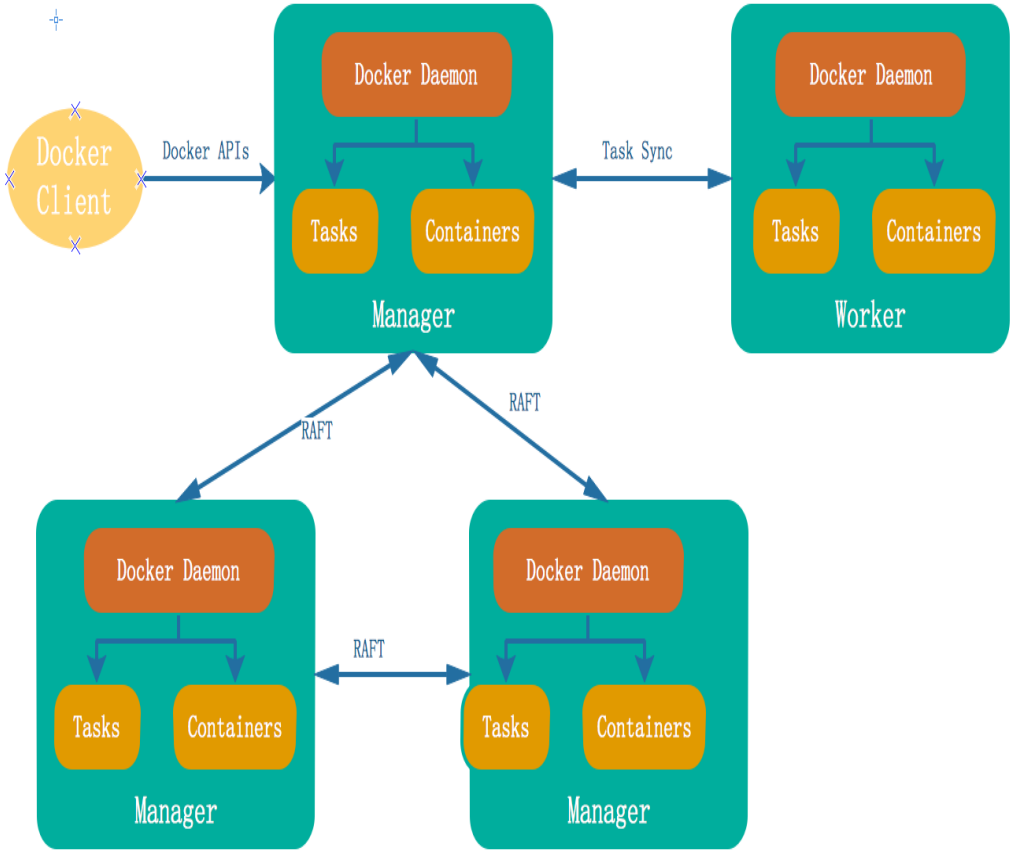
- 节点约束
  - 节点名：constraint:node==XXX
  - 标签：constraint:key==value
- 亲和性
  - 镜像：affinity:image==foo
  - 服务：affinity:service==foo

- 部署简洁
  - 只依赖KV Store和Docker Daemon
  - 所有组件都容器化
- 高效友好的用户交互
  - 高度兼容Docker Engine API，可直接使用Docker Client
- 灵活的约束与亲和性描述

- 不足
  - 容器级别的API，抽象层次较低
  - 响应式设计，不方便执行常驻后台的操作
  - 依赖定期同步跟Docker Engine保持一致状态

- Docker 1.12版开始提供
- 将Swarm的集群管理、容器调度功能集成进Docker Engine
- 提供Service级别抽象
- 自带负载均衡

- 无任何外部依赖
- Daemon身兼Engine、Manager、Agent三职
- Managers之间通过RAFT协议组成分布式强一致性KV Store
- Manager与Worker的Daemon不通信



- 高可用设计
  - Manager数量需要>=3
  - 一主多热备
- 动态添加 / 删除Manager

- 集群管理类
  - init、join、leave
  - token
- 服务类
  - get/list
  - create、delete、update
  - inspect、ps

- 两类API
  - Swarm、Service、network类，只有Manager能处理
  - 容器、镜像、数据卷类，所有节点都能处理
- 高度兼容旧API

