ALIBABA CLOUD

阿里云

容器服务Kubernetes版

最佳实践

文档版本：20220701

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.集群 10
1.1. ECS选型 10
1.2. 高可靠推荐配置 11
1.3. Kubernetes集群网络规划 15
1.4. 混合云部署Kubernetes集群 18
1.5. 提升ACK专有集群的etcd存储容量上限 19
1.6. ACK容器平台集群安全控制 21
1.7. ACK集群神龙资源错峰利用 21
2.节点池 23
2.1. 抢占式实例节点池最佳实践 23
3.网络 28
3.1. 使用Ambassador Edge Stack管理Ingress资源 28
3.2. 通过Ambassador暴露应用API 31
3.3. 优化大规模Tewary集群NetworkPolicy的扩展性 35
3.4. 搭建高性能网络ACK集群 38
3.5. 金融行业从经典网络向VPC容器化改造 39
3.6. DNS最佳实践 40
3.7. Nginx Ingress最佳实践 49
4.安全 54
4.1. 安全概述 54
4.2. 身份认证和访问控制 56
4.3. Pod安全 59
4.4. Runtime安全 64
4.5. 供应链安全 65
4.6. 数据加密和密钥管理 68
4.7. 网络安全 69

4.8. 日志和审计 73
4.9. 多租户安全 75
4.10. 主机安全 78
5.存储 80
5.1. 实现StatefulSet持久化存储的最佳实践-CSI 80
5.2. 实现StatefulSet持久化存储的最佳实践-Flexvolume 86
5.3. 本地存储卷的初始化、挂载与自动扩容 92
5.4. 通过CNFS自动收集异常退出的JVM转储文件 103
6.运维 109
6.1. ACK对接外部LDAP验证源 109
6.2. 云原生ACK应用管理 116
7.监控 118
7.1. 使用Prometheus Client监控应用 118
8.弹性伸缩 124
8.1. 使用ack-autoscaling-placeholder实现容器秒级伸缩 124
8.2. 弹性优化之自定义镜像 126
8.3. 多可用区实现同时快速弹性扩容 135
8.4. 基于阿里云Prometheus指标的容器水平伸缩 142
8.5. 通过Nginx Ingress对多个应用进行HPA 153
9.服务网格 160
9.1. 通过ASM管理ACK虚拟节点上的ECI Pod应用 160
9.2. 通过ASM管理外部注册Kubernetes集群应用 161
9.3. 部署应用示例到包含同VPC多集群的ASM实例 163
9.4. 通过ASM实现跨地域容灾和流量负载均衡 173
9.5. 使用ASM指标实现工作负载的自动弹性伸缩 179
9.6. 通过入口网关访问网格内gRPC服务 186
10.架构方案 194
10.1. 容器跨可用区高可用 194

10.2. 基于ECI的ACK集群高弹性架构 195
10.3. ACK集群低成本高弹性架构 197
10.4. 混合云部署Kubernetes集群 197
11.DevOps 199
11.1. 在Eclipse中安装Cloud Toolkit插件并部署应用到容器服务Kubern... 199
11.2. 快速搭建Jenkins环境并完成流水线作业 208
11.3. 使用GitLab CI运行GitLab Runner并执行Pipeline 222
11.4. 容器应用DevOps for ACK集群 232
11.5. ASK弹性低成本CI/CD 232
11.6. 使用Bamboo部署Bamboo Agent并执行构建任务 233
11.7. 基于K8s多集群隔离环境下的DevOps实现 246
12.自建Kubernetes迁移ACK 248
12.1. 上云须知 248
12.2. Kubernetes迁移方案概述 249
12.3. 使用自定义镜像创建ACK集群 252
12.4. 自建Kubernetes迁移至ACK弹性裸金属集群 259
12.5. 自建Kubernetes集群迁移镜像及应用至ACK 260
12.6. 容器Swarm集群向ACK集群灰度迁移 261
12.7. 源服务器迁移至容器镜像 262
12.8. 容器镜像迁移 266
12.8.1. 通过image-syncer工具迁移容器镜像 266
12.8.2. 从自建Harbor同步镜像到ACR默认实例版 268
12.8.3. 从自建Harbor同步镜像到ACR企业版 271
12.9. Kubernetes应用迁移 272
13.Swarm迁移Kubernetes 383
13.1. 容器服务swarm集群与Kubernetes集群的主要功能比对 383
13.1.1. 概述 383
13.1.2. 概念比对 383

13.1.3. 基本配置比对（使用镜像创建应用） 386
13.1.4. 网络配置比对（使用镜像创建应用） 387
13.1.5. 数据卷及环境变量配置比对（使用镜像创建应用） 393
13.1.6. 容器配置及标签比对（使用镜像创建应用） 394
13.1.7. 健康检查及自动伸缩比对（使用镜像创建应用） 395
13.1.8. 使用YAML文件创建应用比对 397
13.1.9. 网络比对 402
13.1.10. 日志及监控比对 402
13.1.11. 应用访问比对 403
13.2. 迁移方案概述 406
13.3. 标准化Swarm集群 407
13.4. 迁移集群配置 413
13.5. 迁移应用配置 420
13.5.1. 迁移应用配置概述 420
13.5.2. 准备迁移环境 421
13.5.3. 预处理Swarm编排文件 423
13.5.4. 转换Swarm编排文件 424
13.5.5. 部署 Kubernetes 资源文件 426
13.5.6. 手动迁移应用配置 428
13.5.7. 应用启动调试 428
13.5.8. 迁移应用日志配置 431
13.5.9. 应用配置迁移异常解决方案 432
13.6. 附录：标签映射 436
13.6.1. 配置类标签 436
13.6.2. 发布类标签 442
13.6.3. 网络配置类标签 456
13.6.4. 日志配置类标签 468
13.7. 附录：标签配置样例 471

13.8. 应用回归测试 486
13.9. SLB灰度引流 488
13.9.1. SLB灰度引流概述 488
13.9.2. 创建Kubernetes NodePort服务 490
13.9.3. 验证Kubernetes NodePort服务 491
13.9.4. 修改Swarm SLB配置 494
13.10. 客户端流量切换 497
13.11. 下线Swarm集群 498
14.基因计算 500
14.1. 基因计算工作流 500
15.微服务 501
15.1. MSE微服务治理最佳实践 501
15.2. 微服务应用问题定位及故障演练 501
15.3. 微服务架构日志采集运维管理 502
15.4. 微服务引擎的线上流量治理 503
15.5. 异地双活场景下的数据双向同步 504
15.6. 微服务分布式事务 505
15.7. 基于容器快速搭建小程序后端服务 506
16.Serverless 508
16.1. 基于Knative的消息流事件处理 508
16.2. 基于ECI和FaaS构建游戏战斗结算服务 508
16.3. 基于ECI、RI和EIP降低成本方案 509
16.4. 混合云IDC自有K8s弹性使用ECI 510
17.AI 512
17.1. ACK集群实现GPU成本优化 512
17.2. ACK集群实现基于弹性计算的AI推理 513
17.3. ACK集群实现弹性裸金属AI训练 513
17.4. ACK集群实现GPU AI模型训练 514

18.云原生架构下日志服务数据预处理 516

1.集群

1.1. ECS选型

本文介绍构建Kubernetes集群时该如何选择ECS类型以及选型的注意事项。

集群规划

目前在创建Kubernetes集群时，存在着使用很多小规格ECS的现象，这样做有以下弊端：

- 小规格Woker ECS的网络资源受限。
- 如果一个容器基本可以占用一个小规格ECS，此ECS的剩余资源就无法利用（构建新的容器或者是恢复失败的容器），在小规格ECS较多的情况下，存在资源浪费。

使用大规格ECS的优势：

- 网络带宽大，对于大带宽类的应用，资源利用率高。
- 容器在一台ECS内建立通信的比例增大，减少网络传输。
- 拉取镜像的效率更高。因为镜像只需要拉取一次就可以被多个容器使用。而对于小规格的ECS拉取镜像的次数就会增多，若需要联动ECS伸缩集群，则需要花费更多的时间，反而达不到立即响应的目的。

选择Master节点规格

通过容器服务创建的Kubernetes集群，Master节点上运行着etcd、kube-apiserver、kube-controller等核心组件，对于Kubernetes集群的稳定性有着至关重要的影响，对于生产环境的集群，必须慎重选择Master规格。Master规格跟集群规模有关，集群规模越大，所需要的Master规格也越高。

> **说明** 您可从多个角度衡量集群规模，例如节点数量、Pod数量、部署频率、访问量。这里简单的认为集群规模就是集群里的节点数量。

对于常见的集群规模，可以参见如下的方式选择Master节点的规格（对于测试环境，规格可以小一些。下面的选择能尽量保证Master负载维持在一个较低的水平上）。

节点规模	Master规格
1~5个节点	4核8 GB（不建议2核4 GB）
6~20个节点	4核16 GB
21~100个节点	8核32 GB
100~200个节点	16核64 GB

选择Worker节点规格

- ECS规格要求：CPU大于等于4核，且内存大于等于8 GiB。
- 确定整个集群的日常使用的总核数以及可用度的容忍度。
  例如：集群总的核数有160核，可以容忍10%的错误。那么最小选择10台16核ECS，并且高峰运行的负荷不要超过160*90%=144核。如果容忍度是20%，那么最小选择5台32核ECS，并且高峰运行的负荷不要超过160*80%=128核。这样就算有一台ECS出现故障，剩余ECS仍可以支持现有业务正常运行。
- 确定CPU：Memory比例。对于使用内存比较多的应用例如Java类应用，建议考虑使用1:8的机型。

选用裸金属神龙服务器

以下两种场景，建议选用裸金属神龙服务器：

- 集群日常规模能够达到1000核。一台神龙服务器至少96核，这样可以通过10台或11台神龙服务器即可构建一个集群。
- 快速扩大较多容器。例如：电商类大促，为应对流量尖峰，可以考虑使用神龙服务来作为新增节点，这样增加一台神龙服务器就可以支持很多个容器运行。

使用神龙服务器构建集群，存在以下优势：

- 超强网络： 配备RDMA（Remote Direct Memory Access）技术。通过Terway容器网络，充分发挥硬件性能，跨宿主机容器带宽超过9 GB。
- 计算性能零抖动：自研芯片取代Hypervisor，无虚拟化开销，无资源抢占。
- 安全：物理级别加密，支持Intel SGX加密，可信计算环境，支持区块链等应用。

1.2. 高可靠推荐配置

为了保证应用可以稳定可靠的运行在Kubernetes里，本文介绍构建Kubernetes集群时的推荐配置。

磁盘类型及大小

磁盘类型

- 推荐选择SSD盘。
- 对于Worker节点，创建集群时推荐选择**挂载数据盘**。该数据盘用于供*/var/lib/docker*存放本地镜像，避免后续镜像过多导致根磁盘容量不够的问题。在运行一段时间后，本地可能存在很多无用的镜像，快速解决该问题的方法是先下线这台机器，重新构建数据盘后再上线。

磁盘大小

Kubernetes节点需要的磁盘空间也不小，Docker镜像、系统日志、应用日志都保存在磁盘上。创建Kubernetes集群的时候，要考虑每个节点上要部署的Pod数量，每个Pod的日志大小、镜像大小、临时数据，再加上一些系统预留的值。

Kubernetes集群中，ECS操作系统占用3G左右的磁盘空间，建议预留ECS操作系统8G的空间。剩余的磁盘空间由Kubernetes资源对象使用。

是否立即构建Worker节点

创建集群时，**节点类型**若选择：

- **按量付费**，可在创建集群时，构建Worker节点。
- **包年包月**，创建集群时，可先不构建Worker节点，根据后续需求，单独购买ECS添加进集群。

网络选择

- 如果需要连接外部的一些服务，如RDS等，则需要考虑复用原有的VPC，而不是创建一个新的VPC。因为VPC间是隔离的，您可以通过创建一个新的交换机，把运行Kubernetes的机器都放在这个交换机下，从而便于管理。
- 在Kubernetes集群创建时提供两种网络插件：Terway和Flannel，可参见Terway与Flannel对比。
- Pod网络CIDR不能设置太小，如果太小，可支持的节点数量就会受限。这个值的设置需要与**高级选项**中的**节点Pod数量**综合考虑。例如：Pod网络CIDR的网段是/16，那么就有256*256个地址，如果每个节点Pod数量是128，则最多可以支持512个节点。

使用多可用区

阿里云支持多Region（地域），每个Region下又有不同的可用区。可用区是指在同一地域内，电力和网络互相独立的物理区域。多可用区能够实现跨区域的容灾能力。同时也会带来额外的网络延时。

声明每个Pod的resource

在使用Kubernetes集群时，经常会遇到：在一个节点上调度了太多的Pod，导致节点负载太高，没法正常对外提供服务的问题。

为避免上述问题，在Kubernetes中部署Pod时，您可以指定这个Pod需要Request及Limit的资源，Kubernetes在部署这个Pod的时候，就会根据Pod的需求找一个具有充足空闲资源的节点部署这个Pod。下面的例子中，声明Nginx这个Pod需要1核CPU，1024M的内存，运行中实际使用不能超过2核CPU和4096M内存。

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - name: nginx
    image: nginx
    resources: # 资源声明
      requests:
        memory: "1024Mi"
        cpu: "1000m"
      limits:
        memory: "4096Mi"
        cpu: "2000m"
```

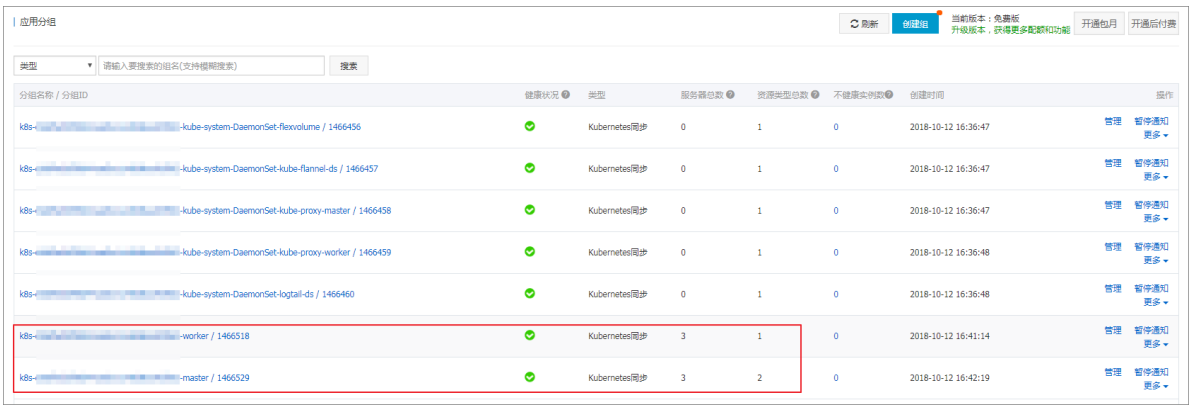
Kubernetes采用静态资源调度方式，对于每个节点上的剩余资源，它是这样计算的：`节点剩余资源=节点总资源-已经分配出去的资源`，并不是实际使用的资源。如果您自己手动运行一个很耗资源的程序，Kubernetes并不能感知到。

另外所有Pod上都要声明resources。对于没有声明resources的Pod，它被调度到某个节点后，Kubernetes也不会在对应节点上扣掉这个Pod使用的资源。可能会导致节点上调度过去太多的Pod。

日常运维

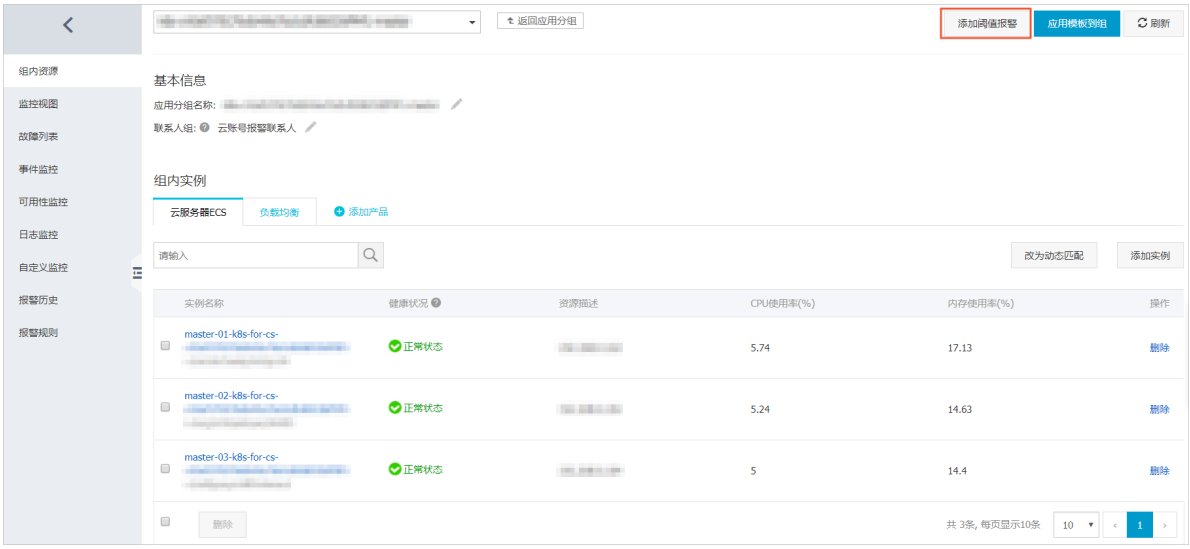
- 日志
  创建集群时，请勾选**日志服务**。
- 监控
  阿里云容器服务与云监控进行集成，通过配置节点监控，提供实时的监控服务。通过添加监控告警规则，节点上的资源使用量很高的时候，可快速定位问题。

通过容器服务创建Kubernetes集群时，会自动在云监控创建两个应用分组：一个对应Master节点，一个对应Worker节点。我们可以在这两个组下面添加一些报警规则，对组里所有的机器生效。后续加入的节点，也会自动出现在组里，不用单独再去配置报警规则。



主要配置ECS资源的报警规则就可以了。

> **说明**
> 对于ECS的监控，日常运维请设置cpu，memory，**磁盘**等的报警规则。且尽量将*/var/lib/docker*放在一个独立的盘上。



启动时等待下游服务，不要直接退出

有些应用可能会有一些外部依赖，例如需要从数据库（DB）读取数据或者依赖另外一个服务的接口。应用启动的时候，外部依赖未必都能满足。手工运维的时候，通常采用依赖不满足立即退出的方式，也就是所谓的failfast，但是在Kubernetes中，这种策略不再适用。原因在于Kubernetes中多数运维操作都是自动的，不需要人工介入，例如部署应用，您不用自己选择节点，再到节点上启动应用，应用fail，也不用手动重启，Kubernetes会自动重启应用。负载增高，还可以通过HPA自动扩容。

针对启动时依赖不满足这个场景，假设有两个应用A和B，A依赖B，刚好运行在同一个节点上。这个节点因为某些原因重启了，重启之后，A首先启动，这个时候B还没启动，对A来说就是依赖不满足。如果A还是按照传统的方式直接退出，当B启动之后，A也不会再启动，必须人工介入处理才行。

Kubernetes的最好的做法是启动时检查依赖，如果不满足，轮询等待，而不是直接退出。可以通过 Init Container完成这个功能。

配置restart policy

Pod运行过程中进程退出是个很常见的问题，无论是代码里的一个bug，还是占用内存太多，都会导致应用进程退出，Pod退出。您可在Pod上配置restartPolicy，就能实现Pod挂掉之后自动启动。

```
apiVersion: v1
kind: Pod
metadata:
  name: tomcat
spec:
  containers:
  - name: tomcat
    image: tomcat
    restartPolicy: OnFailure #
```

restartPolicy有三个可选值：

- *Always*： 总是自动重启。
- *OnFailure*：异常退出才自动重启 （进程退出状态非0）。
- *Never*：从不重启。

配置Liveness Probe和Readiness Probe

Pod处于Running状态和Pod能正常提供服务是完全不同的概念，一个Running状态的Pod，里面的进程可能发生了死锁而无法提供服务。但是因为Pod还是Running的，Kubernetes也不会自动重启这个Pod。所以我们要在所有Pod上配置Liveness Probe，探测Pod是否真的存活，是否还能提供服务。如果Liveness Probe发现了问题，Kubernetes会重启Pod。

Readiness Probe用于探测Pod是不是可以对外提供服务。应用启动过程中需要一些时间完成初始化，在这个过程中是没法对外提供服务的，通过Readiness Probe，可以告诉Ingress或者Service能不能把流量转发到这个Pod上。当Pod出现问题的时候，Readiness Probe能避免新流量继续转发到这个Pod。

```
apiVersion: v1
kind: Pod
metadata:
  name: tomcat
spec:
  containers:
  - name: tomcat
    image: tomcat
    livenessProbe:
      httpGet:
        path: /index.jsp
        port: 8080
      initialDelaySeconds: 3
      periodSeconds: 3
    readinessProbe:
      httpGet:
        path: /index.jsp
        port: 8080
```

每个进程一个容器

很多刚刚接触容器的人喜欢按照旧习惯把容器当作虚拟机（VM）使用，在一个容器里放多个进程：监控进程、日志进程、sshd进程、甚至整个Systemd。这样操作存在两个问题：

- 判断Pod整体的资源占用会变复杂，不方便实施前面提到resource limit。

- 容器内只有一个进程的情况，进程挂了，外面的容器引擎可以清楚的感知到，然后重启容器。如果容器内有多个进程，某个进程挂了，容器未必受影响，外部的容器引擎感知不到容器内有进程退出，也不会对容器做任何操作，但是实际上容器已经不能正常工作了。

如果有几个进程需要协同工作，在Kubernetes里也可以实现，例如：nginx和php-fpm，通过Unix domain socket通信，我们可以用一个包含两个容器的Pod，unix socket放在两个容器的共享volume中。

确保不存在SPOF（Single Point of Failure）

如果应用只有一个实例，当实例失败的时候，虽然Kubernetes能够重启实例，但是中间不可避免地存在一段时间的不可用。甚至更新应用，发布一个新版本的时候，也会出现这种情况。在Kubernetes里，尽量避免直接使用Pod，尽可能使用Deployment/StatefulSet，并且让应用的Pod在两个以上。

为节点池配置部署集

部署集是控制ECS实例分布的策略，该策略将ECS实例分散部署在不同的物理服务器上，提升业务的高可用性和底层容灾能力。通过为节点池配置部署集，能够保证节点池弹出的ECS实例不会分布于同一物理机上，并通过亲和性配置，使您的应用对底层的节点拓扑进行感知，使其均匀地分布在不同节点上。具体操作，请参见为节点池指定部署集。

合理部署Nginx Ingress Controller

在部署Nginx Ingress Controller时，请确保Nginx Ingress Controller分布在不同的节点上，避免不同Nginx Ingress Controller之间资源的抢占和单点故障。您也可以为其使用独占节点来保证性能与稳定性，具体操作，请参见使用独占节点保证Nginx Ingress性能与稳定性。

建议您不要为Nginx Ingress Controller设定资源限制，避免OOM所带来的流量中断。如果确实有限制的需要，建议资源限制CPU不低于1000 Milicore（YAML配置里格式为1000m）和内存不低于2 GiB。关于Nginx Ingress Controller的更多配置建议，请参见Nginx Ingress最佳实践。

合理部署CoreDNS

建议您在部署CoreDNS副本时，将CoreDNS副本打散在不同可用区、不同集群节点上，避免单节点、单可用区故障。CoreDNS默认配置了按节点的弱反亲和性，可能会因为节点资源不足导致部分或全部副本部署在同一节点上，如果遇到这种情况，请删除Pod重新触发其调度来调整。

CoreDNS所运行的集群节点应避免CPU、内存用满的情况，否则会影响域名解析的QPS和响应延迟。关于CoreDNS的更多配置建议，请参见DNS最佳实践。

1.3. Kubernetes集群网络规划

在创建ACK Kubernetes集群时，您需要指定专有网络VPC、虚拟交换机、Pod网络CIDR（地址段）和Service CIDR（地址段）。因此建议您提前规划ECS地址、Kubernetes Pod地址和Service地址。本文将介绍阿里云专有网络VPC环境下ACK Kubernetes集群里各种地址的作用，以及地址段该如何规划。

专有网络VPC网段和Kubernetes网段关系

专有网络VPC（下文简称为VPC或专有网络）的网段规划包含VPC自身网段和虚拟交换机网段，Kubernetes网段规划包含Pod地址段和Service地址段。ACK网络支持Terway和Flannel两种网络模式。

Terway网络模式 | Flannel网络模式

Terway网络模式图

配置Terway模式网络时，需要设置的参数及参数网段配置的注意事项如下：

- **专有网络**
  您在创建VPC时需要选择网段，只能从10.0.0.0/8、172.16.0.0/12、192.168.0.0/16三者当中选择一个。
- **虚拟交换机**

ECS使用的交换机，用于节点间网络通信。在VPC里创建交换机时指定的网段，必须是当前VPC网段的子集（可以和VPC网段一样，但不能超过）。配置网段时，请注意：

- 虚拟交换机是VPC交换机。
- 交换机下ECS所分配到的地址，就是从这个交换机网段内获取的。
- 一个VPC下，可以创建多个交换机，但交换机网段不能重叠。
- 虚拟交换机和Pod虚拟交换机需要在一个可用区下。

- **Pod虚拟交换机**
  Pod地址从该交换机分配，用于Pod网络通信。Pod是Kubernetes内的概念，每个Pod具有一个IP地址。在VPC里创建交换机时指定的网段，必须是当前VPC网段的子集。配置网段时，请注意：
  - Pod虚拟交换机是VPC交换机。
  - Terway网络模式下，Pod分配的Pod IP就是从这个交换机网段内获取的。
  - 该地址段不能和**虚拟交换机**网段重叠。
  - 该地址段不能和**Service CIDR**网段重叠。
  - 虚拟交换机和Pod虚拟交换机需要在一个可用区下。
- **Service CIDR**
  Service地址段。Service是Kubernetes内的概念，对应的是Service类型为ClusterIP时Service使用的地址，每个Service有自己的地址。配置网段时，请注意：
  - Service地址只在Kubernetes集群内使用，不能在集群外使用。
  - Service地址段不能和**虚拟交换机**地址段重叠。
  - Service地址段不能和**Pod虚拟交换机**地址段重叠。

网络规划

在阿里云环境下使用ACK支持的Kubernetes集群，首先需要根据业务场景、集群规模进行网络规划。您可以按下表规格进行规划（未包含场景，请根据实际需要自行调整）。

集群节点规模	目的	VPC规划	可用区
小于100个节点	一般性业务	单VPC	1个
任意	需要多可用区	单VPC	2个及以上
任意	对可靠性有极致要求、需要多地域	多VPC	2个及以上

本文针对Flannel和Terway网络场景，规划容器网络：

- **Flannel配置示例**

专有网络网段	虚拟交换机网段	Pod网络CIDR网段	Service CIDR网段	最大可分配Pod地址数
192.168.0.0/16	192.168.0.0/24	172.20.0.0/16	172.21.0.0/20	65536

- **Terway配置示例**

- Terway Pod独占模式或IPVlan模式

专有网络网段	虚拟交换机网段	Pod虚拟交换机网段	Service CIDR网段	最大可分配Pod地址数
192.168.0.0/16	192.168.0.0/19	192.168.32.0/19	172.21.0.0/20	8192

- **Terway多可用区配置**

专有网络网段	虚拟交换机网段	Pod虚拟交换机网段	Service CIDR网段	最大可分配Pod地址数
192.168.0.0/16	可用区I 192.168.0.0/19	192.168.32.0/19	172.21.0.0/20	8192
	可用区J 192.168.64.0/19	192.168.96.0/19		8192

VPC网络规划

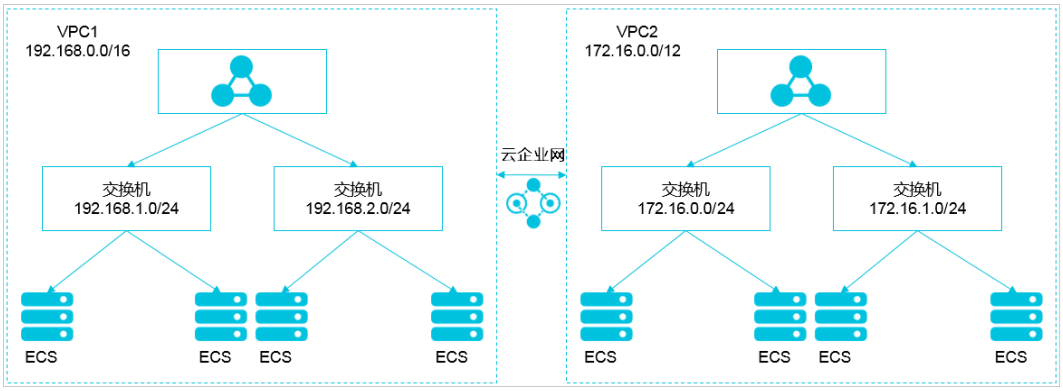
容器网络规划

如何选择地址段？

- **场景1：单VPC+单Kubernetes集群**
  这是最简单的情形。VPC地址在创建VPC的时候就已经确定，创建Kubernetes集群时，选择的Pod及Service地址网段和当前VPC不一样的地址段即可。
- **场景2：单VPC+多Kubernetes集群**
  一个VPC下创建多个Kubernetes集群。
  - VPC地址是在创建VPC时已经确定。创建Kubernetes集群时，每个集群内的VPC地址段、Service地址段和Pod地址段彼此间不能重叠。
  - 所有Kubernetes集群之间的Pod地址段不能重叠，但Service地址段可以重叠。
  - 在默认的网络模式下（Flannel），Pod的报文需要通过VPC路由转发，容器服务会自动在VPC路由上配置到每个Pod地址段的路由表。

  > **说明** 这种情况下Kubernetes集群部分互通，一个集群的Pod可以直接访问另外一个集群的Pod和ECS，但不能访问另外一个集群的Service。

- **场景3：VPC互联**
  两个VPC网络互联的情况下，可以通过路由表配置哪些报文要发送到对端VPC里。如下表所示，VPC 1使用地址段192.168.0.0/16，VPC 2使用地址段172.16.0.0/12，您可以通过路由表，指定在VPC 1里把目的地址为172.16.0.0/12的报文都发送到VPC 2。



VPC互联场景

类别	地址段	目的端	转发到
VPC 1	192.168.0.0/16	172.16.0.0/12	VPC 2
VPC 2	172.16.0.0/12	192.168.0.0/16	VPC 1

在这种情况下，VPC 1和VPC 2里创建的Kubernetes集群有以下限制：

- 不能和VPC 1的地址段重叠
- 不能和VPC 2的地址段重叠
- 不能和其他集群的地址段重叠
- 不能和Pod的地址段重叠
- 不能和Service的地址段重叠

此例子中，Kubernetes集群Pod地址段可以选择10.0.0.0/8下的某个子段。

> **说明** 您需特别关注转发到VPC 2的地址段，可以把这部分地址理解成已经占用的地址，Kubernetes集群不能和已经占用的地址重叠。

如果VPC 2里要访问VPC 1的Kubernetes Pod，则需要在VPC 2里配置到VPC 1 Kubernetes集群Pod地址的路由。

- **场景4：VPC网络到IDC**
  和VPC互联场景类似，同样存在VPC里部分地址段路由到IDC，Kubernetes集群的Pod地址就不能和这部分地址重叠。IDC里如果需要访问Kubernetes里的Pod地址，同样需要在IDC端配置到专线VBR的路由表。

相关文档

- 网络概述
- 使用Terway网络插件
- 使用网络策略Network Policy
- 网络管理FAQ

1.4. 混合云部署Kubernetes集群

本方案演示线下IDC与阿里云以混合云的形式部署Kubernetes集群，实现云下云上服务交叉调用，云上资源可以根据业务请求峰谷值做弹性伸缩。

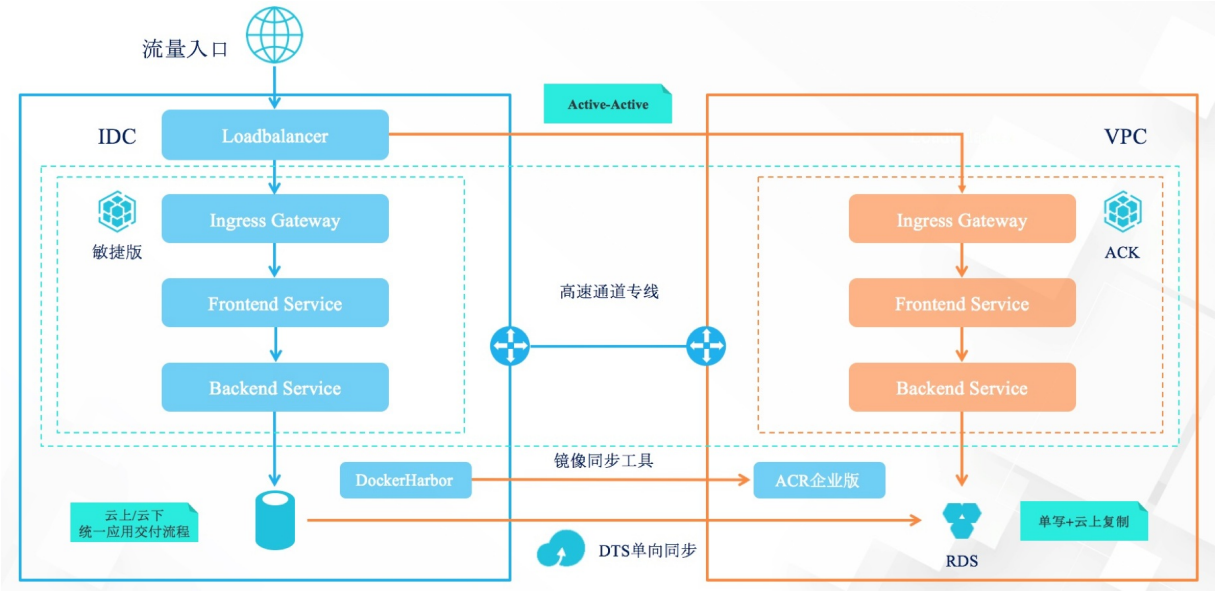
背景信息

使用DTS实现IDC与云上数据库单向同步，将云上数据库作为混合云场景下的冗余和灾备，可以在云上部署可接受一定读延迟的业务应用，缓解IDC机房业务压力。IDC本地镜像文件可以同步至阿里云镜像仓库ACR企业版，实现云上应用本地就近拉取镜像，加速应用部署。

通过该方案，可以解决以下问题：

- 通过云上冗余部署和快速弹性伸缩，低成本解决IDC弹性伸缩难问题。
- 云上可以部署可接受一定读延迟的应用，降低IDC主服务的压力。
- ACR企业版支持多地域同步部署，加速镜像分发。
- DTS单向同步数据灾备，提升IDC机房数据库高可用。

实现原理



- IDC与云上独立部署Kubernetes集群，通过高速通道打通内网环境。
- 云上集群可冗余或独立部署应用，调用IDC提供的数据服务接口访问IDC数据库。
- IDC部署的Docker harbor等镜像仓库同步镜像文件至阿里云ACR企业版，云上集群节点直接从ACR就近拉取Docker镜像，实现快速部署。
- 云下数据库通过DTS单向同步任务同步数据至云上RDS做灾备，云上也可以部署接受一定读延迟的业务应用直接访问云上数据库。

解决方案

关于完整的最佳实践，请参见混合云部署Kubernetes集群。

1.5. 提升ACK专有集群的etcd存储容量上限

ACK专有集群（Dedicated Kubernetes Cluster）最初使用的etcd版本是社区v3.3.8，其支持的最大Backend DB Size是2 GB。当集群中写入的etcd存储超过2 GB时，则会出现无法写入etcd的情况。本文介绍如何通过升级etcd版本到v3.4.3来提升ACK专有集群的etcd存储容量上限。

前提条件

- ACK专有集群的etcd版本小于v3.4.3。
- 对etcd存储的需求大于2 GB。如果没有此需求，则可以不用升级。

背景信息

etcd社区v3.4.3版本允许写入100 GB的etcd存储，可以解决etcd存储不足的问题。

操作步骤

1. 通过SSH依次登录到etcd所在的Master节点，确认当前etcd版本是v3.3.8。
2. 执行下面的Shell脚本，脚本将会下载etcd-v3.4.3 binary并且启动新版本。

> **说明**
> - 请对节点进行逐个升级，确保升级节点Ready后再升级其他节点的etcd。
> - 因为etcd具有高可用性，所以升级不会导致业务访问etcd异常。

```
!/usr/bin/env bash
```

```
etcdbin=http://aliacs-k8s-cn-hangzhou.oss.aliyuncs.com/etcd/etcd-v3.4.3/etcd
etcdctlbin=http://aliacs-k8s-cn-hangzhou.oss.aliyuncs.com/etcd/etcd-v3.4.3/etcdctl
function download(){
    wget -O etcd ${etcdbin}
    wget -O etcdctl ${etcdctlbin}
    chmod +x {etcd,etcdctl}
    mv etcd /usr/bin/etcd
    mv etcdctl /usr/bin/etcdctl
    etcd --version
}
function config() {
    ETCD_FILE=/lib/systemd/system/etcd.service
    sed -i "/ETCD_EXPERIMENTAL_BACKEND_BBOLT_FREELIST_TYPE/ d" ${ETCD_FILE}
    sed -i "/ETCD_QUOTA_BACKEND_BYTES/ d" ${ETCD_FILE}
    sed -i "/^\[Service\]/a\Environment=\"ETCD_EXPERIMENTAL_BACKEND_BBOLT_FREELIST_TYPE
=map\"" ${ETCD_FILE}
    sed -i "/^\[Service\]/a\Environment=\"ETCD_QUOTA_BACKEND_BYTES=100000000000\"" ${ET
CD_FILE}
    sed -i "s/initial-cluster-state new/initial-cluster-state existing/g" ${ETCD_FILE}
    systemctl daemon-reload
    systemctl restart etcd
}
download; config
ENDPOINTS=`ps -eaf|grep etcd-servers|grep -v grep|awk -F "=" '{print $22}'|awk -F " " '
{print $1}'`
ETCDCTL_API=3 etcdctl \
        --endpoints=${ENDPOINTS}         \
        --cacert=/var/lib/etcd/cert/ca.pem         \
        --cert=/var/lib/etcd/cert/etcd-client.pem         \
        --key=/var/lib/etcd/cert/etcd-client-key.pem \
        member list
```

3. 执行以下命令，确认etcd进程启动成功。

```
ps aux|grep etcd
```

后续步骤

检查和确认etcd的健康状况。

```
ETCDCTL_API=3 etcdctl --endpoints=${ENDPOINTS} \
      --cacert=/var/lib/etcd/cert/ca.pem \
      --cert=/var/lib/etcd/cert/etcd-client.pem \
      --key=/var/lib/etcd/cert/etcd-client-key.pem endpoint health
```

```
ENDPOINTS is healthy
```

1.6. ACK容器平台集群安全控制

本文介绍ACK容器平台集群安全控制的场景描述、解决问题、架构图及操作参考链接。

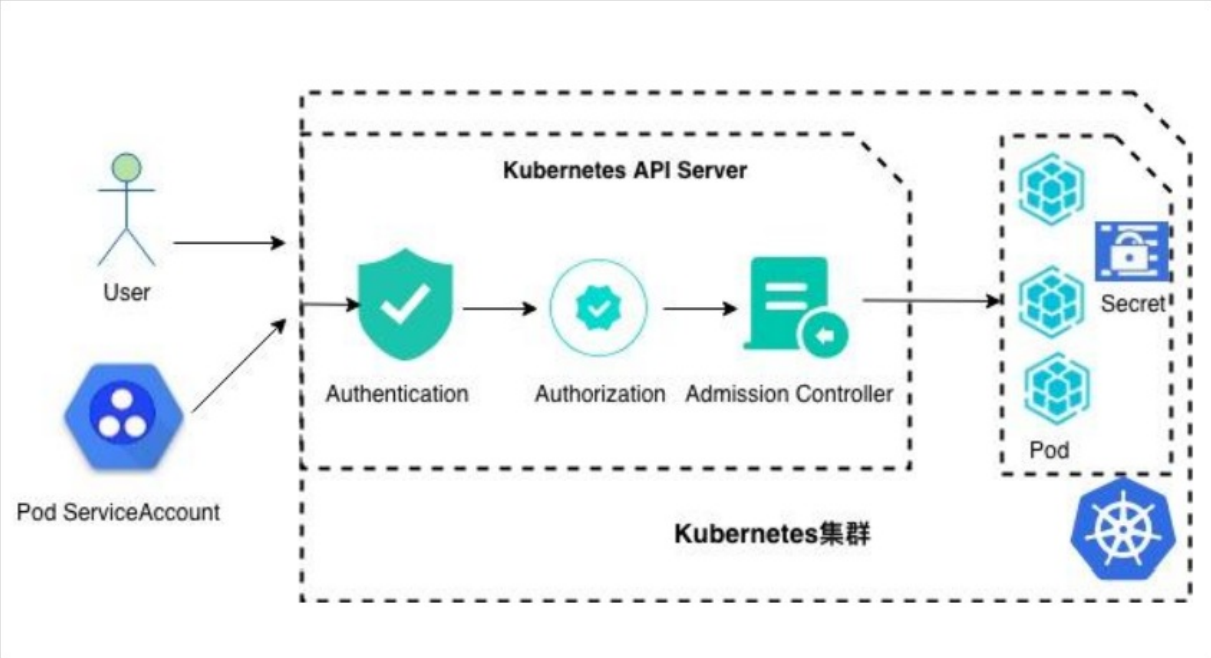
场景描述

本方案主要是通过一些实践示例来介绍您对于在阿里云上使用ACK集群服务的容器平台安全管控的实践验证与使用建议。

解决问题

- 容器集群API Server的安全访问控制。
- 容器服务多租户场景下的授权管理。
- 容器中的敏感信息数据的存储。
- 容器服务集群安全策略配置管理。

架构图



参考链接

有关ACK容器平台集群安全控制的详情，请参见ACK容器平台集群安全控制。

1.7. ACK集群神龙资源错峰利用

本文介绍ACK容器平台集群安全控制的场景描述、解决问题、架构图及操作参考链接。

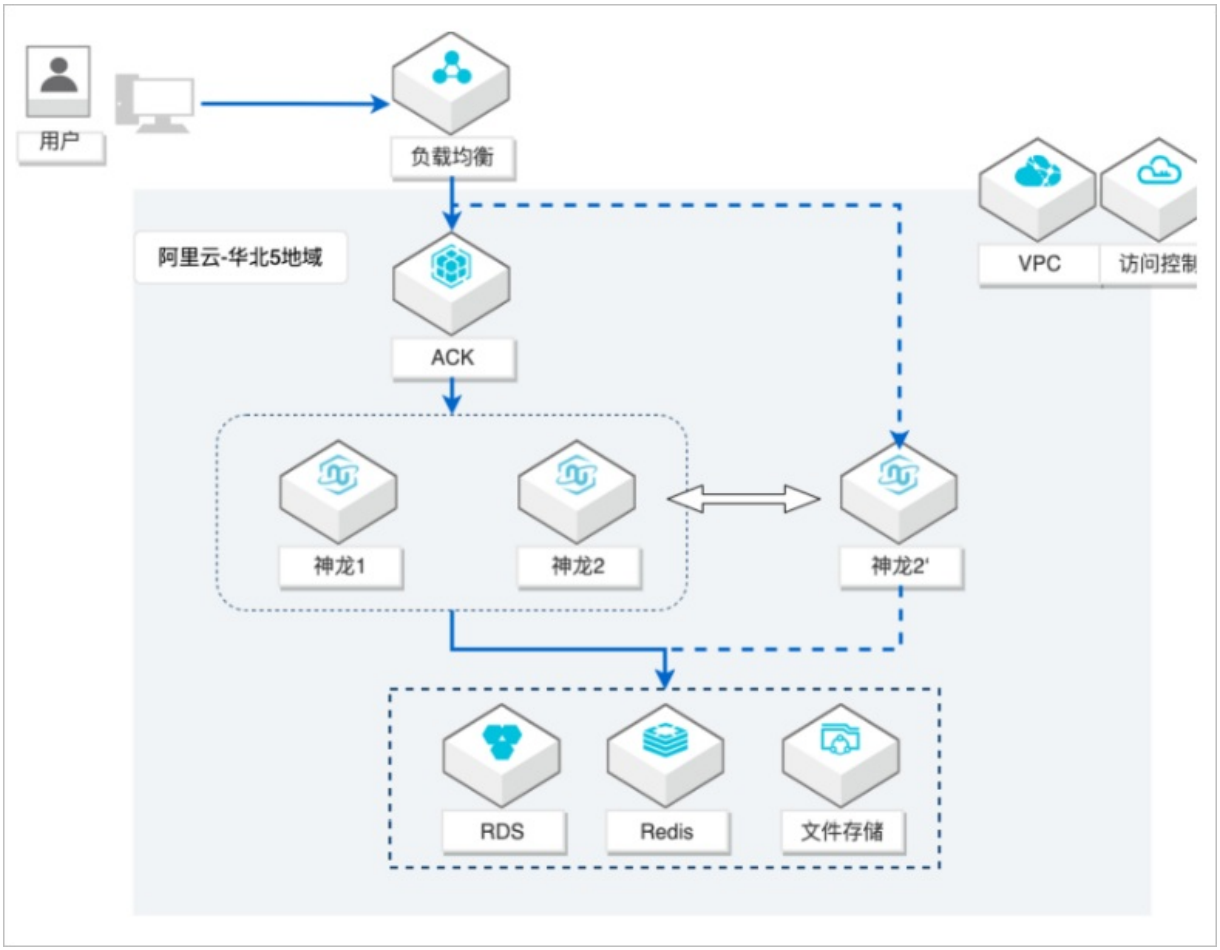
场景描述

使用ACK构建容器集群环境，神龙资源为集群节点资源部署应用。在主业务低谷期，通过将部分神龙节点从容器集群中移除，更换镜像切换操作系统及应用环境， 使得这部分神龙实例资源服务于其他业务。在主业务高峰期前将神龙资源重新加入ACK集群。从而达到错峰利用神龙资源的目的，以便充分利用神龙资源，降低资源成本。

解决问题

- 基于ACK及神龙资源构建容器集群环境，典型部署应用。
- ACK内挂载NAS存储。
- ACK集群神龙实例节点移除和重新加入集群。

架构图



参考链接

有关ACK集群神龙资源错峰利用的详情，请参见ACK集群神龙资源错峰利用。

2.节点池

2.1. 抢占式实例节点池最佳实践

抢占式实例是一种按需使用的实例，相对于按量付费实例价格有一定的折扣。抢占式实例节点池是由抢占式实例、按量付费实例按照一定比例组合而成的节点池，使用抢占式实例节点池，可以节省一定的费用。本文介绍抢占式实例节点池的概念、适用场景，以及如何配置抢占式实例组合等内容。

背景信息

抢占式实例采用按量付费的计费方式，即先使用后付费。费用根据市场价格和计费时长进行计算。更多信息，请参见抢占式实例。

抢占式实例节点池介绍

抢占式实例（Spot Instance）节点池是由抢占式实例、按量付费实例按照一定比例组合而成的节点池。

抢占式实例是一种特殊按量付费实例，价格随着库存等因素动态波动，价格成本较低，相比按量实例节点最多可以节省90%成本。抢占式实例的市场价格会随供需变化而浮动，您需要在创建抢占式实例时指定出价模式，当指定实例规格的实时市场价格低于出价且库存充足时，就能成功创建抢占式实例。

抢占式实例创建成功后，操作与按量付费实例相同，您也可以将抢占式实例和其他云产品组合使用，例如云盘、公网IP地址等。抢占式实例默认有1小时的保护期。超过保护期后，每5分钟检测一次实例规格的实时市场价格和库存，如果某一时刻的市场价格高于出价或实例规格库存不足，抢占式实例会被释放。

适用场景分析

- **抢占式实例节点池**
  抢占式实例节点池由于使用抢占式实例，节点可能在不确定时刻到期被回收，因此适合用于无状态、容错性较好的应用。其中包括批处理和机器学习培训工作负载、大数据ETL（例如Apache Spark）、队列处理应用和无状态API应用等。
  部署在抢占式实例节点池的工作负载需要容忍所需节点资源不可用的时段，对无法容忍的应用负载建议使用按量实例或者包年包月实例的节点池。这类无法容忍的负载一般包含但不限于以下应用：
  - 集群管理工具，例如监控和操作工具。
  - 需要有状态工作负载的部署或应用程序，例如数据库。

- **开启自动伸缩的抢占式实例节点池**
  如果工作负载不仅可部署在抢占式实例节点池，还具有较明显的业务高低峰窗口期，建议您使用开启自动伸缩的抢占式实例节点池。
  开启自动伸缩后，集群节点自动伸缩组件会检查是否需要扩容抢占式实例节点池来部署集群中Pod，以及当节点达到缩容条件时进行自动缩容。开启自动伸缩的抢占式节点池在扩容时触发扩容速度更快，对闲置资源的释放更及时。在节点池内实例可弹出情况下，快速的弹出弥补了一部分抢占式节点池被动回收的不足，及时释放闲置资源加强了节省成本的优势。

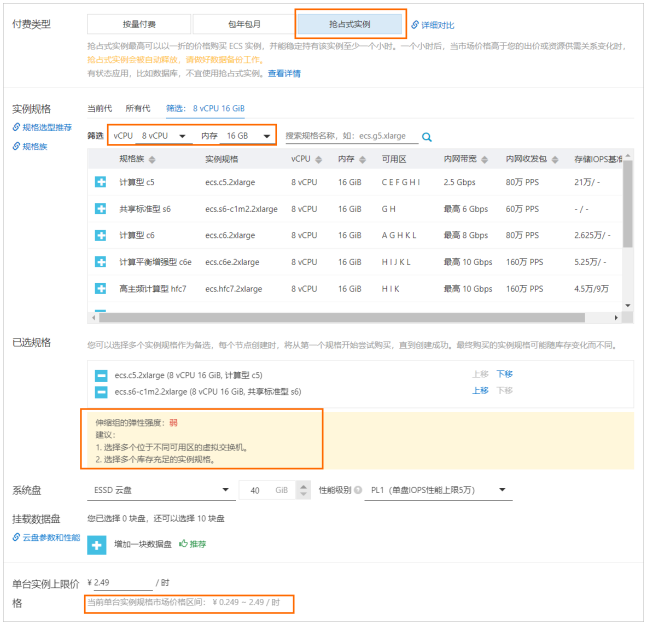
抢占式实例组合的选择和配置

在实例规格选择上没有“最优解”，建议您选择最适合业务类型的，且在库存、成本和性能间达到最佳平衡的方案。为了满足不同业务，阿里云ECS提供了大量的实例类型，如何在实例规格中选出您需要的组合方案，特别是在竞价场景下选择出对业务可能影响最小的实例组合，是用好抢占式实例节点池的第一步。

您可以通过以下方式选择和配置抢占式实例组合：

- **根据容器服务管理控制台推荐**

容器服务管理控制台提供了实例的选择建议。在创建或编辑节点池时，控制台会根据所选地域给出当前时刻所在地域有库存的实例规格，您可以通过所需资源对实例规格进行进一步筛选。在选出实例规格后，控制台会同时算出弹性强度和实例的价格区间，您可以参照弹性强度建议，添加实例规格和设置实例价格上限。关于如何创建或编辑节点池，请参见创建节点池。



- **通过spot-instance-advisor命令行**

  容器服务开源了命令行 `spot-instance-advisor`，以便您通过命令行获取抢占式实例的历史价格波动情况和当前价格信息。`spot-instance-advisor` 的原理是通过API获取一个地域的实例规格与历史价格曲线，通过统计分析的方式，计算排序最低核时的机型，并通过离散度的判断计算实例价格的熵值，熵值越高表示机型的价格波动越频繁。建议您选择熵值低的机型。

  `spot-instance-advisor` 支持以下过滤参数：

