酷划商业平台容器化历程

作者：Lord-X
原文链接：https://ld246.com/article/1565926249145
来源网站：链滴
许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

□rose□rose 如果您觉得我的文章对您有帮助的话，记得在GitHub上star一波哈□ose□rose

□rose□rose GitHub_awesome-it-blog □rose□rose

---

0 酷划商业平台容器化历程

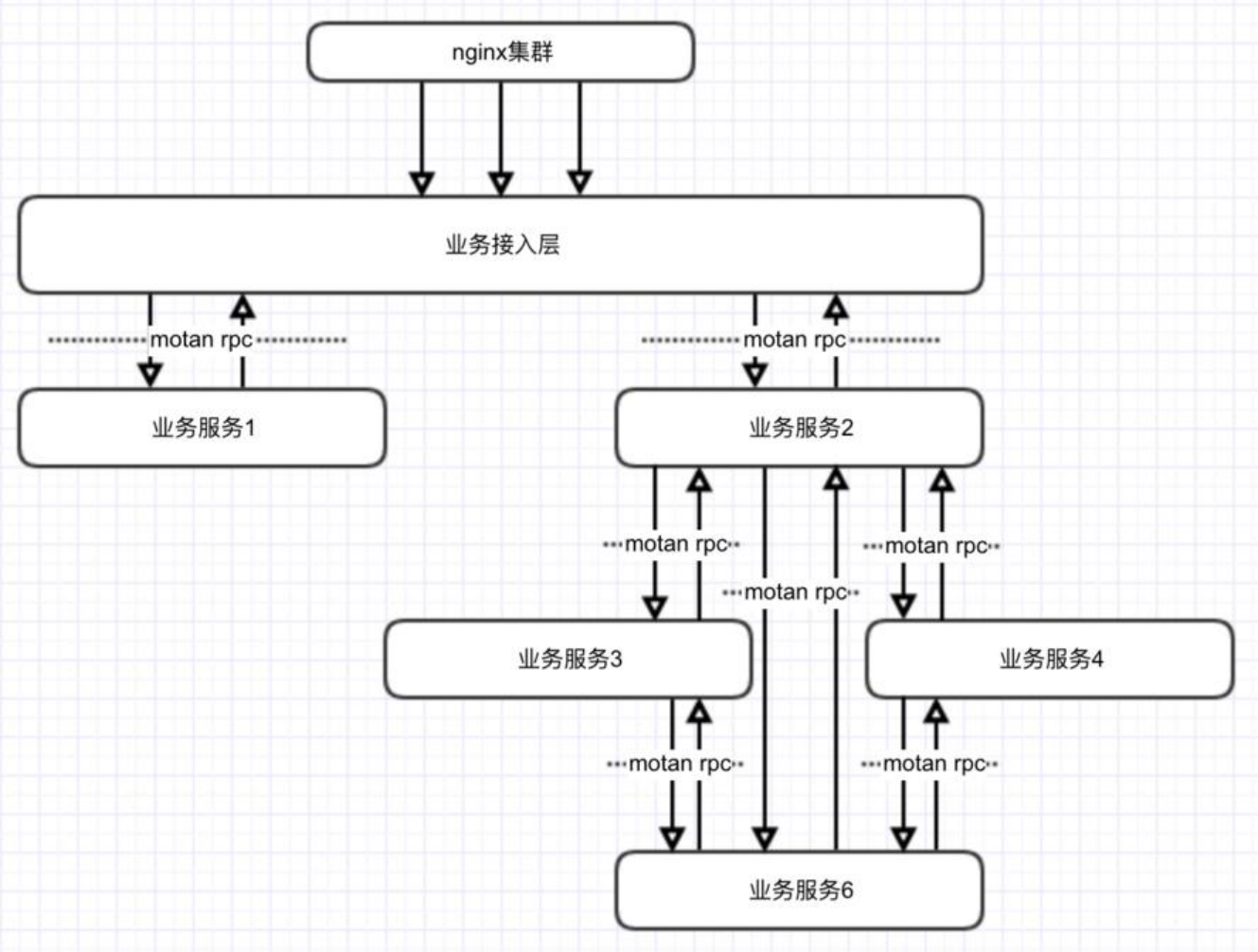
本文会先介绍酷划商业平台容器化前的主链路业务架构及技术架构，然后根据现有的架构，设计具体容器化实施方案。

