amazon web services | Webinar

intel

运行在ECS上的微服务架构

Running Microservices on Amazon ECS

李磊，AWS解决方案架构师
Leon Li, Solutions Architect, Amazon Web Services

议程

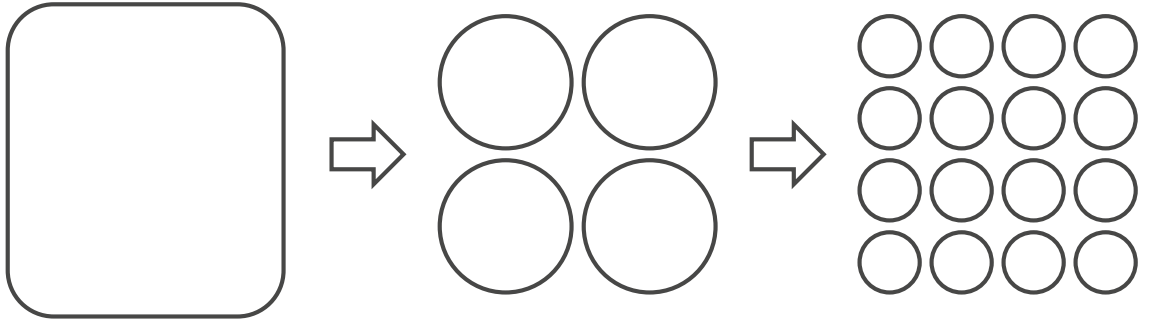
- 面向微服务的架构和SOA以及整合的架构有何不同，为什么我们需要面向微服务的架构
- 探讨微服务架构在扩展中所面临的挑战有哪些
- 在AWS上如何利用ECS管理微服务的应用
- Instacart的Global客户案例分享

微服务架构概览

什么是微服务

“**A software architecture style** in which complex applications are **composed of small, independent processes** communicating with each other using language-agnostic APIs. These services are **small, highly decoupled** and focus on **doing a small task**, facilitating a **modular approach** to system-building.” - Wikipedia

整合 vs SOA vs 微服务

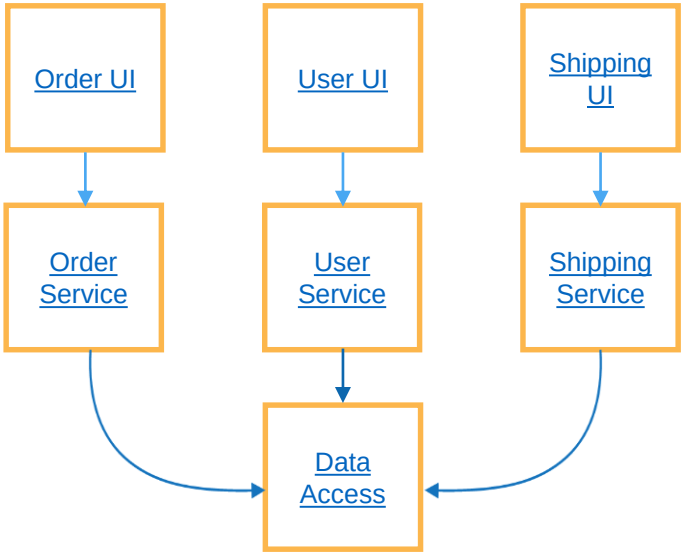


Monolithic
Single Unit

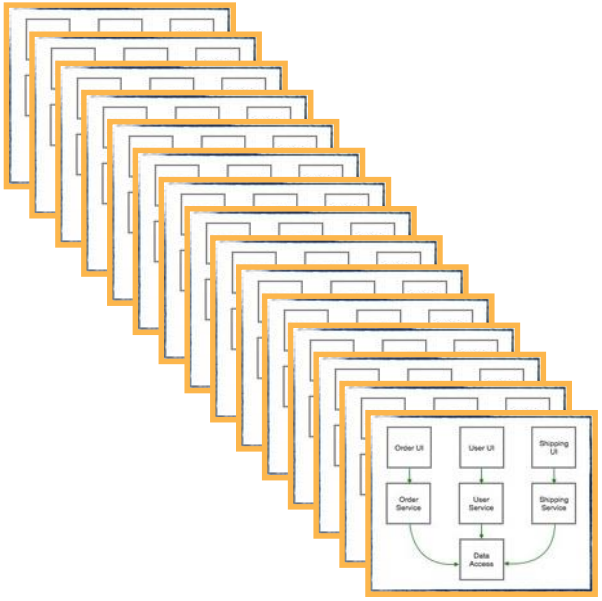
SOA
Coarse-grained

Microservices
Fine-grained

整体型架构(模块化)