```
Usage of ./spot-instance-advisor:
  -accessKeyId string
        Your accessKeyId of cloud account
  -accessKeySecret string
        Your accessKeySecret of cloud account
  -cutoff int
        Discount of the spot instance prices (default 2)
  -family string
        The spot instance family you want (e.g. ecs.n1,ecs.n2)
  -limit int
        Limit of the spot instances (default 20)
  -maxcpu int
        Max cores of spot instances  (default 32)
  -maxmem int
        Max memory of spot instances (default 64)
  -mincpu int
        Min cores of spot instances (default 1)
  -minmem int
        Min memory of spot instances (default 2)
  -region string
        The region of spot instances (default "cn-hangzhou")
  -resolution int
        The window of price history analysis (default 7)
```

执行以下命令，即可获得当前地域内最适合的实例规格配置：

```
./spot-instance-advisor --accessKeyId=<id> --accessKeySecret=<secret> --region=<cn-zhangjiakou>
```

> **说明** `accessKeyId`、`accessKeySecret`和`region`为必填参数，请根据您的实际业务场景填写。

预期输出：

```
Initialize cache ready with 619 kinds of instanceTypes
Filter 93 of 98 kinds of instanceTypes.
Fetch 93 kinds of InstanceTypes prices successfully.
Successfully compare 199 kinds of instanceTypes
      InstanceTypeId               ZoneId     Price(Core)        Discount            ratio
        ecs.c6.large     cn-zhangjiakou-c          0.0135             1.0              0.0
        ecs.c6.large     cn-zhangjiakou-a          0.0135             1.0              0.0
      ecs.c6.2xlarge     cn-zhangjiakou-a          0.0136             1.0              0.0
      ecs.c6.2xlarge     cn-zhangjiakou-c          0.0136             1.0              0.0
      ecs.c6.3xlarge     cn-zhangjiakou-a          0.0137             1.0              0.0
      ecs.c6.3xlarge     cn-zhangjiakou-c          0.0137             1.0              0.0
       ecs.c6.xlarge     cn-zhangjiakou-c          0.0138             1.0              0.0
       ecs.c6.xlarge     cn-zhangjiakou-a          0.0138             1.0              0.0
     ecs.hfc6.xlarge     cn-zhangjiakou-a          0.0158             1.0              0.0
      ecs.hfc6.large     cn-zhangjiakou-a          0.0160             1.0              0.0
      ecs.hfc6.large     cn-zhangjiakou-c          0.0160             1.0              0.0
      ecs.g6.3xlarge     cn-zhangjiakou-a          0.0175             1.0              0.0
      ecs.g6.3xlarge     cn-zhangjiakou-c          0.0175             1.0              0.0
        ecs.g6.large     cn-zhangjiakou-a          0.0175             1.0              0.0
       ecs.g6.xlarge     cn-zhangjiakou-a          0.0175             1.0              0.0
      ecs.g6.2xlarge     cn-zhangjiakou-a          0.0175             1.0              1.0
      ecs.g6.2xlarge     cn-zhangjiakou-c          0.0175             1.0              3.0
        ecs.g6.large     cn-zhangjiakou-c          0.0175             1.0             30.
8
       ecs.g6.xlarge     cn-zhangjiakou-c          0.0175             1.0              9.7
      ecs.hfg6.large     cn-zhangjiakou-c          0.0195             1.0              0.2
```

由以上输出发现排名前几位的价格、混沌系数（对应 `ratio` ）都相对比较平稳，而后面几个实例的价格虽然也处在1折，但是混沌系数（对应 `ratio` ）相对而言会比较高，因此在配置实例规格时，可以优先考虑前面价格较低且混沌系数较低的规格组合。

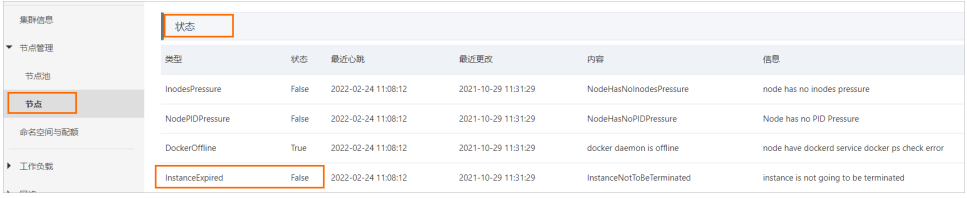
抢占式实例到期的优雅处理方式

抢占式实例到期的优雅处理方式主要包括：监控与通知、节点预补偿与策略、自定义处理行为。

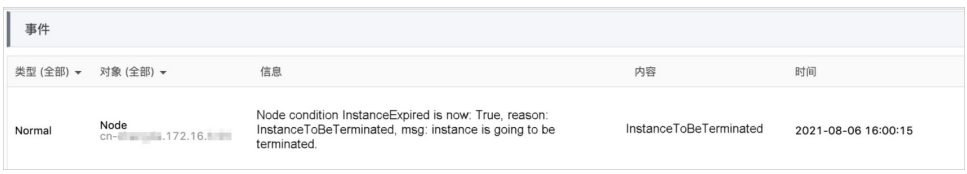
- **监控与通知**

  为了让节点池中抢占式实例释放消息尽早通知到您，ACK集群通过组件NPD监控Spot实例的预释放消息。

  - 没有监控到该抢占式实例的预释放消息时，该节点的状态中**InstanceExpired**值为**False**。



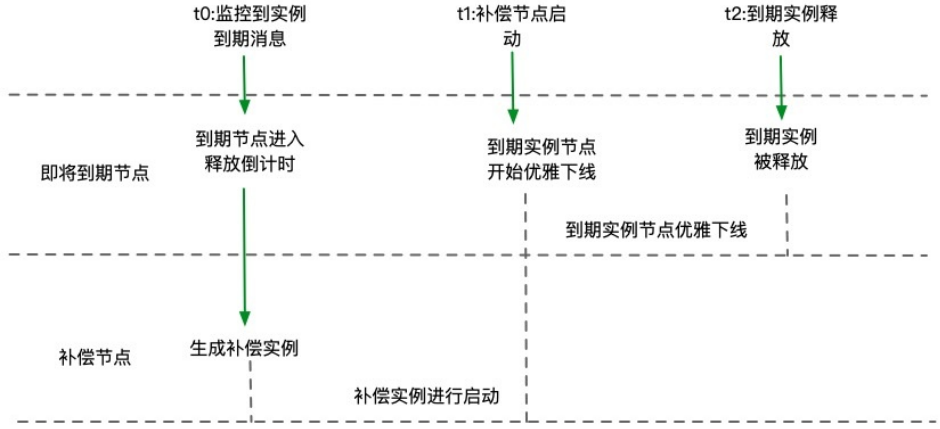
  - 抢占式实例**InstanceExpired**值为**True**时，表示Spot实例即将到期，即将被释放。ACK会通过集群事件（Kubernetes Events）通知您抢占式实例即将释放的消息。



- **节点预补偿与策略**

抢占式实例的到期释放是影响节点上业务负载稳定性的一个关键因素。容器服务ACK已经从配置、开启自动伸缩到监控通知各种维度提供了方法去快速响应抢占式实例到期释放事件。这些处理方式的处理时间点都在抢占式实例到期回收之后，在回收到新实例补充这段时间集群可用资源仍然会减少。为尽量缩短甚至消除这段时间影响，ACK利用节点预补偿功能，在到期实例还未回收前就触发弹出补偿实例。
开启节点预补偿后，ACK会自动监控节点实例是否即将被释放。当ACK监控到节点实例即将被释放时，会自动触发弹出新节点的伸缩活动。这个为了补充即将被释放实例而弹出的实例，称为补偿实例。补偿实例成功运行后，会触发到期抢占式实例的缩容和释放，缩容释放策略包括cordon节点（将节点设置为不可调度）、排水节点、将节点移除等节点优雅下线处理，尽量让到期抢占式实例节点上的业务负载平稳迁移到集群中其他节点上，避免业务受到实例到期的影响。

> **说明** 节点预补偿的结果不会影响到期抢占式实例的回收。无论是否开启节点预补偿，到期实例仍然会在预回收的5分钟后被回收释放掉。



- **自定义处理行为**
  在许多实际业务场景中，节点下线需要执行比一般优雅下线更多的步骤，例如即将下线的节点信息需要从注册的DNS中心移除。结合监控与通知，此类需求建议您监控节点状态中的**InstanceExpired**或者监听**InstanceToBeTerminated**事件。收到节点实例到期或将被释放消息后，即可把该节点作为即将下线处理，然后执行自定义的处理行为。关于监控Spot实例到期状态的具体操作，请参见查看Spot实例到期状态。

相关文档

- 抢占式实例
- 配置节点池内抢占式实例和存量实例的比例
- 管理节点池
- 查看Spot实例到期状态

3.网络

3.1. 使用Ambassador Edge Stack管理Ingress资源

Ambassador Edge Stack（AES）是一个基于Envoy Proxy实现的高性能的Ingress Controller和API网关。AES通过Custom Resource Definitions（CRD）使用Enovy提供的功能，集成了速率限制、身份认证、负载均衡和可观测性等功能。本文将介绍如何使用AES管理K8s Ingress资源。

前提条件

- 创建Kubernetes托管版集群。
- 安装和设置kubectl。

安装部署AES

ACK默认不支持部署AES，您可以自行根据需要进行部署。下文以YAML方式为例介绍如何安装部署AES，更多其他安装部署方式请参见AES官方文档。

1. 执行以下命令部署AES。

```
kubectl apply -f https://www.getambassador.io/yaml/aes-crds.yaml && \
kubectl wait --for condition=established --timeout=90s crd -lproduct=aes && \
kubectl apply -f https://www.getambassador.io/yaml/aes.yaml && \
kubectl -n ambassador wait --for condition=available --timeout=90s deploy -lproduct=aes
```

输出以下内容即表示部署成功。

```
customresourcedefinition.apiextensions.k8s.io/authservices.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/consulresolvers.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/hosts.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/kubernetesendpointresolvers.getambassador
.io created
customresourcedefinition.apiextensions.k8s.io/kubernetesserviceresolvers.getambassador.
io created
customresourcedefinition.apiextensions.k8s.io/logservices.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/mappings.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/modules.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/ratelimitservices.getambassador.io create
d
customresourcedefinition.apiextensions.k8s.io/tcpmappings.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/tlscontexts.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/tracingservices.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/filterpolicies.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/filters.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/ratelimits.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/projectcontrollers.getambassador.io creat
ed
customresourcedefinition.apiextensions.k8s.io/projects.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/projectrevisions.getambassador.io created
customresourcedefinition.apiextensions.k8s.io/authservices.getambassador.io condition m
et
customresourcedefinition.apiextensions.k8s.io/consulresolvers.getambassador.io conditio
```

```
n met
customresourcedefinition.apiextensions.k8s.io/filterpolicies.getambassador.io condition
met
customresourcedefinition.apiextensions.k8s.io/filters.getambassador.io condition met
customresourcedefinition.apiextensions.k8s.io/hosts.getambassador.io condition met
customresourcedefinition.apiextensions.k8s.io/kubernetesendpointresolvers.getambassador
.io condition met
customresourcedefinition.apiextensions.k8s.io/kubernetesserviceresolvers.getambassador.
io condition met
customresourcedefinition.apiextensions.k8s.io/logservices.getambassador.io condition me
t
customresourcedefinition.apiextensions.k8s.io/mappings.getambassador.io condition met
customresourcedefinition.apiextensions.k8s.io/modules.getambassador.io condition met
customresourcedefinition.apiextensions.k8s.io/projectcontrollers.getambassador.io condi
tion met
customresourcedefinition.apiextensions.k8s.io/projectrevisions.getambassador.io conditi
on met
customresourcedefinition.apiextensions.k8s.io/projects.getambassador.io condition met
customresourcedefinition.apiextensions.k8s.io/ratelimits.getambassador.io condition met
customresourcedefinition.apiextensions.k8s.io/ratelimitservices.getambassador.io condit
ion met
customresourcedefinition.apiextensions.k8s.io/tcpmappings.getambassador.io condition me
t
customresourcedefinition.apiextensions.k8s.io/tlscontexts.getambassador.io condition me
t
customresourcedefinition.apiextensions.k8s.io/tracingservices.getambassador.io conditio
n met
Warning: kubectl apply should be used on resource created by either kubectl create --sa
ve-config or kubectl apply
namespace/ambassador configured
serviceaccount/ambassador created
clusterrole.rbac.authorization.k8s.io/ambassador created
clusterrolebinding.rbac.authorization.k8s.io/ambassador created
clusterrole.rbac.authorization.k8s.io/ambassador-projects created
clusterrolebinding.rbac.authorization.k8s.io/ambassador-projects created
service/ambassador-redis created
deployment.apps/ambassador-redis created
ratelimitservice.getambassador.io/ambassador-edge-stack-ratelimit created
authservice.getambassador.io/ambassador-edge-stack-auth created
secret/ambassador-edge-stack created
mapping.getambassador.io/ambassador-devportal created
mapping.getambassador.io/ambassador-devportal-api created
service/ambassador created
service/ambassador-admin created
deployment.apps/ambassador created
deployment.extensions/ambassador condition met
deployment.extensions/ambassador-redis condition met
```

使用AES测试Ingress Controller功能

为了测试AES的Ingress Controller功能，您需要部署一个测试用的Deployment。

1. 执行以下命令，创建Deployment部署配置。

```
cat <<-EOF | kubectl apply -f -
apiVersion: v1
kind: Service
  metadata:
    name: quote
spec:
  ports:
  - name: http
    port: 80
    targetPort: 8080
  selector:
    app: quote
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: quote
spec:
  replicas: 1
  selector:
    matchLabels:
      app: quote
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        app: quote
    spec:
      containers:
      - name: backend
        image: quay.io/datawire/quote:0.3.0
        ports:
        - name: http
          containerPort: 8080
EOF
```

2. 执行以下命令，创建Ingress配置。

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  annotations:
    kubernetes.io/ingress.class: ambassador
  name: test-ingress
spec:
  rules:
  - http:
      paths:
      - path: /backend/
        backend:
          service:
            name: quote
            port:
              number: 80
        pathType: ImplementationSpecific
EOF
```

AES通过CRD提供了更多功能，如下面的配置CRD完全等同如上Ingress的配置。更多信息，请参见官方文档。

```
cat > quote-backend.yaml <<-EOF
apiVersion: getambassador.io/v2
kind: Mapping
metadata:
    name: backend
spec:
    prefix: /backend/
    service: quote
EOF
```

3. 执行以下命令，获取IP地址。

```
kubectl get -n ambassador service ambassador -o "go-template={{range .status.loadBalanc
er.ingress}}{{or .ip .hostname}}{{end}}"
```

4. 执行以下命令，测试AES的Ingress Controller功能。

```
curl -k https://{{AMBASSADOR_IP}}/backend/    #替换AMBASSADOR_IP为上面获取的IP地址。
```

输出类似以下代码即表示成功。

```
  {
    "server": "icky-grapefruit-xar5if66",
    "quote": "A small mercy is nothing at all?",
    "time": "2020-07-17T09:00:57.646315605Z"
  }
```

3.2. 通过Ambassador暴露应用API

Ambassador Edge Stack（AES）是一个基于Envoy Proxy实现的高性能的Ingress Controller和API网关。AES通过Custom Resource Definitions（CRD）使用Enovy提供的功能，集成了速率限制、身份认证、负载均衡和可观测性等功能。本文介绍如何通过Ambassador Edge Stack（AES）暴露部署在ACK集群中的应用API。

前提条件

- 创建Kubernetes托管版集群
- 安装和设置Kubectl

步骤一：部署AES

1. 执行以下命令部署AES。

```
kubectl apply -f https://www.getambassador.io/yaml/aes-crds.yaml
kubectl wait --for condition=established --timeout=90s crd -lproduct=aes
kubectl apply -f https://www.getambassador.io/yaml/aes.yaml
kubectl -n ambassador wait --for condition=available --timeout=90s deploy -lproduct=aes
```

2. 执行以下命令确认AES部署成功。

```
kubectl get pod -n ambassador
```

预期输出：

```
NAME                                READY   STATUS    RESTARTS   AGE
ambassador-566d496b77-tg6ng         1/1     Running   0          2m
ambassador-redis-5fbd59f4fb-88gm8   1/1     Running   0          2m
```

部署Ambassador的Pod显示Running，表示AES部署成功。

步骤二：暴露应用API

1. 使用以下YAML示例创建一个测试应用及对应的Service。

```
cat <<-EOF | kubectl apply -f -
---
apiVersion: v1
kind: Service
metadata:
  name: quote
  namespace: ambassador
spec:
  ports:
  - name: http
    port: 80
    targetPort: 8080
  selector:
    app: quote
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: quote
  namespace: ambassador
spec:
  replicas: 1
  selector:
    matchLabels:
      app: quote
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        app: quote
    spec:
      containers:
      - name: backend
        image: docker.io/datawire/quote:0.4.1
        ports:
        - name: http
          containerPort: 8080
EOF
```

2. 确认应用及对应的Service创建成功。
   - 确认应用部署成功。

     ```
     kubectl get pod -n ambassador -l app=quote
     ```

     预期输出：

     ```
     NAME                     READY   STATUS    RESTARTS   AGE
     quote-d57b799b4-j****    1/1     Running   0          6m29s
     ```

     部署应用的Pod显示running，表示应用部署成功。
   - 确认服务部署成功。

```
kubectl get svc -n ambassador quote
```

预期输出：

```
NAME    TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)    AGE
quote   ClusterIP   172.23.XX.XX   <none>        80/TCP     5m45s
```

返回结果中显示服务名称，表示服务部署成功。

3. 创建Mapping CRD，暴露应用API。

```
cat <<-EOF | kubectl apply -f -
apiVersion: getambassador.io/v2
kind: Mapping
metadata:
    name: backend
    namespace: ambassador
spec:
    prefix: /backend/
    service: quote
EOF
```

4. 确认Mapping CRD资源创建成功。

```
kubectl get mappings -n ambassador
```

预期输出：

```
NAME                       PREFIX      SERVICE          STATE   REASON
ambassador-devportal       /docs/      12*.*.*.*:8500
ambassador-devportal-api   /openapi/   12*.*.*.*:8500
quote-backend              /backend/   quote
```

返回结果中显示Mapping CRD资源，表示Mapping CRD资源部署成功。

5. 获取AES服务的IP地址。

```
kubectl get -n ambassador service ambassador -o "go-template={{range .status.loadBalanc
er.ingress}}{{or .ip .hostname}}{{end}}"
```

预期输出：

```
47.94.XX.XX
```

6. 访问暴露的应用API。

```
curl -k https://{{IP}}/backend/
{
  "server": "frosty-kiwi-ewe7vk96",
  "quote": "A principal idea is omnipresent, much like candy.",
  "time": "2020-08-01T08:41:37.054259819Z"
}
```

请将 `${IP}` 替换为步骤获取的AES服务IP地址。

相关文档

- Ambassador Edge Stack

- 使用Ambassador Edge Stack管理Ingress资源

3.3. 优化大规模Tewary集群NetworkPolicy的扩展性

在Terway的集群中，可以使用NetworkPolicy来控制Pod之间的访问。当Terway的集群Node数量多了之后（节点大于100），NetworkPolicy的代理会给Kubernetes的管控造成不小的压力，所以在大规模集群中需要对NetworkPolicy做针对大规模集群的优化调整。本文介绍如何优化大规模Tewary集群中NetworkPolicy的性能。

前提条件

- 集群网络模式为Terway且集群规模大于100个节点以上，详情请参见创建Kubernetes托管版集群。
- 通过kubectl工具连接集群。

背景信息

Terway使用Calico的Felix作为NetworkPolicy功能的实现。在大规模（节点大于100）的集群中，每个节点上的Felix都从API Server获取规则，这样会给API Server带来不小的压力，所以在大规模集群中可以采用关闭NetworkPolicy的功能或者部署中继组件Typha的方式降低Felix对API Server的压力。

您可以通过两种方式优化大规模集群中NetworkPolicy的性能：

- 部署NetworkPolicy中继。
- 关闭集群的NetworkPolicy能力。

> **说明** 关闭集群的NetworkPolicy能力会导致NetworkPolicy的功能不可用。

部署NetworkPolicy中继

1. 登录容器服务管理控制台。
2. 升级集群的Terway组件至最新版本，具体步骤请参见管理组件。
3. 使用以下模板并执行 `kubectl apply -f` 部署NetworkPolicy中继组件Typha。

```
apiVersion: v1
kind: Service
metadata:
  name: calico-typha
  namespace: kube-system
  labels:
    k8s-app: calico-typha
spec:
  ports:
    - port: 5473
      protocol: TCP
      targetPort: calico-typha
      name: calico-typha
  selector:
    k8s-app: calico-typha
---
apiVersion: apps/v1
kind: Deployment
metadata:
```

```
  name: calico-typha
  namespace: kube-system
  labels:
    k8s-app: calico-typha
spec:
  replicas: 3
  revisionHistoryLimit: 2
  selector:
    matchLabels:
      k8s-app: calico-typha
  template:
    metadata:
      labels:
        k8s-app: calico-typha
      annotations:
        cluster-autoscaler.kubernetes.io/safe-to-evict: 'true'
    spec:
      nodeSelector:
        kubernetes.io/os: linux
      hostNetwork: true
      tolerations:
        - operator: Exists
      serviceAccountName: terway
      priorityClassName: system-cluster-critical
      containers:
      - image: registry-vpc.{REGION-ID}.aliyuncs.com/acs/typha:v3.20.2
        name: calico-typha
        ports:
        - containerPort: 5473
          name: calico-typha
          protocol: TCP
        env:
          - name: TYPHA_LOGSEVERITYSCREEN
            value: "info"
          - name: TYPHA_LOGFILEPATH
            value: "none"
          - name: TYPHA_LOGSEVERITYSYS
            value: "none"
          - name: TYPHA_CONNECTIONREBALANCINGMODE
            value: "kubernetes"
          - name: TYPHA_DATASTORETYPE
            value: "kubernetes"
          - name: TYPHA_HEALTHENABLED
            value: "true"
        livenessProbe:
          httpGet:
            path: /liveness
            port: 9098
            host: localhost
          periodSeconds: 30
          initialDelaySeconds: 30
        readinessProbe:
          httpGet:
            path: /readiness
```

```
            port: 9098
            host: localhost
          periodSeconds: 10
---
apiVersion: policy/v1beta1
kind: PodDisruptionBudget
metadata:
  name: calico-typha
  namespace: kube-system
  labels:
    k8s-app: calico-typha
spec:
  maxUnavailable: 1
  selector:
    matchLabels:
      k8s-app: calico-typha
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  name: bgppeers.crd.projectcalico.org
spec:
  scope: Cluster
  group: crd.projectcalico.org
  versions:
  - name: v1
    served: true
    storage: true
    schema:
      openAPIV3Schema:
        type: object
        properties:
          apiVersion:
            type: string
  names:
    kind: BGPPeer
    plural: bgppeers
    singular: bgppeer
```

说明

- 修改模板中的{REGION-ID}为对应的地域。
- 根据集群规模修改其中的replicas的副本数（一个副本对应200个节点，最小3副本）。

4. 执行以下命令修改Terway的配置项文件*eni-config*，增加中继配置felix_relay_service: calico-typha。

```
kubectl edit cm eni-config -n kube-system
#在打开的文件中增加以下中继配置，与eni_conf参数对齐。
felix_relay_service: calico-typha
```

5. 执行以下命令重启集群中的Terway。

```
kubectl get pod -n kube-system  | grep terway | awk '{print $1}' | xargs kubectl delete
-n kube-system pod
```

预计输出：

```
pod "terway-eniip-8hmz7" deleted
pod "terway-eniip-dclfn" deleted
pod "terway-eniip-rmctm" deleted
```

关闭集群NetworkPolicy

当您不需要NetworkPolicy的功能时，也可以采用直接关闭集群的NetworkPolicy功能，从而降低NetworkPolicy的代理对API Server的压力。

1. 修改Terway的配置项文件*eni-config*，增加禁用NetworkPolicy的配置disable_network_policy: "true"。

   ```
   kubectl edit cm -n kube-system eni-config
   # 修改（如果有以下key）或者新增：
   disable_network_policy: "true"
   ```

2. 执行以下命令重启集群中的Terway。

   ```
   kubectl get pod -n kube-system  | grep terway | awk '{print $1}' | xargs kubectl delete
   -n kube-system pod
   ```

   预计输出：

   ```
   pod "terway-eniip-8hmz7" deleted
   pod "terway-eniip-dclfn" deleted
   pod "terway-eniip-rmctm" deleted
   ```

执行结果

上述操作完成后，NetworkPolicy的代理会使用中继组件而不会再对Kubernetes的API Server造成过大压力。通过观察集群API Server的SLB的监控情况，API Server的负载下降。

3.4. 搭建高性能网络ACK集群

本文介绍搭建高性能网络ACK集群的场景描述、解决问题、架构图及操作参考链接。

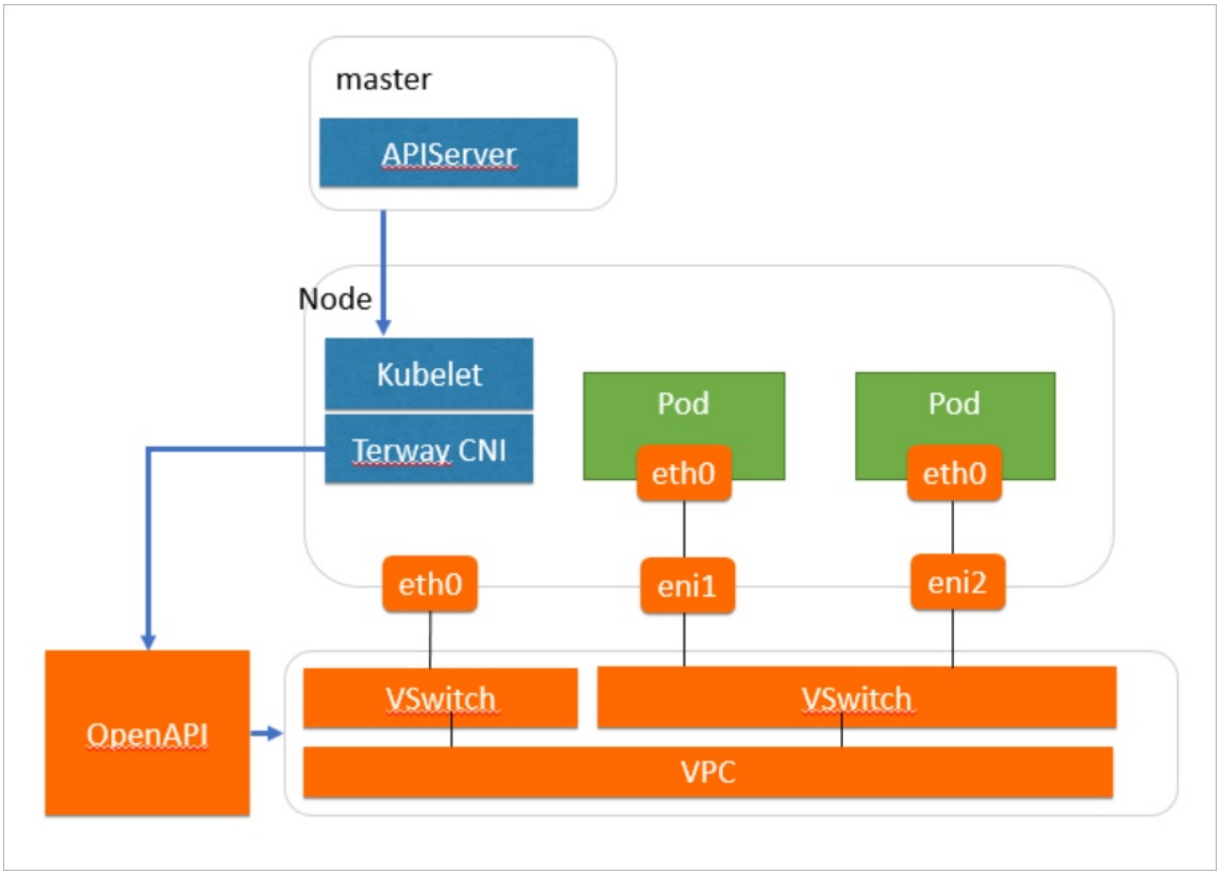
场景描述

容器的网络协议栈实现方式，导致容器之间的网络性能，相比服务器之间直接通信方式，会有一 定程度的下降。阿里云托管版容器服务ACK集群， 支持自研的Terway网络插件，该插件可以有效减少因容器而引入的网络性能下降，可以基本达到服务器之间直接通信的网络性能。本最佳实践主要是一个性能测试方面的实践，指导客户进行POC测试等。

解决问题

- Pod之间网络性能提升。
- 性能测试方法。

架构图



参考链接

有关搭建高性能网络ACK集群的详情，请参见搭建高性能网络ACK集群。

3.5. 金融行业从经典网络向VPC容器化改造

本文介绍金融行业从经典网络向VPC容器化改造的场景描述、解决问题、架构图及操作参考链接。

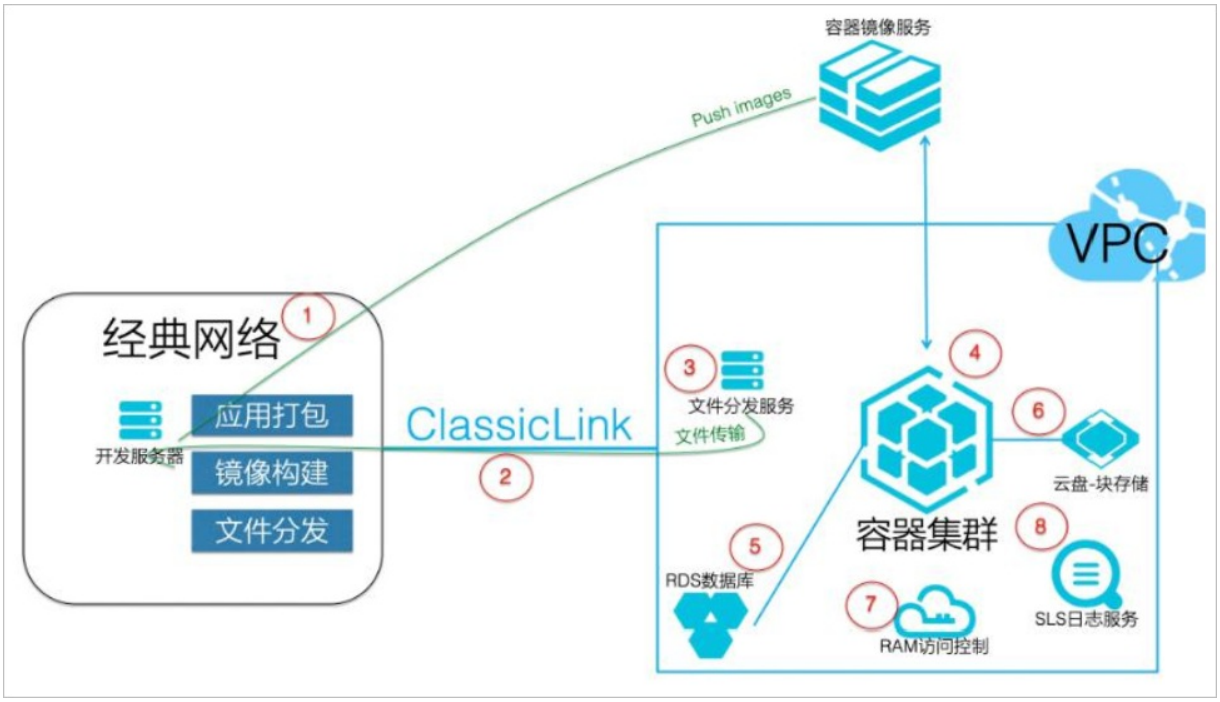
场景描述

本实践介绍经典网络向VPC容器化改造实践以及配置步骤，可适用于金融等行业。

解决问题

- 经典网络管理困难。
- 应用发布不灵活。
- 运维效率低。

架构图



参考链接

有关金融行业从经典网络向VPC容器化改造的详情，请参见金融行业从经典网络向VPC容器化改造。

3.6. DNS最佳实践

DNS是Kubernetes集群中至关重要的基础服务之一，在客户端设置不合理、集群规模较大等情况下DNS容易出现解析超时、解析失败等现象。本文介绍Kubernetes集群中DNS的最佳实践，帮助您避免此类问题。

前提条件

- 创建Kubernetes托管版集群
- 通过kubectl工具连接集群

背景信息

DNS最佳实践包含客户端和服务端的内容：

- 在客户端，您可以通过优化域名解析请求降低解析延迟，通过使用合适的容器镜像、合适的节点操作系统、节点DNS缓存NodeLocal DNSCache等方式来减少解析异常。
  - 优化域名解析请求
  - 使用合适的容器镜像
  - 避免IPVS缺陷导致的DNS概率性解析超时问题
  - 使用节点DNS缓存NodeLocal DNSCache
  - 使用合适的CoreDNS版本
- 在CoreDNS服务端，您可以通过监控CoreDNS运行状态第一时间识别DNS异常，快速定位异常根因，通过合理调整集群CoreDNS部署状态来提升集群CoreDNS高可用和QPS吞吐。
  - 监控CoreDNS运行状态
    - 监控指标
    - 运行日志

- 合理调整集群CoreDNS部署状态
  - 合理调整CoreDNS副本数
  - 合理分配CoreDNS副本运行的位置
  - 手动扩容副本数
  - 自动扩容副本数（cluster-autoscaler）
  - 基于CPU负载指标自动扩容副本数（HPA）
- 合理配置CoreDNS
  - 关闭kube-dns服务的亲和性配置
  - 关闭Autopath插件
  - 配置CoreDNS优雅退出
  - 配置Forward插件与上游VPC DNS服务器的默认协议

有关CoreDNS的更多信息，请参见CoreDNS官方文档。

优化域名解析请求

DNS域名解析请求是Kubernetes最高频的网络行为之一，其中很多请求是可以优化和避免的。您可以通过以下方式优化域名解析请求：

- （推荐）使用连接池：当一个容器应用需要频繁请求另一服务时，推荐使用连接池。连接池可以将请求上游服务的链接缓存在内存中，避免每次访问时域名解析和TCP建连的开销。
- 使用DNS缓存：
  - （推荐）当您的应用无法改造成通过连接池连接另一服务时，可以考虑在应用侧缓存DNS解析结果，具体操作，请参见使用节点DNS缓存NodeLocal DNSCache。
  - 如果NodeLocal DNSCache无法适用的，可以在容器内置NSCD（Name Service Cache Daemon）缓存。关于如何使用NSCD缓存，请参见在Kubernetes集群中使用NSCD。
- 优化*resolv.conf*文件：由于*resolv.conf*文件中ndots和search两个参数的机制作用，容器内配置域名的不同写法决定了域名解析的效率，关于ndots和search两个参数的机制详情，请参见DNS原理和配置说明。
- 优化域名配置：当容器内应用需要访问某域名时，该域名按以下原则配置，可以最大程度减少域名解析尝试次数，继而减少域名解析耗时。
  - Pod访问同命名空间的Service，优先使用`<service-name>`访问，其中`service-name`代指Service名称。
  - Pod跨命名空间访问Service，优先使用`<service-name>.<namespace-name>`访问，其中`namespace-name`代指Service所处的命名空间。
  - Pod访问集群外部域名时，优先使用FQDN类型域名访问，这类域名通过常见域名最后加半角句号（.）的方式来指定地址，可以避免`search`搜索域拼接带来的多次无效搜索，例如需要访问*www.aliyun.com*，则优先使用FQDN类型域名*www.aliyun.com.*来访问。

使用合适的容器镜像

Alpine容器镜像内置的musl libc库与标准glibc的实现存在以下差异：

- 3.3及更早版本Alpine不支持search参数，不支持搜索域，无法完成服务发现。
- 并发请求*/etc/resolv.conf*中配置的多个DNS服务器，导致NodeLocal DNSCache优化失效。
- 并发使用同一Socket请求A和AAAA记录，在旧版本内核上触发Conntrack源端口冲突导致丢包问题。

关于以上问题的更多信息，请参见musl libc。
当Kubernetes集群中部署的容器采用了Alpine作为基础镜像时，可能会因为上述musl libc特性而无法正常解析域名，建议尝试更换基础镜像，如Debian、CentOS等。

避免IPVS缺陷导致的DNS概率性解析超时问题

当集群使用IPVS作为kube-proxy负载均衡模式时，您可能会在CoreDNS缩容或重启时遇到DNS概率性解析超时的问题。该问题由社区Linux内核缺陷导致，具体信息，请参见IPVS。

您可以通过以下任意方式降低IPVS缺陷的影响：

- 使用节点DNS缓存NodeLocal DNSCache，具体操作，请参见使用节点DNS缓存NodeLocal DNSCache。
- 修改kube-proxy中IPVS UDP会话保持的超时时间，具体操作，请参见如何修改kube-proxy中IPVS UDP会话保持的超时时间？。

使用节点DNS缓存NodeLocal DNSCache

在ACK集群中部署NodeLocal DNSCache可以提升服务发现的稳定性和性能，NodeLocal DNSCache通过在集群节点上作为DaemonSet运行DNS缓存代理来提高集群DNS性能。

关于更多NodeLocal DNSCache的介绍及如何在ACK集群中部署NodeLocal DNSCache的具体步骤，请参见使用NodeLocal DNSCache。

使用合适的CoreDNS版本

CoreDNS对Kubernetes版本实现了较好的向后兼容，建议您保持CoreDNS版本为较新的稳定版本。ACK组件管理中心提供了CoreDNS的安装、升级能力，您可以关注组件管理中组件状态，若CoreDNS组件显示可升级，请尽快选择业务低峰期进行升级。

- 关于升级的具体操作，请参见CoreDNS自动升级。
- 关于CoreDNS版本的发布记录，请参见CoreDNS。

CoreDNS v1.7.0以下的版本存在风险隐患，包括且不仅限于以下：

- CoreDNS与APIServer连通性异常（例如APIServer重启、APIServer迁移、网络抖动）时，CoreDNS会因错误日志写入失败导致容器重启。更多信息，请参见Set klog's logtostderr flag。
- 启动CoreDNS时会占用额外内存，默认采用的Memory Limit在较大规模集群下可能触发OOM（Out Of Memory）问题，严重时可能导致CoreDNS Pod反复重启无法自动恢复。更多信息，请参见CoreDNS uses a lot memory during initialization phase。
- CoreDNS存在若干可能影响Headless Service域名、集群外部域名解析的问题。更多信息，请参见plugin/kubernetes: handle tombstones in default processor和Data is not synced when CoreDNS reconnects to kubernetes api server after protracted disconnection。
- 在集群节点异常情况下，部分旧版本CoreDNS默认采用的容忍策略，可能会导致CoreDNS Pod部署在异常节点上，且CoreDNS Pod无法被自动驱逐，继而导致域名解析异常。

不同Kubernetes集群推荐的CoreDNS版本如下表：

Kubernetes版本	推荐CoreDNS版本
v1.14.8以下（停止维护）	v1.6.2
v1.14.8及以上，1.20.4以下	v1.7.0.0-f59c03d-aliyun
1.20.4及以上	v1.8.4.1-3a376cc-aliyun

> **说明** Kubernetes 1.14.8以下的版本现已停止维护，请尽快升级至更高版本后，升级CoreDNS。

监控CoreDNS运行状态

CoreDNS通过标准的Prometheus接口暴露出解析结果等健康指标，第一时间发现CoreDNS服务端甚至上游DNS服务器的异常。

阿里云应用实时监控服务ARMS Prometheus监控默认内置了CoreDNS相关的指标监控和告警规则，您可以在开启Prometheus及仪表盘功能。具体操作，请参见ARMS Prometheus监控。

若您是自建Prometheus监控Kubernetes集群，可以在Prometheus观测相关指标并对以下重点指标设置告警。具体操作，请参见CoreDNS Prometheus官方文档。重点指标如下：

指标类型	指标说明	告警设置
coredns_dns_requests_total	请求次数	可针对总量进行告警，判断当前域名解析QPS是否过高。
coredns_dns_responses_total	响应次数	可针对不同状态码RCODE的响应次数进行告警，例如服务端异常SERVFAIL出现时，可进行告警。
coredns_panics_total	CoreDNS程序异常退出的次数	大于0则说明异常发生，应进行告警。
coredns_dns_request_duration_seconds	域名解析延迟	延迟过高时应进行告警。

在DNS异常发生的情况下，CoreDNS日志有助于您快速诊断异常根因。建议您开启CoreDNS域名解析日志和其SLS日志采集，具体操作，请参见分析和监控CoreDNS日志。

监控指标

运行日志

合理调整集群CoreDNS部署状态

CoreDNS应部署于您的Kubernetes集群中，默认情况下与您的业务容器运行在同样的集群节点上，注意事项如下：

- 合理调整CoreDNS副本数
- 合理分配CoreDNS副本运行的位置
- 手动扩容副本数
- 自动扩容副本数（cluster-autoscaler）
- 基于CPU负载指标自动扩容副本数（HPA）

建议您在任何情况下设置CoreDNS副本数应至少为2，且副本数维持在一个合适的水位以承载整个集群的解析。

CoreDNS所能提供的域名解析QPS与CPU消耗成正相关，开启缓存的情况下，单个CPU可以支撑10000+ QPS的域名解析请求。不同类型的业务对域名请求的QPS需求存在较大差异，您可以观察每个CoreDNS副本的峰值CPU使用量，如果其在业务峰值期间占用CPU大于一核，建议您对CoreDNS进行副本扩容。无法确定峰值CPU使用量时，可以保守采用副本数和集群节点数1：8的比值来部署，即每扩容8个集群节点，增加一个CoreDNS副本，但副本数不应大于10。针对100节点以上的集群，推荐使用节点DNS缓存NodeLocal DNSCache。具体操作，请参见使用节点DNS缓存NodeLocal DNSCache。

- 当集群节点数目长时间较为固定时，可以手动扩容副本数。具体操作，请参见手动扩容副本数。
- 如果集群节点数持续增长，可以设置自动扩容副本数。具体操作，请参见自动扩容副本数（cluster-autoscaler）。

> **说明** UDP报文缺少重传机制，在CoreDNS副本停止过程中，CoreDNS任意副本缩容或重启可能会导致接收到的UDP报文丢失，触发整个集群域名解析超时或异常。当集群节点存在IPVS UDP缺陷导致的丢包风险时，CoreDNS任意副本缩容或重启可能会导致长达五分钟的整个集群域名解析超时或异常。关于IPVS缺陷导致解析异常的解决方案，请参见IPVS缺陷导致解析异常。

建议您在部署CoreDNS副本时，应将CoreDNS副本打散在不同可用区、不同集群节点上，避免单节点、单可用区故障。CoreDNS默认配置了按节点的弱反亲和性，可能会因为节点资源不足导致部分或全部副本部署在同一节点上，如果遇到这种情况，请删除Pod重新触发其调度来调整。

CoreDNS所运行的集群节点应避免CPU、内存用满的情况，否则会影响域名解析的QPS和响应延迟。

当集群节点数长时间较为固定时，您可以通过以下命令扩容CoreDNS副本数。

```
kubectl scale --replicas={target} deployment/coredns -n kube-system
```