- Service、Task、Container三级概念
  - Service：相同功能的一组容器
  - Task：任务调度单元，由Manager生成，同步至Worker
  - Container：Task落地
- Rolling Update

- Replicated Service
  - 用户指定副本数
  - Reconciled：自动确保副本数
  - constraint
    - node.id node.hostname
  - node.role
  - node.labels engine.labels

- Global Service
  - 每个节点有且仅有一个容器
  - 添加加点时自动扩展
  - 可附加constraint
- 网络模型
  - 支持overlay网络，同一网络内，服务名、容器名可解析
  - 一个服务一个网络
  - 服务发现：不同服务可加入同一个网络

- Service自带的负载均衡
  - 基于LVS
- 两种模式
  - VIP：每个服务一个VIP，通过LVS实现；服务名解析至VIP
  - DNS：服务名解析至容器IP，RoundRobin方式
- 服务发生变化时，自动调整后端

Swarm Mode：总结

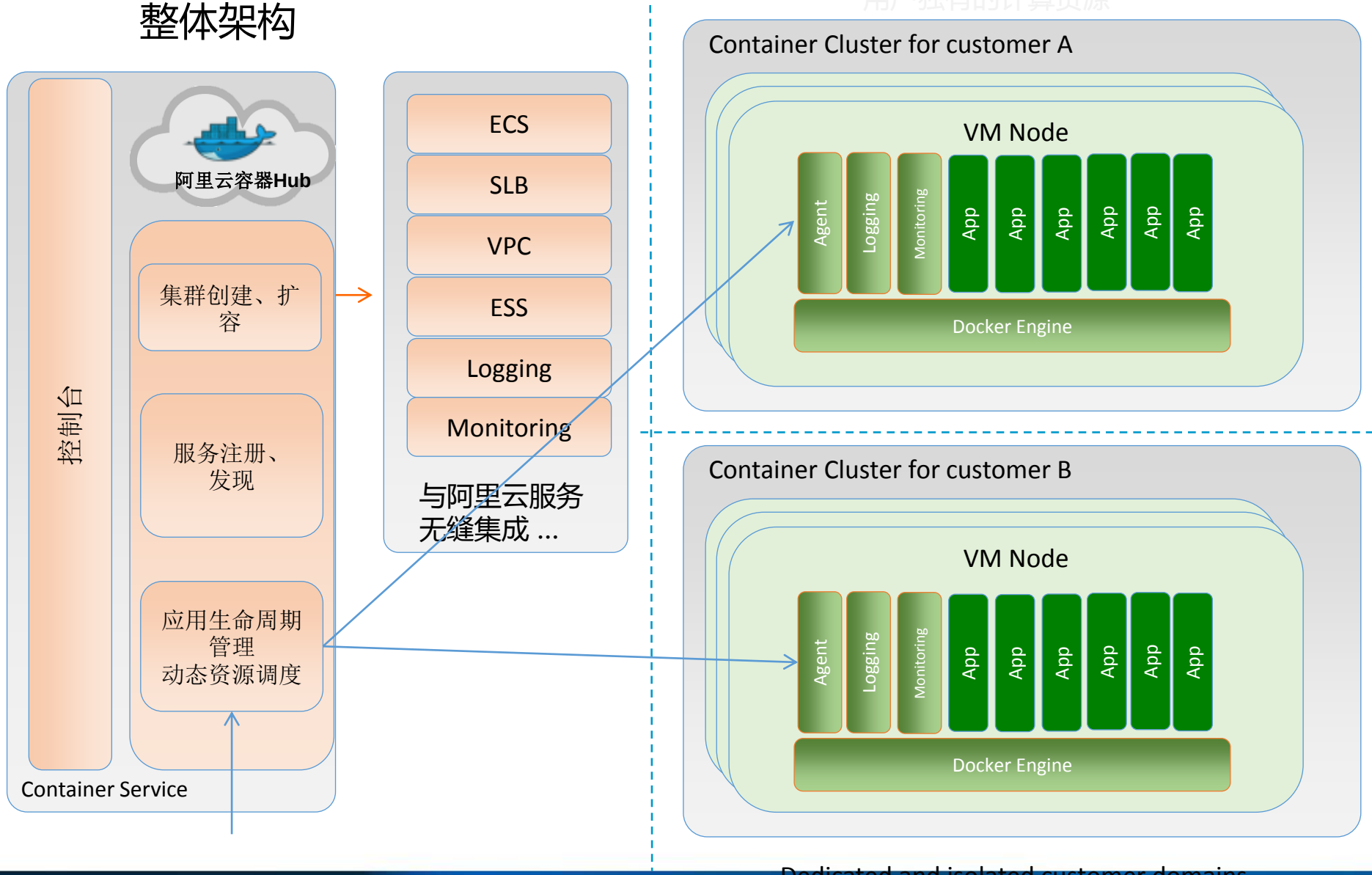
- 部署特别简洁
  - 无任何依赖，只需安装Engine + 一个命令
  - 无中心架构
- 部署高可用服务
  - Service API + RoutingMesh
- Secure by default：
  - 自带证书颁发、更新功能，Manager与Worker之间通过SSL连接