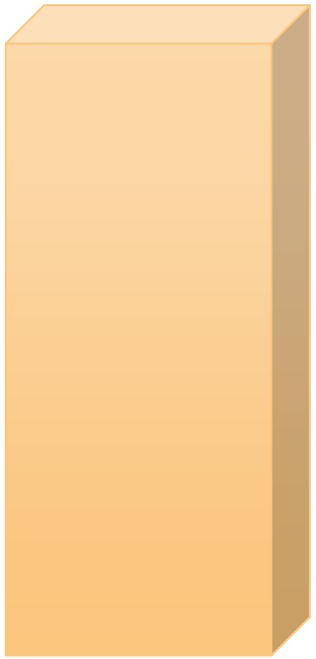


整体型架构 – 扩展



整体型架构的开发和部署流程

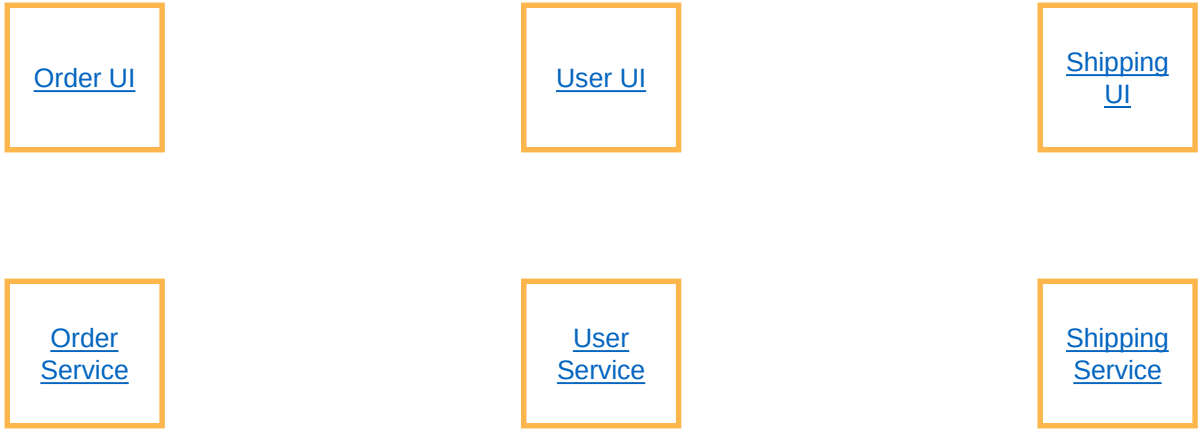
开发小组



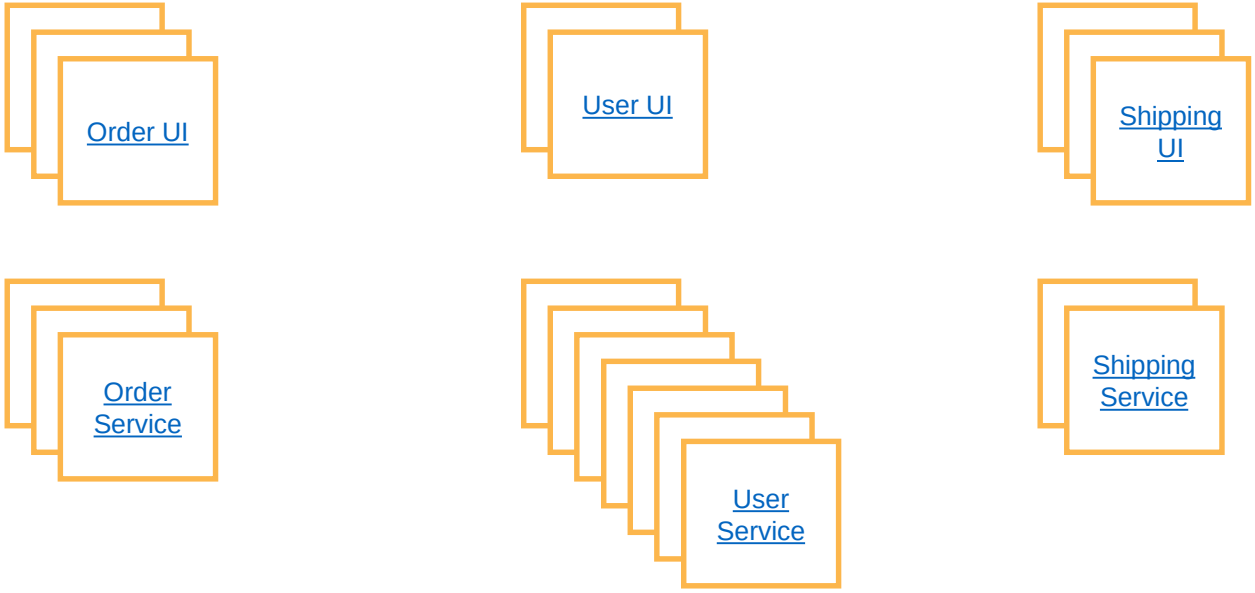
应用程序

开发流

微服务的架构

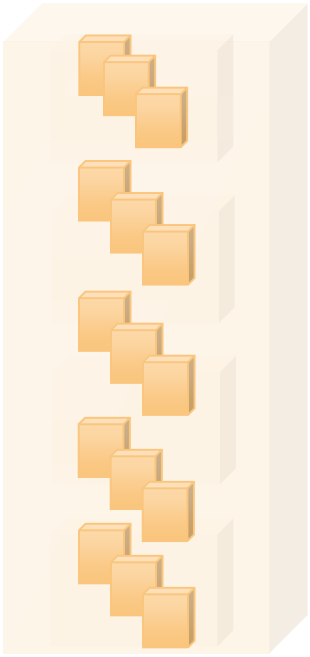


微服务的架构 – 扩展



微服务的开发部署流程

developers



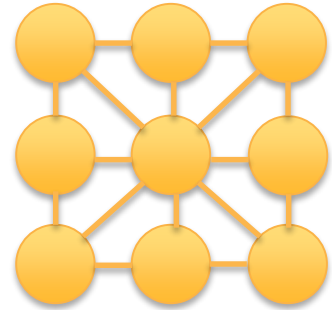
microservices

delivery pipeline

微服务架构的典型特征

去中心化

多语言集成



无依赖

努力做好一件事情

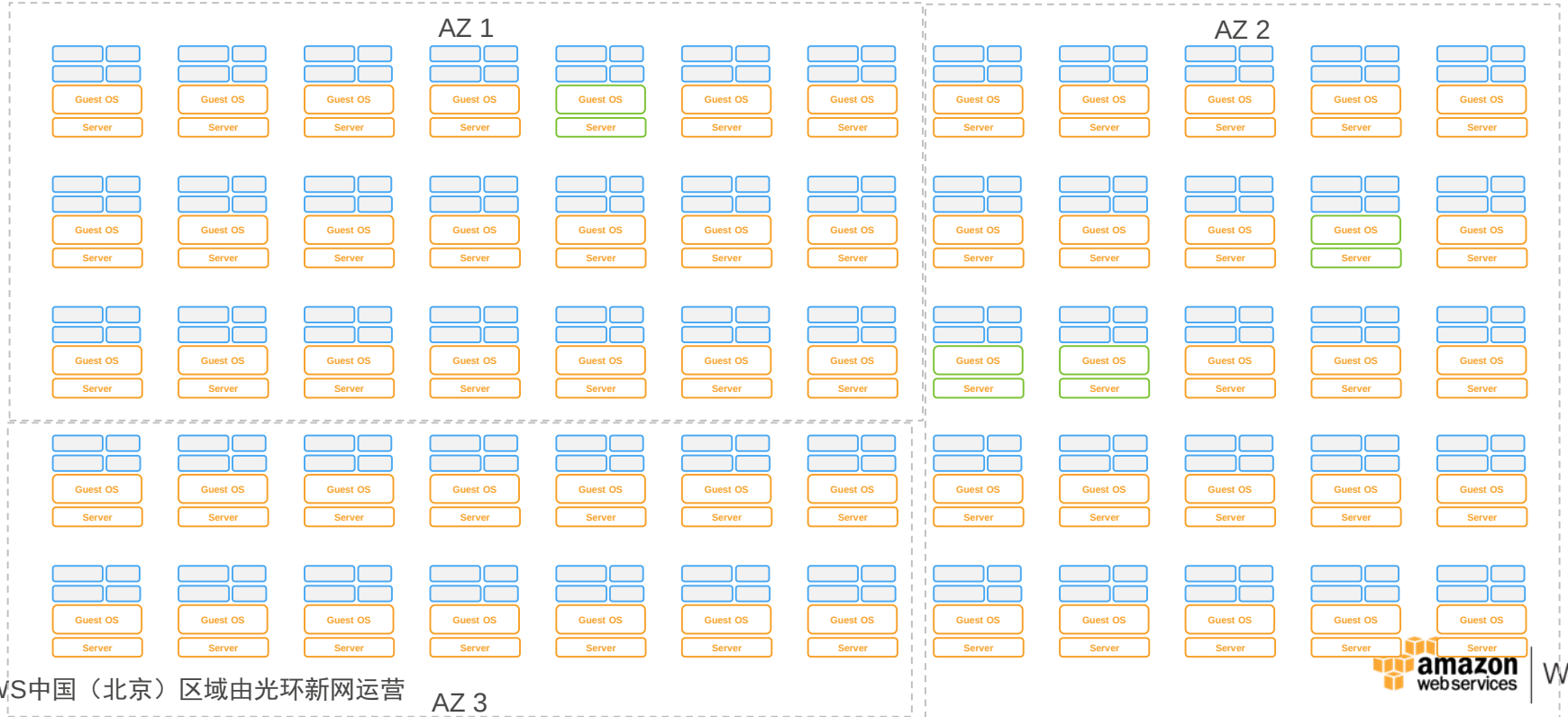
黑盒子

谁建立的，谁负责跑

运行微服务所面临的挑战

微服务架构挑战 #1 – 资源管理

我们无法手动管理一个大型的集群



微服务架构挑战 #2 – 监控

在一整个微服务的架构中可能存在10, 100, 1000, 甚至是10,000个独立的服务.