> **说明** 将目标副本数 `{target}` 设置成目标值。

当集群节点数不断增长时，您可以通过以下YAML部署集群水平伸缩器cluster-proportional-autoscaler动态扩容副本数量：

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: dns-autoscaler
  namespace: kube-system
  labels:
    k8s-app: dns-autoscaler
spec:
  selector:
    matchLabels:
      k8s-app: dns-autoscaler
  template:
    metadata:
      labels:
        k8s-app: dns-autoscaler
    spec:
      serviceAccountName: admin
      containers:
      - name: autoscaler
        image: registry.cn-hangzhou.aliyuncs.com/acs/cluster-proportional-autoscaler:1.8.4
        resources:
          requests:
            cpu: "200m"
            memory: "150Mi"
        command:
        - /cluster-proportional-autoscaler
        - --namespace=kube-system
        - --configmap=dns-autoscaler
        - --nodelabels=type!=virtual-kubelet
        - --target=Deployment/coredns
        - --default-params={"linear":{"coresPerReplica":64,"nodesPerReplica":8,"min":2,"max
":100,"preventSinglePointFailure":true}}
        - --logtostderr=true
        - --v=9
```

上述使用线程伸缩策略中，CoreDNS副本数的计算公式为replicas = max (ceil (cores × 1/coresPerReplica), ceil (nodes × 1/nodesPerReplica))，且CoreDNS副本数受到 `max` ， `min` 限制。线程伸缩策略参数如下。

```
{
      "coresPerReplica": 64,
      "nodesPerReplica": 8,
      "min": 2,
      "max": 100,
      "preventSinglePointFailure": true
}
```

由于HPA会频繁触发CoreDNS副本缩容，建议您不要使用容器水平扩缩容（HPA），如果您的场景下必须依赖于HPA，请参考以下基于CPU负载的策略配置：

```
---
apiVersion: autoscaling/v2beta1
kind: HorizontalPodAutoscaler
metadata:
  name: coredns-hpa
  namespace: kube-system
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: coredns
  minReplicas: 2
  maxReplicas: 10
  metrics:
  - type: Resource
    resource:
      name: cpu
      targetAverageUtilization: 50
```

> **说明** 关于HPA使用方式的更多信息，请参见容器水平伸缩（HPA）。

合理调整CoreDNS副本数

合理分配CoreDNS副本运行的位置

手动扩容副本数

自动扩容副本数（cluster-autoscaler）

基于CPU负载指标自动扩容副本数（HPA）

合理配置CoreDNS

容器服务ACK仅提供CoreDNS的默认配置，您应关注配置中的各个参数、优化其配置，以使CoreDNS可以为您的业务容器正常提供DNS服务。CoreDNS的配置非常灵活，具体操作，请参见DNS原理和配置说明和CoreDNS官方文档。

早期版本随Kubernetes集群部署的CoreDNS默认配置可能存在一些风险，推荐您按以下方式检查和优化：

- 关闭kube-dns服务的亲和性配置
- 关闭Autopath插件
- 配置CoreDNS优雅退出
- 配置Forward插件与上游VPC DNS服务器的默认协议

亲和性配置可能导致CoreDNS不同副本间存在较大负载差异，建议按以下步骤关闭：

- **控制台操作方式**
  i.
  ii. 在控制台左侧导航栏中，单击**集群**。
  iii. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。

iv. 在集群管理页左侧导航栏中，选择**网络 > 服务**。

v. 在kube-system命名空间下，单击服务**kube-dns**右侧的**查看YAML**。

- 如果发现sessionAffinity字段为 `None` ，则无需进行以下步骤。
- 如果发现sessionAffinity为 `ClientIP` ，则进行以下步骤。

vi. 删除sessionAffinity、sessionAffinityConfig及所有子键，然后单击**更新**。

```
#删除以下所有内容。
sessionAffinity: ClientIP
  sessionAffinityConfig:
    clientIP:
      timeoutSeconds: 10800
```

vii. 再次单击服务**kube-dns**右侧的**查看YAML**，校验sessionAffinity字段是否为 `None` ，为 `None` 则Kube-DNS服务变更成功。

- **命令行操作方式**

i. 执行以下命令查看kube-dns服务配置信息。

```
kubectl -n kube-system get svc kube-dns -o yaml
```

- 如果发现sessionAffinity字段为 `None` ，则无需执行以下步骤。
- 如果发现sessionAffinity为 `ClientIP` ，则执行以下步骤。

ii. 执行以下命令打开并编辑名为kube-dns的服务。

```
kubectl -n kube-system edit service kube-dns
```

iii. 删除sessionAffinity相关设置（sessionAffinity、sessionAffinityConfig及所有子键），并保存退出。

```
#删除以下所有内容。
sessionAffinity: ClientIP
  sessionAffinityConfig:
    clientIP:
      timeoutSeconds: 10800
```

iv. 修改完成后，再次运行以下命令查看sessionAffinity字段是否为 `None` ，为 `None` 则Kube-DNS服务变更成功。

```
kubectl -n kube-system get svc kube-dns -o yaml
```

部分早期版本的CoreDNS开启了Autopath插件，该插件在一些极端场景下会导致解析结果出错，请确认其是否处于开启状态，并编辑配置文件将其关闭。更多信息，请参见#3765。

> **说明** 关闭Autopath插件后，客户端发起的域名解析请求QPS最高会增加3倍，解析单个域名耗时最高增加3倍，请关注CoreDNS负载和业务影响。

1. 执行 `kubectl -n kube-system edit configmap coredns` 命令，打开CoreDNS配置文件。
2. 删除 `autopath @kubernetes` 一行后保存退出。
3. 检查CoreDNS Pod运行状态和运行日志，运行日志中出现 `reload` 字样后说明修改成功。

> **说明** CoreDNS刷新配置过程中，可能会占用额外内存。修改CoreDNS配置项后，请观察Pod运行状态，如果出现Pod内存不足的情况，请及时修改CoreDNS Deployment中容器内存限制，建议将内存调整至2 GB。

- **控制台操作方式**
  i.
  ii. 在控制台左侧导航栏中，单击**集群**。
  iii. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
  iv. 在集群管理页左侧导航栏中，选择**配置管理 > 配置项**。
  v. 在kube-system命名空间下，单击配置项*coredns*右侧的**YAML编辑**。
  vi. 参考下列的YAML文件，保证health插件开启，并调整lameduck参数为 `15s` ，然后单击**确定**。

```
.:53 {
        errors
        #health插件在不同的CoreDNS版本中可能有不同的设置情形。
        #情形一：默认未启用health插件。
        #情形二：默认启用health插件，但未设置lameduck时间。
        health
        #情形三：默认启用health插件，设置了lameduck时间为5s。
        health {
            lameduck 5s
        }
        #对于以上三种情形，应统一修改成以下，调整lameduck参数为15s。
        health {
            lameduck 15s
        }
        #其它插件不需要修改，此处省略。
    }
```

  如果CoreDNS Pod正常运行则说明CoreDNS优雅退出的配置更新成功。如果CoreDNS Pod出现异常，可以通过查看Pod事件及日志定位原因。

- **命令行操作方式**
  i. 执行以下命令打开CoreDNS配置文件。

```
kubectl -n kube-system edit configmap coredns
```

  ii. 参考下列的YAML文件，保证 `health` 插件开启，并调整lameduck参数为 `15s` 。

```
.:53 {
        errors
        #health插件在不同的CoreDNS版本中可能有不同的设置情形。
        #情形一：默认未启用health插件。
        #情形二：默认启用health插件，但未设置lameduck时间。
        health
        #情形三：默认启用health插件，设置了lameduck时间为5s。
        health {
            lameduck 5s
        }
        #对于以上三种情形，应统一修改成以下，调整lameduck参数为15s。
        health {
            lameduck 15s
        }
        #其它插件不需要修改，此处省略。
    }
```

iii. 修改CoreDNS配置文件后保存退出。
如果CoreDNS正常运行则说明CoreDNS优雅退出的配置更新成功。如果CoreDNS Pod出现异常，可以通过查看Pod事件及日志定位原因。

NodeLocal DNSCache采用TCP协议与CoreDNS进行通信，CoreDNS会根据请求来源使用的协议与上游DNS服务器进行通信。因此默认情况下，来自业务容器的集群外部域名解析请求会依次经过NodeLocal DNSCache、CoreDNS，最终以TCP协议请求VPC内DNS服务器，即ECS上默认配置的100.100.2.136和100.100.2.13两个 IP。

VPC内DNS服务器对TCP协议支持有限，如果您使用了NodeLocal DNSCache，您需要修改CoreDNS配置，让其总是优先采用UDP协议与上游DNS服务器进行通信，避免解析异常。建议您使用以下方式修改CoreDNS配置文件，即修改命名空间kube-system下名为*coredns*的Deployment。具体操作，请参见管理配置项。在 `forward` 插件中指定请求上游的协议为 `perfer_udp` ，修改之后CoreDNS会优先使用UDP协议与上游通信。修改方式如下所示：

```
#修改前
forward . /etc/resolv.conf
#修改后
forward . /etc/resolv.conf {
  prefer_udp
}
```

关闭Autopath插件

配置CoreDNS优雅退出

配置Forward插件与上游VPC DNS服务器的默认协议

相关文档

- DNS概述
- DNS原理和配置说明
- 使用NodeLocal DNSCache

3.7. Nginx Ingress最佳实践

Nginx Ingress Controller是部署于集群内部的Ingress控制器，可以为您提供性能更好，且定制性更高的使用方式。ACK集群提供的Nginx Ingress Controller在社区版本的基础上，整合了阿里云产品的一系列功能，提供了更加便捷的使用体验。由于Nginx Ingress Controller部署在集群内部，因此它的稳定性与使用时的配置、当前集群状态密切相关。本文介绍Nginx Ingress的最佳实践，您可以参考以下最佳实践，对集群内的Ingress Controller进行配置，获得最佳的使用效果。

背景信息

Nginx Ingress最佳实践包含以下内容：

- Nginx Ingress Controller的性能和稳定性
  - 使用合适的副本数和资源限制
  - 使用独占节点保证Nginx Ingress性能与稳定性
  - Nginx Ingress性能调优
  - 按照负载配置HPA进行自动扩容
- 提升Nginx Ingress Controller的可观测性
  - 使用SLS和ARMS Prometheus服务提升可观测性
  - 使用AHAS流量防护实现流控
- Nginx Ingress Controller进阶功能
  - 使用多套Nginx Ingress Controller
  - 在集群内部访问Nginx Ingress Controller
  - 使用WAF或透明WAF
  - 通过Nginx Ingress Controller进行应用的蓝绿或灰度发布
  - 通过Nginx Ingress Controller代理非HTTP请求

Nginx Ingress Controller的性能和稳定性

使用合适的副本数和资源限制

默认情况下，通过集群创建或从组件中心安装的Nginx Ingress Controller的副本数为2，您可以根据业务的实际需要进行调整。

在部署Nginx Ingress Controller时，请确保Nginx Ingress Controller分布在不同的节点上，避免不同Nginx Ingress Controller之间资源的抢占和单点故障。您也可以为其使用独占节点来保证性能与稳定性，具体操作，请参见使用独占节点保证Nginx Ingress性能与稳定性。同时建议您不要为Nginx Ingress Controller设定资源限制，避免OOM所带来的流量中断。如果确实有限制的需要，建议资源限制CPU不低于1000 Milicore（YAML配置里格式为 `1000m` ）和内存不低于2 GiB。

使用独占节点保证Nginx Ingress性能与稳定性

- 如果您对Ingress Controller有较高稳定性要求，可以为其分配独占节点以避免资源的抢占。具体操作，请参见部署高可靠Ingress Controller。
- 对于高负载场景，您也可以通过设置Ingress Controller来支撑高负载应用。具体操作，请参见部署高负载场景的Nginx Ingress Controller。

Nginx Ingress性能调优

Nginx Ingress Controller性能调优主要分为系统参数调优和Nginx参数调优：

- 系统参数调优：阿里云上的操作系统已经默认优化了一些常见参数，其他还需要调优的系统参数主要包括系统最大Backlog数和可用端口的最大范围。系统参数调优后可以保证Nginx处理高并发请求的能力，以及在连接后端时不会因为端口耗尽而失败。

- Nginx参数调优：
  - 调整单个Worker的最大连接数：Nginx参数主要可以调整单个Worker的最大连接数来保证Nginx Ingress Controller处理高并发请求的能力。
  - 增加连接超时时间：Nginx Ingress Controller与Nginx默认表现不同，会使用长连接对后端业务Pod发起请求，因此Nginx参数还需要增加连接超时时间来让单条连接能够处理更多的请求，减少连接创建时的开销。
  - 设置长连接超时时间：请您确保后端的业务长连接的超时时间不低于Nginx Ingress Controller的连接超时时间（ACK集群中默认为900s）。

Nginx Ingress组件默认已经置入了相关的调优项，在一般场景中能够达到较优的表现。如果您有特殊需要，可以通过ConfigMap中的相关字段进一步优化系统参数和Nginx参数。关于ConfigMap的更多信息，请参见ConfigMaps。

按照负载配置HPA进行自动扩容

一般情况下，Nginx Ingress Controller已经有足够的能力应对业务的突发流量。如果在高负载情况下仍不满足您的要求，也可以配置HPA对Ingress Controller进行扩容。具体操作，请参见容器水平伸缩（HPA）。

> **注意** 在Pod扩缩容时可能会导致部分业务连接的中断情况，因此请谨慎配置。

配置的YAML示例如下所示：

```
apiVersion: autoscaling/v2beta1
kind: HorizontalPodAutoscaler
metadata:
  name: nginx-ingress-controller-hpa
  namespace: kube-system
spec:
   scaleTargetRef:
     apiVersion: apps/v1
     kind: Deployment
     name: nginx-ingress-controller
   minReplicas: 2
   maxReplicas: 5
   metrics:
    - type: Resource
      resource:
        name: cpu
        targetAverageUtilization: 50
```

提升Nginx Ingress Controller的可观测性

使用SLS和ARMS Prometheus服务提升可观测性

Nginx Ingress Controller提供了基于SLS日志以及Prometheus的监控大盘，帮助您更好地了解您的业务的流量情况。

- SLS日志：
  - 创建集群时，如果您已经选中了启用日志组件及Ingress Dashboard，您就可以直接通过容器服务管理控制台**网络 > 路由**页面的**Ingress概览**中查看到基于SLS日志服务提供的大盘服务，您也可以直接在**运维管理 > 日志中心**查看到Nginx Ingress Controller产生的相关日志。具体操作，请参见Nginx Ingress访问日志分析与监控。

- 如果您在创建时没有选中日志以及Ingress的相关选项，也可以手动配置日志收集组件和规则。具体操作，请参见Nginx Ingress访问日志分析与监控。关于监控的更多信息，请参见Ingress Dashboard监控。

- ARMS Prometheus监控：ARMS Prometheus监控可以在创建集群时选择安装，或者在创建集群后通过**运维管理 > Prometheus监控**安装或查看。具体操作，请参见ARMS Prometheus监控。

  > **说明** 使用ARMS Prometheus监控时，请为集群中的Ingress资源添加 `host` 字段，否则在默认情况下，该Ingress的部分metrics将会缺失。您也可以通过在Nginx Ingress Controller的Deployment中修改 `controller` 启动参数，加入 `--metrics-per-host=false` 来解决该问题。

使用AHAS流量防护实现流控

Nginx Ingress Controller整合了阿里云AHAS流量防护的产品功能，可以实现更加精细的流量控制。当有请求流量时，您可以在AHAS控制台中查看Nginx网关请求的实时QPS和RT等数据。具体开启的操作步骤，请参见使用Nginx-Ingress-sentinel实现流控。

Nginx Ingress Controller进阶功能

使用多套Nginx Ingress Controller

在应用中，您可能会因为内外网隔离等需要，在集群中部署多套Nginx Ingress Controller。在容器服务管理控制台除去通过组件中心安装之外，您也可以通过应用市场部署额外的Nginx Ingress Controller。具体操作，请参见部署多个Ingress Controller。

在集群内部访问Nginx Ingress Controller

在集群中，LoadBalancer Service的外部地址（Nginx Ingress Controller的公网 IP）通常会被Iptables或IPVS拦截并转发。当 `externalTrafficPolicy` 为 `Local` ，且节点上没有对应Nginx Ingress Pod时，会发生网络不通的问题。ACK集群中的Nginx Ingress Controller默认使用了Local模式的LoadBalancer Service，因此在集群内访问Nginx Ingress Controller绑定的CLB地址时，可能会出现网络不通的情况。

建议您在集群内部有访问Nginx Ingress Controller（通过公网IP或绑定公网IP的域名）的需要时，使用Service ClusterIP或内部域名（ `nginx-ingress-lb.kube-system` ）访问。同时请避免在Nginx Ingress Controller中对自身进行访问，也可能会因为Hairpin问题导致网络不通。关于该问题的解决方案，请参见Kubernetes集群中访问LoadBalancer暴露出去的SLB地址不通。

使用WAF或透明WAF

为了阻挡恶意请求，您可以在Web应用防火墙控制台或应用型负载均衡ALB控制台为集群Nginx Ingress Controller所使用的CLB开启WAF或透明WAF功能。在开启HTTPS端口上的WAF或透明WAF功能时，需要在控制台上配置所使用的证书。这种情况下，可能会出现以下问题：

- TLS请求会在WAF或透明WAF上进行截断，因此集群内通过Secret配置的证书将不会被暴露在公网出口上。
- 在集群内通过CLB IP或Service ClusterIP访问443端口可能不会经过WAF或透明WAF，导致证书返回错误。
- 在开启WAF或透明WAF的情况下，Nginx Ingress Controller将默认无法获得真实的客户端IP。您可以通过在ConfigMap中（通过组件管理安装的Nginx Ingress Controller默认为kube-system命名空间下的nginx-configuration）添加以下内容，配置启用Nginx的Realip模块，使用 `X-Forwarded-For` 头作为真实客户端地址。

```
use-forwarded-headers: "true" # 0.30.0及更旧版本使用该选项。
enable-real-ip: "true" # 0.44.0及更新版本使用该选项。
proxy-real-ip-cidr: <您从WAF获取到的回源IP段>
```

通过Nginx Ingress Controller进行应用的蓝绿或灰度发布

您可以通过容器服务管理控制台的灰度发布或手动添加Annotation的方式来使用Nginx Ingress Controller提供的灰度功能，具体操作，请参见通过Nginx Ingress实现灰度发布和蓝绿发布。

> **注意** 请确保需要灰度所使用的服务（包括原服务和灰度服务）没有同时被灰度Ingress以外的Ingress资源引用。否则，可能会出现灰度规则的冲突，从而引发流量路由错误。

通过Nginx Ingress Controller代理非HTTP请求

Nginx Ingress Controller 默认使用HTTP协议连接到后端服务，但同时提供了对多种后端协议的支持，其中比较常用的协议有WebSocket、HTTPS和gRPC。关于支持的后端协议具体类型，请参见Backend Protocol。

- WebSocket：Nginx Ingress Controller提供了对WebSocket的原生支持，您不需要进行任何配置即可转发WebSocket连接。如果您有持续较长的WebSocket连接，可以通过Annotation适当地调整后端连接的超时时间，防止业务因为超时而断连。关于调整的具体操作，请参见Custom timeouts。
- HTTPS：针对使用HTTPS的后端服务，您可以在Ingress中添加 `nginx.ingress.kubernetes.io/backend-protocol:"HTTPS"` 的Annotation切换为HTTPS连接。
- gRPC：gRPC仅支持通过TLS端口访问。因此，请确保您的业务通过Nginx Ingress Controller访问gRPC服务时，使用的是加密的TLS端口。关于配置gRPC的具体操作，请参见通过Ingress Controller实现gRPC服务访问。

相关文档

- Ingress概述
- Ingress FAQ

4.安全

4.1. 安全概述

本系列文档旨在为企业安全管理运维人员提供使用阿里云ACK产品时可以参考的安全最佳实践，对使用ACK集群中您可能面临的安全挑战做了基本的分类，每小类中包含了对安全问题的基本介绍和基于ACK的安全最佳实践。

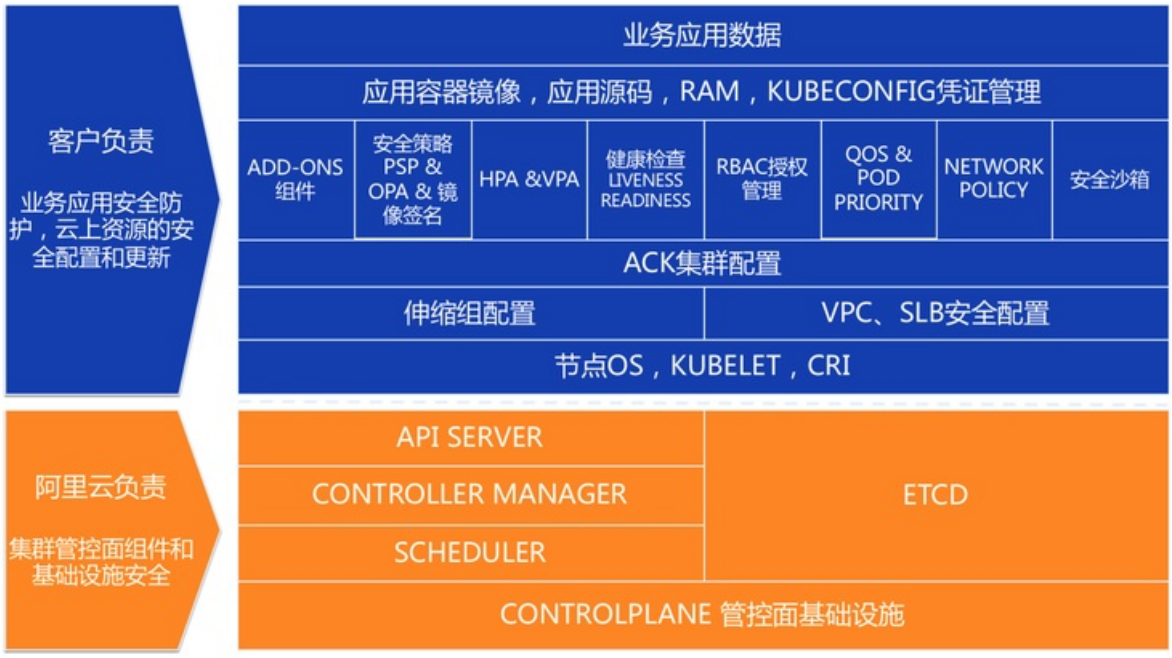
目前包括的相关分类如下：

- 身份认证和访问控制
- Pod安全
- Runtime安全
- 供应链安全
- 数据加密和密钥管理
- 网络安全
- 日志和审计
- 多租户安全
- 主机安全

理解责任共担模型

安全合规在ACK集群托管架构下遵循责任共担原则，其中ACK容器服务负责集群管控面组件（包括Kubernetes Master组件和etcd）以及集群服务相关阿里云基础设施的默认安全性。

ACK需要基于Kubernetes和阿里云提供的安全能力，负责诸如RAM访问控制、Pod安全、运行时安全、网络安全等方向上的安全能力实施，同时基于阿里云提供的修复方案指导，及时更新集群节点OS、Kubernetes组件等相关安全漏洞，您可以参考该系列文章中的推荐方案进行相关方向上的安全加固。



云服务商侧

对于云服务商，首先需要依托于云平台自身的安全能力，构建安全稳定的容器基础设施平台，并且面向容器应用从构建，部署到运行时刻的全生命周期构建对应的安全防护手段。整个安全体系的构建需要遵循如下基本原则：

- **保证容器管控平台基础设施层的默认安全**
  容器平台基础设施层承载了企业应用的管控服务，是保障业务应用正常运行的关键，容器平台的安全性是云服务商应该格外关注的。
  - 完备的平台安全能力：首先云服务商自身基础设施的安全性是容器平台是否安全的基础，例如VPC的安全配置能力、SLB的访问控制、DDoS能力和账号系统对云资源的访问控制能力等都是平台侧面向企业应用需要提供的基础安全能力。
  - 版本更新和漏洞应急响应机制：虚拟机OS的版本更新和漏洞补丁的安装能力也是保证基础设施安全的基本防护措施，除此之外如K8s等容器相关开源社区的风险漏洞，都可能成为恶意攻击者首选的攻击路径，需要厂商提供漏洞的分级响应机制并提供必要的版本升级能力。
  - 平台的安全合规性：这也是很多金融企业和政府部门应用上云的硬性前提条件。云服务商需要基于业界通用的安全合规标准，保证服务组件配置的默认安全性，同时面向平台和安全审计人员，提供完备的审计机制。

- **面向容器应用侧提供纵深防御能力**
  云服务商不仅要在自身管控侧建立完善的安全武装，同时也需要面向业务应用负载，提供适合云原生场景下容器应用的安全防护手段，帮助终端在应用生命周期各阶段都能有对应的安全治理方案。由于云原生具有动态弹性的基础设施，分布式的应用架构和创新的应用交付运维方式等特点，这就要求云服务商能够结合自身平台的基础安全能力，将云原生能力特性赋能于传统的安全模型中，构建面向云原生的新安全体系架构。

企业安全侧

企业的安全管理和运维人员，需要理解云上安全的责任共担模型边界，即企业自身需要承担的安全责任。云原生微服务架构下企业应用在IDC和云上进行部署和交互，传统的网络安全边界已经不复存在，企业应用侧的网络安全架构需要遵循零信任安全模型，基于认证和授权重构访问控制的信任基础。对于企业安全管理人员可以根据以下方向加固企业应用生命周期中的生产安全：

- **保证应用制品的供应链安全**
  云原生的发展使得越来越多的大规模容器应用开始在企业生产环境上部署，也丰富了云原生应用制品的多样性，例如容器镜像和Helm charts都是常见的制品格式。对于企业，制品供应链环节的安全性是企业应用生产安全的源头，一方面需要在应用构建阶段保证制品的安全性；另一方面需要在制品入库、分发和部署时建立对应的访问控制、安全扫描、审计和准入校验机制，保证制品源头的安全性。
- **权限配置和凭证下发遵循权限最小化原则**
  基于统一的身份标识体系进行认证授权是在零信任安全模型下构建访问控制能力的基础。对于企业安全管理人员，需要利用云服务商提供的访问控制能力，结合企业内部的权限账号体系，严格遵循权限最小化原则配置对云上资源和容器侧应用资源的访问控制策略；另外严格控制资源访问凭证的下发，对于可能造成越权攻击行为的已下发凭证要及时吊销。另外要避免容器应用模板中配置过大权限的容器，确保最小化攻击面。
- **关注应用数据和应用运行时刻安全**
  应用的成功部署上线并不意味着安全工作的结束。除了配置完备的资源请求审计外，安全管理运维人员还需要利用厂商提供的运行时刻监控告警和事件通知等机制，保持对容器应用运行时安全的关注，及时发现安全攻击事件和安全隐患。对于企业应用自身依赖的敏感数据（例如数据库密码、应用证书私钥等）需要根据应用数据的安防等级采用对应的密钥加密机制，利用云上的密钥管理方案和落盘加密、机密计算等能力，保证数据在传输和落盘链路上的数据安全性。
- **及时修复安全漏洞和进行版本更新**

无论是虚拟机系统，容器镜像或是容器平台自身的安全漏洞，都有可能被恶意攻击者利用成为入侵应用内部的跳板，企业安全管理运维人员需要根据云服务商推荐的指导方案进行安全漏洞的修复和版本更新（例如K8s集群版本、应用镜像版本等）。此外企业要负责内部员工的安全培训工作，居安思危，提升安全防护意识也是企业安全生产的基础要务。

4.2. 身份认证和访问控制

本文主要介绍身份认证和访问控制包含的两部分基本功能：认证和授权。认证即判断用户是否为能够访问集群的合法用户，授权负责管理指定用户身份对集群资源的访问权限。

ACK集群的认证和授权

Kubernetes使用X.509证书、Bearer Tokens、身份验证代理或者OIDC认证等方式来对API请求进行身份验证。关于ACK原生、访问ACK集群的两种方式，请参见客户端证书、服务账户令牌。

客户端证书Kubeconfig可以通过阿里云控制台、ACK的Openapi接口等获取。更多信息，请参见获取集群KubeConfig接口和使用ServiceAccount Token访问Kubernetes集群。

容器服务授权包含RAM授权和RBAC授权两部分。当您需要对集群进行可见性、扩缩容、添加节点等操作时，需要进行自定义RAM授权策略。目前RAM主要支持RAM系统策略授权和RAM自定义策略授权两种授权方式。当RAM用户需要操作指定集群内K8s资源时（例如，获取集群Pod和Node信息），需要在的授权管理页面对指定RAM用户进行数据平面资源的RBAC授权。更多信息，请参见授权概述。

- **使用临时Kubeconfig和Apiserver认证**
  ACK默认返回的集群访问凭证Kubeconfig中的证书有效期是3年，若Kubeconfig发生泄露，攻击者可以直接获得集群Apiserver的访问权限，默认的ServiceAccount Token同样是一种长期有效的静态凭据，如果泄露攻击者同样可以获得该ServiceAccount绑定的集群访问权限，直到该ServiceAccount被删除。
  因此请您及时吊销可能发生泄露的已下发Kubeconfig凭证。具体操作，请参见吊销集群的KubeConfig凭证。
  关于ACK集群的访问凭证，请参见生成临时的KubeConfig。
- **遵循权限最小化原则访问阿里云资源**
  如果某个用户需要访问ACK集群，无需为用户分配访问其他云产品资源的权限，保证用户访问云产品资源权限最小化。可以通过配置RAM和RBAC为该用户授予访问ACK集群的权限。
- **在创建RoleBindings和ClusterRoleBindings时使用最低访问权限**
  和授予访问云资源的权限一样，在创建 `RoleBindings` 和 `ClusterRoleBindings` 时，同样需要遵循权限最小化原则。应该尽量避免在创建 `Role` 和 `ClusterRole` 时使用 `["*"]` ，应该明确到具体的 `Verbs` 。如果您不确定要分配哪些权限，请考虑使用类似audit2rbac之类的工具，根据 `Kubernetes` 审计日志中观察到的API调用自动生成角色和绑定。
- **控制ACK集群Endpoint的访问范围**
  默认情况下，当您配置ACK集群时，集群的Apiserver Endpoint仅为内网访问，即仅可以集群和VPC内部访问。关于创建出来的Service能够实现公网访问并控制Endpoint的访问范围，请参见通过Annotation配置负载均衡。
- **定期对集群的访问权限进行审计**
  对集群的访问权限可能会随着时间的推移而改变。需要集群的安全管理运维人员定期审核RAM配置和配置RAM用户RBAC权限来查看指定RAM用户是否被分配了合适的权限。或关于使用开源工具来检查绑定到特定服务帐户、用户或组的RBAC权限，并及时收敛不符合需要的过大权限配置。请参见kubectl-who-can和rbac-lookup。

Pods认证

在Kubernetes集群中运行的某些应用程序需要调用Kubernetes API的权限才能正常运行。ACK CCM组件需要对Nodes资源进行增删改查权限。

- **Kubernetes Service Accounts**

服务帐户（Service Accounts）是一种特殊类型的对象，允许您将Kubernetes RBAC角色分配给Pod。K8s会为集群中的每个命名空间自动创建一个默认服务帐户。当您在不引用特定服务帐户的情况下将Pod部署到命名空间时，该命名空间的默认服务帐户将自动分配给Pod和Secret，即该服务帐户的服务帐户 (JWT) 令牌，将挂载到Pod作为一个卷位于*/var/run/secrets/kubernetes.io/serviceaccount*。解码该目录中的服务帐户令牌将显示以下元数据：

```
{
  "iss": "kubernetes/serviceaccount",
  "kubernetes.io/serviceaccount/namespace": "default",
  "kubernetes.io/serviceaccount/secret.name": "default-token-vpc2x",
  "kubernetes.io/serviceaccount/service-account.name": "default",
  "kubernetes.io/serviceaccount/service-account.uid": "1d059c50-0818-4b15-905d-bbf05e1d**
**",
  "sub": "system:serviceaccount:default:default"
}
```

默认服务帐户对Kubernetes API具有以下权限：

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  annotations:
    rbac.authorization.kubernetes.io/autoupdate: "true"
  labels:
    kubernetes.io/bootstrapping: rbac-defaults
  name: system:discovery
rules:
- nonResourceURLs:
  - /api
  - /api/*
  - /apis
  - /apis/*
  - /healthz
  - /openapi
  - /openapi/*
  - /version
  - /version/
  verbs:
  - get
```

此角色授予未经身份验证和经过身份验证的用户读取API信息，并被视为可公开访问的安全角色。
当在Pod中运行的应用程序调用Kubernetes API时，需要为Pod分配一个服务帐户，明确授予其调用API的权限。绑定到服务帐户的Role或 ClusterRole应仅限于应用程序运行所需的API资源和方法，遵循权限最小化原则。要使用非默认服务帐户，只需将Pod的 `spec.serviceAccountName` 字段设置为您希望使用的服务帐户。有关创建服务帐户的其他信息，请参见ServiceAccount permissions。

- **部署服务账户令牌卷投影**
  您可以在容器服务ACK中开启服务账户令牌卷投影功能以降低在Pod中使用ServiceAccount遇到的安全性问题，此功能可使得Kubelet支持基于Pod粒度的Token签发，并且支持Token audience和过期时间的配置。当Token即将过期（过期时间的80%或者Token存在大于24小时），Kubelet支持对Token的自动刷新。具体操作，请参见部署服务账户令牌卷投影。
- **限制节点对实例元数据API的访问**

ECS实例元数据概述包含了ECS实例在阿里云系统中的信息，您可以在运行中的实例内方便地查看实例元数据，并基于实例元数据配置或管理实例。实例元数据中能够查询到用户使用的云资源和用户数据等敏感信息，若不加限制的暴露给节点，在多租等场景下很容易被攻击者利用。因此，需要限制Pod对于meta-server的访问，可以通过Network Policy限制Pod对于meta-server的访问。可以通过如下规则使用 `podSelector` 控制指定Pod对meta-server的访问：

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-metadata-access
  namespace: example
spec:
  podSelector:
    matchLabels:
      app: myapp
  policyTypes:
  - Egress
  egress:
  - to:
    - ipBlock:
        cidr: 100.100.100.200/32
```

更多信息，请参见使用网络策略Network Policy。

- **禁止Pod对Service Account令牌的自动挂载**
  可通过以下两种方式取消Pod对Service Account令牌的自动挂载。
  方式一：通过修改YAML将应用程序PodSpec字段中的 `automountServiceAccountToken` 属性设置为 `false` 。

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  serviceAccountName: build-robot
  automountServiceAccountToken: false
  ...
```

  方式二：执行以下命令，patch命名空间中的默认服务帐户，将 `automountServiceAccountToken` 属性设置为 `false` 。

```
kubectl patch serviceaccount default -p $'automountServiceAccountToken: false'
```

- **为每个应用分配独立的Service Account**
  每个应用程序都应该有自己专用的服务帐户，便于不同应用之间的授权管理，实现应用维度的细粒度权限隔离。关于为应用分配独立的服务账户，请参见为Pod配置服务账户。
- **以非root用户运行应用**
  默认情况下，容器以root身份运行。这允许容器中的进程对容器中的文件进行任意的读取和写入，但以root身份运行容器并不是最佳安全实践。作为替代方案，请考虑
  将 `spec.securityContext.runAsUser` 属性添加到 `PodSpec` 。 `runAsUser` 的值为任意值。
  在以下示例中，Pod中的所有进程都将在 `runAsUser` 字段中指定的用户ID下运行。

```
apiVersion: v1
kind: Pod
metadata:
  name: security-test
spec:
  securityContext:
    runAsUser: 1000
    runAsGroup: 3000
  containers:
  - name: sec-test
    image: busybox
    command: [ "sh", "-c", "sleep 1h" ]
```

此时，对于容器中需要root权限才能访问的文件，容器中的进程则无法访问。如果您更新容器的 `securityContext` 以包含 `fsgroup=65534 [Nobody]` ，将允许读取容器中的文件。

```
spec:
  securityContext:
    fsGroup: 65534
```

- **遵循权限最小化原则分配应用对云上阿里云资源的访问权限**
  避免为应用所在节点分配不必要的RAM权限，遵循权限最小化原则细粒度定制应用所需的RAM策略，避免在权限策略中出现 `["*"]` 等可能扩大访问权限的配置。

相关文档

- 自定义RAM授权策略
- 配置RAM用户或RAM角色RBAC权限
- ECS实例元数据概述

4.3. Pod安全

本文主要介绍如何正确配置Pod安全来加固集群的安全性，防止集群成为攻击者利用的目标。

防止进程逃离容器边界并获得权限

作为使用Kubernetes的开发或者运维人员，您的主要关注点应该是如何防止在容器中运行的进程逃离容器的隔离边界并获得对宿主机的访问权限。这样做主要有两个原因。首先，容器内运行的进程默认在 `[Linux] root` 用户的上下文中运行。尽管 `root` 在容器中的操作部分受到Docker分配给容器的 `Linux capabilities` 的限制，但这些默认权限可能允许攻击者提权或者访问到宿主机的敏感信息，包括 `Secrets` 和 `ConfigMap` 等敏感资源。下面是分配给Docker容器的默认 `capabilities` 列表。更多信息，请参见capabilities(7) — Linux manual page。

`cap_chown, cap_dac_override, cap_fowner, cap_fsetid, cap_kill, cap_setgid, cap_setuid, cap_setpcap, cap_net_bind_service, cap_net_raw, cap_sys_chroot, cap_mknod, cap_audit_write, cap_setfcap` .

应尽可能避免使用以特权身份（ `privileged` ）运行Pod，因为其拥有与宿主机上root关联的所有Linux `capabilities` 。

其次，所有Kubernetes工作节点都使用一种称为节点授权者的授权模式。节点授权者授权所有源自Kubelet的API请求，并允许节点执行以下操作：

读操作：

- Services

- Endpoints
- Nodes
- Pods
- 与绑定到Kubelet节点的Pod相关的Secrets、Configmaps、PV和PVC

写操作：

- 节点和节点状态（启用 `NodeRestriction` 准入插件以限制Kubelet修改自己的节点）
- Pods和Pod状态（启用 `NodeRestriction` 准入插件以限制Kubelet修改绑定到自身的Pod）
- Events

Auth相关操作：

- 对用于TLS引导的 `CertificateSigningRequest (CSR)` API的读/写访问权限
- 能够为委托的身份验证/授权检查创建 `TokenReview` 和 `SubjectAccessReview`

ACK集群默认使用节点限制准入控制器。该控制器仅允许节点修改绑定到节点的一组有限节点属性和Pod对象，但是设法访问主机的攻击者仍然能够从Kubernetes API收集环境中的敏感信息。更多信息，请参见节点限制准入控制器。

PSP（PodSecurityPolicy）在Kubernetes1.21版本中被设置为Deprecated状态，在使用中的用户可以在1.25版本前有一个替换的缓冲期。社区正在计划通过新的内置准入控制器方案来取代PSP，ACK容器服务也将通过基于OPA的策略治理方案逐步替换正在使用的PSP。

Pod安全配置建议

- **限制容器以特权模式运行**
  如前所述，以特权身份运行的容器继承了分配给主机上root的所有Linux capabilities。大多数场景下，容器并不是必须这些权限才能保证业务运行。您可以通过创建Pod安全策略来拒绝容器配置为以特权模式运行的Pod。您可以将Pod安全策略视为Pod在创建之前必须满足的一组安全约束。Kubernetes的Pod安全策略（Pod Security Policy）准入控制组件会基于您定义的规则验证在集群上创建和更新Pod的请求。如果创建或更新Pod的请求不符合定义的规则，系统将拒绝该请求并返回错误。
  ACK集群默认启用PSP准入控制插件，并配置一个名为 `ack.privileged` 的Pod安全策略。这个安全策略将放行任意类型的Pod。作为安全最佳实践，我们建议您根据权限最小化原则以集群命名空间为维度细粒度管理应用对应的PSP策略，例如约束指定命名空间下禁止部署特权容器，只能使用只读的根文件系统，或者只能挂载指定范围的host目录。一个PSP策略模板如下：

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
    name: restricted
    annotations:
        seccomp.security.alpha.kubernetes.io/allowedProfileNames: 'docker/default,runtime
/default'
        apparmor.security.beta.kubernetes.io/allowedProfileNames: 'runtime/default'
        seccomp.security.alpha.kubernetes.io/defaultProfileName:  'runtime/default'
        apparmor.security.beta.kubernetes.io/defaultProfileName:  'runtime/default'
spec:
    privileged: false
    # 防止升级到根目录所需。
    allowPrivilegeEscalation: false
    requiredDropCapabilities:
    - ALL
    # 允许核心卷类型。
    volumes:
    - 'configMap'
    - 'emptyDir'
    - 'projected'
    - 'secret'
    - 'downwardAPI'
    # 假设集群管理员设置的PersistentVolume可以安全使用。
    - 'persistentVolumeClaim'
    hostNetwork: false
    hostIPC: false
    hostPID: false
    runAsUser:
        # 要求容器在没有root权限的情况下运行。
        rule: 'MustRunAsNonRoot'
    seLinux:
        # 此策略假定节点使用AppArmor而不是SELinux。
        rule: 'RunAsAny'
    supplementalGroups:
        rule: 'MustRunAs'
        ranges:
        # 禁止添加根组。
        - min: 1
          max: 65535
    fsGroup:
        rule: 'MustRunAs'
        ranges:
        # 禁止添加根组。
        - min: 1
          max: 65535
    readOnlyRootFilesystem: false
```

上述策略实例可防止Pod以特权或特权提升的模式运行。还限制了可以挂载的卷类型和可以添加的根补充组。您可以根据需要进一步加强策略约束，更多信息，请参见Pod Security Policies。

- **限制应用程序进程以root身份运行**

默认情况下容器都以 root 身份运行。如果攻击者能够利用应用程序中的漏洞并获得正在运行的容器的 shell 访问权限，这可能会出现安全问题。您可以通过多种方式缓解此类风险。一种方式是，通过从容器镜像中删除 shell 。另一种方式是，将 USER 指令添加到您的 Dockerfile 或以非 root 用户身份在Pod中运行容器。Kubernetes podSpec 在 spec.securityContext 下包含和 runAsUser 和 runAsGroup 两个字段，允许您指定运行应用程序的用户和组。您可以通过创建Pod安全策略来强制使用这些字段。更多信息，请参见Users and groups。

- **禁止以Docker in Docker的方式运行容器或者在容器中挂载Docker.sock**
  使用嵌套容器的方式或者挂载Docker.sock可以方便地在Docker容器中构建/运行容器镜像，但您将节点的控制权交给了在容器中运行的进程。关于在Kubernetes上构建容器镜像，请参见使用企业版实例构建镜像、Kaniko、img。
- **限制使用HostPath，如果需要使用HostPath，限制只可以挂载指定前缀的目录并将卷配置为只读**
  使用 HostPath 可以直接将宿主机的目录挂载到容器中。很少有业务场景的Pod用到此功能特性。但如果确实有业务需要，您需要了解其中的风险。默认情况下，以root身份运行的Pod将拥有对 HostPath 暴露的文件系统的写访问权限。这可能允许攻击者修改 Kubelet 设置，创建指向未直接通过 HostPath 暴露的目录或文件的符号链接，例如 /etc/shadow 、安装Ssh密钥、读取挂载到主机的密钥以及进行恶意操作。为了降低使用 HostPath 的风险，请将 spec.containers.volumeMounts 配置为只读，例如：

  ```
  volumeMounts:
  - name: hostPath-volume
      readOnly: true
      mountPath: /host-path
  ```

  您还可以使用Pod安全策略来限制 hostPath 卷可以挂载的目录。例如，以下PSP策略仅允许挂载宿主机中以 /foo 开头的路径。

  ```
  allowedHostPaths:
  # This allows "/foo", "/foo/", "/foo/bar" etc., but
  # disallows "/fool", "/etc/foo" etc.
  # "/foo/../" is never valid.
  - pathPrefix: "/foo"
      readOnly: true # only allow read-only mounts
  ```

- **为每个容器设置请求和资源限制，避免资源争夺或DoS攻击**
  没有请求或资源限制的Pod理论上可以消耗掉主机上的所有可用资源。当有Pod被调度到此节点上时，该节点可能会遭遇CPU或内存不足的情况，这可能导致 Kubelet 崩溃或从节点驱逐Pod。虽然无法完全避免这种情况的发生，但设置请求和资源限制将有助于最大程度地减少资源争夺，并降低应用程序编写不当导致资源消耗过多所带来的风险。
  PodSpec允许您限制CPU和内存的使用。您可以通过在命名空间上设置Resource Quota或创建Limit Range来强制对请求和资源进行限制。资源配额允许您指定分配给命名空间的资源总量，例如CPU和RAM。当应用于命名空间时，会强制您为部署到该命名空间中的所有容器指定请求和资源的限制。相比之下，限制范围可让您更精细地控制资源分配。通过限制范围，您可以为命名空间内的每个Pod或每个容器的CPU和内存资源设置最小值/最大值。如果没有提供，您还可以设置默认请求/限制值。更多信息，请参见Managing Resources for Containers。
- **禁止使用特权提升配置**
  特权提升允许进程更改其运行所在的安全上下文。例如 sudo ，带有 SUID 或 SGID 位的二进制文件也是如此。特权升级基本上是用户以另一个用户或组的权限执行文件的一种方式。 您可以通过将 allowPriviledgedEscalation 设置为 false 的 pod 安全策略或通过在 podSpec 中设置 securityContext.allowPrivilegedEscalation 来阻止容器进行特权提升。
- **禁用Service Account令牌自动挂载**

对于不需要访问Kubernetes API的Pod，您可以在 `PodSpec` 上禁止自动挂载 `ServiceAccount` 令牌，或禁用所有使用特定 `ServiceAccount` 的Pod。

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-no-automount
spec:
  automountServiceAccountToken: false
```

禁用 `ServiceAccount` 自动挂载不会阻止Pod对Kubernetes API的网络访问。为阻止Pod对Kubernetes API进行网络访问，您需要修改ACK集群 `Endpoint` 访问并使用网络策略Network Policy来阻止Pod对Kubernetes API进行网络访问。具体操作，请参见使用网络策略Network Policy。

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: sa-no-automount
automountServiceAccountToken: false
```

- **禁用服务发现**
  对于不需要查找或调用集群服务的Pod，您可以减少提供给Pod的信息量。您可以将Pod的DNS策略设置为不使用CoreDNS，并且不将命名空间中的Service暴露为Pod中的环境变量。更多信息，请参见Environment variables。
  Pod的DNS策略的默认值是 `ClusterFirst` ，使用集群内DNS，而非默认值 `Default` 使用底层节点的DNS解析。更多信息，请参见Kubernetes docs on Pod DNS policy。
  禁用服务链接和更改Pod的DNS策略不会阻止Pod对集群内DNS服务进行网络访问。攻击者仍然可以通过访问集群内DNS服务来枚举集群中的服务（例如： `dig SRV *.*.svc.cluster.local @$CLUSTER_DNS_IP` ）。关于阻止集群内服务发现，请参见使用网络策略Network Policy。

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-no-service-info
spec:
    dnsPolicy: Default # "默认值"不是真正的默认值。
    enableServiceLinks: false
```

- **配置镜像为只读文件系统**
  将您的镜像配置为只读文件系统可防止攻击者覆盖您的应用程序使用的文件系统上的文件。如果您的应用程序必须写入文件系统，请考虑写入临时目录或挂载附加卷。您可以通过如下设置Pod的 `SecurityContext` 来强制执行此操作：

```
...
securityContext:
  readOnlyRootFilesystem: true
...
```