- 不足
  - 只有Service级抽象，Stack级抽象仍无API
  - 部署有状态服务较复杂
- Service API有很多容器特性不支持，如host network、host pid、privileged等
- 无法自举，需要手工init

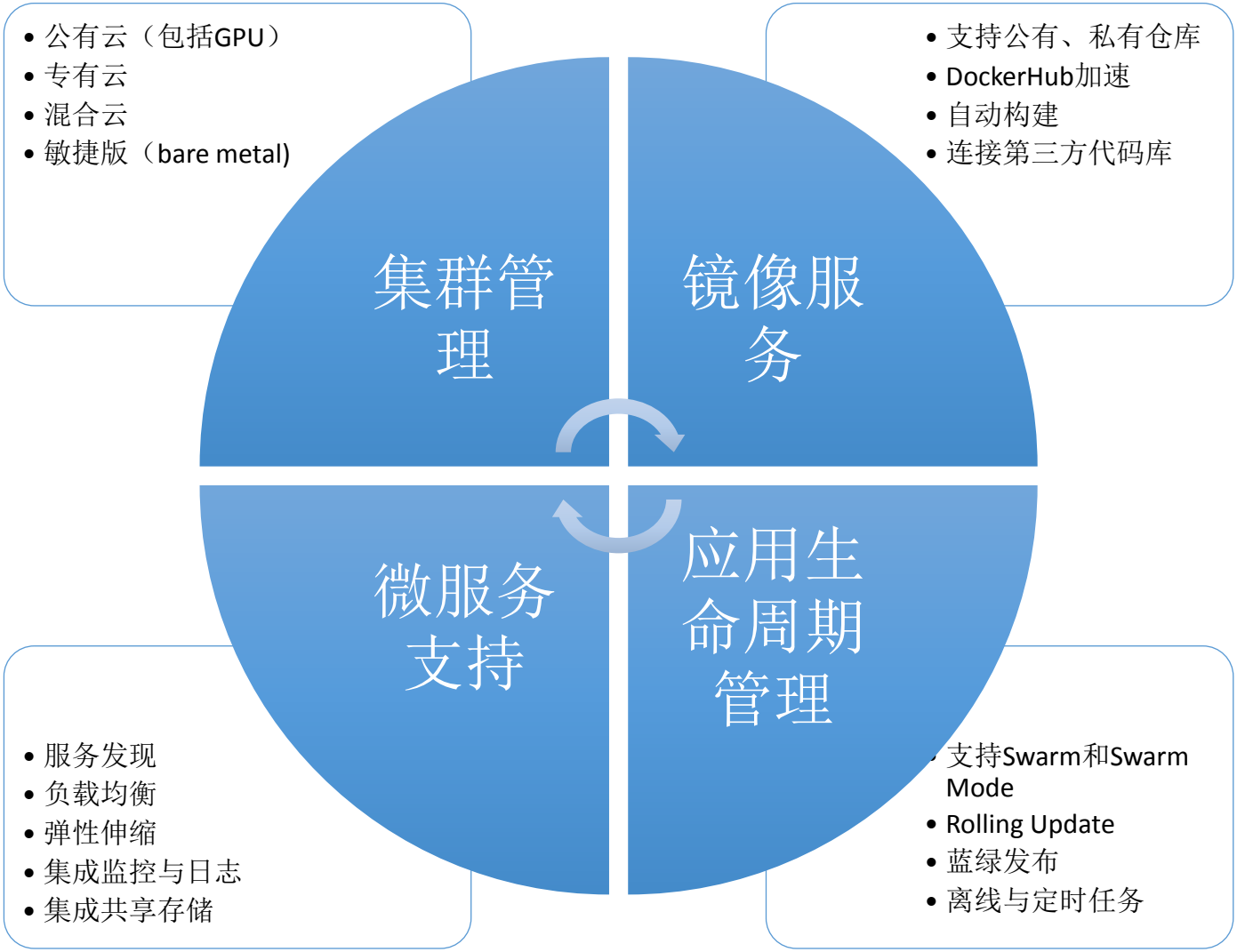
Swarm在阿里的应用

- 阿里内部应用Docker化
  - Swarm统一管理集群
  - 阿里云专有云输出
- 阿里云容器服务
- 阿里云高性能计算、HPC

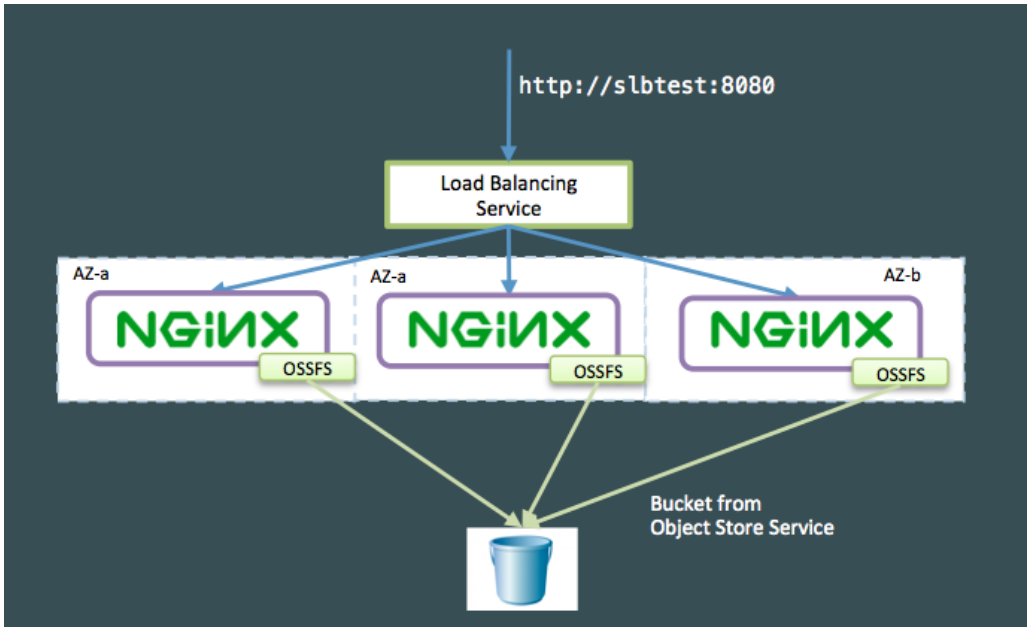
整体架构


用户独有的计算资源
Container Cluster for customer A
VM Node
Agent
Logging
Monitoring
App
App
App
App
App
App
Docker Engine
Container Cluster for customer B
VM Node
Agent
Logging
Monitoring
App
App
App
App
App
App
Docker Engine
控制台
阿里云容器Hub
集群创建、扩容
服务注册、发现
应用生命周期管理
动态资源调度
Container Service
ECS
SLB
VPC
ESS
Logging
Monitoring
与阿里云服务无缝集成 ...
Dedicated and isolated customer domains