- 你如何能够知道每个服务的健康状态?
- 你如何知道每个服务的性能指标是否正常?
- 你如何诊断和调试每个服务?

微服务架构挑战 #3: 服务发现

每组服务在扩容或者缩容的时候都可能涉及到其他服务.

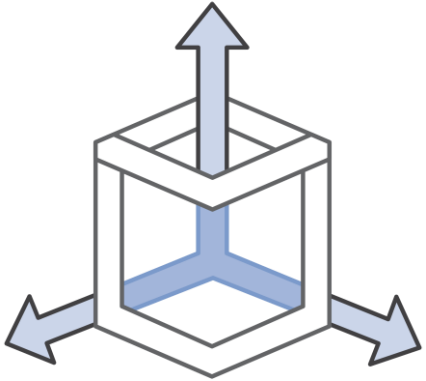
- 服务A怎样知道服务B所有下面挂载实例的URL?
- 你如何保证服务在扩容过程中还是在使用负载均衡器作业?
- 一个新的服务的实例如何让别人发现它?

微服务架构挑战 #4: 部署

在一整个微服务的架构中可能存在10, 100, 1000, 甚至是10,000个独立的服务.

- 每个服务都将会被开发，测试和部署在一条各自的流水线上，那么你如何管理数目众多的这些流水线？
- 服务是多语言的环境 – 不同的开发语言，以及不同的依赖库，我们如何有效的构建和部署它们？
- 你如何决定服务被部署在哪个host上?