相关文档

- resource quota
- limit range

4.4. Runtime安全

Runtime安全为运行中的容器提供主动防护，主要实现方法是检测并阻止容器内发生恶意活动。本文主要介绍如何使用Seccomp减少应用程序的攻击面。

防止容器化应用程序对内核进行系统调用

- **添加注释来配置容器或Pod以使用此配置文件**
  Linux操作系统有几百个系统调用，但其中大部分不是运行容器所必需的。要开始使用Seccomp，请使用Strace生成堆栈跟踪以查看您的应用程序正在进行哪些系统调用，然后使用诸如Syscall2seccomp之类的工具根据从Strace收集的数据创建Seccomp配置文件。更多信息，请参见strace和syscall2seccomp。
  与SELinux不同，Seccomp并非旨在将容器彼此隔离，而是保护主机内核免受未经授权的系统调用。工作原理是拦截系统调用并只对列入白名单的系统调用放行。Docker有一个默认的Seccomp配置文件，适用于大多数通用工作负载。更多信息，请参见默认的seccomp配置文件。
  您可以通过向容器或PodSpec添加以下注释来配置容器或Pod以使用此配置文件。
  - Kubernetes 1.19之前版本：

    ```
    annotations:
      seccomp.security.alpha.kubernetes.io/pod: "runtime/default"
    ```

  - Kubernetes 1.19及之后版本：

    ```
    securityContext:
      seccompProfile:
        type: RuntimeDefault
    ```

- **为需要特殊权限的应用程序Pod创建自己的配置文件**
  Seccomp配置文件是Kubelet Alpha功能。您需要将 `--seccomp-profile-root` 标志添加到Kubelet启动参数中才能使用此功能。
  AppArmor和Seccomp类似，只是通过限制了包括访问部分文件系统的容器的Capabilities来达到防护容器运行时安全的目的。可以在强制或告警模式下运行。关于构建AppArmor配置文件，请参见bane。
  Apparmor仅适用于Ubuntu/Debian发行版的Linux系统。
  Kubernetes目前不提供将AppArmor或Seccomp配置文件自动加载到节点上的功能，需要手动加载或者在安装系统引导程序时安装到节点上。这些操作必须在创建Pod之前完成，因为K8s调度器不知道哪些节点具有这些配置。

Seccomp操作要点

- **使用第三方解决方案进行Seccomp和Apparmor配置的维护**
  如果您对Linux安全不熟悉，那么创建和管理Seccomp和Apparmor配置文件将会很困难。如果您没有资源自行维护这些配置文件，请考虑使用商业解决方案。这些三方的解决方案中有许多比Apparmor和Seccomp等静态配置防护更加优越，已经开始使用机器学习来对异常活动进行阻断或告警。
- **编写Seccomp策略之前考虑添加/删除Linux Capabilities**
  Capabilities涉及对系统调用可访问的内核函数的各种检查。如果检查失败，系统调用通常会返回错误。检查可以在特定系统调用开始时进行，也可以在内核中多个不同系统调用（例如写入特定特权文件）可访问的区域中进行。另一方面，Seccomp是一个系统调用过滤器，在运行之前应用于所有系统调用。一个进程可以设置一个过滤器，这允许Seccomp撤销运行某些系统调用或某些系统调用的特定参数的权限。
  在使用Seccomp之前，请考虑添加/删除Linux capabilities是否能正确控制应用程序的行为。更多信息，请参见Set capabilities for a Container。
- **检查是否可以通过使用Pod安全策略 (PSP) 来实现运行时防护**
  Pod安全策略提供了许多不同的方式来改善集群的安全状况，而不会引入过多的复杂性。在尝试构建Seccomp和Apparmor配置文件之前，请先探索PSP中的配置项是否能满足需求。

PSP在Kubernetes1.21版本中被设置为Deprecated状态，在使用中的用户可以在1.25版本前有一个替换的缓冲期。社区正在计划通过新的内置准入控制器方案来取代PSP，ACK容器服务也将通过基于OPA的策略治理方案逐步替换正在使用的PSP。

- **使用阿里云云安全中心**
  在云原生应用运行阶段，可实现基于云安全中心的应用运行时威胁检测与阻断，实时保障每个应用Pod的安全运行。云安全中心基于云原生的部署能力，实现威胁的数据自动化采集、识别、分析、响应、处置和统一的安全管控。利用多日志关联和上下文分析方案，实时检测命令执行、代码执行、SQL注入、数据泄露等风险，覆盖业务漏洞入侵场景。结合K8s日志和云平台操作日志实时进行行为审计和风险识别，实现容器服务和编排平台存在的容器逃逸、AccessKey泄露、未授权访问风险。更多信息，请参见什么是云安全中心。

相关文档

- AppArmor Loader
- Setting up nodes with profiles
- zaz
- seccomp-operator
- Aqua
- Qualys
- Stackrox
- Sysdig Secure
- Twistlock

4.5. 供应链安全

本文主要介绍如何测试、构建、部署运行整个交付链，以保证集群生命周期的安全性。

背景信息

软件供应链包括三个阶段：软件研发阶段、软件交付阶段、软件使用阶段，不同阶段的攻击面如下：

阶段	攻击面
软件研发阶段	• IDE开发工具污染攻击 • 三方库漏洞和后门攻击 • 直接源码污染攻击
软件交付阶段	• 软件存储替换和篡改攻击 • 传输劫持和捆绑下载攻击
软件使用阶段	• 升级劫持污染攻击 • 运行环境后门和漏洞攻击 • 三方库0Day漏洞攻击

在应用构建测试中，制品安全是您防御黑客攻击的第一道防线。一个不安全的、恶意构造的镜像可以让攻击者进行容器逃逸并获得对宿主机的访问权限。进入主机，攻击者就可以访问敏感信息或在集群内使用您的阿里云账户横向移动。在应用部署时刻，K8s原生的Admission准入机制可以帮助您有效验证部署实例的安全性。应用的成功部署并不意味着安全工作的结束，您需要在应用运行时刻实时监控业务应用安全，及时发现并处理运行时安全事件。

保证集群安全性建议

在Kubernetes环境中，遵循以下建议，以保证集群生命周期的安全性。

- **创建最小化的容器镜像**
  首先从容器镜像中删除所有无关的二进制文件。若使用的是来自Dockerhub的未知镜像，请使用Dive等应用程序检查该镜像，该应用程序可以向您展示每个镜像层的内容。更多信息，请参见Dive。
  在将镜像推送到ACR（Alibaba Cloud Container Registry）后，ACR控制台中也会展示镜像中层信息。删除所有带有 `SETUID` 和 `SETGID` 位的二进制文件，因为这些信息可用于提升权限，并考虑删除所有可用于恶意目的的Shell和应用程序，如 `nc` 和 `curl` 。您可以使用以下命令找到带有 `SETUID` 和 `SETGID` 位的文件：

  ```
  find / -perm /6000 -type f -exec ls -ld {} \;
  ```

  要从这些文件中删除特殊权限，请将以下指令添加到您的容器镜像中：

  ```
  RUN find / -xdev -perm /6000 -type f -exec chmod a-s {} \; || true
  ```

- **使用多阶段构建**
  使用多阶段构建是一种创建最小镜像的方法。多阶段构建可用于 `CI/CD` 流程中进行自动化的持续集成。多阶段构建指在 `Dockerfile` 中使用多个FROM语句，每个FROM指令都可以使用不同的基础镜像，并且是一个独立的子构建阶段。使用多阶段构建打包应用具有构建安全、构建速度快、镜像文件体积小等优点。更多信息，请参见在Dockerfile中使用多阶段构建打包Java应用。

- **为ACR仓库创建RAM策略**
  一个组织中的多个开发运维团队会使用同一个阿里云账户管理云资源。如果这些团队不需要共享资产，您可能需要创建一组RAM策略来限制每个团队可以访问的命名空间或者仓库。例如，`cr:ListInstance*` 表示授予 `cr:ListInstance` 开头的所有Action，设置 `acs:cr:*:*:repository/$instanceid/$namespace/*` 为 `acs:cr:cn-hangzhou:1234567:repository/cri-123456/ns/*` ，表示授予cn-hangzhou账号ID为1234567下ID为cri-123456的实例查询命名空间下镜像仓库的权限：

  ```
  {
      "Statement": [
          {
              "Effect": "Allow",
              "Action": [
                  "cr:ListRepository",
                  "cr:GetImageLayer",
                  "cr:GetRepoTag"
              ],
              "Resource": "*"
          },
          {
              "Action": [
                  "cr:List*"
              ],
              "Effect": "Allow",
              "Resource": [
                  "acs:cr:cn-hangzhou:1234567:repository/cri-123456/ns/*",
              ]
          }
      ],
      "Version": "1"
  }
  ```

在ACR中配置RAM策略的方法，请参见配置仓库的RAM访问控制和配置使用自定义OSS Bucket时的RAM访问控制。

- **优先使用ACR企业版**
  ACR企业版支持云原生制品加密存储，支持镜像安全扫描及多维度漏洞报告，保障存储及内容安全。分别提供容器镜像和Helm Chart的网络访问控制管理，细粒度的操作审计，保障制品访问安全。建议生产环境优先使用ACR企业版，并将仓库访问策略默认置为私有，使用内网VPC域名进行访问，不打开公网访问入口，并配置ACL白名单控制访问来源。更多信息，请参见创建企业版实例。
- **使用ACR云原生应用交付链**
  通过容器镜像服务的云原生应用交付链功能，您可以自由组合镜像构建、镜像扫描、镜像全球同步和镜像分发等任务，提供全链路可观测、可追踪、安全防护能力。更多信息，请参见创建交付链。
  ACR能在推送完成后自动进行镜像安全扫描，若您设置过安全阻断策略，其会识别镜像的安全风险并阻断高风险的容器镜像。通过安全策略的容器镜像才会进行交付链后续的分发和部署环节。交付链能保证容器应用的安全交付和高效部署。您也可以集成安全扫描的相关API，实现自定义周期的镜像安全扫描功能。
- **定期扫描镜像中的漏洞**
  与创建虚拟机使用的系统镜像一样，容器镜像同样包含带有漏洞的二进制文件和应用程序，或者随着时间的推移出现漏洞。因此，最佳的做法是使用镜像安全扫描工具定期对您使用的容器镜像进行漏洞扫描。可以在ACR中对新推送的镜像或者存量的镜像进行定期扫描（每隔24小时扫描一次）。应删除或重建具有 `HIGH` 或 `CRITICAL` 漏洞的镜像。若已部署的镜像出现漏洞，应尽快更换。
  `Kubernetes validation webhook` 也可用于验证镜像是否存在严重漏洞。在使用 `Kubernetes API` 之前调用 `validation webhook` ，可用于拒绝不符合 `webhook` 中定义的验证标准的请求。可以通过调用ACR提供的CreateRepoTagScanTask来确定Pod是否正在拉取具有严重漏洞的镜像。如果发现漏洞，则拒绝Pod的创建，并以事件形式返回带有CVE列表的消息。更多信息，请参见CreateRepoTagScanTask。
- **将USER指令添加到您的Dockerfiles中并以非root用户运行镜像**
  如Pod安全部分所述，您应该避免以root身份运行容器。您可以将其配置为PodSpec的一部分，但在Dockerfile中使用USER指令是一个好习惯。当设置USER指令后，会以指定的用户运行RUN、ENTRYPOINT或CMD。
- **从可信软件源下载依赖包**
  在软件研发阶段，您应该尽量避免从不可信的软件源引用软件包。关于配置阿里云可信的镜像和软件制品源或者下载研发需要的软件包，请参见阿里云镜像站。
  您可以使用阿里云云效搭建属于自己的软件仓库。云效制品仓库 Packages是阿里云出品的一款企业级私有仓库服务，提供基于Maven、Gradle、NPM等软件包管理工具的企业级私有仓库服务，用于管理企业级依赖托管。仓库支持管理Maven制品和NPM制品，并支持配置远程仓库，一键迁移现有私库。提供租户隔离、权限控制、高可用存储等服务，全面保障企业制品安全。具体操作，请参见在云效中使用自己搭建的私有Maven仓库。
- **使用镜像加签并配置验签策略**
  在应用部署时刻对目标镜像的安全性校验能够保证集群中只部署可信授权方签名的容器镜像，降低您环境中发生意外或恶意代码的风险。
  阿里云容器镜像服务ACR支持镜像加签功能，保障镜像从分发到部署的全链路一致性，避免中间人攻击和非法镜像的更新及运行。ACR支持命名空间级别的自动加签，每次推送容器镜像后都会匹配加签规则自动加签，保障您的容器镜像内容可信。更多信息，请参见使用容器镜像加签。
  在ACK集群中，您可以通过组件管理安装Kritis-validation-hook组件，组件在开源Kritis阿里云容器镜像服务ACR软件的基础上增加了对的深度集成，支持验证通过阿里云密钥管理服务KMS签名的容器镜像，实现自动验证容器镜像签名。更多信息，请参见使用kritis-validation-hook组件实现自动验证容器镜像签名、kritis、阿里云密钥管理服务KMS。
  在开启镜像签名之后，您可以配置验签策略，只部署通过验签规则的可信镜像到您的ACK集群，并且支持配置验签规则的白名单，解决第三方组件自动注入的Sidecar容器的镜像无法通过镜像验签，导致无法部署Pod的问题。更多信息，请参见kritis-validation-hook组件介绍。
- **使用阿里云云安全中心**

在云原生应用运行阶段，可实现基于云安全中心的应用运行时威胁检测与阻断，实时保障每个应用Pod的安全运行。云安全中心基于云原生的部署能力，实现威胁的数据自动化采集、识别、分析、响应、处置和统一的安全管控。利用多日志关联和上下文分析方案，实时检测命令执行、代码执行、SQL注入、数据泄露等风险，覆盖业务漏洞入侵场景。结合K8s日志和云平台操作日志实时进行行为审计和风险识别，实现容器服务和编排平台存在的容器逃逸、AccessKey泄露、未授权访问风险。更多信息，请参见什么是云安全中心。

4.6. 数据加密和密钥管理

通过加密存储在阿里云ECS上的数据就能保护数据的隐私性和自主性。本文主要介绍加密云盘的方法。

使用KMS中的指定密钥对指定云盘存储卷进行加密

阿里云ACK集群适用于有高安全性或合规性要求的应用场景，您无需自建和维护密钥管理基础设施，通过加密保护存储在阿里云ECS上的数据就能保护数据的隐私性和自主性。更多信息，请参见加密云盘存储卷。

本文将以配置加密云盘数据卷为例来说明，在创建存储资源实例时，您可以使用KMS中的指定密钥对指定云盘存储卷进行加密。

1. 配置StorageClass参数。
   i. 将以下内容复制到*sc-kms.yaml*文件中。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-disk
provisioner: diskplugin.csi.alibabacloud.com
parameters:
    fsType: ext4
    type: cloud_ssd
    encrypted: "true"
    kmsKeyId: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
reclaimPolicy: Delete
```

   ii. 执行以下命令创建StorageClass。

```
kubectl create -f sc-kms.yaml
```

2. 创建PVC。
   i. 将以下内容复制到*sc-pvc.yaml*文件中。

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: disk-pvc
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 20Gi
  storageClassName: csi-disk
```

   ii. 执行以下命令创建PVC。

```
kubectl create -f sc-pvc.yaml
```

云盘加密建议

- **开启云盘加密特性**
  对业务数据静态加密是安全的最佳实践，具体操作，请参见加密云盘存储卷。
- **定期轮换CMK**
  您可以通过密钥版本化和定期轮转来加强密钥使用的安全性，实现数据保护的安全策略和最佳实践。具体操作，请参见自动轮转密钥。

密钥管理

Kubernetes Secrets模型一般用于存储业务应用涉及的敏感信息，例如应用密码、证书、API访问凭据等敏感信息，所有Secrets对象在原生的K8s集群会以Base64编码的形式存储在集群etcd中。在ACK托管集群管控侧，所有控制平面中的etcd节点云盘均开启了数据盘加密特性，保证用户业务数据的基本隐私性。您可以通过环境变量或挂载存储卷的方式在业务Pod中使用指定的Secret对象。更多信息，请参见Secrets。

- **使用阿里云KMS进行Secret的落盘加密**
  在ACK Pro集群中，您可以使用在密钥管理服务（KMS）中创建的密钥加密Kubernetes Secret，加密过程基于Kubernetes提供的KMS Encryption Provider机制，使用信封加密的方式对存储在etcd中的Kubernetes Secret密钥进行自动加密和解密。更多信息，请参见KMS Encryption Provider机制、什么是信封加密、使用阿里云KMS进行Secret的落盘加密。
- **为密钥创建单独的Namespace并与不同的应用程序隔离**
  如果您的密钥不需要在命名空间中的应用程序之间共享，请为每个应用程序创建一个独立的命名空间并严格控制授予Secret的读写权限。
- **使用卷挂载Secret而不是存储在环境变量中**
  环境变量的值可能会无意中出现在日志中。作为卷挂载的 `Secret` 被实例化为 `tmpfs` 卷，当Pod被删除时，这些卷会自动从节点中删除。
- **使用外部密钥管理系统**
  您可以选择将应用密钥管理在外部的专业密钥管理系统中，这些外部密钥管理系统可以提供细粒度的访问控制，多种加密算法以及密钥自动轮转等高级特性。更多信息，请参见凭据管家概述和Vault。
  在应用需要使用指定密钥时，可以将外部密钥系统中存储的指定密文实时同步到ACK集群中，并以K8s原生Secret的形式集成在应用中。例如，将阿里云凭据管家中管理的密钥以Secret形式实时同步到ACK集群。具体操作，请参见ack-secret-manager。
- **使用ACK-TEE机密计算**
  ACK支持创建基于Intel SGX2.0技术的加密计算托管集群，可以帮助您保护数据使用（计算）过程中的安全性、完整性和机密性，同时简化了可信或机密应用的开发、交付和管理成本。机密计算可以让您把重要的数据和代码放在一个特殊的可信执行加密环境（Trusted Execution Environment，TEE）中，而不会暴露给系统其他部分。其他应用、BIOS、OS、Kernel、管理员、运维人员、云厂商、甚至除了CPU以外的硬件均无法访问机密计算平台数据，极大减少泄露敏感数据的风险，有更好的控制、透明度和隐秘性。更多信息，请参见ACK-TEE机密计算介绍。

4.7. 网络安全

本文主要介绍网络安全防护涉及的两个方面：服务之间网络流量的访问控制和传输过程中的流量加密。

网络策略

在Kubernetes集群中，默认情况下允许所有Pod间的通信。在生产环境中这样的默认配置是不安全的。 `Kubernetes网络策略（Network Policy）` 为您提供了一种机制来限制Pod间的网络流量（通常称为东/西向流量）以及Pod和外部服务之间的网络流量。Network Policy使用Pod Selectors和Labels标签标识源和目的Pod，同时在策略中支持指定的IP地址，端口号和协议类型及组合。使用Terway容器网络时，如果您希望在IP地址或者端口层面控制网络流量，您可以为集群中特定应用配置网络策略。具体操作，请参见使用网络策略Network Policy和Kubernetes Network Policy Recipes。

> **注意** 在大规模生产环境中使用Network Policy会对API Server造成压力，请谨慎使用。

创建默认拒绝策略

与 `RBAC` 策略一样，创建 `Network policy` 同样应遵循最低权限访问原则。应创建一个默认拒绝（Deny All）策略限制来自命名空间的所有入站和出站流量，或者使用Calico创建一个全局策略。

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: default-deny
  namespace: default
spec:
  podSelector: {}
  policyTypes:
  - Ingress
  - Egress
```

创建允许DNS查询的规则

设置了默认的Deny All规则，您就可以开始对其他规则进行分层，例如允许Pod查询CoreDNS进行名称解析的全局规则。

1. 执行以下命令对命名空间添加标签。

   ```
   kubectl label namespace kube-system name=kube-system
   ```

2. 使用以下YAML示例创建 `Network Policy` 。

   ```
   apiVersion: networking.k8s.io/v1
   kind: NetworkPolicy
   metadata:
     name: allow-dns-access
     namespace: default
   spec:
     podSelector:
       matchLabels: {}
     policyTypes:
     - Egress
     egress:
     - to:
       - namespaceSelector:
           matchLabels:
             name: kube-system
       ports:
       - protocol: UDP
         port: 53
   ```

   > **注意** 关于如何在Kubernetes Network Policy中配置Pod间的流量控制，请参见官方文档。

以下示例说明了如何将网络策略与服务帐户ali-sa关联，以及如何防止与readonly-sa-group RBAC组绑定的readonly-sa-role角色来编辑默认命名空间中的服务账户ali-sa。

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: ali-sa
  namespace: default
  labels:
    name: ali-sa
---
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  namespace: default
  name: readonly-sa-role
rules:
# 允许受试者读取名为ali sa的服务帐户。
- apiGroups: [""]
  resources: ["serviceaccounts"]
  resourceNames: ["ali-sa"]
  verbs: ["get", "watch", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  namespace: default
  name: readonly-sa-rolebinding
# 将只读sa角色绑定到名为只读sa组的RBAC组。
subjects:
- kind: Group
  name: readonly-sa-group
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: readonly-sa-role
  apiGroup: rbac.authorization.k8s.io
---
apiVersion: crd.projectcalico.org/v1
kind: NetworkPolicy
metadata:
  name: netpol-sa-demo
  namespace: default
# 允许对引用的默认命名空间中的服务的所有入口流量。
# 参考名为ali sa的服务帐户。
spec:
  ingress:
    - action: Allow
      source:
        serviceAccounts:
          selector: 'name == "ali-sa"'
  selector: all()
```

增量添加规则以选择性地允许Namespaces/Pods之间的流量

首先允许命名空间内的Pod相互通信，然后添加自定义规则，进一步限制该命名空间内的Pod和Pod间的通信。更多信息，请参见Kubernetes Network Policy Recipes。

对网络流量数据进行分析监控

阿里云VPC提供流日志功能，可以记录VPC网络中弹性网卡ENI（Elastic Network Interface）传入和传出的流量信息，帮助您检查访问控制规则、监控网络流量和排查网络故障。通过分析这些信息可以查找VPC内资源（包括Pod）之间的异常流量。更多信息，请参见流日志概述。

安全组

ACK使用安全组来约束控制平面节点和节点之间的网络流量。安全组还用于控制节点、其他VPC资源和外部IP地址之间的流量。当创建ACK集群时，会默认自动为您分配一个安全组。该安全组允许集群内部节点之间的访问不受限制。出于安全考虑您还可以继续收敛安全组规则，您可以参考下面表格中的最小化出入方向配置，在此基础上根据业务出入网需要进行增量配置：

出入方向流量	协议	端口范围	源	目的
最小化入方向流量（来自控制平面和其他节点）	TCP 您希望用于节点间通信的协议	443，10250您希望用于节点间通信的端口	集群安全组	无
推荐入方向流量	ALL/TCP	ALL/443, 1025-65535	集群安全组	无
最小化出方向流量	TCP	443	无	集群安全组
推荐出方向流量	ALL	ALL	无	0.0.0.0/0

更多信息，请参见ECS安全组配置案例和配置安全组。

传输加密

- **阿里云服务网格ASM**
  阿里云服务网格ASM为可帮助您实现服务之间的传输加密，除了服务之间的双向TLS加密，也支持使用SDS（Envoy' s Secret Discovery Service）为服务网关提供HTTPS安全支持、证书动态加载，从而提升服务网关整体安全性。同时ASM支持配合阿里云应用高可用服务AHAS对部署在服务网格内的应用进行流量控制，通过集成阿里云链路追踪服务Tracing Analysis，为分布式应用的开发者提供了完整的调用链路还原、调用请求量统计、链路拓扑、应用依赖分析等能力，可以帮助开发者快速分析和诊断分布式应用架构下的性能瓶颈，提升开发诊断效率。
- **通过Secret配置TLS证书实现HTTPS访问**
  集群中通过Ingress暴露的服务需要开启HTTPS访问，具体操作，请参加通过Secret配置TLS证书实现HTTPS访问。

相关文档

- Cilium
- NetworkPolicy Editor
- Kinvolk's Network Policy Advisor
- 阿里云服务网格ASM
- 什么是服务网格ASM?
- 通过服务网关启用HTTPS安全服务
- 使用链路追踪实现网格内外应用的一体化追踪

4.8. 日志和审计

本文主要介绍ACK容器服务对K8s多种组件日志和审计的采集和分析，包括API Server、Ingress，管控平面组件和K8s关键Events事件等日志审计的收集和分析。以方便日志在出现安全问题或集群出现问题时进行问题溯源。

使用集群审计功能

在Kubernetes集群中，API Server的审计日志可以帮助集群管理人员记录或追溯不同用户的日常操作，是集群安全运维中重要的环节。关于通过日志服务收集、分析审计日志、为审计日志设置自定义的告警规则、关闭集群审计功能，请参见使用集群审计功能。

阿里云服务容器ACK的审计策略如下：

```
apiVersion: audit.k8s.io/v1beta1 # This is required.
kind: Policy
#不要为RequestReceived阶段中的所有请求生成审核事件。
omitStages:
- "RequestReceived"
rules:
# 以下请求被手动确定为高容量和低风险，因此请取消这些请求。
- level: None
  users: ["system:kube-proxy"]
  verbs: ["watch"]
  resources:
    - group: "" # core
      resources: ["endpoints", "services"]
- level: None
  users: ["system:unsecured"]
  namespaces: ["kube-system"]
  verbs: ["get"]
  resources:
    - group: "" # core
      resources: ["configmaps"]
- level: None
  users: ["kubelet"] # legacy kubelet identity
  verbs: ["get"]
  resources:
    - group: "" # core
      resources: ["nodes"]
- level: None
  userGroups: ["system:nodes"]
  verbs: ["get"]
  resources:
    - group: "" # core
      resources: ["nodes"]
- level: None
  users:
    - system:kube-controller-manager
    - system:kube-scheduler
    - system:serviceaccount:kube-system:endpoint-controller
  verbs: ["get", "update"]
  namespaces: ["kube-system"]
  resources:
    - group: "" # core
```

```
      resources: ["endpoints"]
- level: None
  users: ["system:apiserver"]
  verbs: ["get"]
  resources:
    - group: "" # core
      resources: ["namespaces"]
#不要记录这些只读URL。
- level: None
  nonResourceURLs:
    - /healthz*
    - /version
    - /swagger*
#不要记录事件请求。
- level: None
  resources:
    - group: "" # core
      resources: ["events"]
# 机密、配置映射和令牌审查可以包含敏感和二进制数据，
# 因此，只能在元数据级别进行日志记录。
- level: Metadata
  resources:
    - group: "" # core
      resources: ["secrets", "configmaps"]
    - group: authentication.k8s.io
      resources: ["tokenreviews"]
- level: Request
  verbs: ["get", "list", "watch"]
  resources:
    - group: "" # core
    - group: "admissionregistration.k8s.io"
    - group: "apps"
    - group: "authentication.k8s.io"
    - group: "authorization.k8s.io"
    - group: "autoscaling"
    - group: "batch"
    - group: "certificates.k8s.io"
    - group: "extensions"
    - group: "networking.k8s.io"
    - group: "policy"
    - group: "rbac.authorization.k8s.io"
    - group: "settings.k8s.io"
    - group: "storage.k8s.io"
# 已知API的默认级别。
- level: RequestResponse
  resources:
    - group: "" # core
    - group: "admissionregistration.k8s.io"
    - group: "apps"
    - group: "authentication.k8s.io"
    - group: "authorization.k8s.io"
    - group: "autoscaling"
    - group: "batch"
    - group: "certificates.k8s.io"
```

```
    - group: "extensions"
    - group: "networking.k8s.io"
    - group: "policy"
    - group: "rbac.authorization.k8s.io"
    - group: "settings.k8s.io"
    - group: "storage.k8s.io"
    - group: "autoscaling.alibabacloud.com"
# 所有其他请求的默认级别。
- level: Metadata
```

利用审计元数据

Kubernetes审计日志包括两个注释，用于指示请求是否获得授权 `authorization.k8s.io/decision` ，以及作出决定的原因 `authorization.k8s.io/reason` 。使用这些属性来确定允许特定API调用的原因。

使用NPD结合SLS的Kubernetes事件中心监控集群可疑事件

NPD（node-problem-detector）是阿里云容器服务ACK维护的Kubernetes节点诊断的工具，可以将节点的异常，例如Docker Engine Hang、Linux Kernel Hang、网络出网异常、文件描述符异常转换为Node的事件，结合kube-eventer可以实现节点事件告警的闭环。除了NPD检测到的集群节点实时问题和故障，Kubernetes集群自身也会因为集群状态的切换产生各种事件，例如Pod驱逐、镜像拉取失败等异常情况。另外对于普通用户使用exec登录到指定容器等可疑运维操作，SLS（Log Service）日志服务的Kubernetes事件中心实时汇聚Kubernetes中的所有事件并提供存储、查询、分析、可视化、告警等能力，帮助安全运维管理人员及时发现集群稳定性风险。具体操作，请参见事件监控。

开启Ingress Dashboard监控

阿里云Ingress组件支持将您的所有HTTP请求日志记录到标准输出中。同时，阿里云打通Ingress组件访问日志服务与阿里云日志服务，从而您可以使用日志服务快速创建日志分析和监控大盘。在Ingress的可视化仪表盘中，您可以方便地查看Ingress的整体状态，包括：PV、UV、流量、响应延迟、Top URL统计等关键信息，帮助您实时掌控业务流量动态、及时发现恶意访问和DOS攻击。具体操作，请参见Ingress Dashboard监控。

开启CoreDNS日志

阿里云容器服务ACK部署了CoreDNS作为集群内的DNS服务器，您可以通过查看CoreDNS日志来分析CoreDNS解析慢、访问高危请求域名等问题。在日志服务仪表盘中可以看到关于CoreDNS的日志分析报表，您可以及时发现可疑的高危域名请求，具体操作，请参见分析和监控CoreDNS日志。

4.9. 多租户安全

本文主要介绍如何同时保证租户之间公平地分配共享集群资源，以最大程度的避免恶意租户对其他租户的攻击。

背景信息

隔离的安全程度分为软隔离(Soft Multi-tenancy)和硬隔离（Hard Multi-tenancy）。

- 软隔离更多面向企业内部的多租需求，该形态下默认不存在恶意租户，隔离是为了内部团队间的业务保护和对可能的安全攻击进行防护。
- 硬隔离更多面向对外提供服务的服务供应商，由于该业务形态下无法保证不同租户中业务使用者的安全背景，默认租户之间以及租户与K8s系统之间是存在互相攻击的可能，因此也需要更严格的隔离作为安全保障。

软多租

您可以使用原生 Kubernetes 特性来实现软多租，例如 namespace 、 roles 、 role bindings 以及 network polices ，在租户之间实现逻辑分离。例如， RBAC 可以防止租户访问或操纵彼此的资源。 qoutas 和 limte ranges 控制每个租户可以消耗的集群资源量，而 network polices 可以防止部署到不同命名空间的应用程序相互通信。

这些控制措施不能阻止来自不同租户的 Pod 共享一个节点。您可以使用 nodeselector 、 anti-affinity 规则、 taints 和 tolerations 来强制将不同租户的Pod调度到不同的节点上，这通常称为独立租户节点。在租户数量很多的场景下，这样做会变得相当复杂且成本过高。

使用 Namespaces 实现的软多租不允许您向租户提供命名空间的过滤列表，因为命名空间是全局范围的资源对象。如果租户能够查看指定命名空间，则可以查看集群内的所有命名空间。

使用软多租，租户保留默认情况下为集群内运行的所有服务查询 CoreDNS 的能力。攻击者可以在集群中的任何 Pod 中运行 dig SRV ..svc.cluster.local 来利用此特性。如果您需要限制对集群内运行的服务的DNS记录的访问，请使用 CoreDNS 的防火墙或策略插件。具体操作，请参见kubernetes-metadata-multi-tenancy-policy。

- **企业内部环境**
  第一种是在企业环境中，该场景下集群的所有用户均来自企业内部，这也是当前很多K8s集群客户的使用模式，因为服务使用者身份的可控性，相对来说这种业务形态的安全风险是相对可控的。每个租户通常会与一个行政部门（例如部门或团队）保持一致。
  在类似这种场景中，集群管理员通常负责创建命名空间和管理策略。还可以实现托管管理模型，在该模型中，某些个人被赋予对命名空间的监管权，允许他们对非策略相关的对象（如 deployments 、 services 、 pod 、 jobs 等）执行 CRUD 操作。
  Docker提供的隔离机制在此场景中是可以接受的，或者需要增加额外的控制，例如Pod安全策略 (PSP)。如果需要更严格的隔离，还需要限制不同命名空间中服务之间的通信。
- **Kubernetes即服务（KaaS）**
  软多租户可用于您希望提供Kubernetes即服务 (KaaS) 的场景之中。使用KaaS，您的应用程序与提供一组PaaS服务的控制器和CRD集合一起托管在共享集群中。租户直接与Kubernetes API服务器交互，并被允许对非策略对象执行CRUD操作。还有自助功能，例如允许租户创建和管理他们自己的命名空间。在类似此种环境中，租户被假定正在运行不可信代码。
  要在此类环境中隔离租户，您可能需要实施严格的Network Policies以及Pod Sandboxing。具体操作，请参见安全容器。
- **软件即服务 (SaaS)**
  在此环境中，每个租户都与在集群中运行的应用程序的特定实例相关联。每个实例通常都有自己的数据，并使用通常独立于 Kubernetes RBAC 的单独的访问控制。
  与其他场景不同，SaaS环境中的租户不直接与 Kubernetes API 交互。而是SaaS应用程序负责与 Kubernetes API 交互以创建每个租户所需要的对象。

Kubernetes原生配置

Kubernetes在架构上是面向单租户的容器编排管理平台，即控制平面的单个实例在集群内的所有租户之间共享。您可以使用各种Kubernetes对象来实现多租户隔离的目的。例如，可以使用命名空间和基于角色的访问控制（ RBAC ），以在逻辑上将租户彼此隔离。同样， Quotas 和 Limit Ranges 可用于控制每个租户可以消耗的集群资源量。然而，集群是唯一提供强大安全边界的结构。这是因为设法获得对集群内主机的访问权的攻击者可以检索所有安装在该主机上的 Secrets 、 ConfigMaps 和 Volumes 。还可以模拟 Kubelet ，这将允许操纵节点的属性或在集群内横向移动。下面的Kubernetes原生配置可以帮助您降低使用像Kubernetes这样的单租户平台的风险，在一定程度上实现上述场景中租户之间的隔离。

- **命名空间**
  Namespaces是实现软多租的基础。Namespaces允许您将集群分为不同的逻辑层。Quotas、Network Policies、Service Accounts和其他资源对象都需要在Namespaces范围内实现多租。
- **AuthN&AuthZ&Admission**

ACK集群的授权分为RAM授权和RBAC授权两个步骤，其中RAM授权作用于集群管理接口的访问控制，包括对集群的CRUD权限（如集群可见性、扩缩容、添加节点等操作），而RBAC授权用于集群内部Kubernetes资源模型的访问控制，可以做到指定资源在命名空间粒度的细化授权。ACK授权管理为租户内用户提供了不同级别的预置角色模板，同时支持绑定多个用户自定义的集群角色，此外支持对批量用户的授权。具体操作，请参见授权概述。

- **网络策略**
  默认情况下，Kubernetes集群中的所有Pod都允许相互通信。使用Network Policies更改此默认设置。
- Network Policies使用标签或IP地址范围限制Pod之间的通信。在需要租户之间严格网络隔离的多租户环境中，需要添加两条规则：
  - 拒绝Pod之间通信的默认规则。
  - 允许所有Pod查询DNS服务器以进行名称解析。
- **资源配额&限制范围**
  Quotas用于定义集群中托管的工作负载的限制。使用Quotas，您可以指定Pod可以消耗的最大CPU和内存量，也可以限制可以在集群或命名空间中分配的资源数量。Limit ranges允许您声明每个限制的最小值、最大值和默认值。
  在共享集群中过度使用资源通常是有益的，因为可以让您最大限度地利用资源。但是，对集群的无限制访问会导致资源匮乏，从而导致性能下降和应用程序可用性损失。如果一个Pod的请求设置得太低，实际资源利用率超过了节点的容量，节点就会开始遇到CPU或内存压力。发生这种情况时，Pod可能会重启或从节点中驱逐。
  为了防止这种情况发生，您应该在多租户环境中对命名空间实施Quotas，以强制租户在集群上调度Pod时指定请求和限制。这样做还可以限制Pod可以消耗的资源量来缓解潜在的拒绝服务风险。
  在KaaS场景中，您可以使用Quotas来分配集群资源以与租户需要的保持一致。
- **Pod优先级和抢占**
  当您想为不同的客户提供不同的服务质量 (QoS) 时，Pod优先级和抢占会很有用。例如，使用Pod优先级，您可以将客户A的Pod配置为以高于客户B的优先级运行。当可用容量不足时，Kubelet会从客户B驱逐低优先级的Pod以容纳客户A的高优先级Pod。在SaaS环境中，通过这种方式为愿意获得更高质量服务从而支付更高价格的客户提供服务方便。

缓解措施

作为多租环境的安全管理员，您主要关心的是防止攻击者获得对底层主机的访问权限。应考虑采取以下控制措施来降低这种风险：

- **安全沙箱**
  相比于原有Docker运行时，安全沙箱为您提供的一种新的容器运行时选项，可以让您的应用运行在一个轻量虚拟机沙箱环境中，拥有独立的内核，具备更好的安全隔离能力。
  安全沙箱特别适合于不可信应用隔离、故障隔离、性能隔离、多用户间负载隔离等场景。在提升安全性的同时，对性能影响非常小，并且具备与Docker容器一样的用户体验，例如日志、监控、弹性等。更多信息，请参见安全沙箱概述。
- **Open Policy Agent (OPA) & Gatekeeper**
  OPA（Open Policy Agent）是一种功能强大的策略引擎，支持解耦式的Policy Decisions服务并且在K8s集群中已经有了广泛应用。当现有RBAC在命名空间粒度的隔离不能够满足企业应用复杂的安全需求时，可以通过OPA提供object模型级别的细粒度访问策略控制。Gatekeeper是一个Kubernetes准入控制器，可以在应用部署时刻执行指定的已实施OPA策略。更多信息，请参考Gatekeeper。
  同时OPA支持七层的NetworkPolicy策略定义及基于Labels/Annotation的跨命名空间访问控制，可以作为K8s原生NetworkPolicy的有效增强。
- **Kyverno**
  Kyverno是一个面向Kubernetes而生的策略引擎，可以为Kubernetes资源产生验证、改变和生成配置的策略。Kyverno支持Kustomize Overlays风格的策略校验和Mutate修改，并且可以基于灵活的触发器跨命名空间克隆资源。更多信息，请参见Kyverno。

您可以使用Kyverno来隔离命名空间、实现Pod安全和其他最佳实践，并生成默认配置（例如网络策略）。具体操作，请参见策略仓库。

硬多租

硬多租可以通过为每个租户配置单独的集群来实现。虽然这在租户之间提供了非常强的隔离，但有如下几个缺点：

- 当您拥有很多租户时，成本会很高。您不仅需要为每个集群支付控制平面成本，而且无法在集群之间共享计算资源。这样会导致碎片化，其中一部分集群未被充分利用，其他集群则被过度利用。
- 您可能需要购买或构建特殊工具来管理这些集群。随着时间的推移，管理成百上千个集群可能会变得过于繁重。
- 和创建命名空间相比，为每个租户创建集群会很慢。在高度监管的行业或需要强隔离的SaaS环境中，需要采用硬多租方法。

未来方向

Kubernetes社区已经认识到软多租目前的缺点以及硬多租的挑战。多租户特别兴趣小组 (SIG) 正尝试通过几个孵化项目来解决这些问题：

- Virtual Cluster提案描述了一种机制，用于为集群中的每个租户（也称为“Kubernetes on Kubernetes”）创建控制平面服务的单独实例，包括API Server、Controller Manage和Scheduler。更多信息，请参见Virtual Cluster。
- HNC提案 (KEP) 描述了一种通过策略对象继承以及租户管理员创建子命名空间的能力在命名空间之间创建父子关系的方法。更多信息，请参见HNC。
- Multi-Tenancy Benchmarks提案提供了使用命名空间进行隔离和分段共享集群的指南，以及命令行工具Kubectl-mtb用于验证是否符合的指南。更多信息，请参见Multi-Tenancy Benchmarks。

相关文档

- k-rail
- kiosk
- Loft
- DevSpace
- 多租户特别兴趣小组

4.10. 主机安全

宿主机安全保证容器的运行，本文主要介绍如何保护ACK中宿主机的安全性。

定期运行基线检查以验证集群是否符合CIS Benchmarks和等保加固标准

CIS（Center for Internet Security）是一个第三方安全组织，致力于采用线上社区的模式与大公司、政府机构、学术机构一起打造优秀的安全实践解决方案。

CIS Kubernetes基线是针对开源Kubernetes发行版本所编写的，旨在尽可能广泛应用于各个发行版本，同时不同的基线版本与特定的Kubernetes版本相关联。更多信息，请参见CIS Kubernetes基线。

通用版本的CIS Kubernetes基线包含了管控面和数据面的相关内容，各大云服务商通常提供了管控面全托管的Kubernetes集群，通用版本的CIS Kubernetes基线并不能适配所有云服务商。为此，阿里云也在CIS社区发布了更符合ACK集群应用配置场景的CIS Alibaba Cloud Container Service For Kubernetes（ACK）Benchmark。

阿里云ACK集群基于CIS Alibaba Cloud Container Service For Kubernetes（ACK）Benchmark实现了默认安全加固。更多信息，请参见使用security-inspector组件实现集群安全CIS基线检查。

对于集群节点宿主机OS系统，当前各大云服务商发布系统大多都已经提供了对应的CIS Benchmark，包括Alibaba Cloud Linux 2、CentOS、Ubuntu等。Alibaba Cloud Linux 2不仅是阿里云官方操作系统镜像，也是ACK的首选默认系统镜像。Alibaba Cloud Linux 2在2019年08月16日正式通过了CIS组织的全部认证流程并对外发布CIS Aliyun Linux 2 Benchmark version 1.0.0。更多信息，请参见CIS Aliyun Linux 2 Benchmark version 1.0.0。

您可以提升ACK集群所有节点的操作系统的安全性。具体操作，请参见ACK CIS加固使用说明。

阿里云根据国家信息安全部发布的《GB/T22239-2019信息安全技术网络安全等级保护基本要求》中对操作系统提出的一些等级保护要求，基于云原生操作系统Alibaba Cloud Linux实现。更多信息，请参见ACK等保加固使用说明。

您可以使用ACK的等保加固配置满足以下等保合规要求：

- 身份鉴别
- 访问控制
- 安全审计
- 入侵防范
- 恶意代码防范

使用阿里云云安全中心安全防范能力

阿里云安全中心支持对ACK集群节点的默认安全提供全方位保护，包括：

- 漏洞修复：支持对主流漏洞类型进行检测并提供一键修复功能。您可在漏洞修复页面查看服务器当前存在的漏洞风险、手动执行一键扫描，帮助您更全面地了解您资产中的漏洞和风险情况。
- 基线检查：基线检查功能针对服务器操作系统、数据库、软件和容器的配置进行安全检测，并提供检测结果说明和加固建议。基线检查功能可以帮您进行系统安全加固，降低入侵风险并满足安全合规要求。
- 云平台配置检查：云平台配置检查从身份认证及权限、网络访问控制、数据安全、日志审计、监控告警、基础安全防护六个维度为您提供云产品安全配置的检查，帮助您及时发现您的云产品配置风险并提供相应的修复方案。
- 镜像安全扫描：支持对镜像中存在的高危系统漏洞、应用漏洞、恶意样本、配置风险和敏感数据进行检测和识别，并提供漏洞修复方案，为您提供一站式漏洞管理能力，让镜像漏洞修复更简单。

最小化对节点的访问权限

当您想要对节点进行远程访问时，可以通过登录容器服务管理控制台，通过Workbench或者VNC进行内网访问，不为节点挂载公网EIP，若要公网访问，应该在ACK的安全组中配置ACK访问策略，限制访问来源。

并且需要在ACK安全组中控制对节点各个端口的暴露，对于需要公网暴露的端口，必须限制访问来源。

遵循ECS安全最佳实践

ACK默认使用Alibaba Cloud Linux 2创建ECS实例作为ACK的节点，关于在阿里云ECS实例使用过程中提高安全性，请参见安全最佳实践。

相关文档

- ACK CIS加固使用说明
- ACK等保加固使用说明
- 使用操作系统Alibaba Cloud Linux 2
- 漏洞修复概述

5.存储

5.1. 实现StatefulSet持久化存储的最佳实践-CSI

有状态服务StatefulSet支持通过VolumeClaimTemplate为每个Pod创建PV和PVC。并且删除和减少Pod时，不会删除StatefulSet的PV和PVC。本文为您介绍如何通过VolumeClaimTemplate实现StatefulSet持久化存储。

前提条件

创建Kubernetes托管版集群

应用场景

有状态服务StatefulSet的应用场景：

- 稳定的部署次序：有序部署或扩展，需要根据定义的顺序依次进行（即从0到N-1，在下一个Pod运行之前，所有之前的Pod必须都是Running和Ready状态）。
- 稳定的缩容次序：有序收缩或删除，需要根据定义的顺序依次进行（即从N-1到0，从N-1到0， 在下一个Pod执行结束之前 ，所有之前的Pod都需要删除完成）。
- 稳定的网络标志：Pod重新调度后其PodName和HostName不变。
- 稳定的持久化存储：基于PVC，Pod重新调度后仍能访问到相同的持久化数据。

有状态服务-StatefulSet的使用方式：

通过 `VolumeClaimTemplates` 自动创建PVC及PV。

创建StatefulSet服务

> **说明** volumeClaimTemplates：表示一类PVC的模板，系统会根据有状态服务-StatefulSet配置的replicas数量，创建相应数量的PVC。这些PVC除了名字不一样之外其他配置都是一样的。

1. 使用以下内容，创建*statefulset.yaml*。

   创建一个Service和StatefulSet，并且设置该StatefulSet包含2个Pod。

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
  labels:
    app: nginx
spec:
  ports:
  - port: 80
    name: web
  clusterIP: None
  selector:
    app: nginx
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: web
spec:
  selector:
    matchLabels:
      app: nginx
  serviceName: "nginx"
  replicas: 2
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
          name: web
        volumeMounts:
        - name: disk-ssd
          mountPath: /data
  volumeClaimTemplates:
  - metadata:
      name: disk-ssd
    spec:
      accessModes: [ "ReadWriteOnce" ]
      storageClassName: "alicloud-disk-ssd"
      resources:
        requests:
          storage: 20Gi
```

- `replicas`：本例中设置为2，表示创建2个Pod。
- `mountPath`：云盘在容器中挂载的位置。
- `accessModes`：访问模式。
- `storageClassName`：本例配置为`alicloud-disk-ssd`，表示使用的是阿里SSD类型的云盘。

- `storage`：声明应用的使用量。

2. 执行以下命令，部署StatefulSet服务。

```
kubectl create -f statefulset.yaml
```

3. 执行以下命令，查看已部署的Pod。

```
kubectl get pod
```

预期输出：

```
NAME                        READY   STATUS    RESTARTS   AGE
web-0                       1/1     Running   0          6m
web-1                       1/1     Running   0          6m
```

4. 执行以下命令，查看PVC。