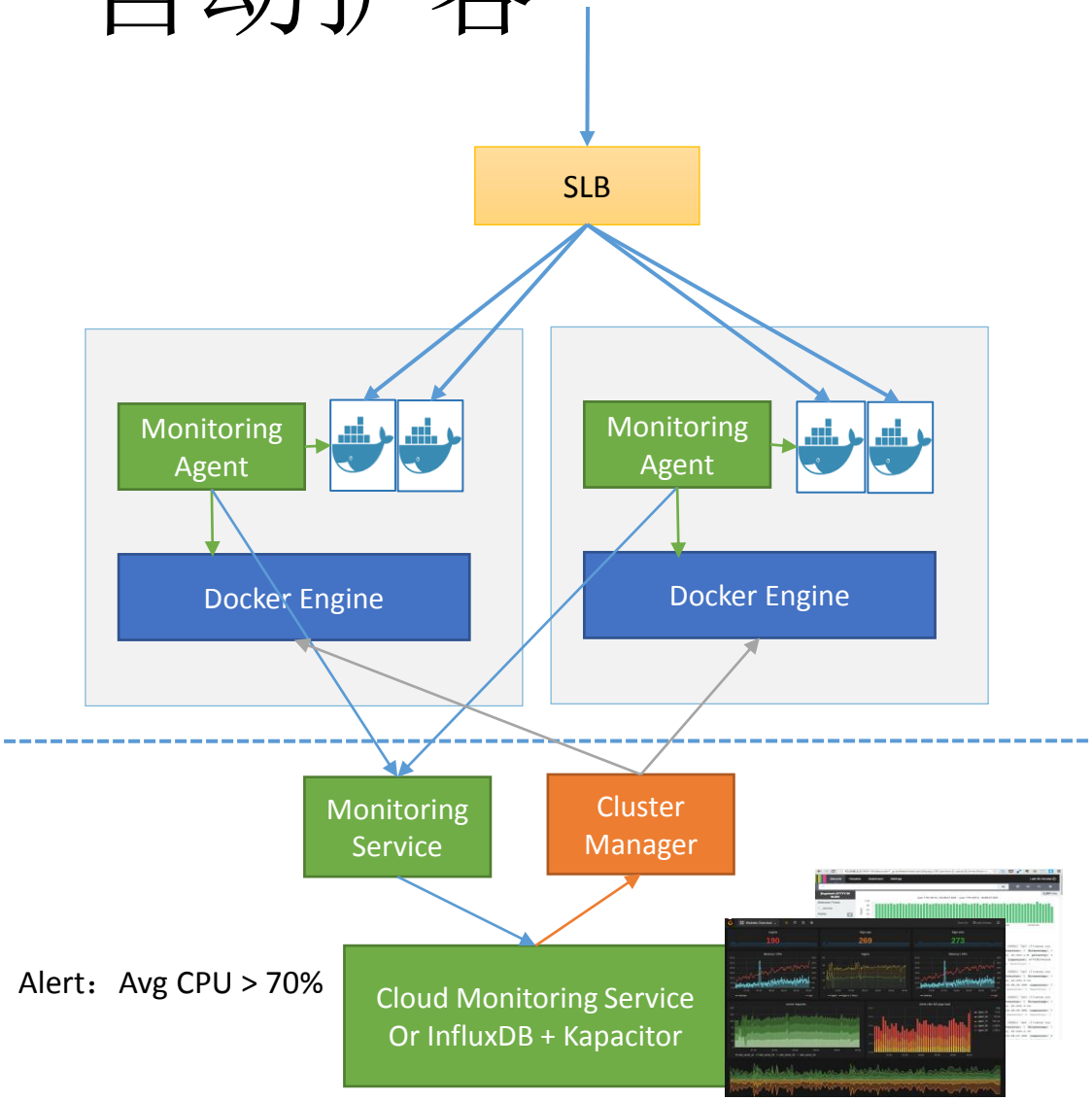
阿里云容器服务


公有云（包括GPU）
专有云
混合云
敏捷版（bare metal)
集群管理
支持公有、私有仓库
DockerHub加速
自动构建
连接第三方代码库
镜像服务
微服务支持
服务发现
负载均衡
弹性伸缩
集成监控与日志
集成共享存储
应用生命周期管理
支持Swarm和Swarm Mode
Rolling Update
蓝绿发布
离线与定时任务

```
version: ”3.0”          #支持v1 v2 v3 compose
模板
services:
  nginx:
    image: nginx:latest
    deploy:
      mode: replicated
      replicas: 3
ports:
      - 8080:80
    labels:
      aliyun.lb.port_8080`: tcp://slbtest:8080
#负载均衡
      aliyun.log_store_dbstdout: stdout
#日志收集
      aliyun.log_store_varlog: /var/log/*.log
    volumes:
      - ’website:/usr/share/nginx/html’
volumes:
  website:
    driver: ossfs  #共享存储数据卷，支持oss、
nas
    driver_opts:
      bucket: acs-sample
```