PS：本文不会讨论容器化本身的东西，只讨论将应用从阿里云ECS迁移到k8s中的实施方案。

1 酷划商业平台主链路业务架构及技术架构

本次迁移过程只涉及业务服务的迁移，所以只涉及具体的业务以及服务间的RPC。不涉及持久层等其组件和中间件。

如下图所示：



由nginx做负载均衡，到接入层（web接口），下面在通过motan串联各个服务。

2 容器化目的

- 降低成本
- 提升产品迭代以及运维效率
- CI/CD，通过镜像持续交付，统一环境，降低因环境不同而导致的问题
- 规范产品迭代交付流程（主要规范测试流程）

3 容器化整体流程

- 全链路迁移预发布环境，并在预发布环境将全链路跑通。
- 全链路迁移到生产环境，通过nginx放小流量进来，观察是否有问题。
- nginx增加放量到50%，至此，ECS保留50%的量，k8s有50%的量，此时要观察是否有报错，并需对比ECS和k8s中的服务性能差异（主要是接口延迟及资源消耗）。
- nginx增加放量到90%，保持最少一周（依据服务的流量变化周期而定）的观察。
- nginx全部切到k8s中。

4 细节

4.1 预发布环境的迁移

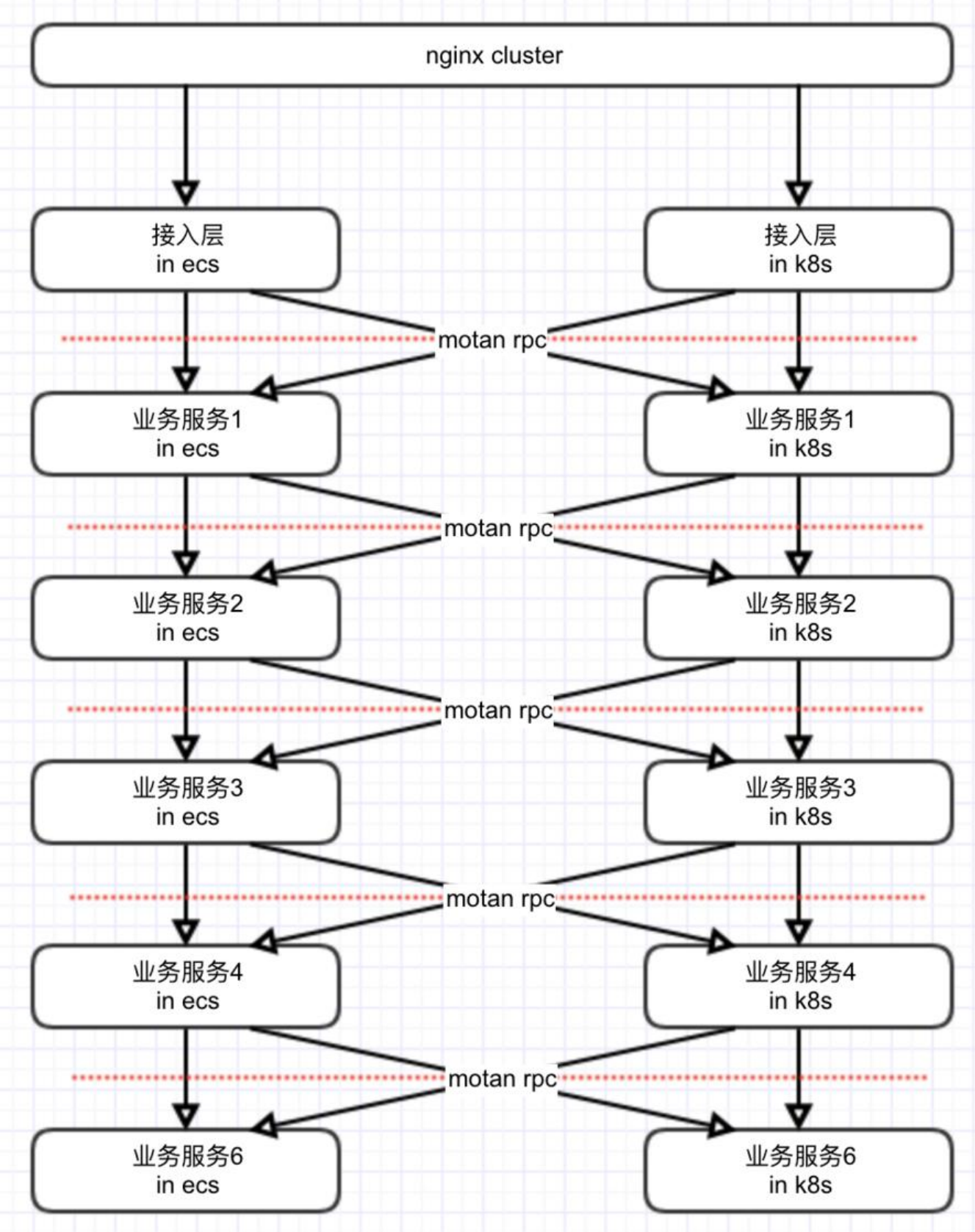
预发布环境指的是上线前，测试同学可以在这个环境上做生产环境验证。

在预发布环境下，DB、Cache、MQ、Config等跟生产环境用的是同一个，能够验证在生产环境的配下，服务是否正常。

预发布环境只有公司内部网络环境可以访问（或挂VPN），依据这个特点，我们优先动预发环境也不对线上产生影响。