```
kubectl get pvc
```

预期输出：

```
NAME             STATUS   VOLUME                   CAPACITY   ACCESS MODES   STORAGECLA
SS        AGE
disk-ssd-web-0   Bound    d-2zegw7et6xc96nbojuoo   20Gi       RWO            alicloud-d
isk-ssd   7m
disk-ssd-web-1   Bound    d-2zefbrqggvkd10xb523h   20Gi       RWO            alicloud-d
isk-ssd   6m
```

验证PVC会根据StatefulSet服务扩容而扩容

1. 执行以下命令，扩容StatefulSet服务到3个Pod。

```
kubectl scale sts web --replicas=3
```

预期输出：

```
statefulset.apps/web scaled
```

2. 执行以下命令，查看扩容后的Pod。

```
kubectl get pod
```

预期输出：

```
NAME                        READY   STATUS    RESTARTS   AGE
web-0                       1/1     Running   0          34m
web-1                       1/1     Running   0          33m
web-2                       1/1     Running   0          26m
```

3. 执行以下命令，查看扩容后的PVC。

```
kubectl get pvc
```

预期输出：

```
NAME             STATUS   VOLUME                   CAPACITY   ACCESS MODES   STORAGECLA
SS        AGE
disk-ssd-web-0   Bound    d-2zegw7et6xc96nbojuoo   20Gi       RWO            alicloud-d
isk-ssd   35m
disk-ssd-web-1   Bound    d-2zefbrqggvkd10xb523h   20Gi       RWO            alicloud-d
isk-ssd   34m
disk-ssd-web-2   Bound    d-2ze4jx1zymn4n9j3pic2   20Gi       RWO            alicloud-d
isk-ssd   27m
```

可以看到，StatefulSet扩容到3个Pod后，PVC也扩容到3个。

验证缩容StatefulSet后，PVC保持不变

1. 执行以下命令，缩减StatefulSet服务到2个Pod。

```
kubectl scale sts web --replicas=2
```

预期输出：

```
statefulset.apps/web scaled
```

2. 执行以下命令，查看缩容后的Pod。

```
kubectl get pod
```

预期输出：

```
NAME                         READY   STATUS    RESTARTS   AGE
web-0                        1/1     Running   0          38m
web-1                        1/1     Running   0          38m
```

Pod的数量已缩减为2。

3. 执行以下命令，查看缩容后的PVC。

```
kubectl get pvc
```

预期输出：

```
NAME             STATUS   VOLUME                   CAPACITY   ACCESS MODES   STORAGECLA
SS        AGE
disk-ssd-web-0   Bound    d-2zegw7et6xc96nbojuoo   20Gi       RWO            alicloud-d
isk-ssd   39m
disk-ssd-web-1   Bound    d-2zefbrqggvkd10xb523h   20Gi       RWO            alicloud-d
isk-ssd   39m
disk-ssd-web-2   Bound    d-2ze4jx1zymn4n9j3pic2   20Gi       RWO            alicloud-d
isk-ssd   31m
```

缩容StatefulSet到2个Pod后，PVC仍为3个。说明PVC不会因StatefulSet缩容而缩减。

验证StatefulSet再次扩容后，PVC保持不变

验证StatefulSet扩容后缩容，再次扩容后，PVC保持不变。

1. 执行以下命令，扩容StatefulSet服务到3个Pod。

```
kubectl scale sts web --replicas=3
```

预期输出：

```
statefulset.apps/web scaled
```

2. 执行以下命令，查看扩容后的Pod。

```
kubectl get pod
```

预期输出：

```
NAME                          READY   STATUS    RESTARTS   AGE
web-0                         1/1     Running   0          1h
web-1                         1/1     Running   0          1h
web-2                         1/1     Running   0          8s
```

3. 执行以下命令，查看扩容后的PVC。

```
kubectl get pvc
```

预期输出：

```
NAME             STATUS   VOLUME                   CAPACITY   ACCESS MODES   STORAGECLA
SS        AGE
disk-ssd-web-0   Bound    d-2zegw7et6xc96nbojuoo   20Gi       RWO            alicloud-d
isk-ssd   1h
disk-ssd-web-1   Bound    d-2zefbrqggvkd10xb523h   20Gi       RWO            alicloud-d
isk-ssd   1h
disk-ssd-web-2   Bound    d-2ze4jx1zymn4n9j3pic2   20Gi       RWO            alicloud-d
isk-ssd   1h
```

扩容后新创建的Pod仍会使用原来的PVC。

验证删除StatefulSet服务的Pod，PVC保持不变

1. 执行以下命令，查看名称为*web-1*的Pod，引用的PVC。

```
kubectl describe pod web-1 | grep ClaimName
```

预期输出：

```
ClaimName:  disk-ssd-web-1
```

2. 执行以下命令，删除名称为*web-1*的Pod。

```
kubectl delete pod web-1
```

预期输出：

```
pod "web-1" deleted
```

3. 执行以下命令，查看Pod。

```
kubectl get pod
```

预期输出：

```
NAME                          READY   STATUS    RESTARTS   AGE
web-0                         1/1     Running   0          1h
web-1                         1/1     Running   0          25s
web-2                         1/1     Running   0          9m
```

重新创建的Pod与删除前的Pod名称一致。

4. 执行以下命令，查看PVC。

```
kubectl get pvc
```

预期输出：

```
NAME             STATUS   VOLUME                   CAPACITY   ACCESS MODES   STORAGECLA
SS        AGE
disk-ssd-web-0   Bound    d-2zegw7et6xc96nbojuoo   20Gi       RWO            alicloud-d
isk-ssd   1h
disk-ssd-web-1   Bound    d-2zefbrqggvkd10xb523h   20Gi       RWO            alicloud-d
isk-ssd   1h
disk-ssd-web-2   Bound    d-2ze4jx1zymn4n9j3pic2   20Gi       RWO            alicloud-d
isk-ssd   1h
```

删除后新创建的Pod仍会使用原来的PVC。

验证StatefulSet服务的持久化存储

1. 执行以下命令，查看*/data*路径下的文件。

```
kubectl exec web-1 ls /data
```

预期输出：

```
lost+found
```

2. 执行以下命令，在*/data*路径下创建文件*statefulset*。

```
kubectl exec web-1 touch /data/statefulset
```

3. 执行以下命令，查看*/data*路径下的文件。

```
kubectl exec web-1 ls /data
```

预期输出：

```
lost+found
statefulset
```

4. 执行以下命令，删除名称为*web-1*的Pod。

```
kubectl delete pod web-1
```

预期输出：

```
pod "web-1" deleted
```

5. 执行以下命令，查看*/data*路径下的文件。

```
kubectl exec web-1 ls /data
```

预期输出：

```
lost+found
statefulset
```

*statefulset*文件仍然存在，说明云盘的数据可持久化保存。

5.2. 实现StatefulSet持久化存储的最佳实践-Flexvolume

有状态服务StatefulSet支持通过VolumeClaimTemplate为每个Pod创建PV和PVC。并且删除和减少Pod时，不会删除StatefulSet的PV和PVC。本文为您介绍如何通过VolumeClaimTemplate实现StatefulSet持久化存储。

背景信息

有状态服务-StatefulSet的应用场景：

- 稳定的部署次序：有序部署或扩展，需要根据定义的顺序依次进行（即从0到N-1，在下一个Pod运行之前，所有之前的Pod必须都是Running和Ready状态）。
- 稳定的扩展次序：有序收缩或删除，需要根据定义的顺序依次进行（即从N-1到0，在下一个Pod运行之前，所有之前的Pod必须都是Running和Ready状态）。
- 稳定的网络标志：Pod重新调度后其PodName和HostName不变。
- 稳定的持久化存储：基于PVC，Pod重新调度后仍能访问到相同的持久化数据。

有状态服务-StatefulSet的使用方式：

通过 `volumeClaimTemplates` 自动创建PVC及PV。

文本将介绍：

- 创建StatefulSet服务
- 伸缩StatefulSet服务
- 删除StatefulSet服务
- StatefulSet服务的持久化存储

前提条件

- 创建Kubernetes托管版集群
- 通过kubectl工具连接集群

部署StatefulSet服务

> **说明** `volumeClaimTemplates`：表示一类PVC的模板，系统会根据有状态服务-StatefulSet配置的replicas数量，创建相应数量的PVC。这些PVC除了名字不一样之外其他配置都是一样的。

1. 创建*statefulset.yaml*文件。

   > **说明** `storageClassName` 配置为*alicloud-disk-ssd*，表示使用的是阿里SSD类型的云盘。

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
  labels:
    app: nginx
spec:
  ports:
  - port: 80
    name: web
  clusterIP: None
  selector:
    app: nginx
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: web
spec:
  selector:
    matchLabels:
      app: nginx
  serviceName: "nginx"
  replicas: 2
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        ports:
        - containerPort: 80
          name: web
        volumeMounts:
        - name: disk-ssd
          mountPath: /data
  volumeClaimTemplates:
  - metadata:
      name: disk-ssd
    spec:
      accessModes: [ "ReadWriteOnce" ]
      storageClassName: "alicloud-disk-ssd"
      resources:
        requests:
          storage: 20Gi
```

2. 执行以下命令，部署StatefulSet服务。

```
kubectl create -f statefulset.yaml
```

3. 同时在另一个窗口中，执行以下命令，查看Pod按照次序部署。

```
kubectl get pod -w -l app=nginx
```

预期输出：

```
NAME          READY   STATUS    RESTARTS   AGE
web-0         0/1     Pending   0          0s
web-0         0/1     Pending   0          0s
web-0         0/1     ContainerCreating   0          0s
web-0         1/1     Running   0          20s
web-1         0/1     Pending   0          0s
web-1         0/1     Pending   0          0s
web-1         0/1     ContainerCreating   0          0s
web-1         1/1     Running   0          7s
```

4. 执行以下命令，查看已部署的Pod。

```
kubectl get pod
```

预期输出：

```
NAME                          READY   STATUS    RESTARTS   AGE
web-0                         1/1     Running   0          6m
web-1                         1/1     Running   0          6m
```

5. 执行以下命令，查看PVC。

```
kubectl get pvc
```

预期输出：

```
NAME             STATUS   VOLUME                   CAPACITY   ACCESS MODES   STORAGECLA
SS        AGE
disk-ssd-web-0   Bound    d-2zegw7et6xc96nbojuoo   20Gi       RWO            alicloud-d
isk-ssd   7m
disk-ssd-web-1   Bound    d-2zefbrqggvkd10xb523h   20Gi       RWO            alicloud-d
isk-ssd   6m
```

伸缩StatefulSet服务

扩容StatefulSet服务

1. 执行以下命令，扩容StatefulSet服务到3个Pod。

```
kubectl scale sts web --replicas=3
```

预期输出：

```
statefulset.apps/web scaled
```

2. 执行以下命令，查看扩容后的Pod。

```
kubectl get pod
```

预期输出：

```
NAME                          READY   STATUS    RESTARTS   AGE
web-0                         1/1     Running   0          34m
web-1                         1/1     Running   0          33m
web-2                         1/1     Running   0          26m
```

3. 执行以下命令，查看扩容后的PVC。

```
kubectl get pvc
```

预期输出：

```
NAME             STATUS   VOLUME                   CAPACITY   ACCESS MODES   STORAGECLA
SS        AGE
disk-ssd-web-0   Bound    d-2zegw7et6xc96nbojuoo   20Gi       RWO            alicloud-d
isk-ssd   35m
disk-ssd-web-1   Bound    d-2zefbrqggvkd10xb523h   20Gi       RWO            alicloud-d
isk-ssd   34m
disk-ssd-web-2   Bound    d-2ze4jx1zymn4n9j3pic2   20Gi       RWO            alicloud-d
isk-ssd   27m
```

缩容StatefulSet服务

1. 执行以下命令，缩减StatefulSet服务到2个Pod。

```
kubectl scale sts web --replicas=2
```

预期输出：

```
statefulset.apps/web scaled
```

2. 执行以下命令，查看缩容后的Pod。

```
kubectl get pod
```

预期输出：

```
NAME                          READY   STATUS    RESTARTS   AGE
web-0                         1/1     Running   0          38m
web-1                         1/1     Running   0          38m
```

Pod的数量已缩减为2。

3. 执行以下命令，查看缩容后的PVC。

```
kubectl get pvc
```

预期输出：

```
NAME             STATUS   VOLUME                   CAPACITY   ACCESS MODES   STORAGECLA
SS        AGE
disk-ssd-web-0   Bound    d-2zegw7et6xc96nbojuoo   20Gi       RWO            alicloud-d
isk-ssd   39m
disk-ssd-web-1   Bound    d-2zefbrqggvkd10xb523h   20Gi       RWO            alicloud-d
isk-ssd   39m
disk-ssd-web-2   Bound    d-2ze4jx1zymn4n9j3pic2   20Gi       RWO            alicloud-d
isk-ssd   31m
```

PVC和PV并没有随Pod一起缩减。

再次扩容StatefulSet服务

1. 执行以下命令，扩容StatefulSet服务到3个Pod。

```
kubectl scale sts web --replicas=3
```

预期输出：

```
statefulset.apps/web scaled
```

2. 执行以下命令，查看扩容后的Pod。

```
kubectl get pod
```

预期输出：

```
NAME                          READY   STATUS    RESTARTS   AGE
web-0                         1/1     Running   0          1h
web-1                         1/1     Running   0          1h
web-2                         1/1     Running   0          8s
```

3. 执行以下命令，查看扩容后的PVC。

```
kubectl get pvc
```

预期输出：

```
NAME             STATUS   VOLUME                   CAPACITY   ACCESS MODES   STORAGECLA
SS        AGE
disk-ssd-web-0   Bound    d-2zegw7et6xc96nbojuoo   20Gi       RWO            alicloud-d
isk-ssd   1h
disk-ssd-web-1   Bound    d-2zefbrqggvkd10xb523h   20Gi       RWO            alicloud-d
isk-ssd   1h
disk-ssd-web-2   Bound    d-2ze4jx1zymn4n9j3pic2   20Gi       RWO            alicloud-d
isk-ssd   1h
```

扩容后新创建的Pod仍会使用原来的PVC和PV。

删除StatefulSet服务

1. 执行以下命令，查看名称为*web-1*的Pod，引用的PVC。

```
kubectl describe pod web-1 | grep ClaimName
```

预期输出：

```
ClaimName:  disk-ssd-web-1
```

2. 执行以下命令，删除名称为*web-1*的Pod。

```
kubectl delete pod web-1
```

预期输出：

```
pod "web-1" deleted
```

3. 执行以下命令，查看Pod。

```
kubectl get pod
```

预期输出：

```
NAME                          READY   STATUS    RESTARTS   AGE
web-0                         1/1     Running   0          1h
web-1                         1/1     Running   0          25s
web-2                         1/1     Running   0          9m
```

重新创建的Pod与删除前的Pod名称一致。

4. 执行以下命令，查看PVC。

```
kubectl get pvc
```

预期输出：

```
NAME             STATUS   VOLUME                   CAPACITY   ACCESS MODES   STORAGECLA
SS        AGE
disk-ssd-web-0   Bound    d-2zegw7et6xc96nbojuoo   20Gi       RWO            alicloud-d
isk-ssd   1h
disk-ssd-web-1   Bound    d-2zefbrqggvkd10xb523h   20Gi       RWO            alicloud-d
isk-ssd   1h
disk-ssd-web-2   Bound    d-2ze4jx1zymn4n9j3pic2   20Gi       RWO            alicloud-d
isk-ssd   1h
```

重新创建的Pod所使用的PVC与删除前的Pod一致。

5. 同时在另一个窗口中，执行以下命令，查看Pod删除和重新创建的过程。

```
kubectl get pod -w -l app=nginx
```

预期输出：

```
NAME    READY   STATUS    RESTARTS   AGE
web-0   1/1     Running   0          102m
web-1   1/1     Running   0          69s
web-2   1/1     Running   0          10m
web-1   1/1   Terminating   0     89s
web-1   0/1   Terminating   0     89s
web-1   0/1   Terminating   0     90s
web-1   0/1   Terminating   0     90s
web-1   0/1   Pending   0     0s
web-1   0/1   Pending   0     0s
web-1   0/1   ContainerCreating   0     0s
web-1   1/1   Running   0     20s
```

StatefulSet服务的持久化存储

1. 执行以下命令，查看*/data*路径下的文件。

```
kubectl exec web-1 ls /data
```

预期输出：

```
lost+found
```

2. 执行以下命令，在*/data*路径下创建文件*statefulset*。

```
kubectl exec web-1 touch /data/statefulset
```

3. 执行以下命令，查看*/data*路径下的文件。

```
kubectl exec web-1 ls /data
```

预期输出：

```
lost+found
statefulset
```

4. 执行以下命令，删除名称为*web-1*的Pod。

```
kubectl delete pod web-1
```

预期输出：

```
pod "web-1" deleted
```

5. 执行以下命令，查看*/data*路径下的文件，刚刚创建的文件*statefulset*并没有被删除，说明云盘的数据可持久保存。

```
kubectl exec web-1 ls /data
```

预期输出：

```
lost+found
statefulset
```

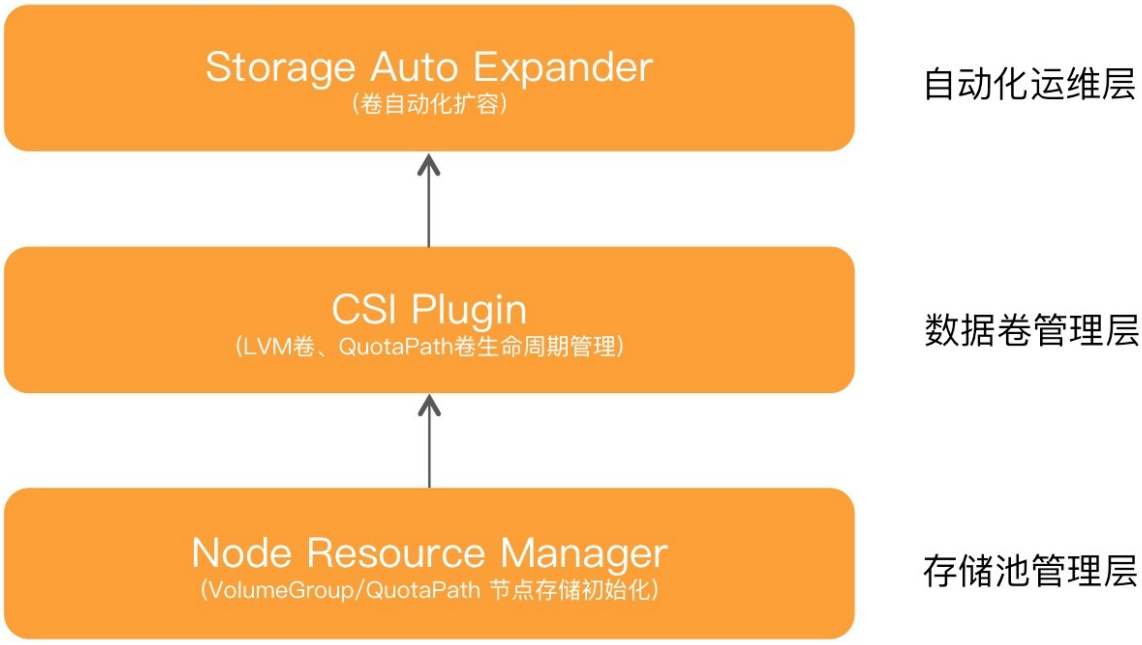
5.3. 本地存储卷的初始化、挂载与自动扩容

阿里云容器服务ACK支持本地存储，提供自动化的逻辑卷生命周期管理能力，且能根据节点本地存储容量进行调度。您可通过PVC/PV方式使用LVM及Quotapath本地存储卷。本文介绍如何在ACK集群中初始化本地存储、本地资源的挂载使用及如何配置本地存储卷自动扩容。

前提条件

已创建Kubernetes集群。具体操作，请参见创建Kubernetes托管版集群。

背景信息



本地存储架构主要由以下三部分组成：

结构	说明
Node Resource Manager	负责维护本地存储的初始化周期。以VolumeGroup为例， 当您在ConfigMap上声明了VolumeGroup相关组成信息时，NodeResourceManager会根据ConfigMap上的定义初始化当前节点上的本地存储。
CSI Plugin	负责维护本地存储卷的生命周期。以LVM为例，当用户创建使用了某个VolumeGroup的PVC时，CSI Plugin会自动生成LogicVolume并绑定到PVC。
Storage Auto Expandor	负责管理本地存储卷的自动扩容，当监控发现当前本地存储卷容量不足时，会自动对本地存储卷扩容。

关于本地存储卷的更多信息，请参见本地存储卷概述。

步骤一：初始化本地存储

通过Node Resource Manager组件初始化本地存储， 通过读取ConfigMap管理集群内所有节点的本地存储，即可以为每个节点自动化创建VolumeGroup及ProjectQuota的根目录。更多信息，请参见node-resource-manager。

1. 在集群的ConfigMap中定义本地存储卷。

   使用以下内容创建ConfigMap。

```
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: ConfigMap
metadata:
  name: node-resource-topo
  namespace: kube-system
data:
  volumegroup: |-
    volumegroup:
    - name: volumegroup1
      key: kubernetes.io/hostname
      operator: In
      value: cn-zhangjiakou.192.168.XX.XX
      topology:
        type: device
        devices:
        - /dev/vdb
        - /dev/vdc
  quotapath: |-
    quotapath:
    - name: /mnt/path1
      key: kubernetes.io/hostname
      operator: In
      value: cn-zhangjiakou.192.168.XX.XX
      topology:
        type: device
        options: prjquota
        fstype: ext4
        devices:
        - /dev/vdd
EOF
```

以上的ConfigMap定义了在 `cn-zhangjiakou.192.168.XX.XX` 节点创建以下两种本地存储卷资源。

- 创建名为 `volumegroup1` 的VolumeGroup资源， 该VolumeGroup资源由宿主机上的*/dev/vdb*及*/dev/vdc*两个块设备组成。
- 创建Quotapath资源， 该Quotapath由*/dev/vdd*块设备格式化挂载在*/mnt/path1*路径下。此处只能声明一个devices。

节点的筛选：通过key、value的方式匹配节点，可以精确到一个节点，也可以是一批节点。

2. 在集群中部署Node Resource Manager组件。

   使用以下内容部署Node Resource Manager组件。

```
cat <<EOF | kubectl apply -f -
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: node-resource-manager
  namespace: kube-system
---
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
```

```
  name: node-resource-manager
rules:
  - apiGroups: [""]
    resources: ["configmaps"]
    verbs: ["get", "watch", "list", "delete", "update", "create"]
  - apiGroups: [""]
    resources: ["nodes"]
    verbs: ["get", "list", "watch"]
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: node-resource-manager-binding
subjects:
  - kind: ServiceAccount
    name: node-resource-manager
    namespace: kube-system
roleRef:
  kind: ClusterRole
  name: node-resource-manager
  apiGroup: rbac.authorization.k8s.io
---
kind: DaemonSet
apiVersion: apps/v1
metadata:
  name: node-resource-manager
  namespace: kube-system
spec:
  selector:
    matchLabels:
      app: node-resource-manager
  template:
    metadata:
      labels:
        app: node-resource-manager
    spec:
      tolerations:
        - operator: "Exists"
      priorityClassName: system-node-critical
      serviceAccountName: node-resource-manager
      hostNetwork: true
      hostPID: true
      containers:
        - name: node-resource-manager
          securityContext:
            privileged: true
            capabilities:
              add: ["SYS_ADMIN"]
            allowPrivilegeEscalation: true
          image: registry.cn-hangzhou.aliyuncs.com/acs/node-resource-manager:v1.18.8.0-
5b1bdc2-aliyun
          imagePullPolicy: "Always"
          args:
            - "--nodeid=$(KUBE_NODE_NAME)"
```

```
          env:
            - name: KUBE_NODE_NAME
              valueFrom:
                fieldRef:
                  apiVersion: v1
                  fieldPath: spec.nodeName
          volumeMounts:
            - mountPath: /dev
              mountPropagation: "HostToContainer"
              name: host-dev
            - mountPath: /var/log/
              name: host-log
            - name: etc
              mountPath: /host/etc
            - name: config
              mountPath: /etc/unified-config
      volumes:
        - name: host-dev
          hostPath:
            path: /dev
        - name: host-log
          hostPath:
            path: /var/log/
        - name: etc
          hostPath:
            path: /etc
        - name: config
          configMap:
            name: node-resource-topo
EOF
```

3. 安装csi-plugin组件。

   csi-plugin组件提供了数据卷的全生命周期管理，包括数据卷的创建、挂载、卸载、删除及扩容等服务。使用以下内容安装csi-plugin组件。

```
cat <<EOF | kubectl apply -f -
apiVersion: storage.k8s.io/v1
kind: CSIDriver
metadata:
  name: localplugin.csi.alibabacloud.com
spec:
  attachRequired: false
  podInfoOnMount: true
---
apiVersion: apps/v1
kind: DaemonSet
metadata:
  labels:
    app: csi-local-plugin
  name: csi-local-plugin
  namespace: kube-system
spec:
  selector:
    matchLabels:
```

```
      app: csi-local-plugin
  template:
    metadata:
      labels:
        app: csi-local-plugin
    spec:
      containers:
      - args:
        - --v=5
        - --csi-address=/csi/csi.sock
        - --kubelet-registration-path=/var/lib/kubelet/csi-plugins/localplugin.csi.alib
abacloud.com/csi.sock
        env:
        - name: KUBE_NODE_NAME
          valueFrom:
            fieldRef:
              apiVersion: v1
              fieldPath: spec.nodeName
        image: registry.cn-hangzhou.aliyuncs.com/acs/csi-node-driver-registrar:v1.2.0
        imagePullPolicy: Always
        name: driver-registrar
        volumeMounts:
        - mountPath: /csi
          name: plugin-dir
        - mountPath: /registration
          name: registration-dir
      - args:
        - --endpoint=$(CSI_ENDPOINT)
        - --v=5
        - --nodeid=$(KUBE_NODE_NAME)
        - --driver=localplugin.csi.alibabacloud.com
        env:
        - name: KUBE_NODE_NAME
          valueFrom:
            fieldRef:
              apiVersion: v1
              fieldPath: spec.nodeName
        - name: SERVICE_PORT
          value: "11290"
        - name: CSI_ENDPOINT
          value: unix://var/lib/kubelet/csi-plugins/localplugin.csi.alibabacloud.com/cs
i.sock
        image: registry.cn-hangzhou.aliyuncs.com/acs/csi-plugin:v1.14.8-40ee9518-local
        imagePullPolicy: Always
        name: csi-localplugin
        securityContext:
          allowPrivilegeEscalation: true
          capabilities:
            add:
            - SYS_ADMIN
          privileged: true
        volumeMounts:
        - mountPath: /var/lib/kubelet
          mountPropagation: Bidirectional
```

```
            name: pods-mount-dir
          - mountPath: /dev
            mountPropagation: HostToContainer
            name: host-dev
          - mountPath: /var/log/
            name: host-log
          - mountPath: /mnt
            mountPropagation: Bidirectional
            name: quota-path-dir
        hostNetwork: true
        hostPID: true
        serviceAccount: csi-admin
        tolerations:
        - operator: Exists
        volumes:
        - hostPath:
            path: /var/lib/kubelet/csi-plugins/localplugin.csi.alibabacloud.com
            type: DirectoryOrCreate
          name: plugin-dir
        - hostPath:
            path: /var/lib/kubelet/plugins_registry
            type: DirectoryOrCreate
          name: registration-dir
        - hostPath:
            path: /var/lib/kubelet
            type: Directory
          name: pods-mount-dir
        - hostPath:
            path: /dev
            type: ""
          name: host-dev
        - hostPath:
            path: /var/log/
            type: ""
          name: host-log
        - hostPath:
            path: /mnt
            type: Directory
          name: quota-path-dir
    updateStrategy:
      rollingUpdate:
        maxUnavailable: 10%
      type: RollingUpdate
EOF
```

步骤二：使用本地存储卷创建应用

方式一：使用LVM本地存储卷创建应用

1. 创建StorageClass。

   使用以下内容创建StorageClass。

```
cat <<EOF | kubectl apply -f -
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
    name: csi-local
provisioner: localplugin.csi.alibabacloud.com
parameters:
    volumeType: LVM
    vgName: volumegroup1
    fsType: ext4
    lvmType: "striping"
reclaimPolicy: Delete
volumeBindingMode: WaitForFirstConsumer
allowVolumeExpansion: true
EOF
```

parameters.vgName为在node-resource-topo configmap中定义的VolumeGroup名称 `volumegroup1` 。更多信息，请参见LVM数据卷。

2. 创建PVC。

```
cat << EOF | kubectl apply -f -
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: lvm-pvc
  annotations:
    volume.kubernetes.io/selected-node: cn-zhangjiakou.192.168.XX.XX
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
  storageClassName: csi-local
EOF
```

annotations下的 `volume.kubernetes.io/selected-node: cn-zhangjiakou.192.168.XX.XX` 表明后续关联PVC的Pod将会被调度到 `cn-zhangjiakou.192.168.XX.XX` 节点上，该节点是之前在ConfigMap中定义的VolumeGroups资源所在节点。

3. 创建应用。

使用以下内容创建应用。

```
cat << EOF | kubectl apply -f -
apiVersion: apps/v1
kind: Deployment
metadata:
  name: deployment-lvm
  labels:
    app: nginx
spec:
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        volumeMounts:
          - name: lvm-pvc
            mountPath: "/data"
      volumes:
        - name: lvm-pvc
          persistentVolumeClaim:
            claimName: lvm-pvc
EOF
```

等待应用启动，容器中的*/data*目录容量等于PVC声明的容量2 GiB。

方式二：使用Quotapath本地存储卷创建应用

1. 创建StorageClass。

   使用以下内容创建StorageClass。

   ```
   cat << EOF | kubectl apply -f -
   apiVersion: storage.k8s.io/v1
   kind: StorageClass
   metadata:
     name: alicloud-local-quota
   parameters:
     volumeType: QuotaPath
     rootPath: /mnt/path1
   provisioner: localplugin.csi.alibabacloud.com
   reclaimPolicy: Delete
   allowVolumeExpansion: true
   volumeBindingMode: WaitForFirstConsumer
   EOF
   ```

   parameters.rootPath为在node-resource-topo configmap中定义的Quotapath资源的名称`/mnt/path1`。更多信息，请参见QuotaPath数据卷。

2. 创建PVC。

```
cat << EOF | kubectl apply -f -
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: csi-quota
  annotations:
    volume.kubernetes.io/selected-node: cn-zhangjiakou.192.168.XX.XX
  labels:
    app: web-quota
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
  storageClassName: alicloud-local-quota
EOF
```

annotations下的`volume.kubernetes.io/selected-node: cn-zhangjiakou.192.168.XX.XX`表明后续关联PVC的Pod将会被调度到`cn-zhangjiakou.192.168.XX.XX`节点上，该节点是之前在ConfigMap中定义的Quotapath资源所在节点。

3. 创建应用。

   使用以下内容创建应用。

```
cat << EOF | kubectl apply -f -
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: web-quota
spec:
  selector:
    matchLabels:
      app: nginx
  serviceName: "nginx"
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
        volumeMounts:
        - name: disk-ssd
          mountPath: /data
      volumes:
        - name: "disk-ssd"
          persistentVolumeClaim:
            claimName: csi-quota
EOF
```

   等待应用启动，容器中的*/data*目录容量等于PVC声明的容量2 GiB。

步骤三：自动扩容本地存储卷

1. 安装自动扩容插件。更多信息，请参见使用storage-operator进行存储组件的部署与升级。

   i. 执行以下命令配置storage-operator。

   ```
   kubectl edit cm storage-operator -n kube-system
   ```

   预期输出：

   ```
     storage-auto-expander: |
       # deploy config
       install: true
       imageTag: "v1.18.8.1-d4301ee-aliyun"
   ```

   ii. 执行以下命令检查自动扩容组件是否启动。

   ```
   kubectl get pod -n kube-system |grep storage-auto-expander
   ```

   预期输出：

   ```
   storage-auto-expander-6bb575b68c-tt4hh                1/1     Running     0
   2m41s
   ```

2. 配置自动扩容策略。

   使用以下内容配置自动扩容策略。

   ```
   cat << EOF | kubectl apply -f -
   apiVersion: storage.alibabacloud.com/v1alpha1
   kind: StorageAutoScalerPolicy
   metadata:
     name: hybrid-expand-policy
   spec:
     pvcSelector:
       matchLabels:
         app: web-quota
     namespaces:
       - default
     conditions:
       - name: condition1
         key: volume-capacity-used-percentage
         operator: Gt
         values:
           - "80"
     actions:
       - name: action
         type: volume-expand
         params:
           scale: 50%
           limits: 15Gi
   EOF
   ```

   从以上命令模板可以看出，当存储容量达到80%以上时执行数据卷扩容，每次扩容50%，最大扩容到15 GiB。

5.4. 通过CNFS自动收集异常退出的JVM转储文件

当您的业务是使用Java开发，且设置的JVM堆空间过小时，程序会发生OOM（Out Of Memory）的问题。此时您可以使用CNFS（Container Network File System）作为记录日志的载体，挂载到容器内相应目录中，当JVM发生OOM时，CNFS可以将日志记录到相应的目录里。本文介绍如何使用CNFS自动收集异常退出的JVM转储文件。

前提条件

- 已创建Kubernetes集群，且存储插件选择为CSI。具体操作，请参见创建Kubernetes托管版集群。
- 已创建容器镜像服务企业版实例。具体操作，请参见创建企业版实例。
- 已使用CNFS托管NAS文件系统。具体操作，请参见使用CNFS托管NAS文件系统。

背景信息

- 阿里云容器服务使用容器网络文件系统CNFS，将阿里云的文件存储抽象为一个K8s对象（CRD）进行独立管理，包括创建、删除、描述、挂载，监控及扩容等运维操作。更多信息，请参见容器网络文件系统CNFS概述。
- 阿里云容器镜像服务ACR（Alibaba Cloud Container Registry）是面向容器镜像、Helm Chart等符合OCI标准的云原生制品安全托管及高效分发平台。更多信息，请参见什么是容器镜像服务ACR。

注意事项

- Java设置的最大Heap值（Xmx）应小于Pod Memory的Limit值，防止JVM未发生OOM，但Pod发生OOM的情况。
- 在使用Java转储时，建议创建个新的CNFS，将业务使用的CNFS与Java转储的CNFS分开，防止*.hprof*文件过大，转储时占用大量业务资源，影响业务。

操作步骤

1. 您可以使用*registry.cn-hangzhou.aliyuncs.com/acs1/java-oom-test:v1.0*镜像作为模拟OOM的Java程序，用于触发JVM的OOM。

   关于如何构建镜像，请参见使用企业版实例构建镜像。

2. 使用以下示例，创建一个名称为*java-application*的Deployment。

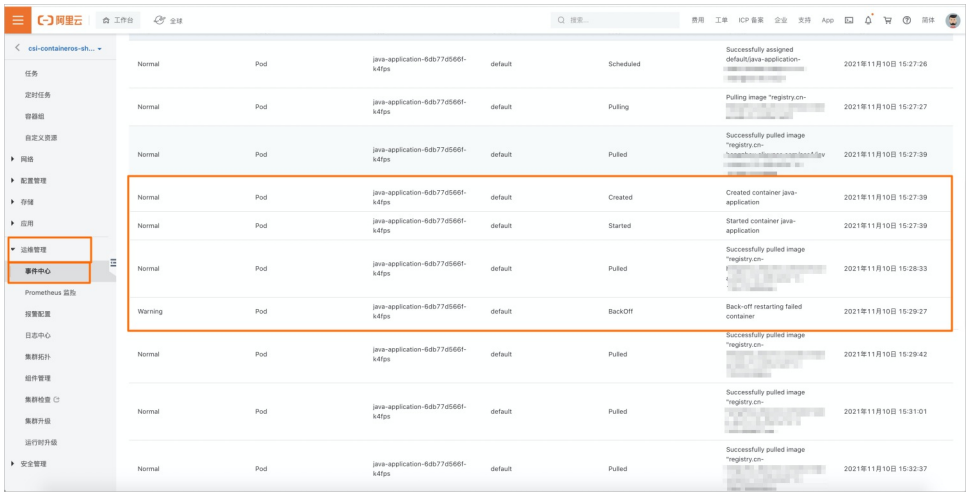
   本示例在启动Java程序Mycode时，设置申请的堆大小为80 MB，堆转储的目录为*/mnt/oom/logs*。当JVM的堆大小不满足时，捕获HeapDumpOnOutOfMemoryError错误。

```
cat << EOF | kubectl apply -f -
apiVersion: apps/v1
kind: Deployment
metadata:
  name: java-application
spec:
  selector:
    matchLabels:
      app: java-application
  template:
    metadata:
      labels:
        app: java-application
    spec:
      containers:
      - name: java-application
        image: registry.cn-hangzhou.aliyuncs.com/acs1/java-oom-test:v1.0  #本文示例程序的镜像地址。
        imagePullPolicy: Always
        env:                                #定义两个键值：POD_NAME为metadata.name，POD_NAMESPACE为metadata.namespace。
        - name: POD_NAME
          valueFrom:
            fieldRef:
              apiVersion: v1
              fieldPath: metadata.name
        - name: POD_NAMESPACE
          valueFrom:
            fieldRef:
              apiVersion: v1
              fieldPath: metadata.namespace
        args:
        - java                              #执行命令。
        - -Xms80m                           #堆内存的最小Heap值。
        - -Xmx80m                           #堆内存的最大Heap值。
        - -XX:HeapDumpPath=/mnt/oom/logs    #发生OOM时，堆内存转储的路径。
        - -XX:+HeapDumpOnOutOfMemoryError   #捕获堆发生OOM的错误。
        - Mycode                            #执行程序。
        volumeMounts:
        - name: java-oom-pv
          mountPath: "/mnt/oom/logs"        #容器内部使用/mnt/oom/logs做为挂载目录。
          subPathExpr: $(POD_NAMESPACE).$(POD_NAME)    #使用$(POD_NAMESPACE).$(POD_NAME)作为创建出子目录，将OOM转储文件生成到子目录中。
      volumes:
      - name: java-oom-pv
        persistentVolumeClaim:
          claimName: cnfs-nas-pvc           #使用CNFS的PVC，名称为cnfs-nas-pvc。
EOF
```

3. 通过容器服务管理控制台的事件中心，查看到该Pod发生了Back-off restarting的告警事件，说明*java-application*应用发生了OOM。
   i. 登录容器服务管理控制台。
   ii. 在控制台左侧导航栏中，单击**集群**。

iii. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。

iv. 在集群管理页左侧导航栏中，选择**运维管理 > 事件中心**。



4. 由于NAS目前没有浏览、上传、下载文件的功能，您可以使用File Browser作为Web端的访问工具。首先将NAS的挂载点挂载到File Browser的*root Dir*上，然后通过 `kubectl port-forward` 命令，将File Browser的容器端口映射到本地，再通过浏览器访问存放在NAS上的文件。

   i. 使用以下模板，创建File Browser的Deployment和File Browser需要使用的ConfigMap，默认开启80端口。

```
cat << EOF | kubectl apply -f -
apiVersion: v1
data:
  .filebrowser.json: |
    {
      "port": 80
    }
kind: ConfigMap
metadata:
  labels:
    app.kubernetes.io/instance: filebrowser
    app.kubernetes.io/name: filebrowser
  name: filebrowser
  namespace: default
---
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app.kubernetes.io/instance: filebrowser
    app.kubernetes.io/name: filebrowser
  name: filebrowser
  namespace: default
spec:
  progressDeadlineSeconds: 600
  replicas: 1
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app.kubernetes.io/instance: filebrowser
```

```
      app.kubernetes.io/name: filebrowser
  template:
    metadata:
      labels:
        app.kubernetes.io/instance: filebrowser
        app.kubernetes.io/name: filebrowser
    spec:
      containers:
      - image: docker.io/filebrowser/filebrowser:v2.18.0
        imagePullPolicy: IfNotPresent
        name: filebrowser
        ports:
        - containerPort: 80
          name: http
          protocol: TCP
        resources: {}
        securityContext: {}
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
        volumeMounts:
        - mountPath: /.filebrowser.json
          name: config
          subPath: .filebrowser.json
        - mountPath: /db
          name: rootdir
        - mountPath: /rootdir
          name: rootdir
      dnsPolicy: ClusterFirst
      restartPolicy: Always
      schedulerName: default-scheduler
      securityContext: {}
      terminationGracePeriodSeconds: 30
      volumes:
      - configMap:
          defaultMode: 420
          name: filebrowser
        name: config
      - name: rootdir
        persistentVolumeClaim:
          claimName: cnfs-nas-pvc
EOF
```

预期输出：

```
configmap/filebrowser unchanged
deployment.apps/filebrowser configured
```

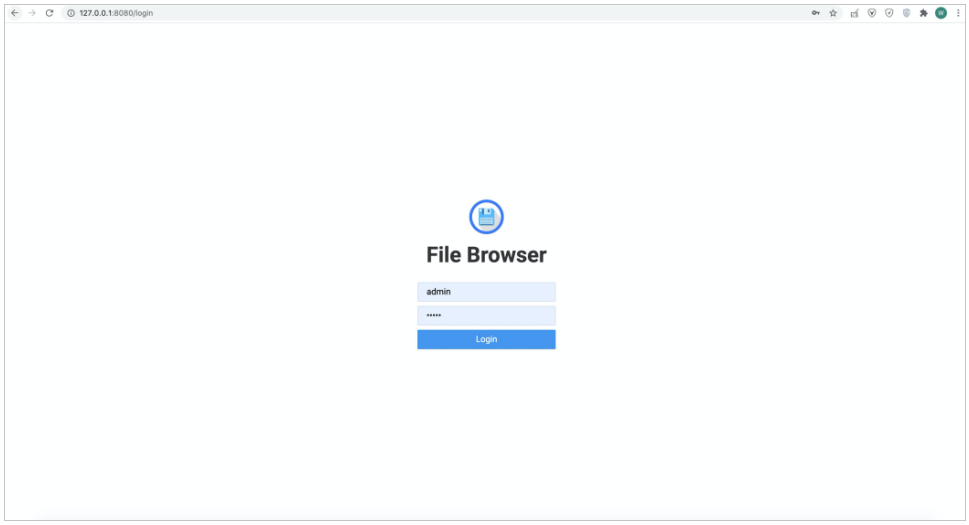
ii. 将File Browser的80端口转发到本地。

```
kubectl port-forward deployment/filebrowser 8080:80
```

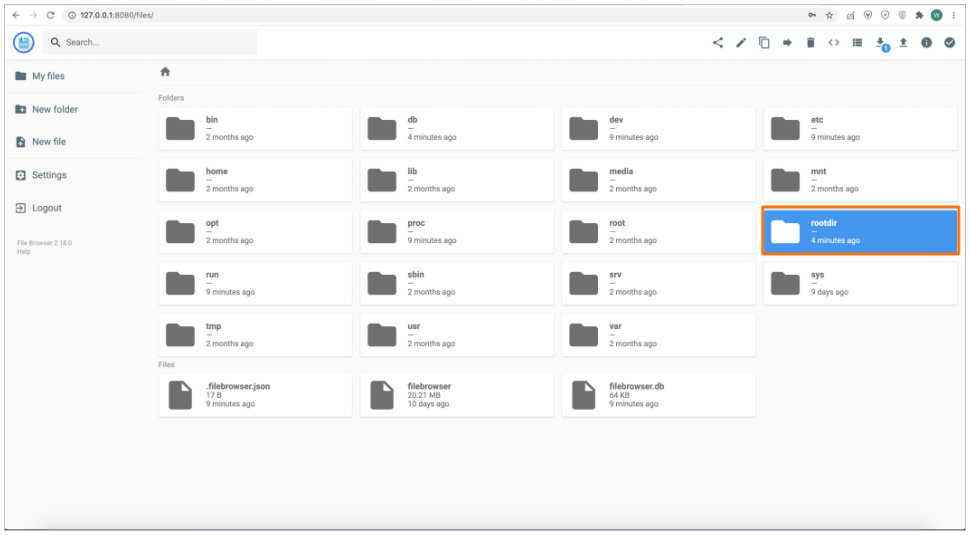
预期输出：

```
Forwarding from 127.0.0.1:8080 -> 80
Forwarding from [::1]:8080 -> 80
```

iii. 打开浏览器，在地址栏输入*127.0.0.1:8080*，可以看到File Browser的登录界面，输入默认账号（admin）和密码（admin），进入到容器内部。

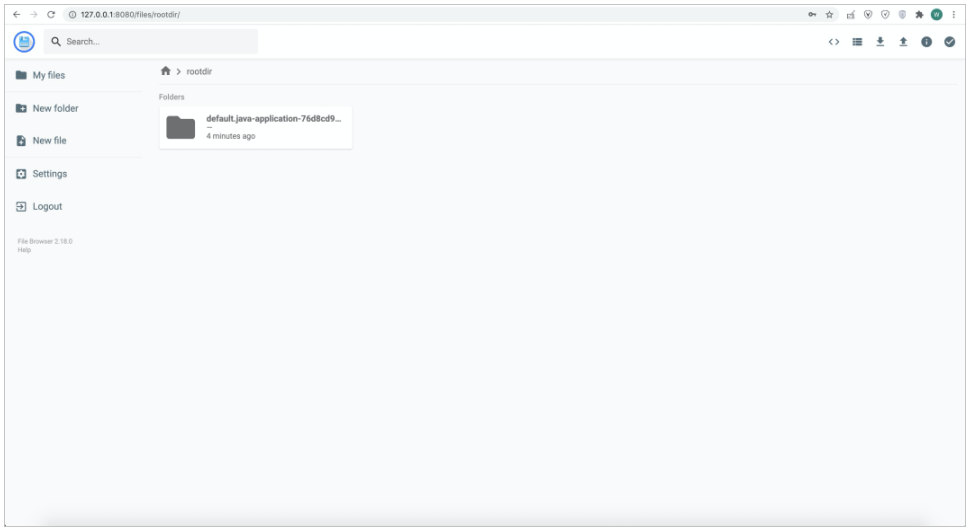


iv. 由于File Browser将名称为*cnfs-nas-pvc*的PVC挂载到*rootDir*下，双击*rootDir*进入到NAS挂载点。

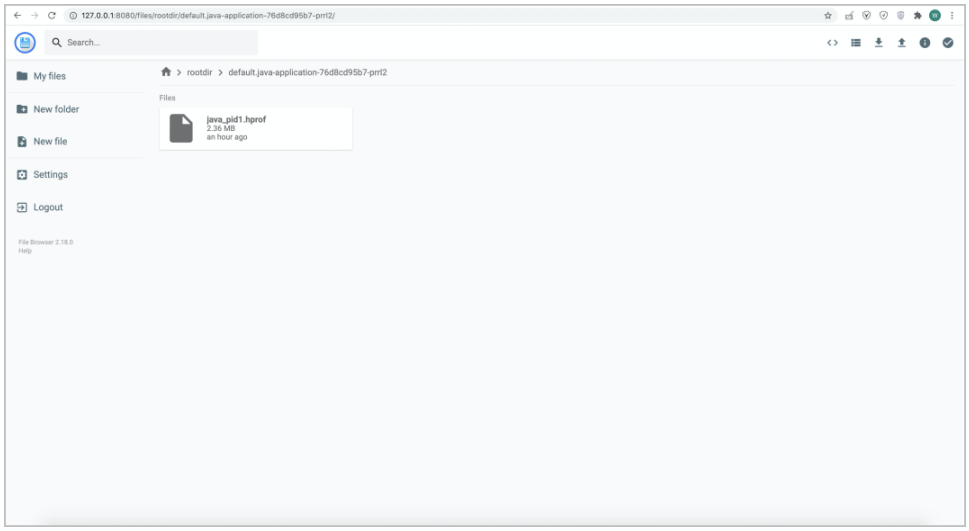


执行结果

在File Browser中可以看到名称为*default-java-application-76d8cd95b7-prrl2*的目录，这个目录是*java-application*的 `subPathExpr: $(POD_NAMESPACE).$(POD_NAME)` 作为规则生成的目录。