自动扩容



- 声明式自动扩容

aliyun.auto_scaling.max_cpu: 70

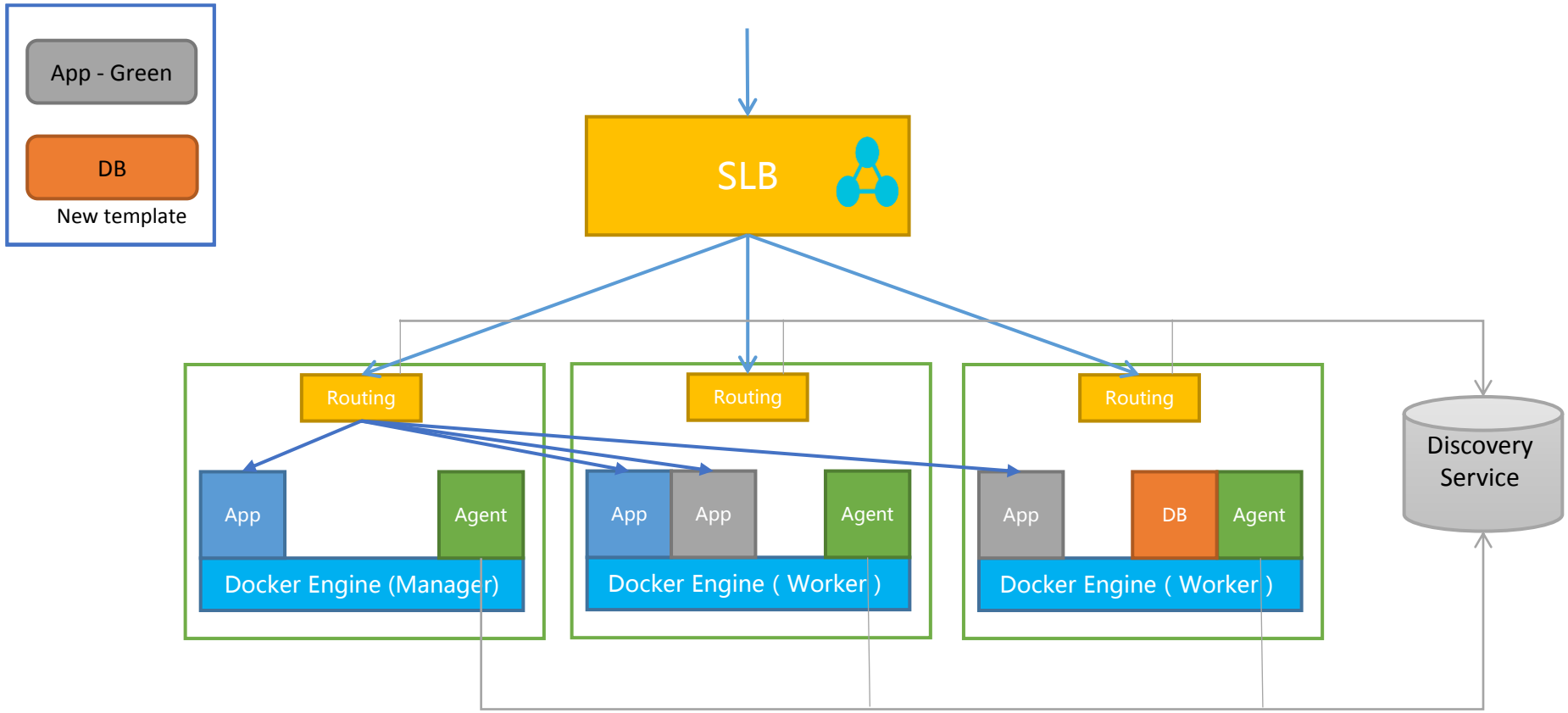
aliyun.auto_scaling.step: 2

- 监控插件

输入: nagios, apache, docker, UDP, ...