预发布环境的迁移方案比较简单粗暴，直接在k8s中将全链路的预发环境部署，然后从nginx层直接转过去。如果有异常再及时切回来，保证环境的可用。

大概是下面这样的部署方式(假设把上面的架构图变成一个调用时序的纵向结构图)：



在ECS和在k8s中的服务同时注册到注册中心中，所以这里呈现出交叉调用的关系图。

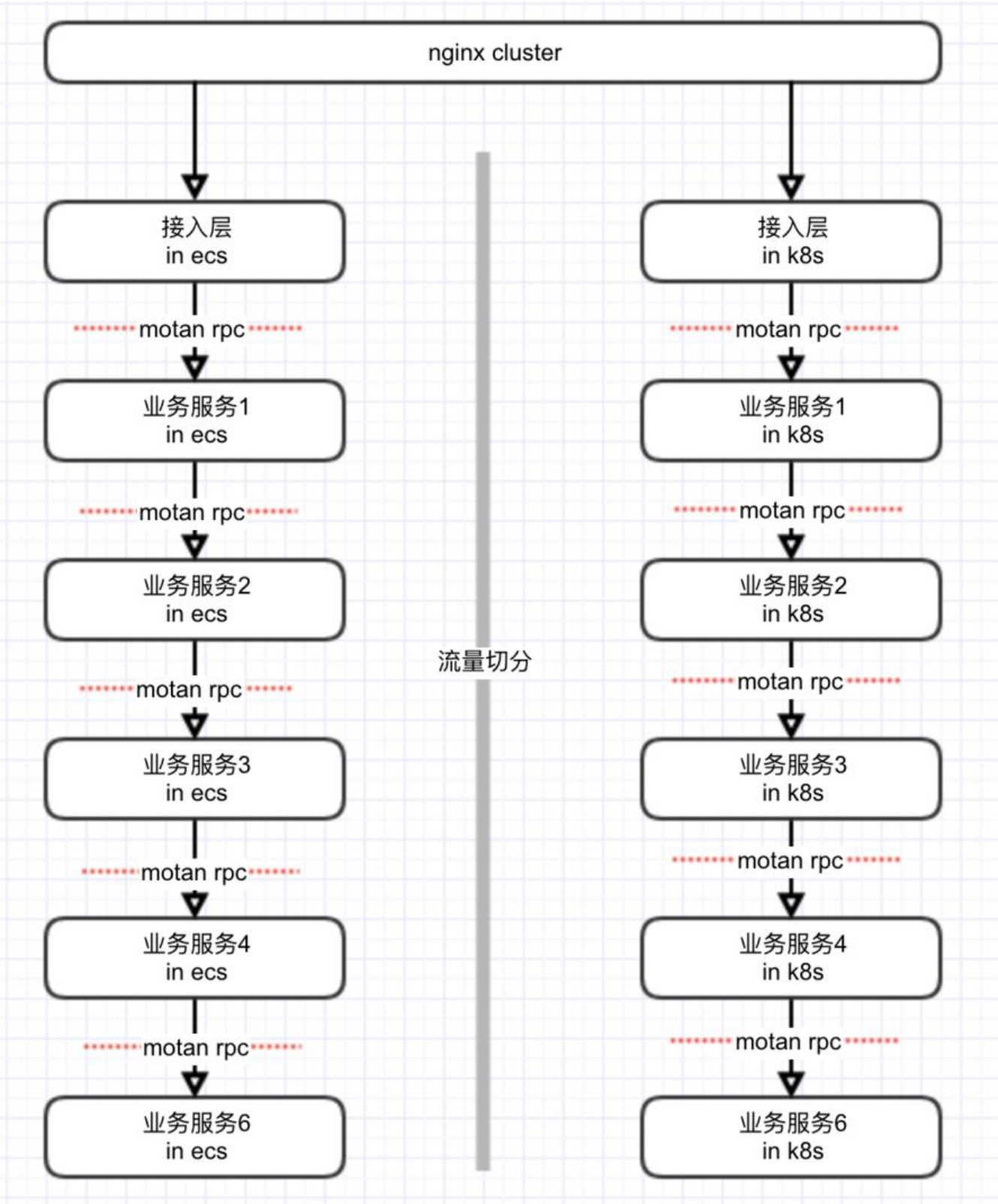
验证通过后，直接将ECS中的服务下线即可。

4.2 生产环境的迁移

生产环境与预发环境不同，不能存在交叉调用的情况。因为，交叉调用的情况下，在大流量的情况下万一有异常，不能很好的控制，另外，也不好观察ECS和k8s的性能差异。

所以，我们需要将流量完全切开。即，流量从nginx进来后，根据比例进入到ECS环境和k8s环境中。

也就是下图这个样子：



原文链接：酷划商业平台容器化历程

流量如何切分?

这需要先了解motan的服务注册发现机制。在motan中，使用group来标识一个服务，即，如果服务供方的group相同，即可注册为一个服务组，调用方使用这个group就能调用这个服务。

这个group其实是一个配置。目前这个group的配置是在Apollo中管理的。

PS：这里只讨论group，motan还有其他的相关配置，这里不讨论。

由于ecs和k8s用的注册中心是同一个，如果某一个k8s的服务上线后，就会注册到注册中心，此时ecs流量也会访问进来。

如何避免这种情况呢?

我们基于ecs和k8s的IP的特点，以及配置中心(Apollo)的灰度功能，得出一套解决方案。