然后进入此目录，可以看到目录中的转储文件*java_pid1.hprof*。如果您需要定位到程序发生OOM的代码行数，可以将*java_pid1.hprof*下载到本地，通过MAT（Eclipse Memory Analyzer Tools）进一步分析JVM堆栈信息。



6.运维

6.1. ACK对接外部LDAP验证源

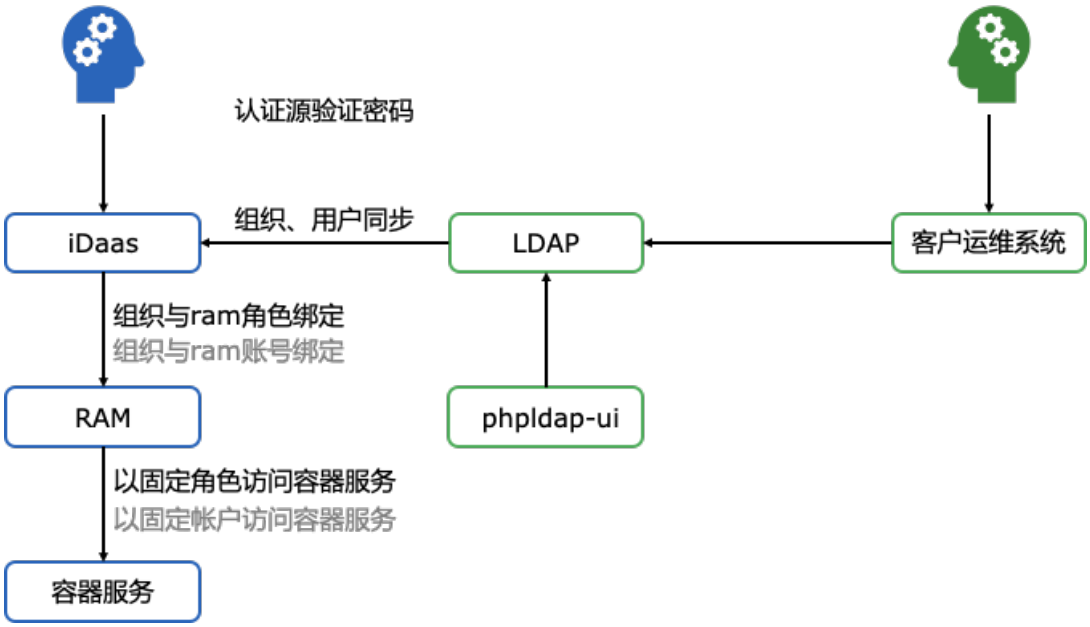
随着企业客户上云进程的推进，有些客户有自己的账户体系，而上云后这些体系如何平滑迁移到云上？如何避免大量账号的注册和多份用户名密码的管理？为解决这些问题，本文介绍如何在阿里云上完成对自建LDAP（轻量目录访问协议）的对接过程。

背景信息

本文涉及到的产品如下：

- 容器服务Kubernetes版（Container Service for Kubernetes）。
- 访问控制RAM（Resource Access Management）。
- 阿里云云盾应用身份服务IDaaS（Alibaba Cloud Identity as a Service）：用于对接您自己的帐户系统和阿里云的帐户系统的产品。
- LDAP轻型目录访问协议：用来保存帐户信息，实现单点登录。OpenLDAP是LDAP的开源实现。

对接原理



阿里云上完成对自建LDAP的对接的原理如下：

1. IDaaS添加LDAP对接配置，同步LDAP账号进入IDaaS，密码不同步。此时使用LDAP用户名登录需要在IDaaS中重新设置密码（如果您不想使用多套密码，可以使用以下SSO方式）。
2. IDaaS中添加应用：
   - 应用需要通过访问密钥（AccessKey）对接RAM中有AliyunRAMFullAccess权限的子账号，用于应用对RAM权限的操控。
   - 应用与RAM中另外的一个Role或RAM账号对接，用于赋权给LDAP中用于登录的账号。
   - 应用需要导入IDaaS中已经导入阿里云系统的LDAP账号，允许这些账号使用本应用，权限为应用中对接的Role或RAM账号权限。
3. 配置IDaaS单点登录对接的LDAP服务。
4. LDAP用户通过SSO登录阿里云平台，获取IDaaS中某应用里对接的Role或RAM账号权限。

5. 主账号在容器平台进行PaaS层赋权。

步骤一：环境准备

模拟LDAP环境操作步骤如下。

1. 执行以下命令搭建OpenLDAP。

   搭建OpenLDAP来模拟LDAP数据源和LDAP的管理工具php-LDAP-admin。

   i. 执行以下命令复制库代码。

   ```
   git clone https://github.com/lilongthinker/demo-base-code.git
   ```

   预期输出：

   ```
   正克隆到 'demo-base-code'...
   remote: Enumerating objects: 12, done.
   remote: Counting objects: 100% (12/12), done.
   remote: Compressing objects: 100% (10/10), done.
   remote: Total 12 (delta 0), reused 9 (delta 0), pack-reused 0
   展开对象中：100% (12/12)，完成.
   ```

   ii. 执行以下命令修改域名。

   ```
   cd demo-base-code/01_ldap
   01_ldap git:(master) tree ./
   ```

   预期输出：

   ```
   ./
   ├── ingress-phpadmin.yaml
   ├── ldap-deploy.yaml
   ├── ldap-secret.yaml
   ├── ldap-service.yaml
   ├── phpldapadmin-deploy.yaml
   └── phpldapadmin-svc.yaml
   0 directories, 6 files
   ################################################
   apiVersion: networking.k8s.io/v1beta1
   kind: Ingress
   metadata:
     name: ldap-ui
     namespace: public-service
   spec:
     rules:
     - host: phpldap.c26a7ab225d1544088735677ed906xxxx.cn-beijing.alicontainer.com #需要按照您的集群信息修改域名。
       http:
         paths:
         - backend:
             serviceName: phpldapadmin
             servicePort: 8080
   ##需要使用vim命令修改上述host的内容为自己集群的域名。
   ################################################
   ```

iii. 执行以下命令创建服务。

```
01_ldap git:(master) kubectl create ns public-service
```

预期输出：

```
namespace/public-service created
```

iv. 执行以下命令部署应用。

```
01_ldap git:(master) kubectl apply -f ./
```

预期输出：

```
ingress.extensions/ldap-ui created
deployment.extensions/ldap created
secret/ldap-secret created
service/ldap-service created
deployment.extensions/phpldapadmin created
service/phpldapadmin created
```

2. 初始化账号。

i. 登录php-LDAP-admin。

a. 执行以下命令获取ingress域名和地址。

```
01_ldap_with_ui git:(master) ✗ kubectl get ing
```

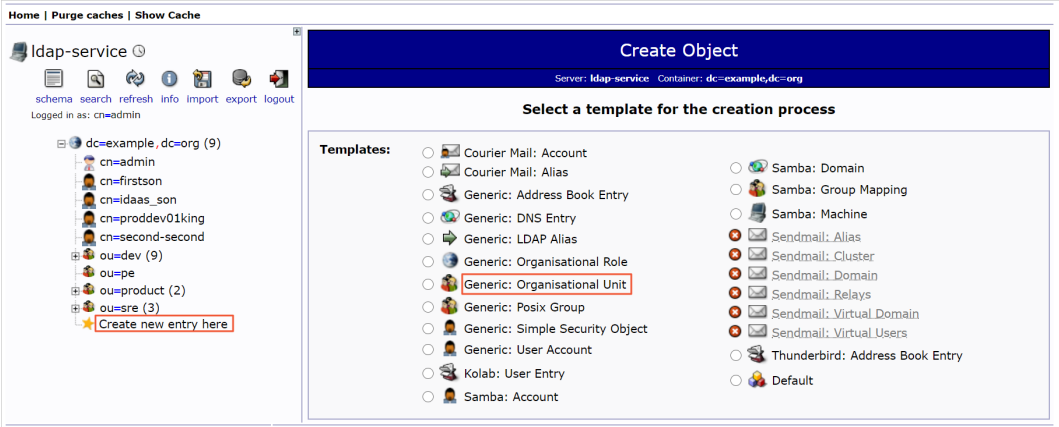
```
NAME      HOSTS
ADDRESS          PORTS   AGE
ldap-ui   phpldap.c26a7ab225d1544088735677ed906xxxx.cn-beijing.alicontainer.com
121.xx.xxx.xxx   80      45s
```

b. 拷贝上述获取的ingress域名至浏览器中，然后用默认的DN和密码登录php-LDAP-admin控制台页面。

> **说明**
>
> - 默认的DN：cn=admin,dc=example,dc=org
> - 默认的密码：admin

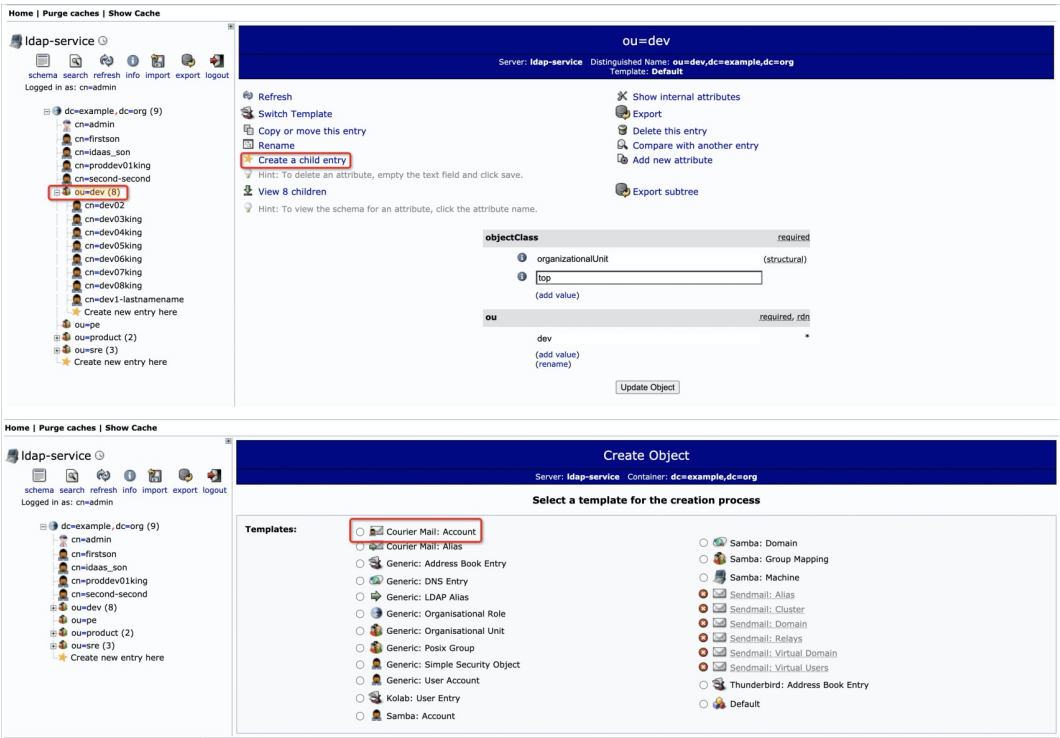
ii. 创建组织和账号。

a. 在php-LDAP-admin控制台左侧导航栏，单击**Create new entry here**，在**Create Object**页面，单击**Generic: Organisational Unit**，然后单击**Create Object**。



b. 输入组织名称，然后单击**Create Oject**。本文组织示例名称为*dev*。

c. 在php-LDAP-admin控制台左侧导航栏，单击上步中创建的ou=dev，单击**Create a child entry**，然后单击**Courier Mail: Account**，配置创建用户信息，然后单击**Create Object**。



> 注意 php-LDAP-admin的账号默认生成的名字中间会有空格，请注意删除空格。

步骤二：配置IDaaS

配置IDaaS，导入组织和账号数据的步骤如下。

1. 登录应用身份管理控制台。
2. 在实例列表中，单击目标实例名称。
3. 在左侧导航栏中，选择**账号 > 机构及组**。
4. 在**查看详情**区域，单击**配置LDAP**。

5. 在**LDAP配置**面板中，单击**新建配置**。
6. 填写LDAP配置的相关信息。有关配置项的详细信息，请参见LDAP账户同步配置。

   LDAP部分配置信息说明如下表。

   - 服务器链接

参数	描述
AD/LDAP名称	自定义名称。
服务器地址	LDAP服务对外的服务地址（非Web页面地址）。
端口号	LDAP服务对外的服务端口号，设置为*389*。
Base DN	设置LDAP的同步目录。
管理员DN	有LDAP管理权限的管理员账号。
密码	LDAP管理员密码。
类型	设置为**Windows AD**或**OpenLDAP**。
LDAP同步至本系统	需设置为**启用**。
本系统同步至LDAP	可设置为**禁用**或**启用**。

   - 字段匹配规则

参数	描述
用户名	LDAP中的用户名定义，一般为CN。
外部ID	■ 如果是Windows AD，设置为*objectGUID*。 ■ 如果是OpenLdap，设置为*uid*。
密码字段	■ 如果是Windows AD，设置为*unicodePwd*。 ■ 如果是OpenLdap，设置为*userPassword*。
用户唯一标识	■ 如果是Windows AD，设置为*DistinguishedName*。 ■ 如果是OpenLdap，设置为*EntryDN*。
邮箱	填写字段名，如*mail*。
手机号	填写字段名，如*telephoneNumber*。
昵称	在IDaaS中账户的显示名称。
默认密码	定时从AD或者LDAP同步账户到IDaaS系统时的默认密码。

> **说明**
> LDAP配置说明如下：
> - LDAP配置中**账号**区域的**管理员DN**和**密码**为：
>   - 管理员DN：cn=admin,dc=example,dc=org
>   - 密码：admin
> - 服务器链接和字段匹配规则配置完后，需分别单击**测试连接**，然后单击**保存**。
> - 如果您只想同步部分的账号到阿里云，可以在**Base DN**处填写需要同步的根目录即可；如果是多个非同一个根目录的部门需要同步，重复进行LDAP配置可以设置多个数据源。

7. 导入组织与账号。具体操作步骤请参见从AD导入账户及组织机构。

步骤三：配置IDaaS单点登录

1. 创建应用。
   i. 在实例详情页左侧导航栏，选择**应用 > 添加应用**。
   ii. 在**添加应用**页面的**应用ID**列下，选择**plugin_aliyun_role** ，然后单击其右侧的**添加应用**。
   iii. 在弹出的面板中，单击**添加SigningKey**，填写相关信息，然后单击**提交**。
   iv. 单击目标应用右侧的**选择**进行LDAP配置。

   > **说明** 此处的AK信息填写为创建有AliyunRAMFullAccess权限的RAM子账户信息。

   v. 单击**提交**。
2. 按照组织授权应用。
   i. 在实例详情页左侧导航栏，选择**授权 > 应用授权**。
   ii. 在**按应用授权组织机构/组**页签，单击目标应用和目标组织机构。
   iii. 在实例详情页左侧导航栏，选择**应用 > 应用列表**，然后单击应用右侧的**详情**，在**应用信息**区域，单击**查看详情**。
   iv. 在**应用详情**面板中，单击**导出IDaaS SAML元配置文件**。
   v. 在RAM控制台左侧导航栏，单击**SSO管理**，然后在**角色SSO**页签，单击**新建身份提供商**。

vi. 在**新建身份提供商**面板，填写提供商名称，单击**上传文件**上传元配置文件，然后单击**确定**。



vii. 创建RAM角色为**身份提供商**并赋权。具体步骤请参见创建可信实体为身份提供商的RAM角色。

viii. 在应用身份管理控制台左侧导航栏，选择**应用 > 应用列表**，在目标应用右侧，单击**详情**，然后在**账户信息 - 子账户**区域，单击**查看应用子账户**。

ix. 在**子账户**页面，单击**添加账号关联**。

x. 在**添加账号关联**面板，填写主账户和子账户信息，然后单击**保存**。

> **说明**
> 主账号为**机构及组**页面的**账户**页签中导入的LDAP用户的名称。
> 子账号登录后的权限，可通过绑定role或者账号获得，可通过文件批量设置子账号对应角色授权ACK的fullAccess权限。

步骤四：为单点登录配置认证源IDaaS

1. 在应用身份管理控制台左侧导航栏，选择**认证 > 认证源**。
2. 在**认证源**页面右上角，单击**添加认证源**。
3. 选择**LDAP**认证源并单击右侧的**添加认证源**。
4. 填写配置信息，然后单击**提交**。

> **注意** 选中是否显示，否则认证源不会在登录框部分显示。有关更多配置信息，请参见LDAP认证登录。

步骤五：查看效果

通过单点登录页面的链接查看效果。

1. 在应用身份管理控制台左侧导航栏，选择目标实例。
2. 将目标实例右侧的用户登录页地址拷贝至浏览器，然后单击LDAP认证源。



有关LDAP认证登录的更多信息，请参见LDAP认证登录。

6.2. 云原生ACK应用管理

本文介绍阿里云容器服务Kubernetes版（ACK）集群应用管理的场景描述、方案优势、解决问题、架构图及操作参考链接。

场景描述

云原生时代的到来，使得在使用Kubernetes进行应用管理的同时也面临着一些痛点。遇到的痛点：

- 应用编排文件如何管理。
- 应用包含哪些K8s资源，能否统一管理。
- 如何管理应用版本，能否快速回滚。

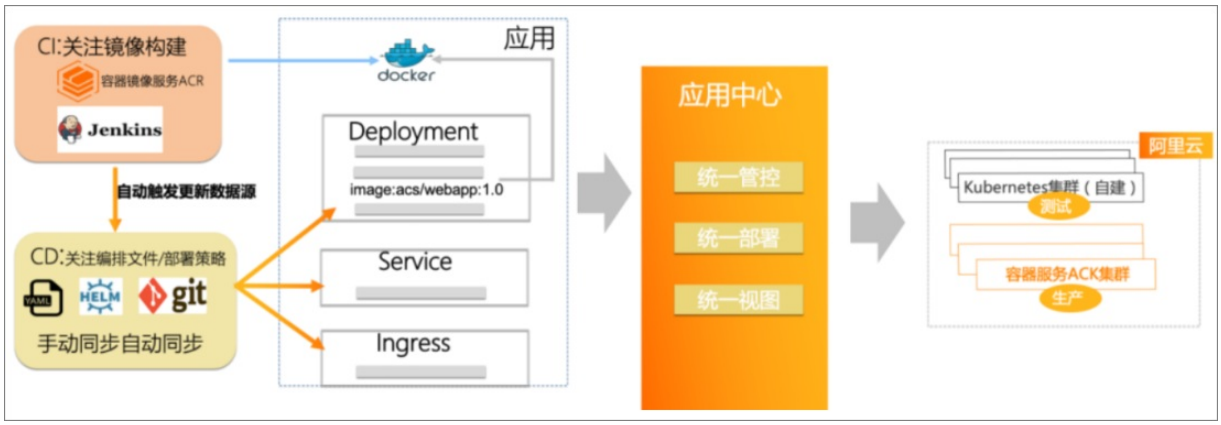
方案优势

- 多集群低成本容灾，日常迭代版本一键同发。
- 国内站、国际站多地域部署业务，版本同发。
- 应用维度查看/管理工作负载（相关资源聚合）。
- 同一套编排，多环境发布，大幅降低出错率。

解决问题

- 应用编排文件如何管理。
- 应用包含哪些K8s资源，能否统一管理问题。
- 如何管理应用版本，能否快速回滚问题。

架构图



参考链接

关于云原生ACK应用管理的详情，请参见云原生ACK应用管理实践。

7.监控

7.1. 使用Prometheus Client监控应用

通过在应用中埋点来暴露应用数据，使用Prometheus Client监控抓取数据，即可实现利用Prometheus监控应用的目的。本文以阿里云容器服务Kubernetes集群和阿里云容器镜像服务为例，介绍如何通过Prometheus Client监控应用。

前提条件

- 已创建Kubernetes集群。具体操作，请参见创建Kubernetes托管版集群。
- 已接入阿里云Prometheus监控。具体操作，请参见开启ARMS Prometheus监控。
- 已创建阿里云容器镜像服务镜像仓库。具体操作，请参见步骤二：创建镜像仓库。

步骤一：对应用埋点

Prometheus Client目前支持大部分编程语言，更多信息，请参见CLIENT LIBRARIES。以下示例通过对应用埋点以暴露Go应用的监控数据：

```
package main
import (
    "flag"
    "fmt"
    "log"
    "math"
    "math/rand"
    "net/http"
    "time"
    "github.com/prometheus/client_golang/prometheus"
    "github.com/prometheus/client_golang/prometheus/promhttp"
)
var (
    addr             = flag.String("listen-address", ":8080", "The address to listen on fo
r HTTP requests.")
    uniformDomain    = flag.Float64("uniform.domain", 0.0002, "The domain for the uniform
distribution.")
    normDomain       = flag.Float64("normal.domain", 0.0002, "The domain for the normal di
stribution.")
    normMean         = flag.Float64("normal.mean", 0.00001, "The mean for the normal distr
ibution.")
    oscillationPeriod = flag.Duration("oscillation-period", 10*time.Minute, "The duration o
f the rate oscillation period.")
)
var (
    // Create a summary to track fictional interservice RPC latencies for three distinct se
rvices with different latency distributions.
    // These services are differentiated via a "service" label.
    rpcDurations = prometheus.NewSummaryVec(
        prometheus.SummaryOpts{
            Name:       "rpc_durations_seconds",
            Help:       "RPC latency distributions.",
            Objectives: map[float64]float64{0.5: 0.05, 0.9: 0.01, 0.99: 0.001},
        },
        []string{"service"},
```

```
        []string{"service"},
    )
    // The same as above, but now as a histogram, and only for the normal
    // distribution. The buckets are targeted to the parameters of the
    // normal distribution, with 20 buckets centered on the mean, each
    // half-sigma wide.
    rpcDurationsHistogram = prometheus.NewHistogram(prometheus.HistogramOpts{
        Name:    "rpc_durations_histogram_seconds",
        Help:    "RPC latency distributions.",
        Buckets: prometheus.LinearBuckets(*normMean-5**normDomain, .5**normDomain, 20),
    })
)
func init() {
    // Register the summary and the histogram with Prometheus's default registry.
    prometheus.MustRegister(rpcDurations)
    prometheus.MustRegister(rpcDurationsHistogram)
    // Add Go module build info.
    prometheus.MustRegister(prometheus.NewBuildInfoCollector())
}
func main() {
    flag.Parse()
    start := time.Now()
    oscillationFactor := func() float64 {
        return 2 + math.Sin(math.Sin(2*math.Pi*float64(time.Since(start))/float64(*oscillat
ionPeriod)))
    }
    // Periodically record some sample latencies for the three services.
    go func() {
        for {
            v := rand.Float64() * *uniformDomain
            rpcDurations.WithLabelValues("uniform").Observe(v)
            time.Sleep(time.Duration(100*oscillationFactor()) * time.Millisecond)
        }
    }()
    go func() {
        for {
            v := (rand.NormFloat64() * *normDomain) + *normMean
            rpcDurations.WithLabelValues("normal").Observe(v)
            // Demonstrate exemplar support with a dummy ID. This
            // would be something like a trace ID in a real
            // application.  Note the necessary type assertion. We
            // already know that rpcDurationsHistogram implements
            // the ExemplarObserver interface and thus don't need to
            // check the outcome of the type assertion.
            rpcDurationsHistogram.(prometheus.ExemplarObserver).ObserveWithExemplar(
                v, prometheus.Labels{"dummyID": fmt.Sprint(rand.Intn(100000))},
            )
            time.Sleep(time.Duration(75*oscillationFactor()) * time.Millisecond)
        }
    }()
    go func() {
        for {
            v := rand.ExpFloat64() / 1e6
            rpcDurations.WithLabelValues("exponential").Observe(v)
            time.Sleep(time.Duration(50*oscillationFactor()) * time.Millisecond)
```

```
        }
    }()
    // Expose the registered metrics via HTTP.
    http.Handle("/metrics", promhttp.HandlerFor(
        prometheus.DefaultGatherer,
        promhttp.HandlerOpts{
            // Opt into OpenMetrics to support exemplars.
            EnableOpenMetrics: true,
        },
    ))
    log.Fatal(http.ListenAndServe(*addr, nil))
}
```

在本示例中，相关参数说明如下：

- 在注册rpc_durations_seconds指标前需要注册一个监控指标prometheus.MustRegister。本示例中rpc_durations_seconds为prometheus.NewSummaryVec类型，更多其他类型，请参见Prometheus。
- rpcDurations是一个全局的单例，在更新监控数据时通过调用rpcDurations.WithLabelValues("uniform").Observe(v)增加监控数据。

关于代码模板，请参见prometheus / client_golang。

步骤二：将应用制作为镜像并上传到镜像仓库

将完成埋点的应用制作成镜像并上传至阿里云容器镜像服务的镜像仓库。

1. 执行以下命令构建镜像。

   ```
   docker build -t <本地临时Docker镜像名称>:<本地临时Docker镜像版本号> . --no-cache
   ```

   示例命令：

   ```
   docker build -t prometheus-demo:v1 . --no-cache
   ```

2. 执行以下命令为镜像打标。

   ```
   sudo docker tag <本地临时Docker镜像名称>:<本地临时Docker镜像版本号> <Registry域名>/<命名空间>/<镜像名称>:<镜像版本号>
   ```

   示例命令：

   ```
   sudo docker tag prometheus-demo:v1 registry.cn-hangzhou.aliyuncs.com/ringtail/prometheus-demo:v1
   ```

3. 执行以下命令将镜像推送至镜像仓库。

   ```
   sudo docker push <Registry域名>/<命名空间>/<镜像名称>:<镜像版本号>
   ```

   示例命令：

   ```
   sudo docker push registry.cn-hangzhou.aliyuncs.com/ringtail/prometheus-demo:v1
   ```

4. 查看已推送的镜像。
   i. 登录容器镜像服务控制台。
   ii. 在顶部菜单栏，选择所需地域。
   iii. 在左侧导航栏，选择**实例列表**。

iv. 在**实例列表**页面单击个人版实例。

v. 在个人版实例管理页面选择**仓库管理 > 镜像仓库**。

vi. 在**镜像仓库**页面单击目标仓库**操作**列的**管理**。

vii. 在左侧导航栏，选择**镜像版本**。

在镜像版本列表中可查看已推送的镜像。

步骤三：将应用部署至容器服务Kubernetes集群

1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，单击**集群**。
3. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
4. 创建容器组。

i. 在集群管理页左侧导航栏中，选择**工作负载 > 容器组**。

ii. 在**容器组**页面，单击**使用YAML创建资源**。

iii. 在**创建**页面的**模板**代码框输入以下内容，然后单击**创建**。

```
apiVersion: apps/v1 # for versions before 1.8.0 use apps/v1beta1
kind: Deployment
metadata:
  name: demo-app
  labels:
    app: demo-app
spec:
  replicas: 2
  selector:
    matchLabels:
      app: demo-app
  template:
    metadata:
      labels:
        app: demo-app
    spec:
      containers:
      - name: demo-app
        image: <Registry域名>/<命名空间>/<镜像名称>:<镜像版本号>
        command:
        - /random
        ports:
        - containerPort: 8080
```

示例代码：

```
apiVersion: apps/v1 # for versions before 1.8.0 use apps/v1beta1
kind: Deployment
metadata:
  name: demo-app
  labels:
    app: demo-app
spec:
  replicas: 2
  selector:
    matchLabels:
      app: demo-app
  template:
    metadata:
      labels:
        app: demo-app
    spec:
      containers:
      - name: demo-app
        image: registry.cn-hangzhou.aliyuncs.com/ringtail/prometheus-demo:v1
        command:
        - /random
        ports:
        - containerPort: 8080
```

在**容器组**页面可查看已创建的容器组。

5. 创建服务。
   i. 在集群管理页左侧导航栏中，选择**网络 > 服务**。
   ii. 在**服务**页面，单击**使用YAML创建资源**。
   iii. 在**创建**页面的**模板**代码框输入以下内容，然后单击**创建**。

```
apiVersion: v1
kind: Service
metadata:
  labels:
    app: demo-app
  name: demo-app
  namespace: default
spec:
  ports:
  - name: http-metrics
    port: 8080
    protocol: TCP
    targetPort: 8080
  selector:
    app: demo-app
  type: ClusterIP
```

在**服务**页面可查看已创建的服务。

步骤四：配置服务发现

配置阿里云Prometheus监控的服务发现以抓取Go应用数据。本示例以ARMS控制台操作为例：

1. 登录Prometheus控制台。
2. 在顶部菜单栏，选择集群所在的地域。
3. 在左侧导航栏单击**监控列表**。
4. 在**Prometheus监控**页面，单击目标集群右侧**操作**列下的**设置**。
5. 在目标集群的设置页面，单击**服务发现**页签，再单击ServiceMonitor页签。
6. 在ServiceMonitor页签下，单击**添加ServiceMonitor**。
7. 在**添加ServiceMonitor**对话框中输入以下内容，然后单击**确定**。

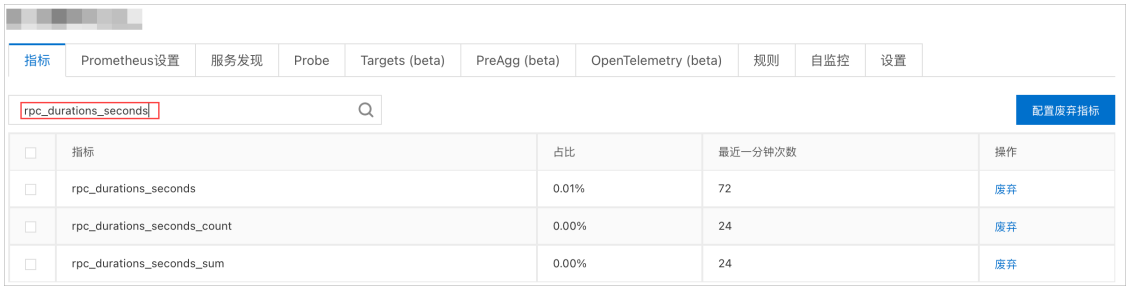
```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  labels:
    app: demo-app
  name: demo-app
  namespace: default
spec:
  endpoints:
  - interval: 30s
    port: http-metrics
  jobLabel: app
  namespaceSelector:
    matchNames:
    - default
  selector:
    matchLabels:
      app: demo-app
```

在ServiceMonitor页签下可查看已配置的服务发现。

步骤五：验证Prometheus Client是否支持通过指标监控应用数据

1. 在设置页面，单击**指标**页签。
2. 在文本框输入步骤一：对应用埋点注册的指标rpc_durations_seconds，单击 。

   若查询到指标rpc_durations_seconds，表示Prometheus Client已通过该指标监控应用数据。



8.弹性伸缩

8.1. 使用ack-autoscaling-placeholder实现容器秒级伸缩

ack-autoscaling-placeholder为集群的自动扩展提供了缓冲区，它适用于工作负载需要快速启动而无需考虑节点资源不足的使用场景。本文介绍如何使用ack-autoscaling-placeholder实现容器秒级伸缩。

前提条件

您已为ACK集群开通自动伸缩。关于开通自动伸缩的操作步骤，请参见节点自动伸缩。

操作步骤

1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，选择**市场 > 应用市场**。
3. 在**应用目录**页签，搜索ack-autoscaling-placeholder，然后单击**ack-autoscaling-placeholder**。
4. 在**ack-autoscaling-placeholder**页面，单击**一键部署**。
5. 在**创建**面板，选择**集群**和**命名空间**，然后单击**下一步**。选择**Chart版本**，编辑**参数**，然后单击**确定**。创建成功后，在**应用 > Helm**页面，可查看到该应用状态为**已部署**。
6. 在集群管理页左侧导航栏中，选择**应用 > Helm**。
7. 在**Helm**页面，单击ack-autoscaling-placeholder**操作**列的**更新**。然后在**更新发布**面板中，更新YAML，然后单击**确定**。

```
nameOverride: ""
fullnameOverride: ""
##
priorityClassDefault:
  enabled: true
  name: default-priority-class
  value: -1
##
deployments:
   - name: ack-place-holder
     replicaCount: 1
     containers:
       - name: placeholder
         image: registry-vpc.cn-shenzhen.aliyuncs.com/acs/pause:3.1
         pullPolicy: IfNotPresent
         resources:
           requests:
             cpu: 4                  #资源占位4C8G
             memory: 8
     imagePullSecrets: {}
     annotations: {}
     nodeSelector:                   #节点选择
       demo: "yes"
     tolerations: []
     affinity: {}
     labels: {}
```

8. 部署工作负载的PriorityClass。

   本文示例定义一个优先级较高的PriorityClass。

```
kubectl apply -f priorityClass.yaml
```

```
apiVersion: scheduling.k8s.io/v1
kind: PriorityClass
metadata:
  name: high-priority
value: 1000000              #配置优先级。
globalDefault: false
description: "This priority class should be used for XYZ service pods only."
```

9. 部署实际的工作负载。

```
kubectl apply -f workload.yaml
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: placeholder-test
  labels:
    app: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      nodeSelector:                          #节点选择。
        demo: "yes"
      priorityClassName: high-priority       #这里写入第8步配置的PriorityClass名称。
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
        resources:
          requests:
            cpu: 3                           #实际负载的资源需求。
            memory: 5
```

从下图可以看到，实际工作负载由于Pod配置了高优先级的PriorityClass。当节点资源不足时，会将占位容器placeHolder进行驱逐，此时占位容器placeHolder处于Pending状态。由于集群开通了自动伸缩，此状态会触发ACK集群进行扩容。实现了工作负载的**秒级弹出**。





实现原理

使用优先级非常低（负数）的占位容器来超额配置，以保留其他Pod可以使用的资源。如果集群没有可用的资源，真正的工作负载也会将占位容器所占用的资源抢占，实现快速启动，然后结合同时使用Cluster-Autoscaler，迫使集群进行节点维度的扩展。

8.2. 弹性优化之自定义镜像

Alicloud Image Builder是阿里云推出的一款镜像构建工具，旨在通过简易的方式自动化构建镜像。利用Alicloud Image Builder构建出的操作系统镜像，再结合ACK集群节点池的自定义镜像功能，可以快速地扩容节点。本文介绍如何在ACK集群中将Alicloud Image Builder通过Job的方式构建自定义操作系统镜像。

前提条件

- 您已创建一个ACK集群。具体操作，请参见创建Kubernetes托管版集群。
- 使用kubectl工具连接集群。具体操作，请参见通过kubectl工具连接集群。

背景信息

容器服务ACK节点池支持ACK集群节点的弹性伸缩，默认创建节点池时，提供的操作系统镜像包括CentOS、Alibaba Cloud Linux 2等操作系统镜像，已经能够满足绝大多数场景的使用。但是针对一些预安装或者高性能场景下，可能原有的基础镜像并不能满足我们的需求。阿里云提供的Alicloud Image Builder，可以帮助您构建属于自己的自定义操作系统镜像，可以提高复杂场景下弹性伸缩的便捷性。

使用Alicloud Image Builder创建自定义镜像时，您可以通过Job或CronJob将镜像构建任务下发到集群完成构建。

使用ACK Job快速构建自定义操作系统镜像

本文通过创建名为build-config的配置项和名为build的Job工作负载为例，说明如何使用Alicloud Image Builder快速构建自定义操作系统镜像。

1. 创建名为build-config的配置项用以配置构建操作系统镜像的参数。
    i. 使用以下YAML内容创建名为*build-config.yaml*的文件。

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: build-config
data:
  ack.json: |-
    {
      "variables": {
        "image_name": "ack-optimized_image-1.20-{{timestamp}}",
        "source_image": "aliyun_2_1903_x64_20G_alibase_20210120.vhd",
        "instance_type": "ecs.gn5i-c4g1.xlarge",
        "region": "{{env `ALICLOUD_REGION`}}",
        "access_key": "{{env `ALICLOUD_ACCESS_KEY`}}",
        "secret_key": "{{env `ALICLOUD_SECRET_KEY`}}",
        "runtime": "{{env `RUNTIME`}}",
        "skip_secrutiy_fix": "{{env `SKIP_SECURITY_FIX`}}"
      },
      "builders": [
        {
          "type": "alicloud-ecs",
          "access_key": "{{user `access_key`}}",
          "secret_key": "{{user `secret_key`}}",
          "region": "{{user `region`}}",
          "image_name": "{{user `image_name`}}",
          "source_image": "{{user `source_image`}}",
          "ssh_username": "root",
          "instance_type": "{{user `instance_type`}}",
          "skip_image_validation": "true",
          "io_optimized": "true"
        }
      ],
      "provisioners": [{
       "type": "file",
       "source": "scripts/ack-optimized-os-1.20.sh",
       "destination": "/root/"
        },
        {
       "type": "shell",
       "inline": [
         "export RUNTIME=docker",
         "export SKIP_SECURITY_FIX=true",
         "bash /root/ack-optimized-os-1.20.sh",
      ]
      }]
    }
```

上述YAML中涉及参数解释如下。

Alicloud Image Builder配置文件的参数解释

参数	示例值	描述
`variables{"<variable1>":"<value>"}`	variables{"access_key":"{{env ALICLOUD_ACCESS_KEY}}"}	定义了Alicloud Image Builder中会用到的变量（`variables`）。 **说明** 如果将重要信息，例如AccessKey（包括`access_key`和`secret_key`）写入配置文件的话，存在信息泄露的风险，但是将其设置成变量后可防止意外，变量的值来源于运行时的输入值。
`builders{"type":"<value>"}`	builders{"type":"alicloud-ecs"}	镜像生成器（`builders`）。当设置type为*aliyun-ecs*时，表示构建镜像时，会临时创建一个ECS实例来完成镜像构建。构建完成后，ECS实例会自动销毁。
`provisioners{"type":"<value>"}`	provisioners{"type":"shell"}	镜像配置器（`provisioners`），用以定义需要在临时实例内执行的操作。当设置type为*Shell*时，说明使用的是Shell Provisioner，表示在连接Linux实例后自动执行一段Shell命令。例如，执行Shell命令`yum install redis.x86_64 -y`安装Redis。 关于Provisioner配置的更多信息，请参见下文的Provisioner配置介绍。

镜像构建涉及的参数解释

参数	示例值	描述	重要度
access_key	LTAInPyXXXXQ****	您的AccessKey ID。更多详情，请参见获取AccessKey。	高
secret_key	CM1ycKrrCekQ0dhXXXXXXXXXl7y****	您的AccessKey Secret。	高
region	cn-beijing	目标自定义镜像的所属地域。	高
image_name	ack-custom_image	目标自定义镜像的名称。不允许与已有镜像重名。	低
source_image	aliyun_2_1903_x64_20G_alibase_20200904.vhd	具有相同操作系统的阿里云公共镜像ID。	高
PRESET_GPU	true	预置安装GPU，加速启动。	高

ii. 执行以下命令部署Alicloud Image Builder到集群。

```
kubectl apply -f build-config.yaml
```

2. 创建Job工作负载以完成自定义操作系统镜像的构建。

i. 使用以下YAML内容创建名为*build.yaml*的文件。

```
apiVersion: batch/v1
kind: Job
metadata:
  name: image-builder
  namespace: default
spec:
  template:
    metadata:
      name: image-builder
    spec:
      containers:
        - name: image-builder
          image: registry.cn-hangzhou.aliyuncs.com/build-test/image-builder:v2
          env:
            - name: ALICLOUD_ACCESS_KEY
              value: xxxxxxx
            - name: ALICLOUD_SECRET_KEY
              value: xxxxxxx
            - name: REGION
              value: cn-hangzhou
          command: ["packer"]
          args:  ["build","/config/ack.json"]
          volumeMounts:
            - name: config
              mountPath: /config
      volumes:
        - name: config
          configMap:
            name: build-config
            items:
              - key: ack.json
                path: ack.json
      restartPolicy: Never
```

ii. 执行以下命令部署Job到集群开始构建操作系统镜像。

```
kubectl apply -f build.yaml
```

3. （可选）登录ACK控制台查看自定义镜像构建日志。

构建镜像时会产生操作日志。日志给出了构建过程中执行的每一个步骤，包括校验参数、创建临时资源、预安装软件、创建目标资源和释放临时资源等。您可以执行以下步骤查看镜像构建日志。

i. 登录容器服务管理控制台。

ii. 在控制台左侧导航栏中，单击**集群**。

iii. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。

iv. 在集群管理页左侧导航栏中，选择**工作负载 > 任务**。

v. 在任务列表中，找到上步创建的任务（Job），并单击其右侧**操作**列下的**详情**。

vi. 在目标任务详情页，单击**日志**页签，然后查看镜像构建日志。

Provisioner配置介绍

Provisioner是在转换为静态操作系统镜像之前，在正在运行的机器中用于安装和配置软件的组件。常用来被执行安装软件到镜像中的主要工作场景包括：

- 安装软件包。
- 修补内核。
- 创建用户。
- 下载应用程序代码。

Provisioner常见操作：

- 执行Shell脚本。

```
"provisioners": [{
    "type": "shell",
    "script": "script.sh"
}]
```

- 使用Ansible执行编排脚本。

```
"provisioners": [
  {
  "type": "ansible",
  "playbook_file": "./playbook.yml"
  }
]
```

- 安装CPFS客户端。
  由于CPFS需要安装的包较多，且一部分安装包涉及现场编译流程，安装过程比较费时。在客户端节点数量较大时，使用自定义镜像可以极大减少批量安装CPFS客户端节点的成本。示例配置如下。

```
    {
      "variables": {
        "region": "{{env `REGION`}}",
        "image_name": "ack-custom_image",
        "source_image": "centos_7_04_64_20G_alibase_201701015.vhd",
        "access_key": "{{env `ALICLOUD_ACCESS_KEY`}}",
        "secret_key": "{{env `ALICLOUD_SECRET_KEY`}}"
      },
    "builders": [
        {
          "type": "alicloud-ecs",
          "access_key": "{{user `access_key`}}",
          "secret_key": "{{user `secret_key`}}",
          "region": "{{user `region`}}",
          "image_name": "{{user `image_name`}}",
          "source_image": "{{user `source_image`}}",
          "ssh_username": "root",
          "instance_type": "ecs.g6.large",
          "skip_image_validation": "true",
          "io_optimized": "true"
        }
      ],
   "provisioners": [{
        "type": "shell",
        "inline": [
            "cd $HOME",
            "wget https://cpfs-client.oss-cn-beijing.aliyuncs.com/kernel/kernel-devel-`un
ame -r`.rpm",
            "rpm -ivh --replacefiles kernel-devel-`uname -r`.rpm"
        ]
      }]
```

- 定制GPU节点系统镜像，加速启动。预置GPU的镜像同样可用于CPU机型。

```
{
      "variables": {
        "region": "{{env `REGION`}}",
        "image_name": "ack-custom_image",
        "source_image": "aliyun_2_1903_x64_20G_qboot_alibase_20210325.vhd",
        "access_key": "{{env `ALICLOUD_ACCESS_KEY`}}",
        "secret_key": "{{env `ALICLOUD_SECRET_KEY`}}"
        "runtime": "{{env `RUNTIME`}}",
        "skip_secrutiy_fix": "{{env `SKIP_SECURITY_FIX`}}"
      },
    "builders": [
        {
          "type": "alicloud-ecs",
          "access_key": "{{user `access_key`}}",
          "secret_key": "{{user `secret_key`}}",
          "region": "{{user `region`}}",
          "image_name": "{{user `image_name`}}",
          "source_image": "{{user `source_image`}}",
          "ssh_username": "root",
          "instance_type": "ecs.gn6i-c4g1.xlarge",    #预置GPU安装需要设置GPU规格的类型。
          "skip_image_validation": "true",
          "io_optimized": "true"
        }
      ],
   "provisioners": [
        {
            "type": "file",
            "source": "scripts/ack-optimized-os-1.20.sh",
            "destination": "/root/"
        },
        {
            "type": "shell",
            "inline": [
                "export RUNTIME=containerd",
                "export PRESET_GPU=true",          #预置GPU安装需要设置PRESET_GPU为true，不需要预置GPU时保持留空或设置为false。
                "export NVIDIA_DRIVER_VERSION=510.47.03",          #设置GPU版本，留空则默认安装460.91.03版本。
                "export SKIP_SECURITY_FIX=true",
                "bash /root/ack-optimized-os-1.20.sh"
            ]
        }
      ]
    }
```

- 将业务镜像缓存到系统镜像中。

```
{
      "variables": {
        "region": "{{env `REGION`}}",
        "image_name": "ack-custom_image",
        "source_image": "aliyun_2_1903_x64_20G_qboot_alibase_20210325.vhd",
        "access_key": "{{env `ALICLOUD_ACCESS_KEY`}}",
        "secret_key": "{{env `ALICLOUD_SECRET_KEY`}}"
        "runtime": "{{env `RUNTIME`}}",
        "skip_secrutiy_fix": "{{env `SKIP_SECURITY_FIX`}}"
      },
    "builders": [
        {
          "type": "alicloud-ecs",
          "access_key": "{{user `access_key`}}",
          "secret_key": "{{user `secret_key`}}",
          "region": "{{user `region`}}",
          "image_name": "{{user `image_name`}}",
          "source_image": "{{user `source_image`}}",
          "ssh_username": "root",
          "instance_type": "ecs.c6.xlarge",
          "skip_image_validation": "true",
          "io_optimized": "true"
        }
      ],
   "provisioners": [
        {
            "type": "file",
            "source": "scripts/ack-optimized-os-1.20.sh",
            "destination": "/root/"
        },
        {
            "type": "shell",
            "inline": [
                "export RUNTIME=docker",
                "export SKIP_SECURITY_FIX=true",
                "bash /root/ack-optimized-os-1.20.sh",
                "docker pull nginx"         #将Nginx镜像固化到系统镜像中。
            ]
        }
      ]
    }
```

- 设置镜像上传、下载并发数。
  i. 登录容器服务管理控制台。
  ii. 在控制台左侧导航栏中，单击**集群**。
  iii. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
  iv. 在集群管理页左侧导航栏中，选择**节点管理 > 节点池**。单击目标节点池名称。
  v. 单击**基本信息**页签，在**节点池信息**区域，单击**伸缩组**后面的链接。
  vi. 单击**实例配置来源**页签。单击目标伸缩配置右侧**操作**列下的**修改**。
  vii. 在弹出的**容器伸缩组提醒**框中单击**确定**。

viii. 在**弹性伸缩Auto Scaling**页面，完成相关配置。然后单击**下一步系统配置**。

ix. 在弹出框中单击**继续**。

x. 在**系统配置**页面，使用Base64编码方式解密**高级选项**区域**实例自定义数据**框中的数据。

xi. 解密完成后，将以下代码加入解密后代码的后面。