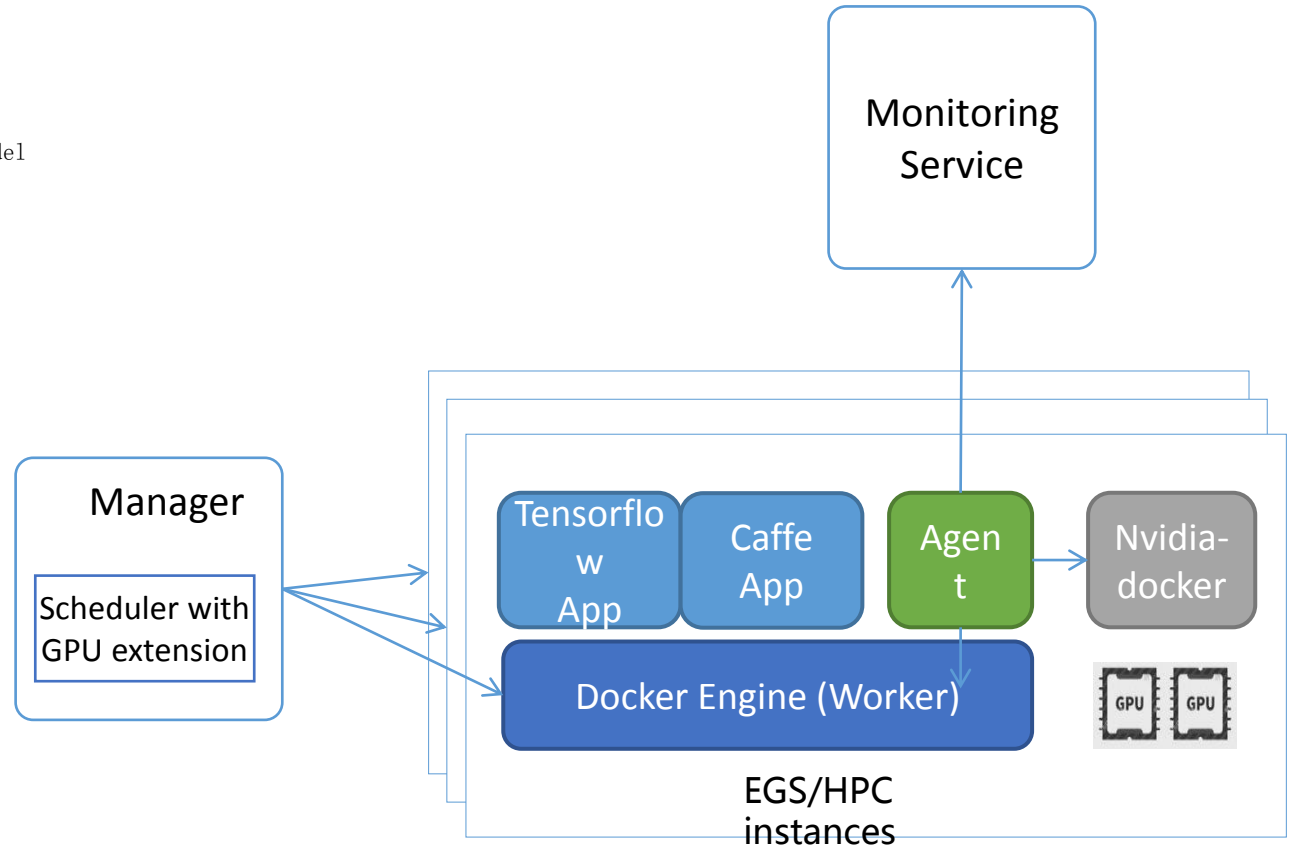
输出: Influxdb, prometheus, kafka

蓝绿发布


App - Green
DB
New template
SLB
Routing
Routing
Routing
App
Agent
Docker Engine (Manager)
App
App
Agent
Docker Engine (Worker)
App
DB
Agent
Docker Engine (Worker)
Discovery Service

机器学习

```
version: '2'
services:
  inception:
    image: acs_sample/inception:demo
    volumes:
       - inception_model/inception_model
    labels:
      - aliyun.gpu=2
    ports:
      - "9000:9000"
volumes:
  inception_model:
    driver: nas
```



云栖社区