核心思路： 利用Apollo的灰度功能，将group拆分为主版本配置和灰度配置，指定IP的机器走灰度配，其他机器走主配置。例如，我们现在需要将group拆分为"service-ecs"和"service-k8s"，ECS的机统一走service-ecs这个group服务发现，k8s的统一走service-k8s服务发现，这样就将两个环境的流完全拆开了。

现在的一个问题是，如何能让ecs和k8s走到指定的配置中（主版本or灰度）。

首先，Apollo可以指定多个机器IP来走灰度配置，但这些IP必须是固定不能变的。然后考虑k8s和ecs IP特点，ECS每次重启IP都不会改变，但k8s的pod每次重启IP都会变化，所以我们必须将ecs的IP配到灰度中，然后灰度的group配置为service-ecs，主版本的group配置为service-k8s，这样一来，容中的IP不管怎么变化，都会走到主版本配置的service-k8s中，而ECS也肯定会走到service-ecs中了。

下面来看下Apollo中具体的配置方式：

假设我们要提供ad-common这个服务，ecs的服务要走到ad-common这个group中，k8s要走到ad-ommon-k8s这个group中。

Apollo中主版本的配置：

下面是灰度版本配置：

还有灰度版本的指定IP配置：

最后，调用方的配置，同理，有主版本和灰度两个版本的配置。灰度版本的配置为ad-common，主本的配置为ad-common-k8s，这样一来k8s中的调用方就肯定能走到ad-common-k8s中，而ecs的用方，由于命中了灰度，所以肯定会走到ad-common中。