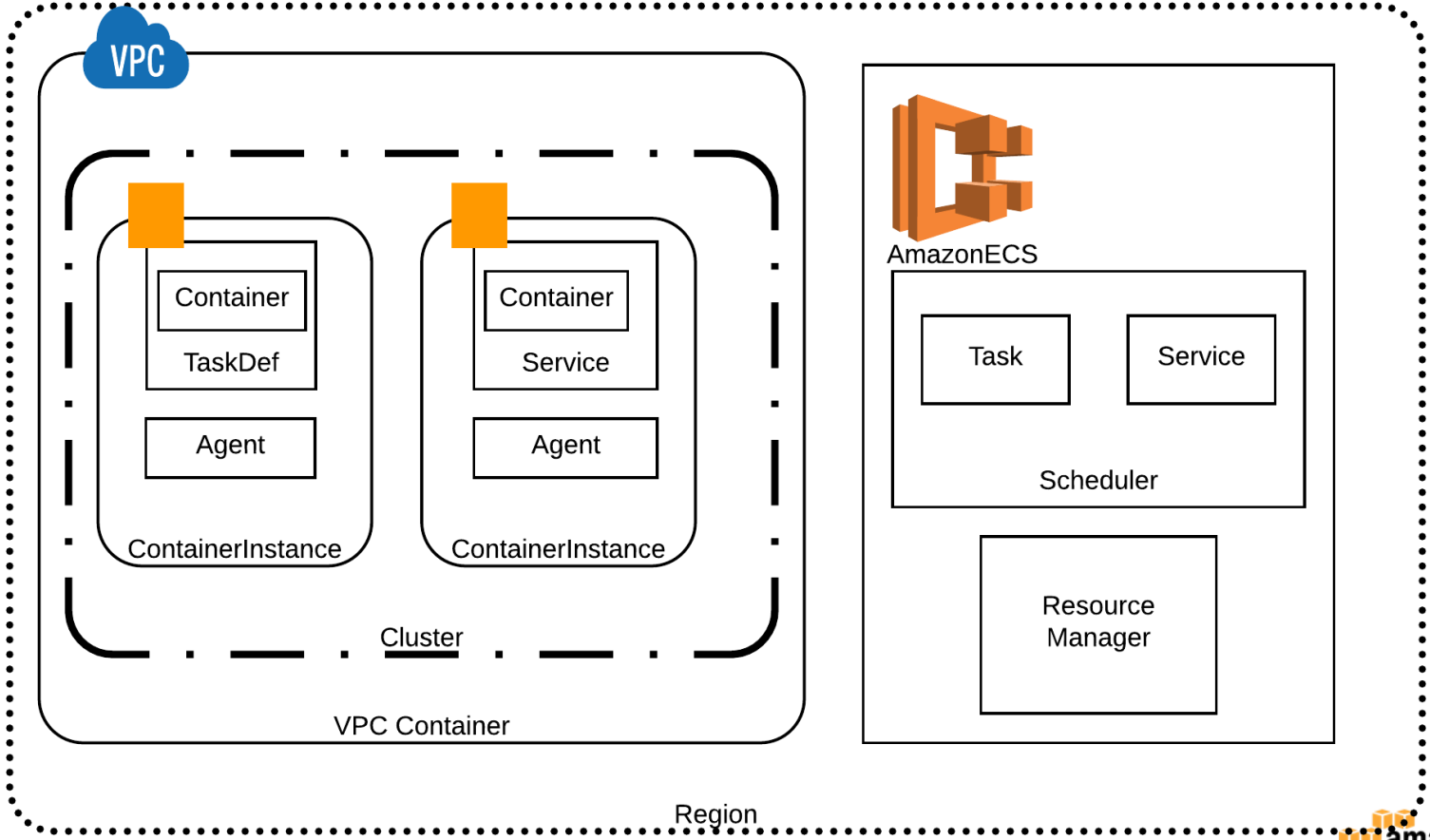
Amazon ECS简介



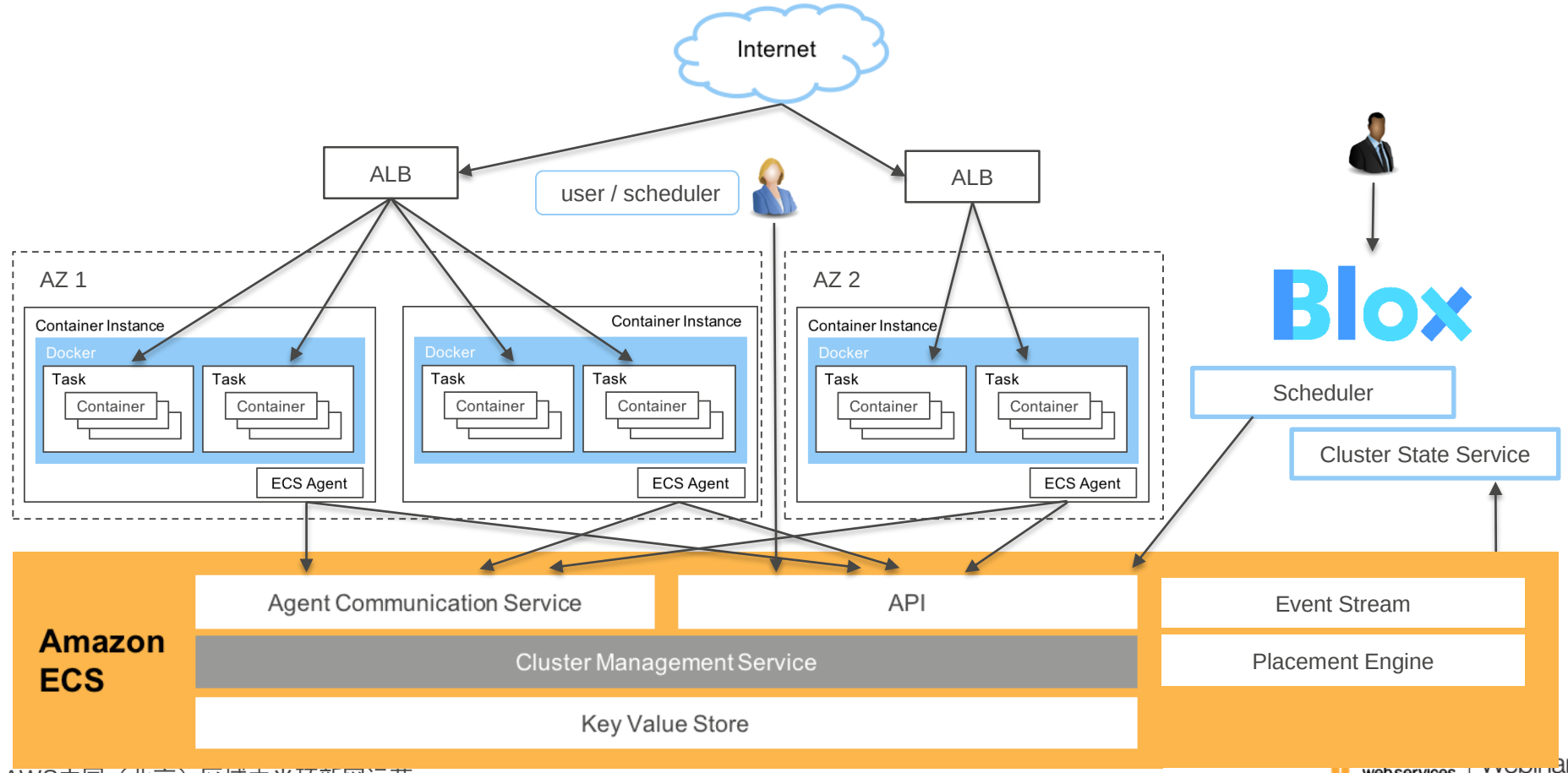
- 全托管的弹性服务 - 当你当集群中的微服务架构扩容时，你无须担心任何事情.
- 集群的资源调度是共享了下层宿主机资源的.
- 资源管理上支持完整的ACID
- 轻松的与CloudWatch服务整合监控与日志管理
- CI/CD工作流轻松与云上的Code*服务结合