```
yum install -y jq
echo "$jq '. += {"max-concurrent-downloads": 20,"max-concurrent-uploads": 20}' /etc/d
ocker/daemon.json" > /etc/docker/daemon.json
service docker restart
```



xii. 使用Base64编码方式加密合成的代码，用加密后的代码替换原来**实例自定义数据**框中的代码。

xiii. 单击**下一步确认配置**，单击**确认修改**。

相关操作

使用Alicloud Image Builder创建好自定义镜像后，您就可以使用自定义的镜像创建弹性伸缩节点池以实现快速扩容节点。有关如何创建弹性节点池，请参见节点自动伸缩。

相关文档

- 使用自定义镜像创建Kubernetes集群

8.3. 多可用区实现同时快速弹性扩容

多可用区均衡是数据类型业务高可用场景下常用的部署方式。当业务压力增大时，有多可用区均衡调度策略的应用希望可以自动扩容出多个可用区的实例来满足集群的调度水位。本文介绍如何在多可用区实现快速弹性扩容。

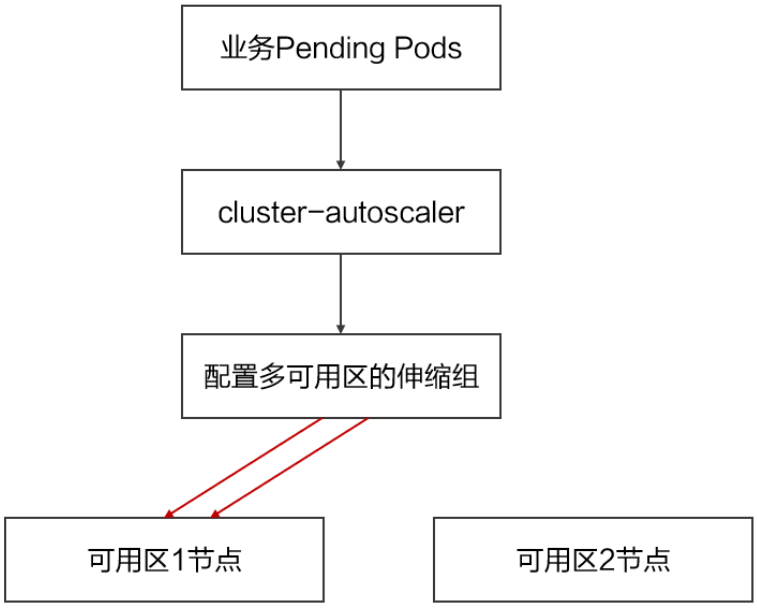
前提条件

已选择多个可用区，并在每个可用区创建好vSwtich，需要在哪些可用区弹出就需要这些可用区至少有一个vSwtich。关于创建可用区和vSwtich的具体操作，请参见创建Kubernetes托管版集群。

背景信息

容器服务节点自动伸缩组件可通过预调度判断服务能否部署在某个伸缩组上，然后将扩容实例个数的请求发给指定伸缩组，最终由ESS伸缩组生成实例。但这种在一个伸缩组上配置多个可用区vSwtich的模型特点，会导致以下问题：

当多可用区业务Pod出现由于集群资源不足无法调度时，节点自动伸缩服务会触发该伸缩组扩容，但是无法将需要扩容的可用区与实例的关系传递到ESS弹性伸缩组，因此可能会连续弹出某一个地域的多个实例，而非在多个vSwtich同时弹出，这样无法满足在多可用区同时扩容的需求。



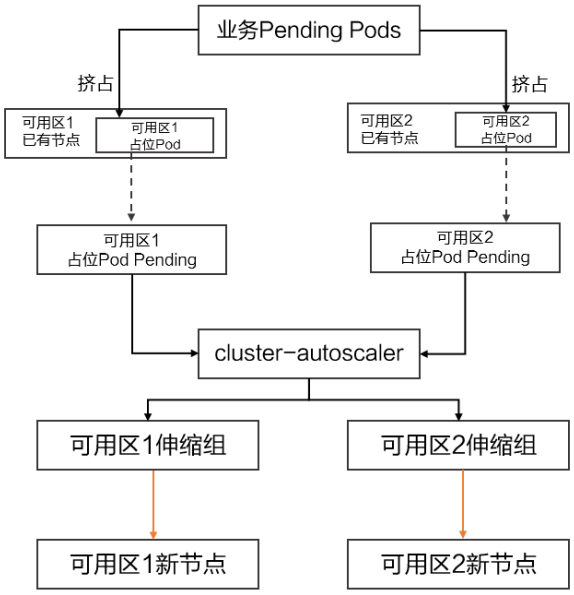
解决方案

为了解决同时扩容多可用区节点的问题，容器服务ACK引入了ack-autoscaling-placeholder组件，通过少量的资源冗余方式，将多可用区的弹性伸缩问题转变为并发节点池的定向伸缩问题。具体操作，请参见使用ack-autoscaling-placeholder实现容器秒级伸缩。

具体原理如下：

1. 首先为每个可用区创建一个节点池，并分别在各个节点池打上可用区的标签。
2. 通过配置可用区标签nodeSelector的方式，使用ack-autoscaling-placehodler为每个可用区创建占位Pod，默认的占位Pod具有比较低权重的PriorityClass，应用Pod的优先级高于占位Pod。
3. 这样业务应用Pod Pending后，会抢占各个可用区占位Pod，带有可用区nodeSelector的多可用区占位Pod处于Pending后，节点自动伸缩组件感知到的调度策略就从多个可用区的anti-affinity变成了可用区的nodeSelector，从而可以轻松处理发出扩容区域的节点的请求。

以下图两个可用区为例，介绍利用现有架构上能满足多可用区同时扩容的方法。



1. 利用ack-autoscaling-placeholder作为业务应用和节点自动伸缩组件之间的桥梁，为每个可用区创建占位Pod，占位Pod的调度优先级需要低于实际业务应用的调度优先级。
2. 应用Pod Pengding后会迅速抢占占位Pod，并部署在各个可用区的已有节点上，同时被抢占的占位Pod会处于Pending状态。
3. 由于占位Pod是带有可用区nodeSelector调度策略的，节点自动伸缩组件就可以并发扩容到对应的可用区。

步骤一：每个可用区创建一个开启自动伸缩的节点池，并配置自定义节点标签

1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，单击**集群**。
3. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
4. 在集群管理页左侧导航栏中，选择**节点管理 > 节点池**。
5. 在**节点池**页面右上角，单击**创建节点池**。

   您还可以在**节点池**页面右上角，单击**创建托管节点池**。
6. 在**创建节点池**页面，设置创建节点池的配置项。

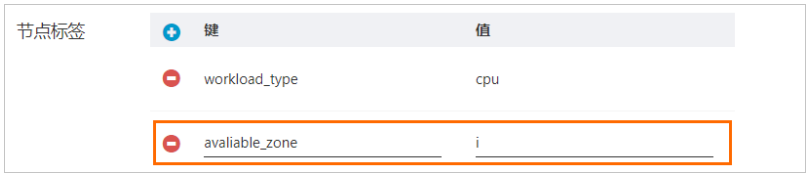
   本文以在可用区I创建开启自动伸缩的节点池auto-zone-I为例。有关配置项的详细说明，请参见创建节点池。

i. 选择可用区I的vSwtich。



ii. 选中**开启自动弹性伸缩**。

iii. 单击**显示高级选项**，设置**节点标签**的**键**为*avaliable_zone*，**值**为*i*。



iv. 配置节点池其他参数，单击**确认配置**，然后单击**确认**。

7. 在**节点池**页面，查看节点池列表。当auto-zone-I节点池状态为**已激活**，表示该节点池创建成功。
8. 重复步骤6，在每个需要扩容的可用区创建开启自动伸缩的节点池。



步骤二：部署Placeholder及占位Deployment

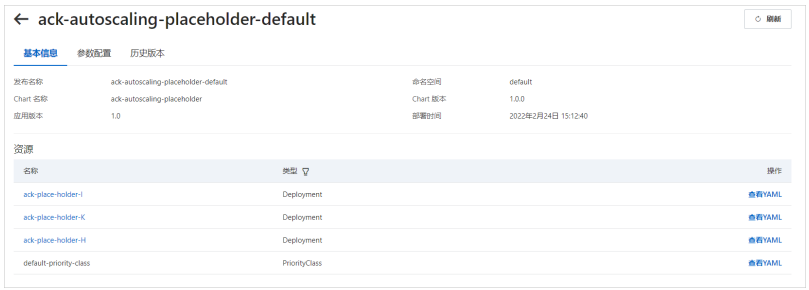
1. 在控制台左侧导航栏中，选择**市场 > 应用市场**。
2. 在**应用目录**页签，搜索ack-autoscaling-placeholder，然后单击**ack-autoscaling-placeholder**。
3. 在**ack-autoscaling-placeholder**页面，单击**一键部署**。
4. 在**创建**面板，选择**集群**和**命名空间**，然后单击**下一步**。选择**Chart版本**，编辑**参数**，然后单击**确定**。创建成功后，在**应用 > Helm**页面，可查看到该应用状态为**已部署**。
5. 在集群管理页左侧导航栏中，选择**应用 > Helm**。
6. 在**Helm**页面，单击ack-autoscaling-placeholder-defalut**操作**列的**更新**。
7. 在**更新发布**面板中，更新YAML，然后单击**确定**。将每个可用区都设置Placehodler，每个区域需要定义一个占位Deployment。

   本文以在可用区I、K、H创建占位Deployment为例，YAML示例如下：

```
deployments:
- affinity: {}
  annotations: {}
  containers:
  - image: registry-vpc.cn-beijing.aliyuncs.com/acs/pause:3.1
    imagePullPolicy: IfNotPresent
    name: placeholder
    resources:
      requests:
        cpu: 3500m      #调度单元CPU。
        memory: 6       #调度单元内存。
  imagePullSecrets: {}
  labels: {}
  name: ack-place-holder-I              #占位Deployment名称。
  nodeSelector: {"avaliable_zone":i}    #可用区标签（需要与步骤一创建节点池中配置的标签一致）。
  replicaCount: 10                      #每次扩容可快速弹出Pod数量。
  tolerations: []
- affinity: {}
  annotations: {}
  containers:
  - image: registry-vpc.cn-beijing.aliyuncs.com/acs/pause:3.1
    imagePullPolicy: IfNotPresent
    name: placeholder
    resources:
      requests:
        cpu: 3500m     #调度单元CPU。
        memory: 6      #调度单元内存。
  imagePullSecrets: {}
  labels: {}
  name: ack-place-holder-K             #占位Deployment名称。
  nodeSelector: {"avaliable_zone":k}   #可用区标签（需要与步骤一创建节点池中配置的标签一致）。
  replicaCount: 10                     #每次扩容可快速弹出Pod数量。
  tolerations: []
- affinity: {}
  annotations: {}
  containers:
  - image: registry-vpc.cn-beijing.aliyuncs.com/acs/pause:3.1
    imagePullPolicy: IfNotPresent
    name: placeholder
    resources:
      requests:
        cpu: 3500m    #调度单元CPU。
        memory: 6     #调度单元内存。
  imagePullSecrets: {}
  labels: {}
  name: ack-place-holder-H           #占位Deployment名称。
  nodeSelector: {"avaliable_zone":h} #可用区标签（需要与步骤一创建节点池中配置的标签一致）。
  replicaCount: 10                   #每次扩容可快速弹出Pod数量。
  tolerations: []
fullnameOverride: ""
nameOverride: ""
podSecurityContext: {}
priorityClassDefault:
  enabled: true
```

```
  name: default-priority-class
  value: -1
```

更新成功后，在各个可用区可以查看到占位Deployment已创建。



步骤三：创建实际负载的PriorityClass

1. 使用以下YAML示例，创建名为priorityClass.yaml的文件。

```
apiVersion: scheduling.k8s.io/v1
kind: PriorityClass
metadata:
  name: high-priority
value: 1000000                #配置优先级，比步骤二的工作负载默认优先级高。
globalDefault: false
description: "This priority class should be used for XYZ service pods only."
```

如果不希望在负载上配置PriorityClass，可以通过设置全局PriorityClass的方式进行全局默认设置，这样只需要下发默认配置后，抢占能力便自动生效。

```
apiVersion: scheduling.k8s.io/v1
kind: PriorityClass
metadata:
  name: global-high-priority
value: 1                                #配置优先级，比步骤二的工作负载默认优先级高。
globalDefault: true
description: "This priority class should be used for XYZ service pods only."
```

2. 执行以下命令，部署工作负载的PriorityClass。

```
kubectl apply -f priorityClass.yaml
```

预期输出：

```
priorityclass.scheduling.k8s.io/high-priority created
```

步骤四：创建实际工作负载

以可用区I为例。

1. 使用以下YAML示例，创建名为*workload.yaml*的文件。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: placeholder-test
  labels:
    app: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      nodeSelector:                          #节点选择。
        avaliable_zone: "i"
      priorityClassName: high-priority       #这里写入步骤三配置的PriorityClass名称，如果全局配置开启，则为非必填。
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
        resources:
          requests:
            cpu: 3                           #实际负载的资源需求。
            memory: 5
```

2. 执行以下命令，部署实际的工作负载。

```
kubectl apply -f workload.yaml
```

预期输出：

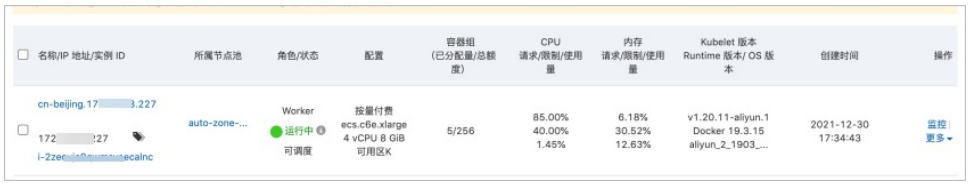
```
deployment.apps/placeholder-test created
```

验证结果

部署后在**工作负载 > 容器组**页面可以发现，由于实际负载的PriorityClass比占位Pod的要高，在弹出节点上，抢占的占位Pod会运行起来。而被抢占的占位Pod会触发节点自动伸缩组件的并发扩容，为下次实际负载扩容做准备。



在**节点管理 > 节点**页面，可以看到负载Pod运行在之前占位Pod的节点上。



相关文档

- 使用ack-autoscaling-placeholder实现容器秒级伸缩

8.4. 基于阿里云Prometheus指标的容器水平伸缩

默认HPA只支持基于CPU和内存的自动伸缩，并不能满足日常的运维需求。阿里云Prometheus监控全面对接开源Prometheus生态，支持类型丰富的组件监控，提供多种开箱即用的预置监控大盘，且提供全面托管的Prometheus服务。本文介绍如何将阿里云Prometheus指标转换成HPA可用的指标，从而为应用提供更加便捷的扩缩机制。

前提条件

在将阿里云Prometheus指标转换成HPA可用的指标前，您需要部署相关组件。

1. 部署阿里云Prometheus监控组件。具体操作，请参见开启ARMS Prometheus监控。
2. 部署alibaba-cloud-metrics-adapter组件。具体操作，请参见部署alibaba-cloud-metrics-adapter组件。

示例

本文举例如何配置alibaba-cloud-metrics-adapter，实现将阿里云Prometheus指标转换为HPA可用指标，并实现该指标自动伸缩。

1. 部署工作负载。
   i. 登录容器服务管理控制台。
   ii. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
   iii. 在集群管理页左侧导航栏中，选择**工作负载 > 无状态**。
   iv. 在**无状态**页面，单击**使用YAML创建资源**。
   v. 在**创建**页面，部署以下YAML文件创建一个名为sample-app的应用及对应的Service，然后单击**创建**。

   > **说明** 此容器暴露出http_requests_total的指标用来标识访问次数。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: sample-app
  labels:
    app: sample-app
spec:
  replicas: 1
  selector:
    matchLabels:
      app: sample-app
  template:
    metadata:
      labels:
        app: sample-app
    spec:
      containers:
      - image: luxas/autoscale-demo:v0.1.2
        name: metrics-provider
        ports:
        - name: http
          containerPort: 8080
---
apiVersion: v1
kind: Service
metadata:
  name: sample-app
  namespace: default
  labels:
    app: sample-app
spec:
  ports:
    - port: 8080
      name: http
      protocol: TCP
      targetPort: 8080
  selector:
    app: sample-app
  type: ClusterIP
```

2. 添加ServiceMonitor。
   i. 登录ARMS控制台。
   ii. 在左侧导航栏选择**Prometheus监控 > Prometheus实例列表**。
   iii. 在**Prometheus监控**页面左上角选择容器服务K8s集群所在的地域，然后单击目标实例名称进入对应实例页面。
   iv. 在左侧导航栏单击**服务发现**，然后单击**配置**页签。
   v. 在**配置**页签下单击ServiceMonitor。

vi. 在**ServiceMonitor**页签下单击**添加ServiceMonitor**创建ServiceMonitor。

模板文件如下所示。

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: sample-app
  namespace: default
spec:
  endpoints:
  - interval: 30s
    port: http
    path: /metrics
  namespaceSelector:
    any: true
  selector:
    matchLabels:
      app: sample-app
```

3. 确认监控状态。
在**服务发现**页面，单击**Targets**，如果看到**default/sample-app/0(1/1 up)**，则说明您成功在阿里云Prometheus监控到了部署的应用。

4. 修改alibaba-cloud-metrics-adapter配置。

i. 登录容器服务管理控制台。

ii. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。

iii. 在集群管理页左侧导航栏中，选择**应用 > Helm**。

iv. 在**Helm**页面的**发布名称**列，找到alibaba-cloud-metrics-adapter，并单击右侧的**更新**。

v. 将alibaba-cloud-metrics-adapter中完整的YAML文件内容复制粘贴至模板中，然后单击**更新**。

以下为部分YAML文件内容。

```
AlibabaCloudMetricsAdapter:
  affinity: {}
  commonLabels: ""
  env:
  - AccessKeyId: ""
  - AccessKeySecret: ""
  - Region: ""
  fullnameOverride: ""
  image:
    pullPolicy: IfNotPresent
    # 此处为公网镜像地址，内网地址请把前缀域名换为registry-vpc.{{RegionId}}.aliyuncs.com
。
    repository: registry.aliyuncs.com/acs/alibaba-cloud-metrics-adapter-amd64
    tag: v0.2.0-alpha-2f697ee
  listenPort: 443
  nameOverride: ""
  nodeSelector: {}
  podAnnotations: {}
  prometheus:
    adapter:
      rules:
```

```
        custom:
        # 添加新的转换规则，请确保阿里云Prometheus中指标标签和此处一致，如果不一致，请参见阿里云Prometheus中指标标签修改。
        - seriesQuery: http_requests_total{namespace!="",pod!=""}
          resources:
            overrides:
              # 此处resource为Kubernetes的API Resource，可通过kubectl api-resources -o wide查看。
              # 此处key对应Prometheus数据中的LabelName，请确认Prometheus指标数据中有此LabelName。
              namespace: {resource: "namespace"}
              pod: {resource: "pod"}
          name:
            matches: ^(.*)_total
            as: ${1}_per_second
          metricsQuery: sum(rate(<<.Series>>{<<.LabelMatchers>>}[2m])) by (<<.GroupBy>>)
        default: false
    #1.这里设置为true，打开Prometheus adapter。
    enabled: true
    logLevel: 2
    metricsRelistInterval: 1m
    tls:
      ca: ""
      certificate: ""
      enable: false
      key: ""
    #2.这里填入阿里云Prometheus监控的地址。 如何获取Prometheus数据请求地址，请参见本文最后如何获取Prometheus数据请求URL。
    url: http://arms-prometheus-proxy.aliyun.com:9090/api/v1/prometheus/8cba801fff65546a3012e9a6843afd/1240538168824185/ce63742a509f948dda8ef18e75e703356/cn-shenzhen
  ramRoleType: restricted
  replicas: 1
  service:
    type: ClusterIP
  serviceAccountName: admin
  tolerations: []
ConfigReloader:
  image:
    # 此处为公网镜像地址，内网地址请把前缀域名换为registry-vpc.{{RegionId}}.aliyuncs.com。
    repository: registry.aliyuncs.com/acs/configmap-reload
    tag: v0.0.1
```

> **说明**
>
> - `url`：填写阿里云Prometheus监控的地址。如何获取Prometheus数据请求URL，请参见如何获取Prometheus数据请求URL。
> - 有关ack-alibaba-cloud-adapter配置文件详细说明，请参见ack-alibaba-cloud-adapter配置文件详解。

5. 部署HPA。

当前版本已支持Prometheus Metrics同时透出Custom Metrics与External Metrics，您可以根据需求选择两种不同方式通过HPA进行容器伸缩。

- **方式一：通过Custom Metrics进行容器伸缩**

  a. 执行以下命令，通过Custom Metrics指标查询方式，查看HPA可用指标详情。

  ```
  kubectl get --raw "/apis/custom.metrics.k8s.io/v1beta1/namespaces/default/pods/*/
  http_requests_per_second" | jq .
  ```

  查询指标结果如下：

  ```
  {
    "kind": "MetricValueList",
    "apiVersion": "custom.metrics.k8s.io/v1beta1",
    "metadata": {
      "selfLink": "/apis/custom.metrics.k8s.io/v1beta1/namespaces/default/pods/%2A/
  http_requests_per_second"
    },
    "items": [
      {
        "describedObject": {
          "kind": "Pod",
          "namespace": "default",
          "name": "sample-app-579bc6774c-rmjfd",
          "apiVersion": "/v1"
        },
        "metricName": "http_requests_per_second",
        "timestamp": "2022-01-28T08:42:58Z",
        "value": "33m",
        "selector": null
      }
    ]
  }
  ```

b. 使用以下YAML文件部署HPA，然后执行 `kubectl apply -f hpa.yaml` 创建HPA应用。

```
kind: HorizontalPodAutoscaler
apiVersion: autoscaling/v2beta1
metadata:
  name: sample-app
spec:
#HPA的伸缩对象描述，HPA会动态修改该对象的Pod数量。
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: sample-app
#HPA的最小Pod数量和最大Pod数量。
  minReplicas: 1
  maxReplicas: 10
#监控的指标数组，支持多种类型的指标共存。
  metrics:
  - type: Pods
    pods:
      #使用指标:pods/http_requests。
      metricName: http_requests_per_second
 # AverageValue类型的目标值，Pods指标类型下只支持AverageValue类型的目标值。
      targetAverageValue: 500m       #当出现了小数点，K8s又需要高精度时，会使用单位m或k
。例如1001m=1.001，1k=1000。
```

c. 在Service中开启负载均衡后，执行压测验证。

a. 执行以下命令进行压测实验。

```
ab -c 50 -n 2000 LoadBalancer(sample-app):8080/
```

b. 执行以下命令查看HPA详情。

```
kubectl get hpa sample-app
```

预期输出：

```
NAME         REFERENCE                TARGETS    MINPODS   MAXPODS   REPLICAS
AGE
sample-app   Deployment/sample-app   33m/500m   1         10        1
7m
```

- **方式二：通过External Metrics进行容器伸缩**

a. 执行以下命令，通过External Metrics指标查询方式，查看HPA可用指标详情。

```
kubectl get --raw "/apis/external.metrics.k8s.io/v1beta1/namespaces/default/http_requests_per_second" | jq .
```

查询指标结果如下：

```
{
  "kind": "ExternalMetricValueList",
  "apiVersion": "external.metrics.k8s.io/v1beta1",
  "metadata": {},
  "items": [
    {
      "metricName": "http_requests_per_second",
      "metricLabels": {},
      "timestamp": "2022-01-28T08:40:20Z",
      "value": "33m"
    }
  ]
}
```

b. 使用以下YAML文件部署HPA，然后执行 `kubectl apply -f hpa.yaml` 创建HPA应用。

```
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: sample-app
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: sample-app
  minReplicas: 1
  maxReplicas: 10
  metrics:
    - type: External
      external:
        metric:
          name: http_requests_per_second
          selector:
            matchLabels:
              job: "sample-app"
#External指标类型下只支持Value和AverageValue类型的目标值。
        target:
          type: AverageValue
          averageValue: 500m
```

c. 在Service中开启负载均衡后，执行压测验证。

a. 执行以下命令进行压测实验。

```
ab -c 50 -n 2000 LoadBalancer(sample-app):8080/
```

b. 执行以下命令查看HPA详情。

```
kubectl get hpa sample-app
```

预期输出：

```
NAME         REFERENCE                TARGETS    MINPODS   MAXPODS   REPLICAS
AGE
sample-app   Deployment/sample-app    33m/500m   1         10        1
7m
```

ack-alibaba-cloud-adapter配置文件详解

ack-alibaba-cloud-adapter通过以下步骤将Prometheus中的指标转换成HPA可用的指标：

1. Discovery：ack-alibaba-cloud-adapter会从Prometheus发现可用的指标。
2. Association：将指标与Kubernetes资源（Pod、Node、Namespace）相关联。
3. Naming：定义转换后的HPA可用指标名称。
4. Querying：定义查询Prometheus语句。

以上文中sample-app容器中暴露出来的 `http_requests_total` 指标转换成HPA中的 `http_requests_per_second` 为例，完整的ack-alibaba-cloud-adapter配置文件如下。

```
- seriesQuery: http_requests_total{namespace!="",pod!=""}
  resources:
    overrides:
      namespace: {resource: "namespace"}
      pod: {resource: "pod"}
  name:
    matches: ^(.*)_total
    as: ${1}_per_second
  metricsQuery: sum(rate(<<.Series>>{<<.LabelMatchers>>}[2m])) by (<<.GroupBy>>)
```

说明

- `seriesQuery`：即PromQL请求数据。
- `metricsQuery`：对 `seriesQuery` 中PromQL请求的数据做聚合操作。
- `resources`：是PromQL里的数据Label，与resource进行匹配，这里的resouce是指集群内的api-resource，比如Pod、Namespace和Node。可通过 `kubectl api-resources -o wide` 查看。此处 `Key` 对应Prometheus数据中的 `LabelName`，请确认Prometheus指标数据中有此LabelName。
- `name`：是指根据正则匹配把Prometheus指标名转为比较可读的指标名，这里是把 `http_request_total` 转为 `http_request_per_second`。

- Discovery

指定待转换的Prometheus指标，您可以通过seriesFilters精确过滤指标。seriesQuery可以根据标签进行查找（示例代码如下）。

```
seriesQuery: http_requests_total{namespace!="",pod!=""}
seriesFilters:
    - isNot: "^container_.*_seconds_total"
```

> **说明** seriesFilters为非必填项，用来过滤指标：
> - is：<regex>，匹配包含该正则表达式的指标。
> - isNot：<regex>，匹配不包含该正则表达式的指标。

- Association
  设置Prometheus指标标签与Kubernetes中的资源映射关系。`http_requests_total`指标的标签包括`kubernetes_namespace!=""`和`kubernetes_pod_name!=""`。

```
- seriesQuery: http_requests_total{namespace!="",pod!=""}
  resources:
    overrides:
      namespace: {resource: "namespace"}
      pod: {resource: "pod"}
```

- Naming
  用于将Prometheus指标名称转换成HPA的指标名称，但不会改变Prometheus本身的指标名称。如果使用Prometheus原来的指标，可以不设置。

> **说明** 您可以通过执行命令`kubectl get --raw "/apis/custom.metrics.k8s.io/v1beta1"`查看HPA可用的所有指标。

```
- seriesQuery: http_requests_total{namespace!="",pod!=""}
  resources:
    overrides:
      namespace: {resource: "namespace"}
      pod: {resource: "pod"}
  name:
    matches: "^(.*)_total"
    as: "${1}_per_second"
```

- Querying
  查询Prometheus API的模板。ack-alibaba-cloud-adapter会根据HPA中的参数，填充参数到此模板中，然后发送给Prometheus API请求，并将获得的值最终提供给HPA进行弹性扩缩。

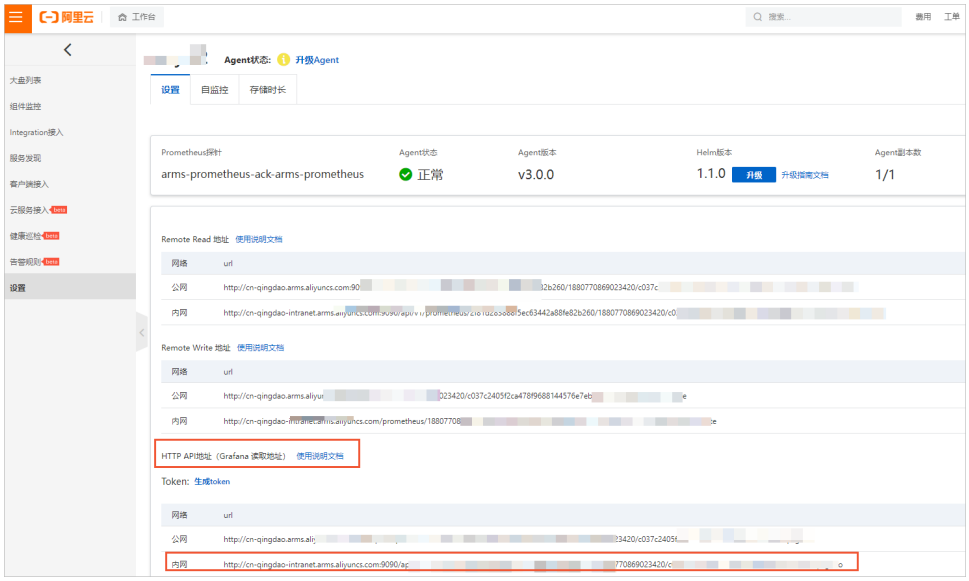
```
- seriesQuery: http_requests_total{namespace!="",pod!=""}
  resources:
    overrides:
      namespace: {resource: "namespace"}
      pod: {resource: "pod"}
  name:
    matches: ^(.*)_total
    as: ${1}_per_second
  metricsQuery: sum(rate(<<.Series>>{<<.LabelMatchers>>}[2m])) by (<<.GroupBy>>)
```

如何获取Prometheus数据请求URL

阿里云Prometheus监控场景

1. 登录ARMS控制台。
2. 在左侧导航栏选择**Prometheus监控 > Prometheus实例列表**。
3. 在**Prometheus监控**页面左上角选择容器服务K8s集群所在的地域，然后单击目标实例名称进入对应实例页面。
4. 在左侧导航栏单击**设置**，然后单击**设置**页签。
5. 在**设置**页签下获取HTTP API地址（Grafana读取地址）。

   推荐使用内网，如无法使用内网时，可使用公网。



开源Prometheus监控场景

1. 部署Prometheus监控方案。
   i. 登录容器服务管理控制台。
   ii. 在控制台左侧导航栏中，选择**市场 > 应用市场**。
   iii. 在**应用市场**页面单击**应用目录**页签，搜索并单击ack-prometheus-operator。
   iv. 在ack-prometheus-operator页面，单击**一键部署**。
   v. 在**创建**面板中，选择集群和命名空间，然后单击**下一步**。

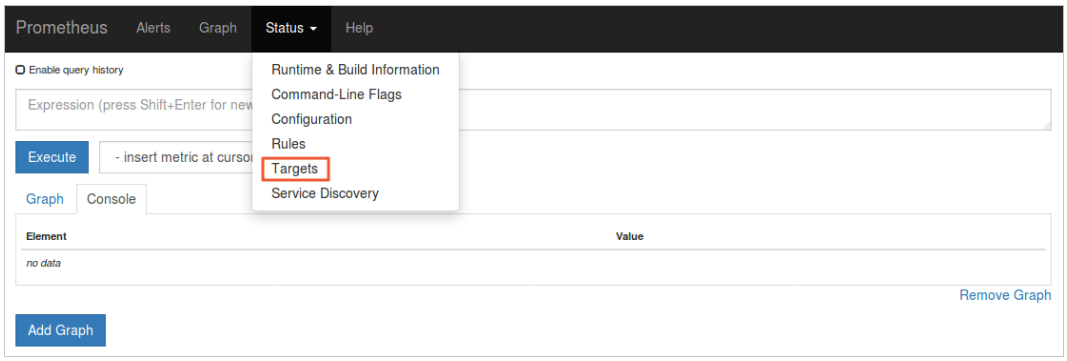
vi. 在**参数配置**页面，设置相应参数，然后单击**确定**。
查看部署结果：

a. 执行以下命令，将集群中的Prometheus映射到本地9090端口。

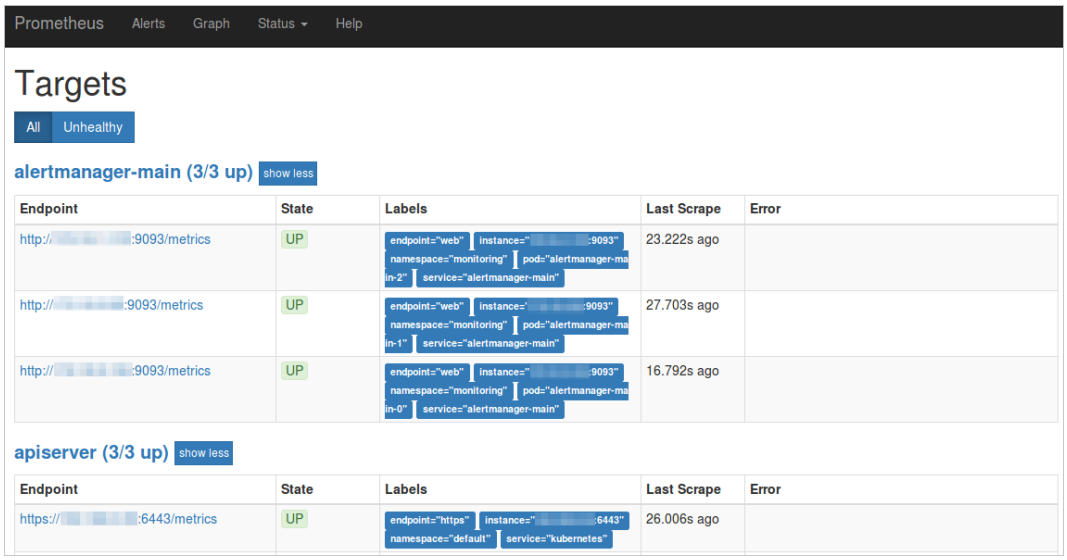
```
kubectl port-forward svc/ack-prometheus-operator-prometheus 9090:9090 -n monito
ring
```

b. 在浏览器中访问*localhost:9090*，即可查看Prometheus。

c. 选择菜单栏**Status > Targets**，查看所有采集任务。



如果所有任务的状态为**UP**，表示所有采集任务均已正常运行。



2. 查看**Labels**中对应的**service**和**namespce**。

以ServiceName是ack-prometheus-operator-prometheus，ServiceNamespace是monitoring为例说明该开源Prometheus数据请求的URL：

```
http://ack-prometheus-operator-prometheus.monitoring.svc.cluster.local:9090
```

相关文档

- ARMS Prometheus监控
- 容器水平伸缩（HPA）

8.5. 通过Nginx Ingress对多个应用进行HPA

多实例部署可以最大程度的保证应用的稳定性，但同时也会造成闲时资源的浪费和高额成本。手动调节方式工作量大还存在一定程度的滞后性。通过Nginx Ingress对多个应用进行HPA，可以保障部署在ACK集群上的应用的稳定性以及达到更好的成本控制。本文介绍通过Nginx Ingress对多个应用进行HPA的方法。

前提条件

通过Nginx Ingress对多个应用进行HPA前，需要将阿里云Prometheus指标转换成HPA可用的指标，您需要部署相关组件。

- 部署阿里云Prometheus监控组件。具体操作，请参见开启ARMS Prometheus监控。
- 部署alibaba-cloud-metrics-adapter组件。具体操作，请参见部署alibaba-cloud-metrics-adapter组件。
- 已安装压力测试工具Apache Benchmark。更多信息，请参见Apache Benchmark。

背景信息

在实际的生产中，您使用请求量来自动扩缩容时，可以通过注册http_requests_total来透出请求量指标，同时推荐使用Ingress组件自带的指标来进行HPA。

Ingress是Kubernetes API中的标准资源类型之一。Ingress实现的功能是将客户端请求的Host名称或请求的URL路径，转发到指定的Service资源中，即将Kubernetes集群外部的请求转发至集群内部的Service中，再被Service转发至Pod，处理客户端的请求。

Nginx Ingress Controller是部署于集群内部的Ingress控制器，可以为您提供性能更好，且定制性更高的使用方式。ACK集群提供的Nginx Ingress Controller在社区版本的基础上，整合了阿里云产品的一系列功能，提供了更加便捷的使用体验。

操作步骤

本教程包含两个ClusterIP型服务，通过Nginx Ingress实现对外的流量路由。基于`nginx_ingress_controller_requests`指标，为应用配置HPA，以实现随着流量的变化为Pod扩缩容的功能。

1. 使用以下YAML文件创建业务Deployment和对应的Service。
   i. 创建*nginx1.yaml*文件。然后执行命令`kubectl apply -f nginx1.yaml`，创建应用test-app和对应的Service。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: test-app
  labels:
    app: test-app
spec:
  replicas: 1
  selector:
    matchLabels:
      app: test-app
  template:
    metadata:
      labels:
        app: test-app
    spec:
      containers:
      - image: skto/sample-app:v2
        name: metrics-provider
        ports:
        - name: http
          containerPort: 8080
---
apiVersion: v1
kind: Service
metadata:
  name: test-app
  namespace: default
  labels:
    app: test-app
spec:
  ports:
    - port: 8080
      name: http
      protocol: TCP
      targetPort: 8080
  selector:
    app: test-app
  type: ClusterIP
```

ii. 创建*nginx2.yaml*文件。然后执行命令 `kubectl apply -f nginx2.yaml` ，创建应用sample-app和对应的Service。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: sample-app
  labels:
    app: sample-app
spec:
  replicas: 1
  selector:
    matchLabels:
      app: sample-app
  template:
    metadata:
      labels:
        app: sample-app
    spec:
      containers:
      - image: skto/sample-app:v2
        name: metrics-provider
        ports:
        - name: http
          containerPort: 8080
---
apiVersion: v1
kind: Service
metadata:
  name: sample-app
  namespace: default
  labels:
    app: sample-app
spec:
  ports:
    - port: 80
      name: http
      protocol: TCP
      targetPort: 8080
  selector:
    app: sample-app
  type: ClusterIP
```

2. 创建*ingress.yaml*文件。然后执行 `kubectl apply -f ingress.yaml` 命令，部署Ingress资源。

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: test-ingress
  namespace: default
spec:
  ingressClassName: nginx
  rules:
    - host: test.cf6828835dbba4a368e67f0b9925d****.cn-qingdao.alicontainer.com
      http:
        paths:
          - backend:
              service:
                name: sample-app
                port:
                  number: 80
            path: /
            pathType: ImplementationSpecific
          - backend:
              service:
                name: test-app
                port:
                  number: 8080
            path: /home
            pathType: ImplementationSpecific
```

- host：指定服务访问域名，本示例使用集群的默认域名。集群默认域名格式为：`*.[cluster-id].[region-id].alicontainer.com`，`cluster-id`、`region-id`请替换为实际值。
- path：指定访问的URL路径。请求到来之后会根据路由规则匹配相应的Service，然后通过Service访问相应的Pod。
- backend：由Service名称和Service端口组成，指定当前`path`转发的Service。

3. 执行以下命令，获取Ingress资源。

```
kubectl get ingress -o wide
```

预期输出：

```
NAME           CLASS   HOSTS
ADDRESS        PORTS   AGE
test-ingress   nginx   test.cf6828835dbba4a368e67f0b9925d****.cn-qingdao.alicontainer.c
om   10.10.10.10   80      55s
```

4. 部署成功后，可以通过`/`和`/home`两个路径分别访问Host地址。Nginx Ingress Controller会根据上面的配置分别访问test-app和sample-app。通过阿里云Prometheus查询指标`nginx_ingress_controller_requests`来获取各应用的请求情况，详情如下。

```
nginx_ingress_controller_requests{canary="",controller_class="k8s.io/ingress-nginx",controller_namespace="kube-system",controller_pod="nginx-ingress-
controller-58ffffbccd-hnvg6",host="test.cf6828835dbba4a368e6[redacted].cn-qingdao.alicontainer.com",ingress="test-
ingress",method="GET",namespace="default",path="/",service="sample-app",status="200"} 3
nginx_ingress_controller_requests{canary="",controller_class="k8s.io/ingress-nginx",controller_namespace="kube-system",controller_pod="nginx-ingress-
controller-58ffffbccd-hnvg6",host="test.cf6828835dbba4a368e67f0b9[redacted].cn-qingdao.alicontainer.com",ingress="test-
ingress",method="GET",namespace="default",path="/home",service="test-app",status="200"} 1
```

5. 修改组件alibaba-cloud-metrics-adapter中的*adapter.config*文件，将Prometheus中的指标转换成HPA可用的指标。

> **说明** 在进行本步骤前，请确保已完成了组件alibaba-cloud-metrics-adapter的部署。

i. 登录容器服务管理控制台。
ii. 在控制台左侧导航栏中，单击**集群**。
iii. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
iv. 在左侧导航栏中，单击**应用 > Helm**。
v. 单击**ack-alibaba-cloud-metrics-adapter**。
vi. 在**资源**区域，单击**adapter-config**。
vii. 在**adapter-config**页面，单击页面右上角的**编辑**。
viii. 用以下代码替换**值**中的代码，单击**确定**。

关于以下配置项详解，请参考ack-alibaba-cloud-adapter配置文件详解。

```
rules:
- metricsQuery: sum(rate(<<.Series>>{<<.LabelMatchers>>}[2m]))
  name:
    as: ${1}_per_second
    matches: ^(.*)_requests
  resources:
    namespaced: false
    overrides:
      controller_namespace:
        resource: namespace
  seriesQuery: nginx_ingress_controller_requests
```

6. 执行以下命令，查看指标输出。

```
kubectl get --raw "/apis/external.metrics.k8s.io/v1beta1/namespaces/*/nginx_ingress_controller_per_second" | jq .
```

查询指标结果如下：

```
{
  "kind": "ExternalMetricValueList",
  "apiVersion": "external.metrics.k8s.io/v1beta1",
  "metadata": {},
  "items": [
    {
      "metricName": "nginx_ingress_controller_per_second",
      "metricLabels": {},
      "timestamp": "2022-03-31T10:11:37Z",
      "value": "0"
    }
  ]
}
```

7. 创建*hpa.yaml*文件。然后执行 `kubectl apply -f hpa.yaml` 命令，对业务应用sample-app和test-app分别部署HPA。

```
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: sample-hpa
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: sample-app
  minReplicas: 1
  maxReplicas: 10
  metrics:
    - type: External
      external:
        metric:
          name: nginx_ingress_controller_per_second
          selector:
            matchLabels:
#可以通过这个字段对指标进行过滤，这里设置的字段会传入adapter.config中的<<.LabelMatchers>>标签。
              service: sample-app
#External指标类型下只支持Value和AverageValue类型的目标值。
        target:
          type: AverageValue
          averageValue: 30
----------
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: test-hpa
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: test-app
  minReplicas: 1
  maxReplicas: 10
  metrics:
    - type: External
      external:
        metric:
          name: nginx_ingress_controller_per_second
          selector:
            matchLabels:
#可以通过这个字段对指标进行过滤，这里设置的字段会传入adapter.config中的<<.LabelMatchers>>标签。
              service: test-app
#External指标类型下只支持Value和AverageValue类型的目标值。
        target:
          type: AverageValue
          averageValue: 30
```

8. 执行以下命令，查看HPA部署情况。

```
kubectl get hpa
```

预期输出：

```
NAME          REFERENCE                TARGETS       MINPODS   MAXPODS   REPLICAS   AGE
sample-hpa    Deployment/sample-app    0/30 (avg)    1         10        1          74s
test-hpa      Deployment/test-app      0/30 (avg)    1         10        1          59m
```

9. HPA部署成功后，进行压测实验，观察业务应用是否会随着请求增大而扩容。
   i. 执行以下命令，对Host下的 `/home` 路径进行压测。

   ```
   ab -c 50 -n 5000 test.cf6828835dbba4a368e67f0b9925d****.cn-qingdao.alicontainer.com
   /home
   ```

   ii. 执行以下命令，查看HPA情况。

   ```
   kubectl get hpa
   ```

   预期输出：

   ```
   NAME          REFERENCE                TARGETS           MINPODS   MAXPODS   REPLICAS
   AGE
   sample-hpa    Deployment/sample-app    0/30 (avg)        1         10        1
   22m
   test-hpa      Deployment/test-app      22096m/30 (avg)   1         10        3
   80m
   ```

   iii. 执行以下命令，对Host的根路径进行压测。

   ```
   ab -c 50 -n 5000 test.cf6828835dbba4a368e67f0b9925d****.cn-qingdao.alicontainer.com
   /
   ```

   iv. 执行以下命令，查看HPA情况。

   ```
   kubectl get hpa
   ```

   预期输出：

   ```
   NAME          REFERENCE                TARGETS           MINPODS   MAXPODS   REPLICAS
   AGE
   sample-hpa    Deployment/sample-app    27778m/30 (avg)   1         10        2
   38m
   test-hpa      Deployment/test-app      0/30 (avg)        1         10        1
   96m
   ```

   从上述结果可以看到，业务应用在请求量增大超过阈值的情况下成功扩容。

9.服务网格

9.1. 通过ASM管理ACK虚拟节点上的ECI Pod应用

阿里云容器服务基于虚拟节点和ECI提供了多种Serverless Container产品形态，例如通过部署ACK虚拟节点组件创建ECI Pod实现了Kubernetes与弹性容器实例ECI的无缝连接。您可以灵活动态地按需创建ECI Pod，免去集群容量规划的麻烦。本文介绍如何在服务网格ASM中管理运行在ACK虚拟节点上的ECI Pod应用。

前提条件

- ASM实例为v1.7.5.41-ge61a01c3-aliyun或以上版本。
- 已在ACK集群中部署ack-virtual-node组件，并确保运行正常。具体操作，请参见步骤一：在ACK集群中部署ack-virtual-node组件。
- 已添加ACK集群到ASM实例。具体操作，请参见添加集群到ASM实例。

启用自动注入

在ASM控制台中启动自动注入功能，可以在创建Pod的过程中，将Sidecar自动注入Proxy容器，以实现数据平面的网格化。

1. 登录ASM控制台。
2. 在左侧导航栏，选择**服务网格 > 网格管理**。
3. 在**网格管理**页面，找到待配置的实例，单击实例的名称或在**操作**列中单击**管理**。
4. 在网格详情页面左侧导航栏选择**网格实例 > 全局命名空间**。
5. 在**命名空间**页面下找到待注入的命名空间（本示例中包括2个命名空间default和vk， 如果不存在需要先创建），在**自动注入**列单击**启用Sidecar自动注入**。
6. 在**确认**对话框，单击**确定**。

创建ECI Pod应用

> **说明** 创建ECI Pod应用后，ASM可以通过Sidecar对ECI Pod应用进行数据平面化管理。

通过配置Pod标签的方式创建ECI Pod应用

给Pod添加 `label alibabacloud.com/eci=true` 的标签，Pod将以ECI方式运行，并且所在的节点是虚拟节点。

1. 执行以下命令，确认default命名空间已包含 `istio-injection=enabled` 标签。

   ```
   kubectl get ns default --show-labels
   ```

   预期输出：

   ```
   NAME      STATUS   AGE   LABELS
   default   Active   84d   istio-injection=enabled,provider=asm
   ```

2. 执行以下命令，部署Nginx应用。

   ```
   kubectl run nginx -n default --image nginx -l alibabacloud.com/eci=true
   ```

3. 执行以下命令，查看虚拟节点上的Pod信息。

```
kubectl get pod -n default -o wide|grep virtual-kubelet
```

通过配置Namespace标签的方式创建ECI Pod应用

给Pod所在的命名空间添加 `label alibabacloud.com/eci=true` 标签，Pod将以ECI方式运行，并且所在的节点是虚拟节点。

1. 执行以下命令，确认vk命名空间已包含 `istio-injection=enabled` 标签。