至此完成流量的切分。然后要做的就是逐渐从nginx控制放量，观察bug以及性能问题等。

4.3 全量切生产环境

基于上述方案，切全量是否可以直接把nginx流量全部切到k8s里就好了？答案是不行的。

为啥呢？因为k8s中的motan的group用的是ad-common-k8s，有很多服务可能还被其他业务线调，而其他业务线调用的依然是ad-common，如果直接全量切到k8s，并将ecs全部下线，其他业务线挂了。

所以我们现在要想办法将k8s中的服务的group改回ad-common。

咋改呢？首先将流量全部切回ECS，保证k8s中没有流量。

然后自底向上开始切（基于刚才那个从上到下的调用时序图）。

操作配置：将灰度版本删除，将主版本的group改为ad-common。

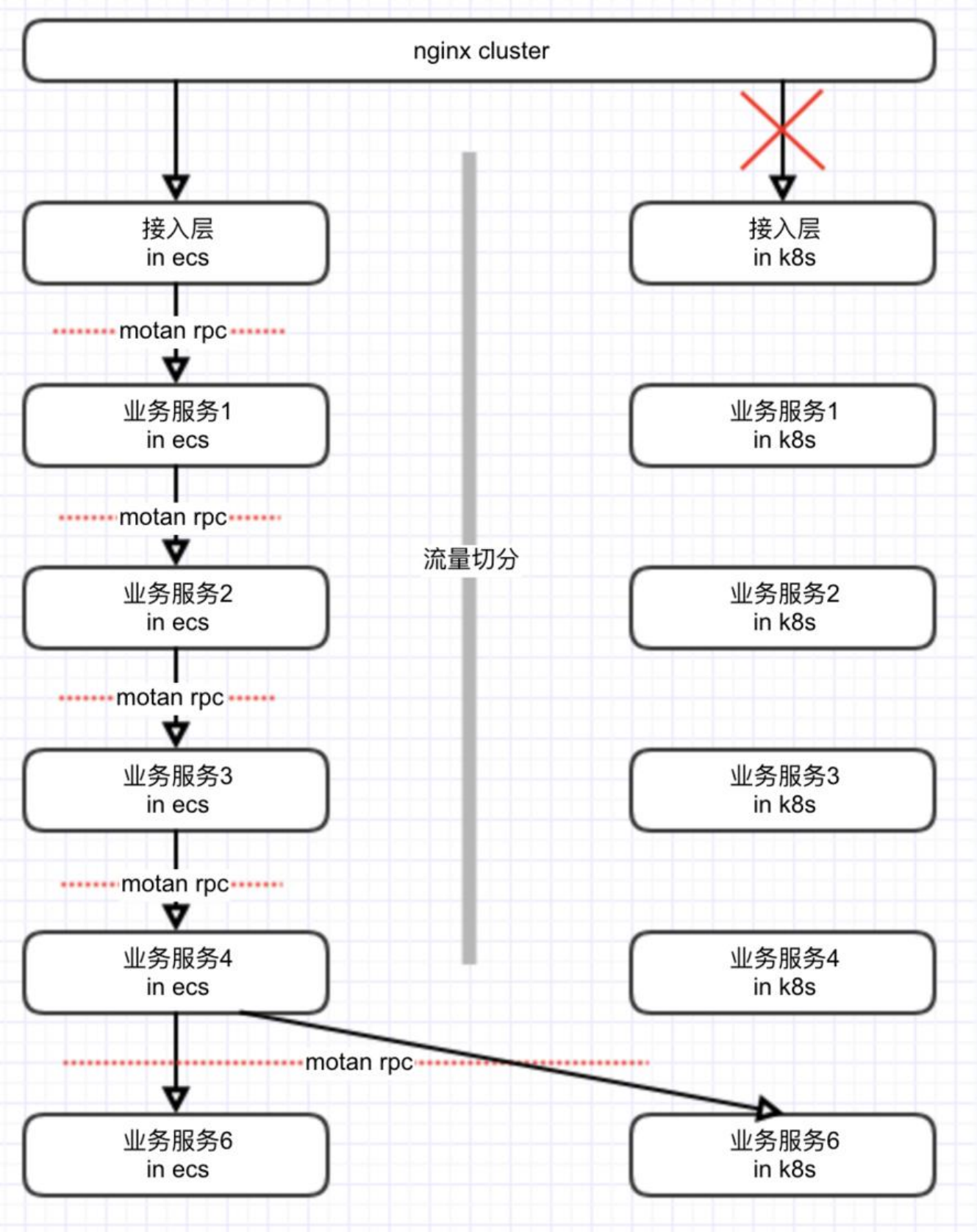
目前正在运行的ECS，注册的group是ad-common，灰度删除后，即使重启，也会命中到主版本的ad common上，保持不变，所以ECS是没问题的。