Amazon ECS 架构

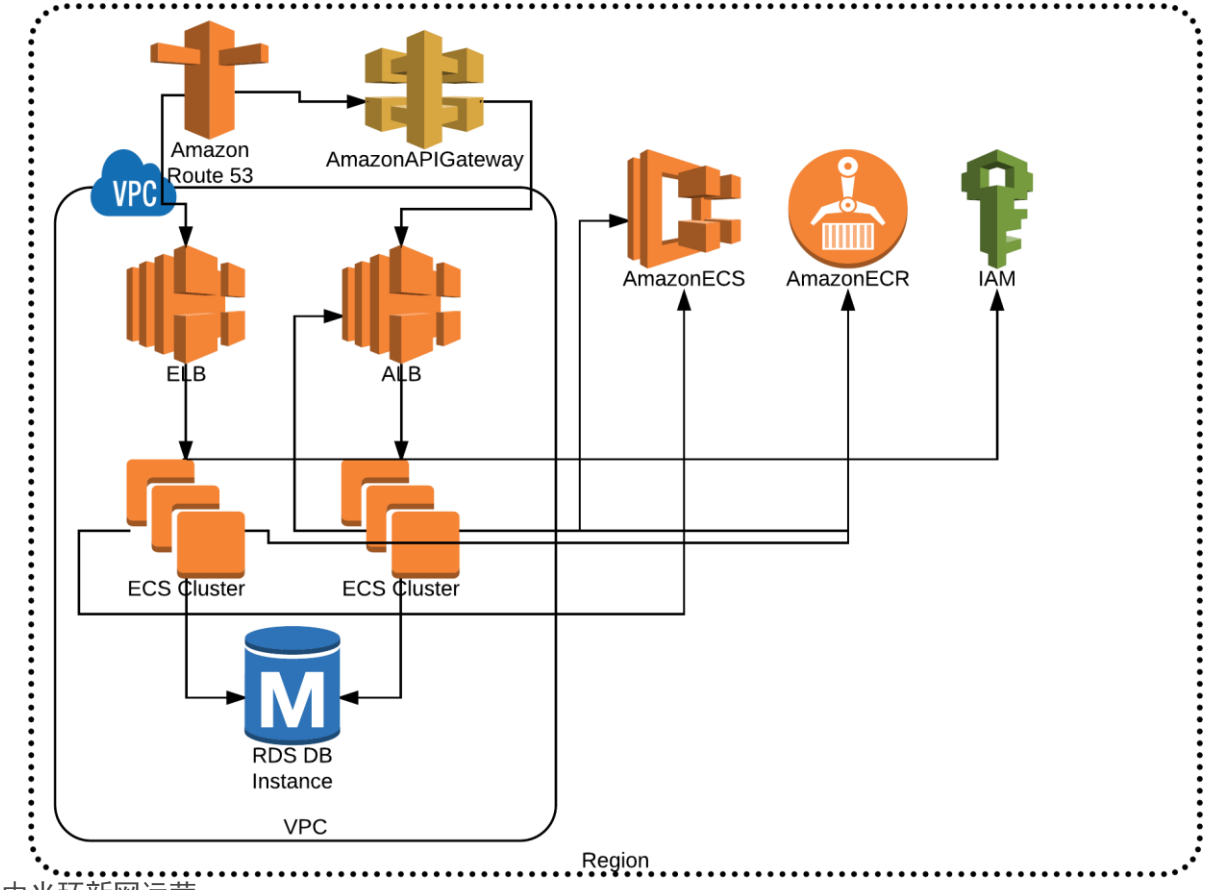
Amazon ECS 架构



Amazon ECS 工作原理



ECS上的微服务架构



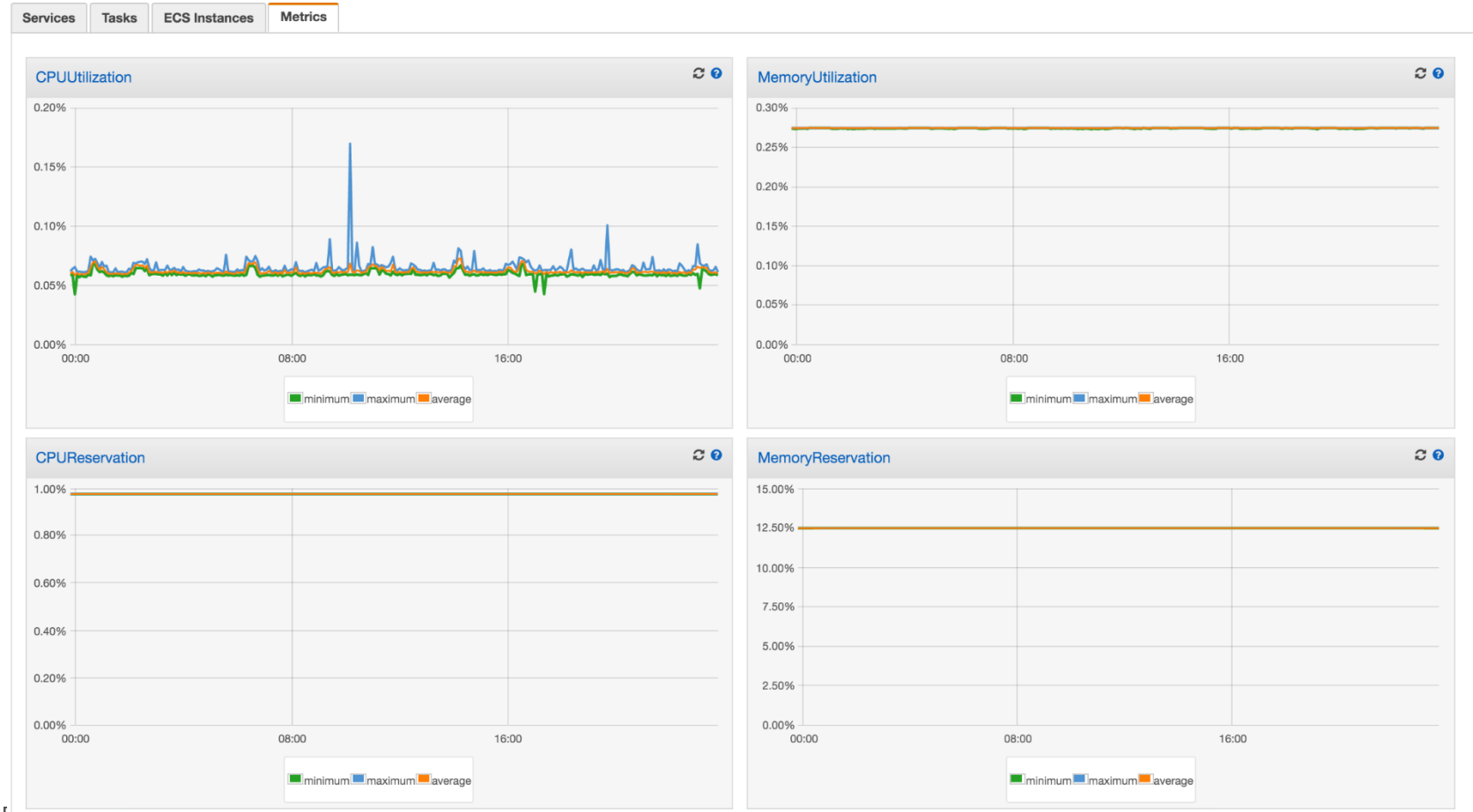
CloudWatch监控集成

CloudWatch可以保存客户机所上传的每分钟的指标采样信息，并自动保存2周的数据采样.

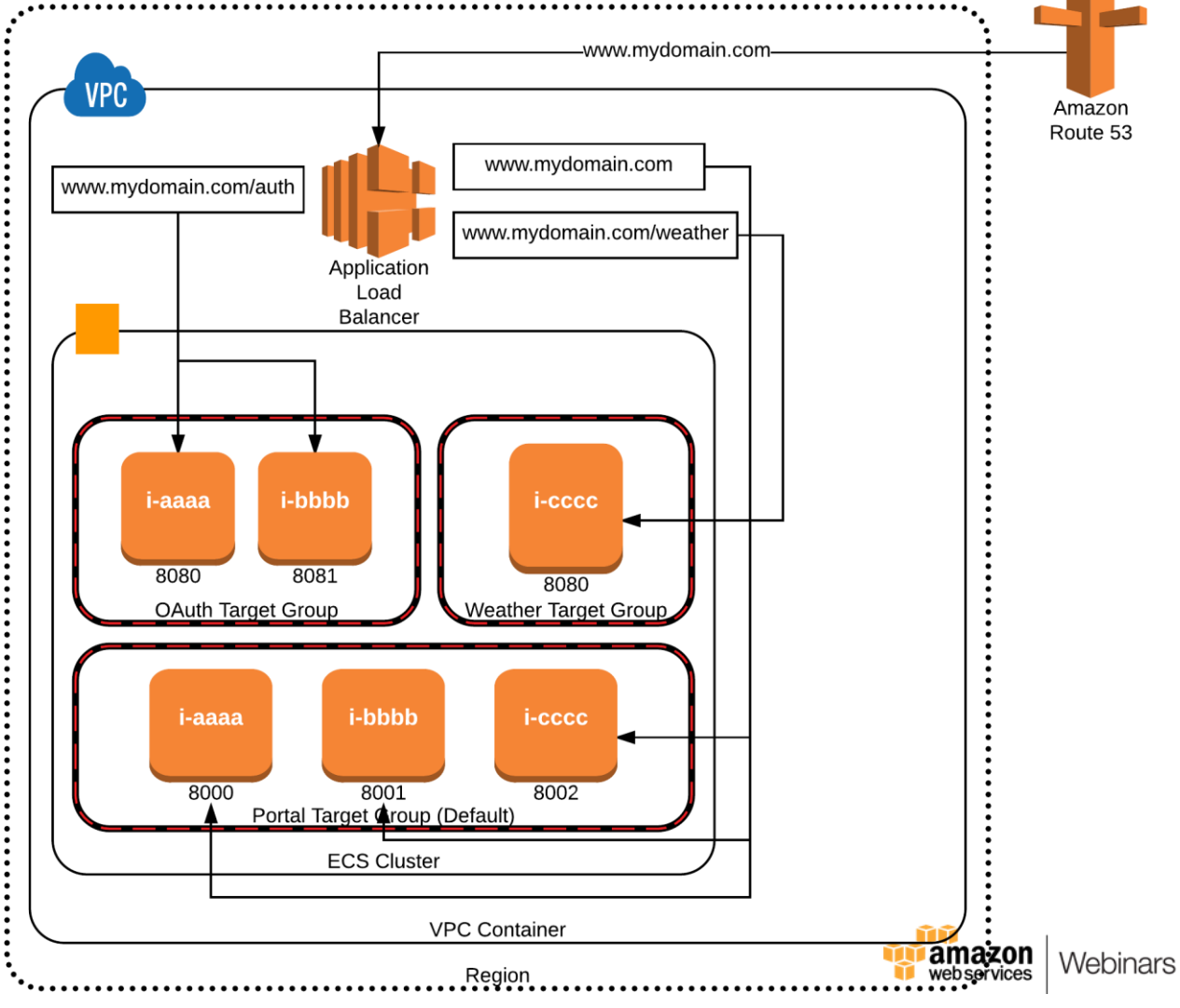
可用的指标包括: `CPUReservation`, `MemoryReservation`, `CPUUtilization`, `MemoryUtilization`

可用的维度包括: `ClusterName`, `ServiceName`

CloudWatch监控指标



利用ALB和Route53来管理集群中的服务发现情况



在ECS中部署容器的调度选择

Batch Jobs	Long-Running Apps
ECS task scheduler	ECS service scheduler
Run tasks once	Health management
Batch jobs	Scale-up and scale-down
RunTask (random)	AZ aware
StartTask (placed)	Grouped containers

容器调度: Long-Running App

利用最小资源代价的部署 *minimumHealthyPercent* = 50%, *maximumPercent* = 100%

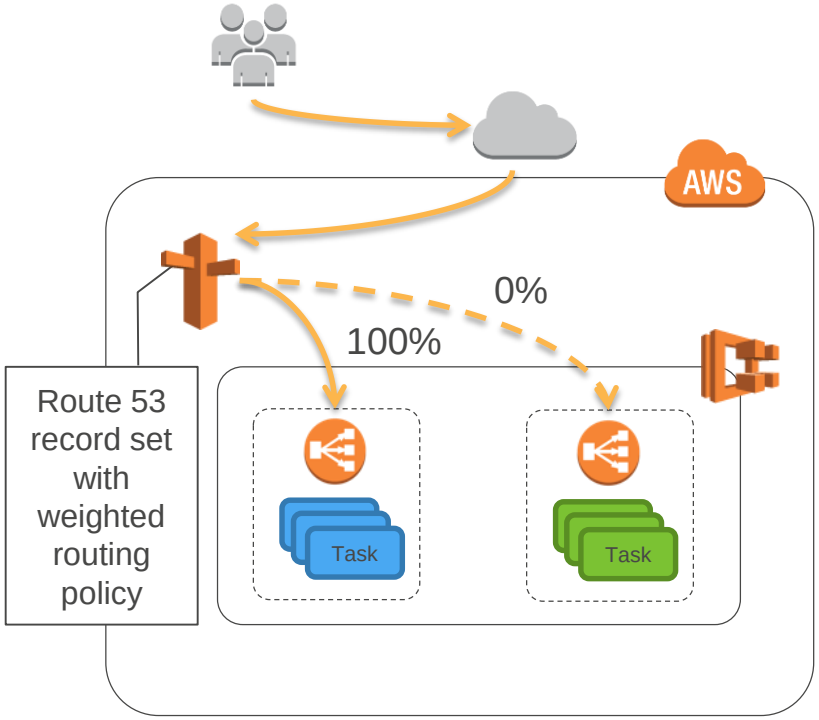
容器调度: Long-Running App

快速部署，不降低服务的容量: *minimumHealthyPercent* = 100%, *maximumPercent* = 200%

容器调度: Long-Running App

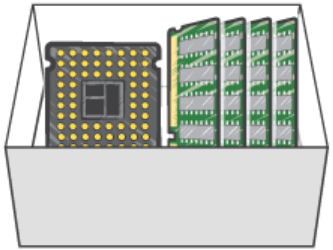
蓝绿部署

- 定义两个ECS服务
- 每个服务拥有各自的负载均衡器
- 利用Route53的权重来设置百分比，一开始是100% vs 0%
- 利用部署的蓝绿服务来切换权重比



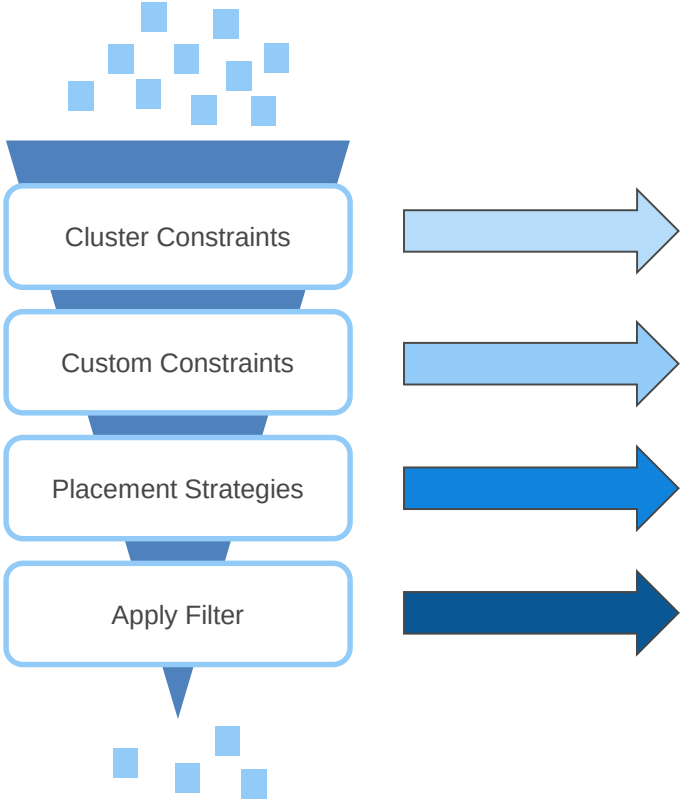
任务置放引擎
Task Placement Engine

置放约束和属性（新）



	名称	示例
✔	AMI ID	attribute:ecs.ami-id == ami-eca289fb
✔	Availability Zone	attribute:ecs.availability-zone == us-east-1a
✔	Instance Type	attribute:ecs.instance-type == t2.small
✔	Distinct Instances	type=“distinctInstance”
✔	Custom	attribute:stack == prod