k8s中，重启后，同样走到主版本的ad-common上。

这样一来，ECS和k8s就同时在ad-common上提供服务了。

然后按照这个方案，继续向上操作即可。

切完ad-common后，服务状态如图所示：



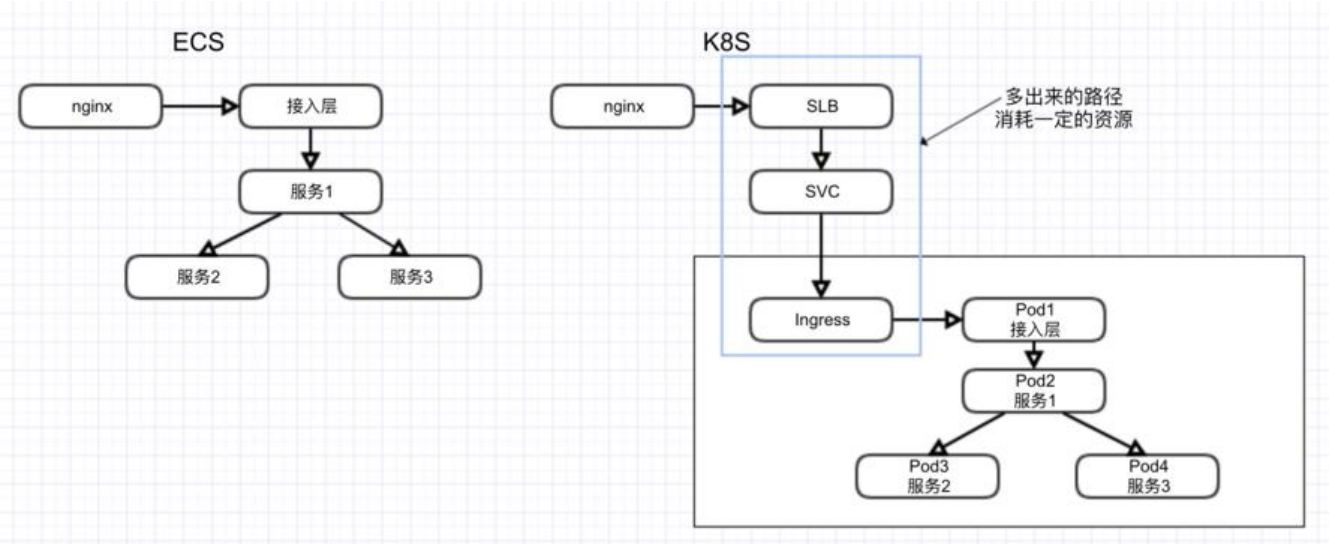
总结

首先，在生产中，小流量接入的测试是很有必要的。在这个阶段，我们遇到并解决了一些在ECS中遇

到的问题。

其次，Ecs和k8s等分流量的测试也是很有必要的，这可以较好的对比两个环境的性能差异。本次容器过程中遇到了如下一些问题：

- 所有接口响应延迟整体变慢。
  - 经验证，与阿里云可用区有关，同可用区下的服务之间访问延迟要远远优于跨可用区情况。
  - 其次，与流量进入k8s和ecs走过的路径不同有关。下面是ecs和k8s流量路径对比：



- 服务启动瞬间，接口响应延迟飙升，最高可达10S。
  - 首先定位到是阿里的Sentinal影响，此服务在应用启动时同步获取配置，而且响应比较慢，导致来的流量全部hang住。

解决方案是，等Sentinal初始化完成，再接入流量。

  - 预热：包括各种连接池的预热（Redis、Motan、MQ、DB、Tomcat等），以及代码预热（JIT。经验证，对于我们这种重度依赖Redis的服务来说，预热Redis连接池收益是很大的。