任务置放的原理



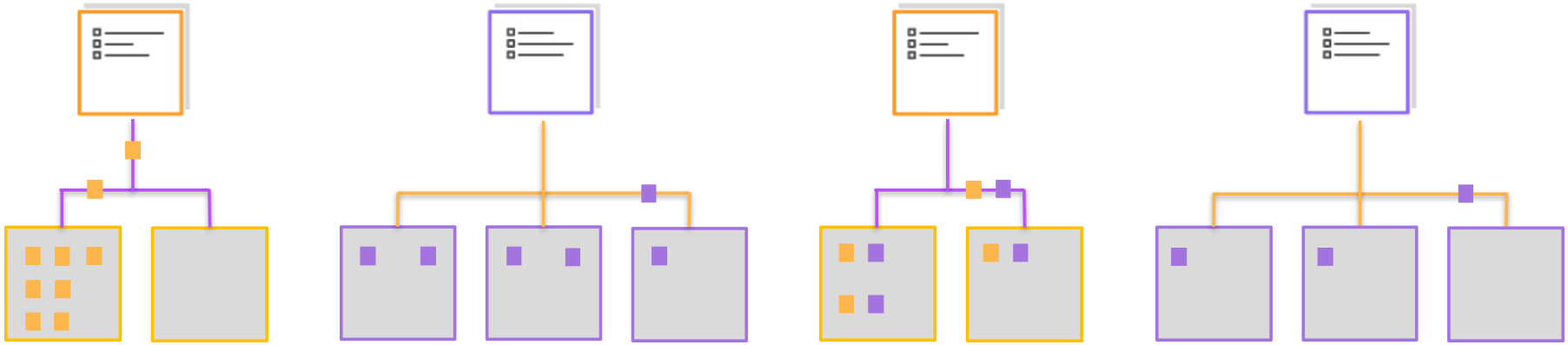
满足 CPU、内存和端口要求

按照位置、虚机实例类型、AMI或者自定义属性约束进行过滤

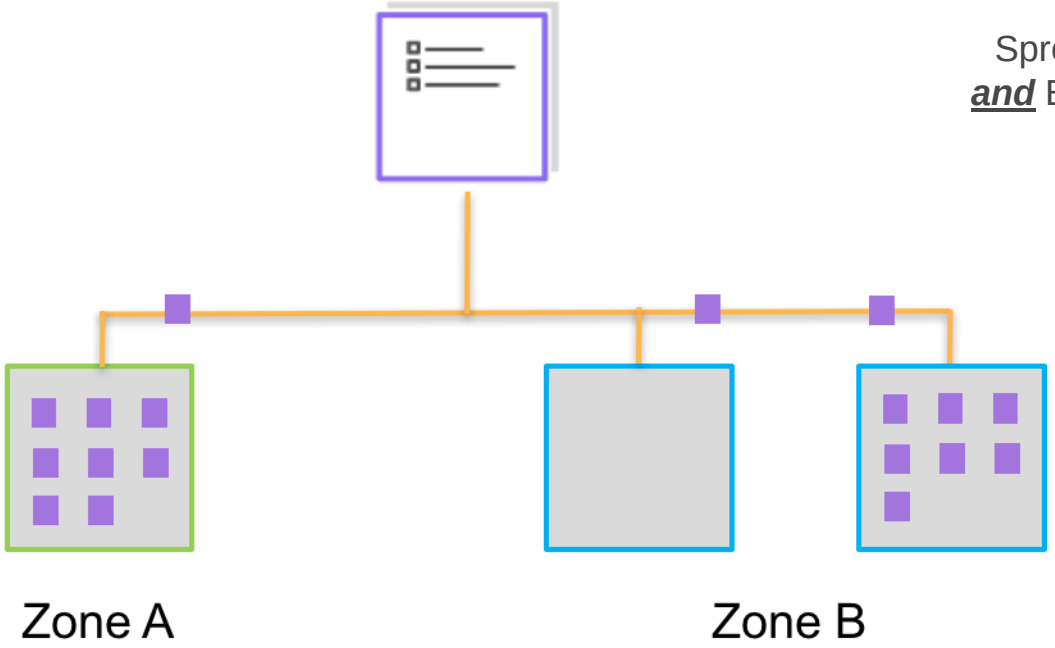
找到满足spread或binpack置放策略的虚机实例

选择最后确定的虚机实例

默认支持的置放策略

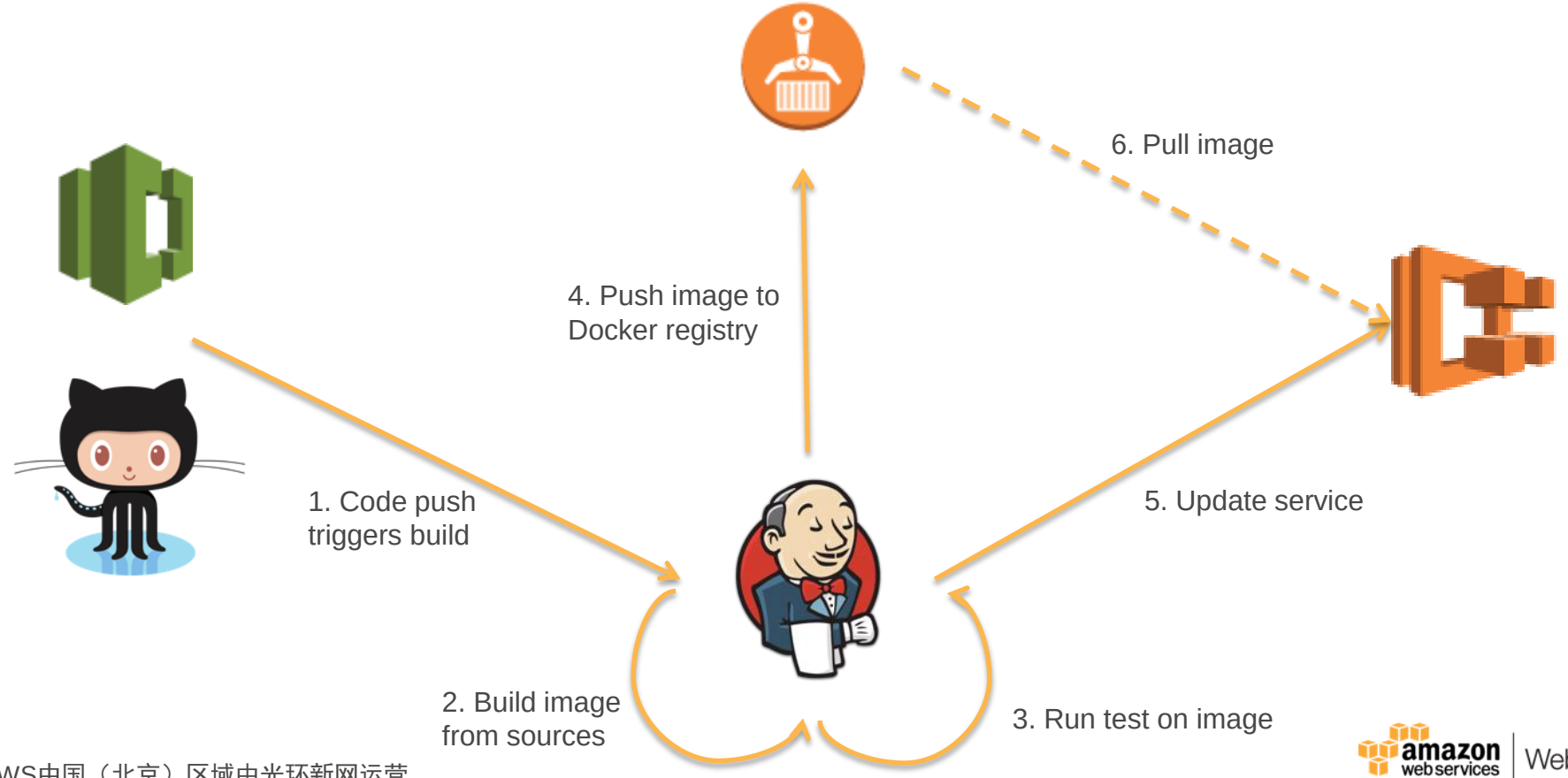


置放策略叠加



Spread tasks across Zones
and Binpack within each Zone

ECS与Jenkins的持续集成



ECS与Jenkins的持续集成

快速的部署

开发人员 – 只需简单的合并代码到master, 搞定！

Jenkins Build步骤

利用webhooks，或者监控以及Lambda来触发

利用Docker image来build并利用插件轻松集成

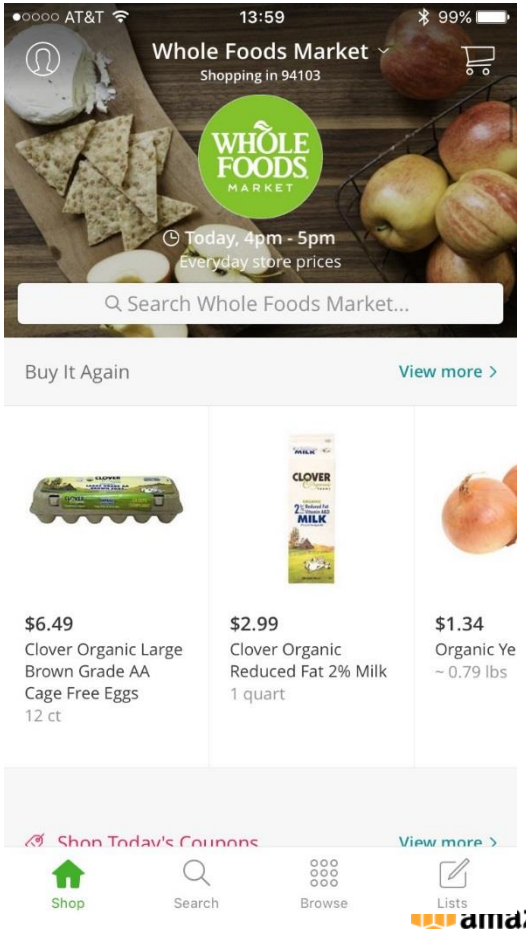
Push Docker image到ECR或者你的私有Docker Registry

利用ECS API来更新Task.

客户案例分享Instacart

Instacart

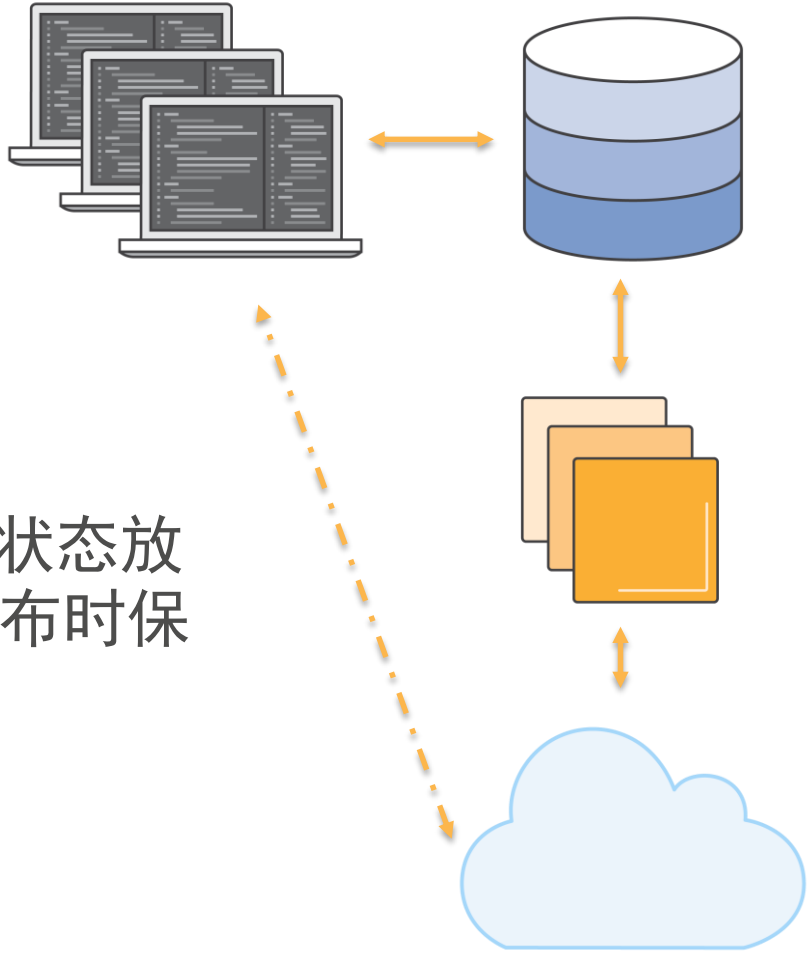
创立于 2012
当天到达的本地商店采购服务
很小的技术和运维团队



技术栈

主要采用 Ruby/Rails/JS, Python, & R
子域名划分, 各自拥有自己的 SLA
子域名又由各自的微服务组成
Monorepo交付模型
自定义的基于ECS的内部PaaS平台

Instacart's PaaS



利用RDS协调数据

在云端和本地采用同样的工具

本地开发就已经把所有需要的状态放到了RDS数据库中，随后在发布时保持设置和环境的统一.

“every minute counts”

Why not EC2?

启动相对较慢
部署和回滚不具备原子性
资源的灵活度受限于EC2所定义的实例大小
对开发者透明: 容器 vs. 实例

部署

原子性的部署通过CloudFormation
瞬间回滚
按照不同的应用类型来自定义”GC”策略
更符合需要部署的应用程序的规格
更加迅速快捷的CI/CD

部署流程

利用Jenkins的Webhooks
首先部署10%, 然后慢慢到100%
利用metrics: 5xx & latency

蓝绿部署

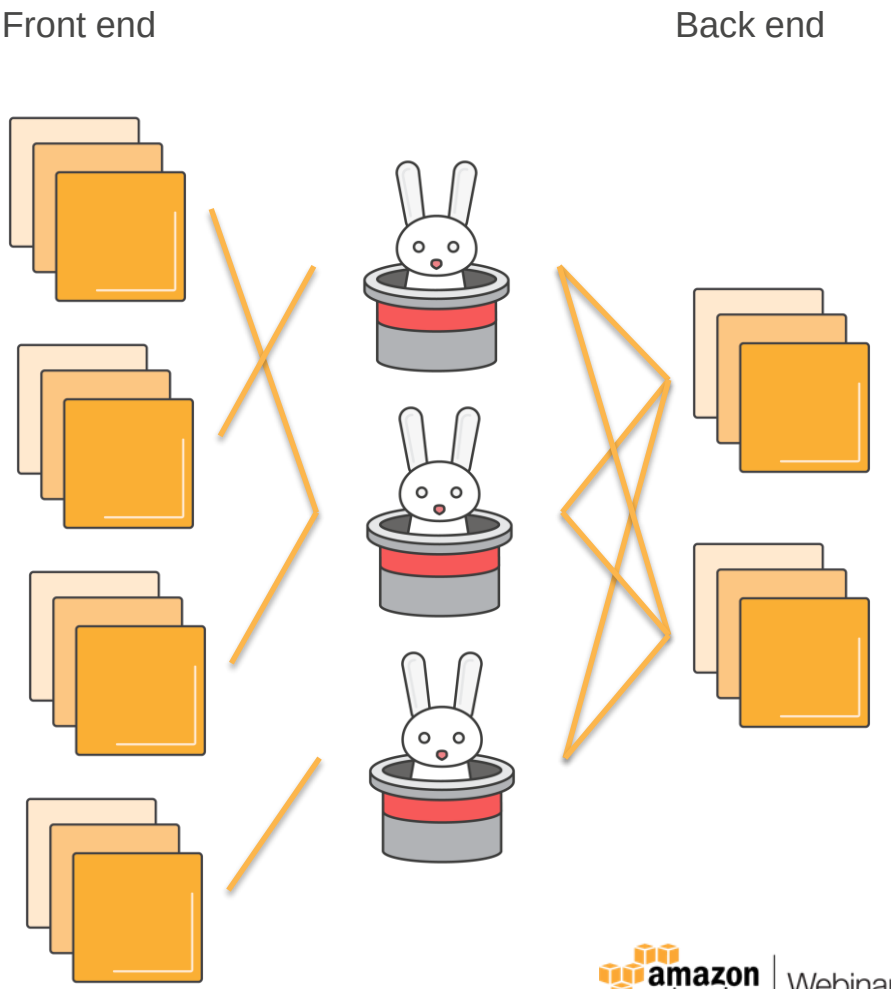
利用ALB的注册/分离来完成蓝/绿
只需要保持一个或多个旧版的运行就可以立即回滚
利用不同的端口部署蓝绿

服务发现

项目初期是通过RabbitMQ松耦合的

通过RabbitMQ的队列来实现服务发现

快速的注册和分离负载均衡器



amazon web services | Webinar

intel

谢谢！

AWS中国（北京）区域由光环新网运营

amazon web services | Webinars