```
kubectl get ns default --show-labels
```

预期输出：

```
NAME      STATUS   AGE   LABELS
default   Active   84d   istio-injection=enabled,provider=asm
```

2. 执行以下命令，为vk命名空间添加标签。

```
kubectl label namespace vk alibabacloud.com/eci=true
```

3. 执行以下命令，部署Nginx应用。

```
kubectl -n vk run nginx --image nginx
```

4. 执行以下命令，查看虚拟节点上的Pod信息。

```
kubectl -n vk get pod -o wide|grep virtual-kubelet
```

9.2. 通过ASM管理外部注册Kubernetes集群应用

您可以通过容器服务控制台注册外部Kubernetes集群，并通过服务网格ASM进行应用管理。

前提条件

- 一个可以访问公网地址的外部Kubernetes集群，并能够通过容器服务控制台进行注册，请参见创建注册集群并接入本地数据中心集群。
- 已开通服务网格功能，请参见创建ASM实例。

操作步骤

1. 登录ASM控制台。
2. 在左侧导航栏，选择**服务网格 > 网格管理**。
3. 在**网格管理**页面，单击**创建新网格**。
4. 在**创建新网格**页面，填写网格的名称、选择相应的地域、专有网络VPC及交换机。

> **说明**
> - 根据注册的外部接入集群所在的地域信息，选择与之更近距离的地域.。
> - 在已有VPC列表中选中注册的外部接入集群所使用的VPC实例。
> - 在交换机列表中选择所需的交换机。如果没有您需要的交换机，单击**创建交换机**进行创建，请参见创建和管理交换机。

5. 设置是否开放**使用公网地址暴露API Server**。

> **说明** ASM实例的运行基于Kubernetes运行时，可以通过API Server定义执行各种网格资源，如虚拟服务、目标规则或者Istio网关等。
> - 如果选择开放，会创建一个EIP，并挂载到私网SLB上。API Server会暴露6443端口，您可以在公网通过kubeconfig来连接和操作集群，从而定义网格资源。
> - 如果选择不开放，则不会创建EIP，您只能在VPC下通过kubeconfig来连接和操作集群，从而定义网格资源。

6. 设置**使用公网地址暴露Istio Pilot**。

> **说明** 请勾选此选项，否则注册的外部接入集群中的Pod无法连接到Pilot，导致应用无法正常使用。

7. 其他设置可保持默认值，暂不启用。单击**确定**，开始创建实例。

> **说明** 一个ASM实例的创建时间约为2到3分钟。

8. 在**网格管理**页面，找到待配置的实例，单击实例的名称或在**操作**列中单击**管理**。
9. 在网格详情页面左侧导航栏选择**集群与工作负载管理 > Kubernetes集群**，然后在右侧页面单击**添加**。
10. 在**添加集群**面板，选中需要添加的外部接入集群，然后单击**确定**。

> **说明** 添加集群之后，ASM实例的状态变为**更新中**。数秒之后（时长与添加的外部接入集群网络访问速度有关），单击页面右上方的**刷新**，网格状态会变为**运行中**。在**Kubernetes集群**页面，可以查看已添加集群的信息。

11. 在网格详情页面左侧导航栏单击**ASM网关**，然后在右侧页面单击**创建**。
12. 在部署入口网关面板配置参数。
    i. 在**部署集群**下拉列表中选择要部署入口网关的外部接入集群。
    ii. 选择负载均衡的类型，公网访问或内网访问。

> **说明** 针对不同的外部接入集群，可能支持不同的负载均衡类型，例如不支持内网访问，需要根据实际情况选择类型。如果不支持负载均衡类型的服务，那么可以暂选择公网类型，待创建之后，修改对应的YAML内容重新指定服务类型，例如指定为Nodeport或者ClusterIP类型。
> 目前针对不同的外部接入集群，只能选择**新建负载均衡**。

iii. 配置端口映射。

> 说明
>
> - 建议容器端口与服务端口一致，并在Istio网关资源定义中启用了该端口。
> - 控制台默认提供了4个Istio常用的端口，但并不意味着必须从中选择，您可以根据需要自行添加或删除端口。

13. 单击**确定**，完成部署。

成功添加入口网关后，可登录到外部接入集群查看详情。

部署应用到外部接入集群

使用Kubectl命令行或者登录到外部接入集群控制台（如果存在），部署应用到外部接入集群中，请参见部署应用到ASM实例。

定义Istio资源

在服务网格ASM控制台中可以定义Istio资源，请参见使用Istio资源实现版本流量路由。

9.3. 部署应用示例到包含同VPC多集群的ASM实例

通过服务网格ASM，可以将一个应用的服务组件部署在同VPC的多个集群上。本文以Bookinfo应用为例，介绍如何将该应用部署到包含两个集群的ASM实例。

前提条件

- 在同一VPC下创建两个ACK集群（本例中m1c1和m1c2），详情请参见创建Kubernetes专有版集群。
- 创建一个ASM实例（本例中mesh1），详情请参见创建ASM实例。

步骤一：修改集群的安全组名称

将两个集群对应的安全组名称修改为易于辨识的名称，本例中为m1c1-sg和m1c2-sg。

1. 登录ECS管理控制台。
2. 在左侧导航栏，选择**网络与安全 > 安全组**。
3. 在顶部菜单栏左上角处，选择地域。
4. 在**安全组**页面中，找到需要修改的安全组，单击**操作**列下的**修改**。
5. 在弹出的对话框中，修改**安全组名称**和**描述**。
6. 单击**确定**。

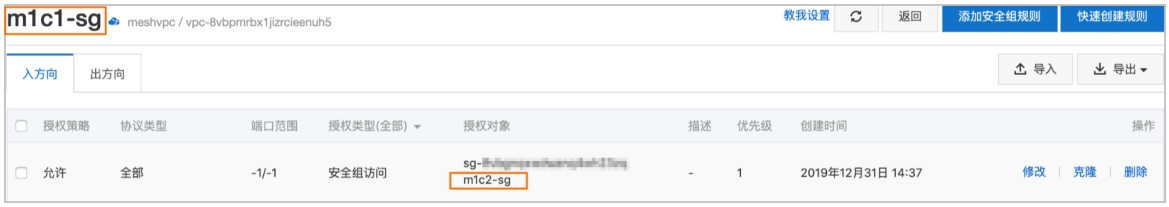
修改后的名称，如下图所示。



步骤二：配置集群的互访联通性

为了使两个集群能够互相访问，需要为彼此添加安全组访问规则。

1. 在m1c1-sg安全组配置界面，添加以m1c2-sg为授权对象的访问规则。详情请参见添加安全组规则。



2. 在m1c2-sg安全组规则配置界面，添加以m1c1-sg为授权对象的访问规则。



步骤三：添加集群到ASM实例并部署集群的入口网关

将两个集群添加到ASM实例后，由于两个集群已实现访问互通，因此只需为一个集群部署入口网关。

1. 将两个集群添加到ASM实例，详情请参见添加集群到ASM实例。
2. 为m1c1集群部署入口网关，详情请参见添加入口网关服务。

步骤四：部署Bookinfo应用

为了演示ASM跨集群的应用部署能力，Bookinfo应用的不同微服务分别部署在两个集群上。

1. 在m1c2中部署不包含review-v3 deployment的Bookinfo应用，详情请参见部署应用到ASM实例。

> **说明** Review-v3 deployment对应的功能是书评中显示红色星。

对应的YAML文件内容如下所示：

```
# Details service
apiVersion: v1
kind: Service
metadata:
  name: details
  labels:
    app: details
    service: details
spec:
  ports:
  - port: 9080
    name: http
  selector:
    app: details
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: bookinfo-details
  labels:
```

```
    account: details
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: details-v1
  labels:
    app: details
    version: v1
spec:
  replicas: 1
  selector:
    matchLabels:
      app: details
      version: v1
  template:
    metadata:
      labels:
        app: details
        version: v1
    spec:
      serviceAccountName: bookinfo-details
      containers:
      - name: details
        image: docker.io/istio/examples-bookinfo-details-v1:1.15.0
        imagePullPolicy: IfNotPresent
        ports:
        - containerPort: 9080
---
# Ratings service
apiVersion: v1
kind: Service
metadata:
  name: ratings
  labels:
    app: ratings
    service: ratings
spec:
  ports:
  - port: 9080
    name: http
  selector:
    app: ratings
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: bookinfo-ratings
  labels:
    account: ratings
---
apiVersion: apps/v1
kind: Deployment
metadata:
```

```
  name: ratings-v1
  labels:
    app: ratings
    version: v1
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ratings
      version: v1
  template:
    metadata:
      labels:
        app: ratings
        version: v1
    spec:
      serviceAccountName: bookinfo-ratings
      containers:
      - name: ratings
        image: docker.io/istio/examples-bookinfo-ratings-v1:1.15.0
        imagePullPolicy: IfNotPresent
        ports:
        - containerPort: 9080
---
# Reviews service
apiVersion: v1
kind: Service
metadata:
  name: reviews
  labels:
    app: reviews
    service: reviews
spec:
  ports:
  - port: 9080
    name: http
  selector:
    app: reviews
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: bookinfo-reviews
  labels:
    account: reviews
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: reviews-v1
  labels:
    app: reviews
    version: v1
spec:
```

```
  replicas: 1
  selector:
    matchLabels:
      app: reviews
      version: v1
  template:
    metadata:
      labels:
        app: reviews
        version: v1
    spec:
      serviceAccountName: bookinfo-reviews
      containers:
      - name: reviews
        image: docker.io/istio/examples-bookinfo-reviews-v1:1.15.0
        imagePullPolicy: IfNotPresent
        ports:
        - containerPort: 9080
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: reviews-v2
  labels:
    app: reviews
    version: v2
spec:
  replicas: 1
  selector:
    matchLabels:
      app: reviews
      version: v2
  template:
    metadata:
      labels:
        app: reviews
        version: v2
    spec:
      serviceAccountName: bookinfo-reviews
      containers:
      - name: reviews
        image: docker.io/istio/examples-bookinfo-reviews-v2:1.15.0
        imagePullPolicy: IfNotPresent
        ports:
        - containerPort: 9080
 ---
# apiVersion: apps/v1
# kind: Deployment
# metadata:
#   name: reviews-v3
#   labels:
#     app: reviews
#     version: v3
# spec:
```

```
#   replicas: 1
#   selector:
#     matchLabels:
#       app: reviews
#       version: v3
#   template:
#     metadata:
#       labels:
#         app: reviews
#         version: v3
#     spec:
#       serviceAccountName: bookinfo-reviews
#       containers:
#       - name: reviews
#         image: docker.io/istio/examples-bookinfo-reviews-v3:1.15.0
#         imagePullPolicy: IfNotPresent
#         ports:
#         - containerPort: 9080
---
# Productpage services
apiVersion: v1
kind: Service
metadata:
  name: productpage
  labels:
    app: productpage
    service: productpage
spec:
  ports:
  - port: 9080
    name: http
  selector:
    app: productpage
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: bookinfo-productpage
  labels:
    account: productpage
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: productpage-v1
  labels:
    app: productpage
    version: v1
spec:
  replicas: 1
  selector:
    matchLabels:
      app: productpage
      version: v1
  template:
```

```
  template:
    metadata:
      labels:
        app: productpage
        version: v1
    spec:
      serviceAccountName: bookinfo-productpage
      containers:
      - name: productpage
        image: docker.io/istio/examples-bookinfo-productpage-v1:1.15.0
        imagePullPolicy: IfNotPresent
        ports:
        - containerPort: 9080
---
```

2. 在m1c1中部署review-v3以及rating service（review依赖的服务）。

   对应的YAML文件内容如下所示：

```
# Reviews service
apiVersion: v1
kind: Service
metadata:
  name: reviews
  labels:
    app: reviews
    service: reviews
spec:
  ports:
  - port: 9080
    name: http
  selector:
    app: reviews
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: bookinfo-reviews
  labels:
    account: reviews
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: reviews-v3
  labels:
    app: reviews
    version: v3
spec:
  replicas: 1
  selector:
    matchLabels:
      app: reviews
      version: v3
  template:
    metadata:
```

```
    metadata:
      labels:
        app: reviews
        version: v3
    spec:
      serviceAccountName: bookinfo-reviews
      containers:
      - name: reviews
        image: docker.io/istio/examples-bookinfo-reviews-v3:1.15.0
        imagePullPolicy: IfNotPresent
        ports:
        - containerPort: 9080
---
# Ratings service
apiVersion: v1
kind: Service
metadata:
  name: ratings
  labels:
    app: ratings
    service: ratings
spec:
  ports:
  - port: 9080
    name: http
  selector:
    app: ratings
```

步骤五：添加虚拟服务和Istio网关

1. 在ASM实例的default命名空间下新建一个虚拟服务，名为bookinfo，详情请参见使用Istio资源实现版本流量路由。

   对应的YAML文件内容如下所示：

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: bookinfo
spec:
  hosts:
  - "*"
  gateways:
  - bookinfo-gateway
  http:
  - match:
    - uri:
        exact: /productpage
    - uri:
        prefix: /static
    - uri:
        exact: /login
    - uri:
        exact: /logout
    - uri:
        prefix: /api/v1/products
    route:
    - destination:
        host: productpage
        port:
          number: 9080
```

2. 在ASM实例的default命名空间下新建一个Istio网关，名为bookinfo-gateway，详情参见使用Istio资源实现版本流量路由。

   对应的YAML文件内容如下所示：

```
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
metadata:
  name: bookinfo-gateway
spec:
  selector:
    istio: ingressgateway # use istio default controller
  servers:
  - port:
      number: 80
      name: http
      protocol: HTTP
    hosts:
    - "*"
```

访问productpage页面，刷新页面时会轮流显示reviews的3个版本。虽然review-v3和其他服务不在同一个集群中，也可以正常显示。

（可选）步骤六：指定reviews总是用v3版本

通过定义目标规则和虚拟服务，可以定义Bookinfo应用的微服务部署策略。本例中将指定Bookinfo总是使用review v3版本。

1. 在ASM实例的default命名空间下新建一个目标规则，名为reviews。

   YAML文件的内容如下所示：

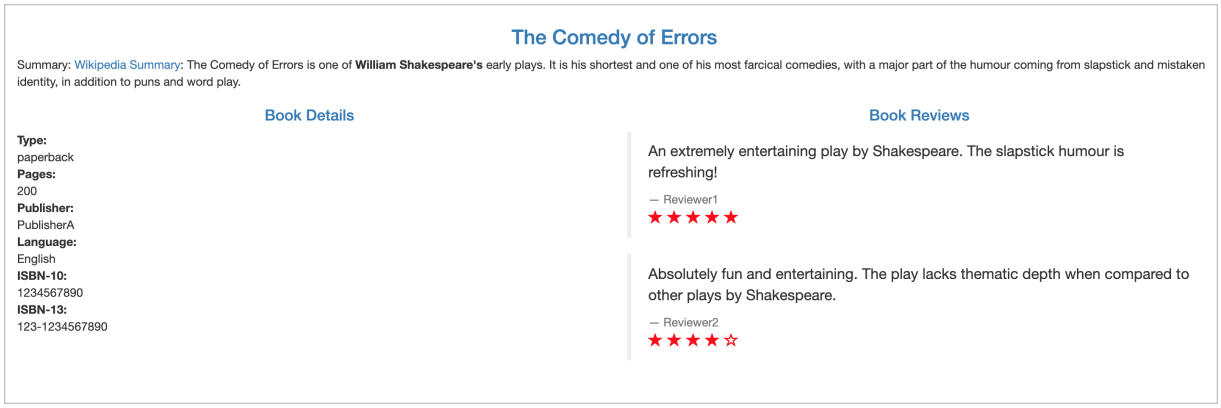
```
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: reviews
spec:
  host: reviews
  subsets:
  - name: v1
    labels:
      version: v1
  - name: v2
    labels:
      version: v2
  - name: v3
    labels:
      version: v3
```

2. 在ASM实例的default命名空间下新建一个虚拟服务，名为reviews。

   对应的YAML文件内容如下所示：

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: reviews
spec:
  hosts:
    - reviews
  http:
  - route:
    - destination:
        host: reviews
        subset: v3
```

此时访问productpage页面，reviews将始终使用v3版本，即书评中为红色星。



9.4. 通过ASM实现跨地域容灾和流量负载均衡

服务网格ASM为应用服务提供了跨地域流量分布和跨地域故障转移能力。跨地域流量分布功能可以将流量按照设定的权重路由至多个集群，实现多地域负载均衡。跨地域故障转移功能可以在某地域服务发生故障时，将该地域流量转移至其他地域，实现跨地域容灾。本文以Bookinfo应用为例，介绍如何使用跨地域故障转移和流量分布能力实现跨地域容灾和流量负载均衡。

前提条件

已创建ASM实例。具体操作，请参见创建ASM实例。

网络规划

在进行操作前，您需要对vSwitch、VPC和集群的网段、名称等信息进行规划，本文规划如下：

- vSwich和VPC的网络规划
  - vSwitch网络规划

    > **注意** 为了避免使用CEN打通VPC网络后产生路由冲突，两个vSwitch不能使用相同的网段。

vSwitch名称	VPC	IPv4网段
vpc-hangzhou-switch-1	vpc-hangzhou	192.168.0.0/24
vpc-shanghai-switch-1	vpc-shanghai	192.168.2.0/24

  - VPC网络规划

VPC名称	Region	IPv4网段
vpc-hangzhou	cn-hangzhou	192.168.0.0/16
vpc-shanghai	cn-shanghai	192.168.0.0/16

- 集群的网络规划

集群名称	Region	VPC	Pod CIDR	Service CIDR
ack-hangzhou	cn-hangzhou	vpc-hangzhou	10.45.0.0/16	172.16.0.0/16
ack-shanghai	cn-shanghai	vpc-shanghai	10.47.0.0/16	172.18.0.0/16

步骤一：创建不同地域的集群

1. 按照以上规划创建2个杭州和上海地域vSwitch，然后使用vSwitch创建VPC。具体操作，请参见创建交换机和创建专有网络和交换机。
2. 使用上文创建的VPC和网络规划创建杭州和上海地域的集群。具体操作，请参见创建Kubernetes托管版集群。

步骤二：使用CEN实现跨地域VPC网络互通

1. 创建CEN实例。
   i. 登录云企业网控制台。
   ii. 在**云企业网实例**页面，单击**创建云企业网实例**。
   iii. 在**创建云企业网实例**面板设置参数，然后单击**确定**。

参数	描述
名称	输入实例名称。 名称长度为2~128个字符，以英文字母或中文开头，可包含数字、下划线（_）和短划线（-）。
描述	输入实例描述信息。
实例类型	实例类型，本文选择专有网络（VPC）。
地域	选择所选实例的地域，本文选择华东1（杭州）。
网络实例	选择要加载的实例，选择上文创建的杭州地域的VPC。

2. 加载网络实例。
   i. 在**企业网实例**页面，找到上文创建的云企业网实例，单击实例ID。
   ii. 在**网络实例管理**页签，单击**加载网络实例**。
   iii. 在**加载网络实例**面板**同账号**页签下设置参数，然后单击**确定**。

参数	描述
实例类型	实例类型，本文选择专有网络（VPC）
地域	选择所选实例的地域，本文选择华东2（上海）。
网络实例	选择要加载的实例，选择上文创建的上海地域的VPC。

3. 购买带宽包，具体操作，请参见购买带宽包。
4. 设置跨地域互通带宽。
   i. 在**企业网实例**页面，找到上文创建的云企业网实例，单击实例ID。
   ii. 单击**跨地域互通带宽管理**页签，然后单击**设置跨地域带宽**。
   iii. 在**设置跨地域带宽**面板设置**带宽包**为上文购买的带宽包，**互通地域**为华东2（上海）到华东1（杭州），然后单击**确定**。
5. 添加安全组规则。

   在两个集群的安全组内添加对方集群的Pod网络CIDR，允许对方集群的Pod网段地址访问本机。
   i. 登录容器服务管理控制台。
   ii. 在**集群列表**页面单击ack-shanghai集群右侧**操作**列下的**详情**。
   iii. 在**集群信息**页面单击**基本信息**页签。

   查看ack-shanghai集群的Pod网络CIDR，然后返回**集群列表**页面。

iv. 在**集群列表**页面单击ack-hangzhou集群右侧**操作**列下的**详情**。

v. 在**集群信息**页面单击**集群资源**页签，然后单击**安全组**右侧的安全组ID。

vi. 在**安全组详情**页签**入方向**下单击**手动添加**。

vii. 设置**协议类型**为**全部**，**源**为ack-shanghai集群的Pod网络CIDR，其他为默认值，然后单击**操作**列下的**保存**。

viii. 重复执行以上步骤，查看ack-hangzhou集群的Pod网络CIDR，然后在ack-shanghai集群的安全组中，添加ack-hangzhou集群的Pod网络CIDR。

步骤三：将Pod路由信息发布至CEN

1. 登录容器服务管理控制台。
2. 在**集群列表**页面单击ack-hangzhou集群右侧**操作**列下的**详情**。
3. 在集群信息页面单击**集群资源**页签，然后单击**虚拟专有网络VPC**右侧VPC ID。
4. 在VPC的详情页面**路由器基本信息**区域查看路由器ID。
5. 在专有网络控制台左侧导航栏单击**路由表**。
6. 在**路由表**页面找到上文获取的路由器ID的路由实例名称，然后单击路由实例名称。
7. 在**路由条目列表**页签下单击**自定义**。
8. 单击Pod CIDR网段的子网段右侧的**发布**，本文为10.45.0.0/16的子网段。
9. 在**发布路由**的对话框单击**确定**。
10. 重复执行以上步骤，将ack-shanghai集群的Pod路由信息发布至CEN。
11. 验证ack-hangzhou和ack-shanghai集群Pod路由信息是否都发布至CEN。

    在杭州地域路由表详情页面**路由条目列表**页签下单击**动态**。可以看到ack-shanghai集群PodCIDR的路由信息和vpc-shanghai-switch-1的IPv4网段的路由信息。在上海地域路由表详情页面也可以看到ack-hangzhou集群PodCIDR的路由信息和vpc-hangzhou-switch-1的IPv4网段的路由信息。说明ack-hangzhou和ack-shanghai集群Pod路由信息都发布至CEN。

步骤四：添加集群到ASM实例。

添加杭州地域和上海地域的集群到同一个ASM实例中。

1. 登录ASM控制台。
2. 在左侧导航栏，选择**服务网格 > 网格管理**。
3. 在**网格管理**页面，找到待配置的实例，单击实例的名称或在**操作**列中单击**管理**。
4. 在网格详情页面左侧导航栏选择**集群与工作负载管理 > Kubernetes集群**，然后在右侧页面单击**添加**。
5. 在**添加集群**面板，选中杭州地域的集群，然后单击**确定**。
6. 在**重要提示**对话框中单击**确定**。
7. 重复执行以上步骤，将上海地域的集群添加到该ASM实例中。

步骤五：在ASM中配置入口网关

1. 查看ack-shanghai集群的ID。

   i. 登录容器服务管理控制台。

   ii. 在**集群列表**页面单击ack-shanghai集群右侧**操作**列下的**详情**。

iii. 在集群信息页面单击**基本信息**页签。

在**基本信息**区域查看集群ID。

2. 登录ASM控制台。
3. 在左侧导航栏，选择**服务网格 > 网格管理**。
4. 在**网格管理**页面，找到待配置的实例，单击实例的名称或在**操作**列中单击**管理**。
5. 在网格详情页面左侧导航栏单击**ASM网关**。
6. 在**ASM网关**页面，单击**创建**。
7. 在**部署入口网关**面板设置**部署集群**为ack-hangzhou集群，**负载均衡类型**为**公网访问**，选择**负载均衡类型**，其他采用默认设置。然后单击**确定**。
8. 在**ASM网关**页面单击ingressgateway右侧**操作**列下的**YAML**。
9. 在**编辑**面板补充ack-shanghai集群的ID，然后单击**确定**。

```
spec:
  clusterIds:
    - ack-hangzhou cluster-id
    - ack-shanghai cluster-id
```

步骤六：部署演示应用BookInfo

1. 使用kubectl连接到ack-hangzhou集群。具体操作，请参见通过kubectl工具连接集群。
2. 使用以下内容，创建*ack-hangzhou-k8s.yaml*。

展开此项，查看YAML详细内容 ❯

3. 执行以下命令，在ack-hangzhou集群部署BookInfo应用。

```
kubectl apply -f ack-hangzhou-k8s.yaml
```

4. 使用kubectl连接到ack-shanghai集群。具体操作，请参见通过kubectl工具连接集群。

> **说明** 使用kubectl连接到ack-shanghai集群时，您需要将ack-hangzhou集群的kubeconfig切换成ack-shanghai集群的kubeconfig。

5. 使用以下内容，创建*ack-shanghai.yaml*。

展开此项，查看YAML详细内容 ❯

6. 执行以下命令，在ack-shanghai集群部署BookInfo应用。

```
kubectl apply -f ack-shanghai.yaml
```

7. 使用kubectl连接到ASM。具体操作，请参见通过kubectl连接ASM实例。

> **说明** 使用kubectl连接到ASM时，您需要将ack-shanghai集群的kubeconfig切换成ASM的kubeconfig。

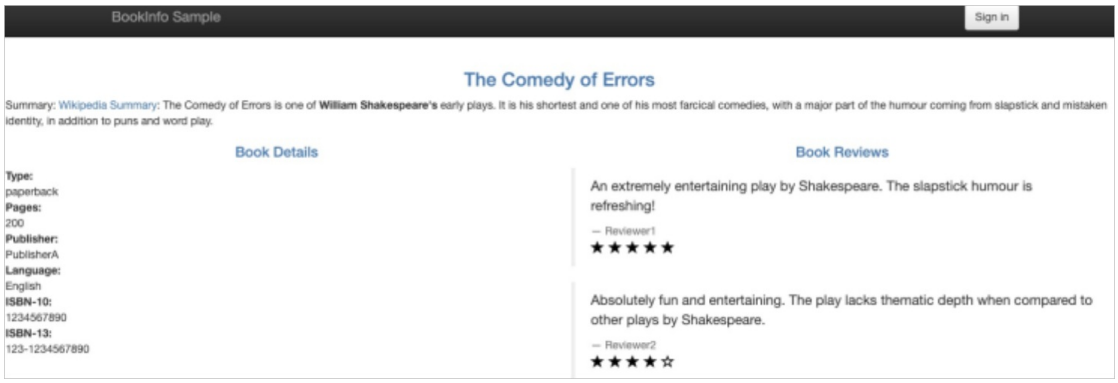
8. 使用以下内容，创建*asm.yaml*。

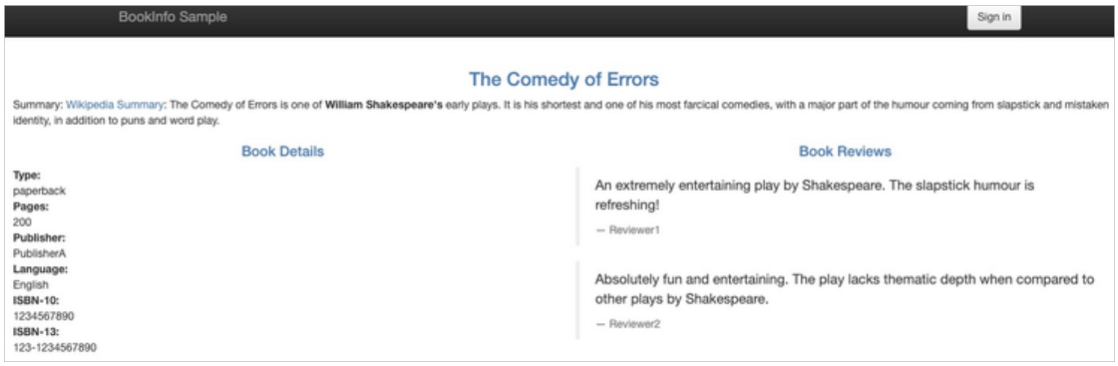
展开此项，查看YAML详细内容 ❯

9. 执行以下命令，在ASM创建路由规则。

```
kubectl apply -f asm.yaml
```

10. 验证Bookinfo应用是否部署成功。
    i. 登录容器服务管理控制台。
    ii. 在控制台左侧导航栏中，单击**集群**。
    iii. 在**集群列表**页面单击ack-hangzhou集群右侧**操作**列下的**详情**。
    iv. 在集群管理页左侧导航栏中，选择**网络 > 服务**。
    v. 在**服务**页面顶部设置**命名空间**为Istio-system，查看istio-ingressgateway右侧**外部端点**下端口为80的IP地址。
    vi. 在浏览器地址栏中输入*入口网关IP地址/productpage*。
    多次刷新页面，可以看到以下图片轮流出现。





步骤七：使用跨地域流量分布和跨地域故障转移

设置跨地域流量分布

1. 登录ASM控制台。
2. 在左侧导航栏，选择**服务网格 > 网格管理**。
3. 在**网格管理**页面，找到待配置的实例，单击实例的名称或在**操作**列中单击**管理**。
4. 在网格信息页面**基本信息**区域单击**跨地域负载均衡**右侧的**启用跨地域流量分布**。

> **说明** 如果您已经启用跨地域故障转移，您需要禁用跨地域故障转移后，才能启用跨地域流量分布。

5. 在**跨地域流量分布**对话框设置**策略**为cn-hangzhou，然后单击**新建策略**。
6. 单击 > 图标，然后单击 ⊕ 图标，设置**目标**为cn-hangzhou，**权重**为90%。

7. 单击⊕图标，设置**目标**为cn-shanghai，**权重**为10%，单击**确认**。

8. 执行以下命令，循环请求10次BookInfo应用，验证跨地域流量分布是否成功。

```
for ((i=1;i<=10;i++));do curl http://<ack-hangzhou集群中80端口的入口网关地址>/productpage 2>&1|grep full.stars;done
```

预期输出：

```
<!-- full stars: -->
<!-- full stars: -->
```

可以看到访问10次，输出2行 `full stars` ，说明10次请求中9次路由到了ack-hangzhou集群的v1版本reviews服务，1次路由到了ack-shanghai集群的v2版本reviews服务，流量按照权重路由到多个集群成功。

设置跨地域故障转移

1. 停用ack-hangzhou集群的review。
   i. 登录容器服务管理控制台。
   ii. 在控制台左侧导航栏中，单击**集群**。
   iii. 在集群管理页左侧导航栏中，选择**工作负载 > 无状态**。
   iv. 在**无状态**页面设置**命名空间**为default，单击reviews-v1右侧**操作**列下的**伸缩**。
   v. 在**伸缩**页面设置**所需容器组数量**为0，然后单击**确定**。
2. 配置DestinationRule。

   配置DestinationRule，当1s内无法请求到reviews服务，该reviews服务将被移除1min。
   i. 登录ASM控制台。
   ii. 在左侧导航栏，选择**服务网格 > 网格管理**。
   iii. 在**网格管理**页面，找到待配置的实例，单击实例的名称或在**操作**列中单击**管理**。
   iv. 在网格详情页面左侧导航栏选择**流量管理 > 目标规则**。
   v. 在**目标规则**页面单击reviews右侧**操作**列下的**YAML**。
   vi. 在**编辑**面板增加以下内容，然后单击**确定**。

```
spec:
  ......
  trafficPolicy:
    connectionPool:
      http:
        maxRequestsPerConnection: 1
    outlierDetection:
      baseEjectionTime: 1m
      consecutive5xxErrors: 1
      interval: 1s
```

   - maxRequestsPerConnection：最大连接数量。
   - baseEjectionTime：最小的移除时间长度。
   - consecutive5xxErrors：连续错误数量。
   - interval：移除检测的时间间隔。

3. 启用跨地域故障转移。
   i. 在网格信息页面**基本信息**区域单击**跨地域负载均衡**右侧的**启用跨地域故障转移**。

   > **说明** 如果您已经启用跨地域流量分布，您需要禁用跨地域流量分布后，才能启用跨地域故障转移。

   ii. 在**跨地域故障转移**对话框中设置当**策略源**是cn-shanghai，源故障转移至cn-hangzhou，当**策略源**是cn-hangzhou，源故障转移至cn-shanghai，然后单击**确认**。
4. 执行以下命令，循环请求10次BookInfo应用，并对路由到v2版本reviews服务的结果数量进行统计。

```
for ((i=1;i<=10;i++));do curl http://<ack-hangzhou集群中80端口的入口网关地址>/productpage
2>&1|grep full.stars;done|wc -l
```

预期输出：

```
20
```

可以看到访问10次，返回20（每次路由到v2版本reviews服务会返回两行包含`full stars`的结果），说明10次请求全部路由到了ack-shanghai集群的v2版本reviews服务，跨地域故障转移成功。

9.5. 使用ASM指标实现工作负载的自动弹性伸缩

服务网格ASM为ACK集群内的服务通信提供了一种非侵入式的生成遥测数据的能力。这种遥测功能提供了服务行为的可观测性，可以帮助运维人员对应用程序进行故障排除、维护和优化，而不会带来任何额外负担。根据监控的四个黄金指标维度（延迟、流量、错误和饱和度），服务网格ASM为管理的服务生成一系列指标。本文介绍如何使用ASM指标实现工作负载的自动弹性伸缩。

前提条件

- 已创建ACK集群。更多信息，请参见创建Kubernetes托管版集群。
- 已创建ASM实例。更多信息，请参见创建ASM实例。
- 已在ACK集群中创建Prometheus实例和Grafana示例。更多信息，请参见开源Prometheus监控。
- 已集成Prometheus实现网格监控。更多信息，请参见集成自建Prometheus实现网格监控。

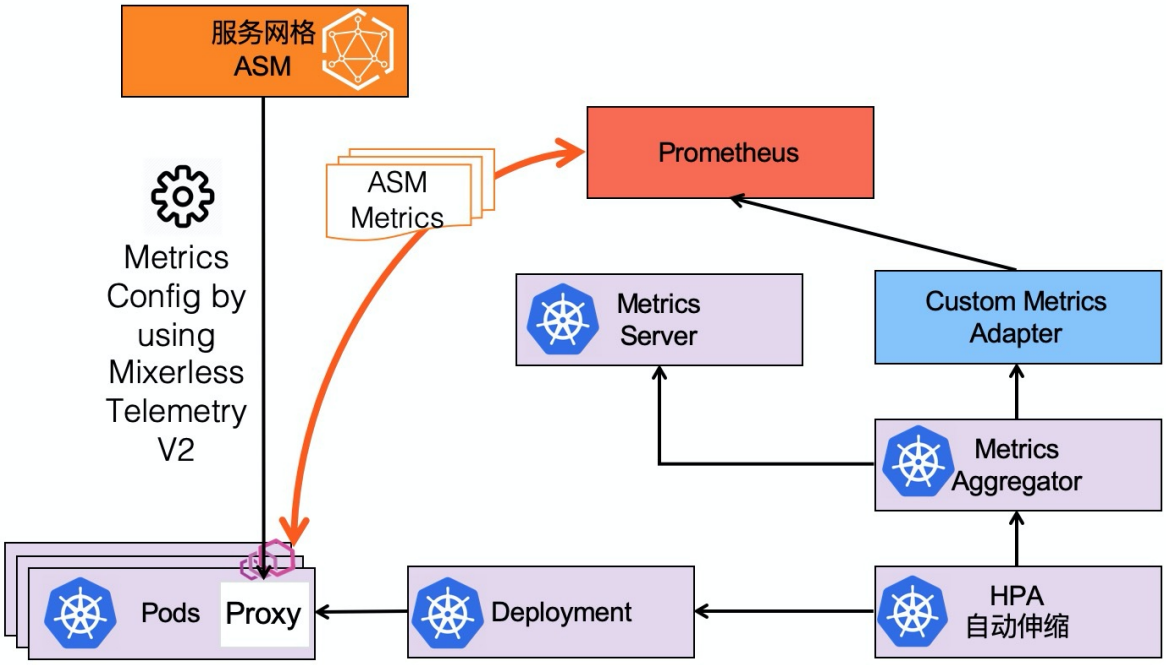
背景信息

服务网格ASM为管理的服务生成一系列指标。更多信息，请参见Istio标准指标。

自动伸缩是一种根据资源使用情况进行自动扩缩工作负载的方法。Kubernetes中的自动伸缩具有以下两个维度：

- 集群自动伸缩器CA（Cluster Autoscaler）：用于处理节点伸缩操作，可以增加或减少节点。
- 水平自动伸缩器HPA（Horizontal Pod Autoscaler）：用于自动伸缩应用部署中的Pod，可以调节Pod的数量。

Kubernetes提供的聚合层允许第三方应用程序将自身注册为API Addon组件来扩展Kubernetes API。这样的Addon组件可以实现Custom Metrics API，并允许HPA访问任意指标。HPA会定期通过Resource Metrics API查询核心指标（例如CPU或内存）以及通过Custom Metrics API获取特定于应用程序的指标，包括ASM提供的可观测性指标。



步骤一：开启采集Prometheus监控指标

1. 登录ASM控制台。
2. 在左侧导航栏，选择**服务网格 > 网格管理**。
3. 在**网格管理**页面，找到待配置的实例，单击实例的名称或在**操作**列中单击**管理**。
4. 在网格管理详情页面单击右上角的**功能设置**。

   > **说明** 请确保ASM实例的Istio为1.6.8.4及以上版本。

5. 在**功能设置更新**面板中选中**开启采集Prometheus监控指标**，然后单击**确定**。

   ASM将自动生成采集Prometheus监控指标相关的EnvoyFilter配置。

步骤二：部署自定义指标API Adapter

1. 下载Adapter安装包，关于Adapter安装包请参见kube-metrics-adapter。然后在ACK集群中安装部署自定义指标API Adapter。

   ```
   ## 如果使用Helm v3。
   helm -n kube-system install asm-custom-metrics ./kube-metrics-adapter  --set prometheus
   .url=http://prometheus.istio-system.svc:9090
   ```

2. 安装完成后，通过以下方式确认kube-metrics-adapter已启用。
   - 确认autoscaling/v2beta已存在。

     ```
     kubectl api-versions |grep "autoscaling/v2beta"
     ```

预期输出：

```
autoscaling/v2beta
```

- 确认kube-metrics-adapter Pod状态。

```
kubectl get po -n kube-system |grep metrics-adapter
```

预期输出：

```
asm-custom-metrics-kube-metrics-adapter-85c6d5d865-2cm57          1/1     Running   0
19s
```

- 列出Prometheus适配器提供的自定义外部指标。

```
kubectl get --raw "/apis/external.metrics.k8s.io/v1beta1" | jq .
```

预期输出：

```
{
  "kind": "APIResourceList",
  "apiVersion": "v1",
  "groupVersion": "external.metrics.k8s.io/v1beta1",
  "resources": []
}
```

步骤三：部署示例应用

1. 创建test命名空间。具体操作，请参见管理命名空间。
2. 启用Sidecar自动注入。具体操作，请参见安装Sidecar代理。
3. 部署示例应用。
   i. 创建*podinfo.yaml*文件。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: podinfo
  namespace: test
  labels:
    app: podinfo
spec:
  minReadySeconds: 5
  strategy:
    rollingUpdate:
      maxUnavailable: 0
    type: RollingUpdate
  selector:
    matchLabels:
      app: podinfo
  template:
    metadata:
      annotations:
        prometheus.io/scrape: "true"
      labels:
        app: podinfo
```

```
    spec:
      containers:
      - name: podinfod
        image: stefanprodan/podinfo:latest
        imagePullPolicy: IfNotPresent
        ports:
        - containerPort: 9898
          name: http
          protocol: TCP
        command:
        - ./podinfo
        - --port=9898
        - --level=info
        livenessProbe:
          exec:
            command:
            - podcli
            - check
            - http
            - localhost:9898/healthz
          initialDelaySeconds: 5
          timeoutSeconds: 5
        readinessProbe:
          exec:
            command:
            - podcli
            - check
            - http
            - localhost:9898/readyz
          initialDelaySeconds: 5
          timeoutSeconds: 5
        resources:
          limits:
            cpu: 2000m
            memory: 512Mi
          requests:
            cpu: 100m
            memory: 64Mi
---
apiVersion: v1
kind: Service
metadata:
  name: podinfo
  namespace: test
  labels:
    app: podinfo
spec:
  type: ClusterIP
  ports:
    - name: http
      port: 9898
      targetPort: 9898
      protocol: TCP
  selector:
```

```
    app: podinfo
```

ii. 部署podinfo。

```
kubectl apply -n test -f podinfo.yaml
```

4. 为了触发自动弹性伸缩，需要在命名空间test中部署负载测试服务，用于触发请求。

i. 创建*loadtester.yaml*文件。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: loadtester
  namespace: test
  labels:
    app: loadtester
spec:
  selector:
    matchLabels:
      app: loadtester
  template:
    metadata:
      labels:
        app: loadtester
      annotations:
        prometheus.io/scrape: "true"
    spec:
      containers:
        - name: loadtester
          image: weaveworks/flagger-loadtester:0.18.0
          imagePullPolicy: IfNotPresent
          ports:
            - name: http
              containerPort: 8080
          command:
            - ./loadtester
            - -port=8080
            - -log-level=info
            - -timeout=1h
          livenessProbe:
            exec:
              command:
                - wget
                - --quiet
                - --tries=1
                - --timeout=4
                - --spider
                - http://localhost:8080/healthz
            timeoutSeconds: 5
          readinessProbe:
            exec:
              command:
                - wget
                - --quiet
                - --tries=1
```

```
                - --timeout=4
                - --spider
                - http://localhost:8080/healthz
            timeoutSeconds: 5
          resources:
            limits:
              memory: "512Mi"
              cpu: "1000m"
            requests:
              memory: "32Mi"
              cpu: "10m"
          securityContext:
            readOnlyRootFilesystem: true
            runAsUser: 10001
---
apiVersion: v1
kind: Service
metadata:
  name: loadtester
  namespace: test
  labels:
    app: loadtester
spec:
  type: ClusterIP
  selector:
    app: loadtester
  ports:
    - name: http
      port: 80
      protocol: TCP
      targetPort: http
```

ii. 部署负载测试服务。

```
kubectl apply -n test -f loadtester.yaml
```

5. 验证部署示例应用和负载测试服务是否成功。

i. 确认Pod状态。

```
kubectl get pod -n test
```

预期输出：

```
NAME                          READY   STATUS    RESTARTS   AGE
loadtester-64df4846b9-nxhvv   2/2     Running   0          2m8s
podinfo-6d845cc8fc-26xbq      2/2     Running   0          11m
```

ii. 进入负载测试器容器，并使用hey命令生成负载。

```
export loadtester=$(kubectl -n test get pod -l "app=loadtester" -o jsonpath='{.item
s[0].metadata.name}')
kubectl -n test exec -it ${loadtester} -c loadtester -- hey -z 5s -c 10 -q 2 http:/
/podinfo.test:9898
```

返回结果，生成负载成功，说明示例应用和负载测试服务部署成功。

步骤四：使用ASM指标配置HPA

定义一个HPA，该HPA将根据每秒接收的请求数来扩缩Podinfo的工作负载数量。当平均请求流量负载超过10 req/sec时，将指示HPA扩大部署。

1. 创建*hpa.yaml*。

```
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: podinfo
  namespace: test
  annotations:
    metric-config.external.prometheus-query.prometheus/processed-requests-per-second: |
      sum(
          rate(
              istio_requests_total{
                destination_workload="podinfo",
                destination_workload_namespace="test",
                reporter="destination"
              }[1m]
          )
      )
spec:
  maxReplicas: 10
  minReplicas: 1
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: podinfo
  metrics:
    - type: External
      external:
        metric:
          name: prometheus-query
          selector:
            matchLabels:
              query-name: processed-requests-per-second
        target:
          type: AverageValue
          averageValue: "10"
```

2. 部署HPA。

```
kubectl apply -f hpa.yaml
```

3. 验证HPA是否部署成功。

   列出Prometheus适配器提供的自定义外部指标。

```
kubectl get --raw "/apis/external.metrics.k8s.io/v1beta1" | jq .
```

   预期输出：

```
{
  "kind": "APIResourceList",
  "apiVersion": "v1",
  "groupVersion": "external.metrics.k8s.io/v1beta1",
  "resources": [
    {
      "name": "prometheus-query",
      "singularName": "",
      "namespaced": true,
      "kind": "ExternalMetricValueList",
      "verbs": [
        "get"
      ]
    }
  ]
}
```

返回结果中包含自定义的ASM指标的资源列表，说明HPA部署成功。

验证自动弹性伸缩

1. 进入测试器容器，并使用hey命令生成工作负载请求。

```
kubectl -n test exec -it ${loadtester} -c loadtester -- sh
~ $ hey -z 5m -c 10 -q 5 http://podinfo.test:9898
```

2. 查看自动伸缩状况。

> **说明** 默认情况下，指标每30秒执行一次同步，并且只有在最近3分钟~5分钟内容器没有重新缩放时，才可以进行放大或缩小。这样，HPA可以防止冲突决策的快速执行，并为集群自动扩展程序预留时间。

```
watch kubectl -n test get hpa/podinfo
```

预期输出：

```
NAME      REFERENCE             TARGETS          MINPODS   MAXPODS   REPLICAS   AGE
podinfo   Deployment/podinfo    8308m/10 (avg)   1         10        6          124m
```

一分钟后，HPA将开始扩大工作负载，直到请求/秒降至目标值以下。负载测试完成后，每秒的请求数将降为零，并且HPA将开始缩减工作负载Pod数量，几分钟后上述命令返回结果中的REPLICAS副本数将恢复为一个。

9.6. 通过入口网关访问网格内gRPC服务

服务网格ASM的流量管理功能支持通过入口网关访问内部的gRPC服务。本文通过示例介绍如何通过ASM入口网关访问内部gRPC服务，并在gRPC的两个版本之间进行流量切换。

前提条件

- 已创建ASM实例。具体操作，请参见创建ASM实例。
- 已创建ACK集群。具体操作，请参见创建Kubernetes托管版集群。
- 添加集群到ASM实例。具体操作，请参见添加集群到ASM实例。
- 已部署入口网关服务。具体操作，请参见添加入口网关服务。

- 已部署应用到ASM实例的集群中。具体操作，请参见部署应用到ASM实例。
- 已创建ASM实例，且版本为专业版、企业版或旗舰版。具体操作，请参见创建ASM实例。

步骤一：部署示例应用

部署名为istio-grpc-server-v1和istio-grpc-server-v2的示例应用。

1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，单击**集群**。
3. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
4. 在集群管理页左侧导航栏中，选择**工作负载 > 无状态**。
5. 在**无状态**页面**命名空间**下拉列表中选择命名空间，单击**使用YAML创建资源**。

   > **说明** 当前选中的命名空间应当已标注自动注入Sidecar，即包含istio-system=enabled标签。具体操作，请参见升级Sidecar代理。

6. 在创建页面将下面的YAML模版粘贴到模板文本框中，单击**创建**。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: istio-grpc-server-v1
  labels:
    app: istio-grpc-server
    version: v1
spec:
  replicas: 1
  selector:
    matchLabels:
      app: istio-grpc-server
      version: v1
  template:
    metadata:
      labels:
        app: istio-grpc-server
        version: v1
    spec:
      containers:
      - args:
        - --address=0.0.0.0:8080
        image: registry.cn-hangzhou.aliyuncs.com/aliacs-app-catalog/istio-grpc-server
        imagePullPolicy: Always
        livenessProbe:
          exec:
            command:
            - /bin/grpc_health_probe
            - -addr=:8080
          initialDelaySeconds: 2
        name: istio-grpc-server
        ports:
        - containerPort: 8080
        readinessProbe:
          exec:
```

```
            command:
            - /bin/grpc_health_probe
            - -addr=:8080
          initialDelaySeconds: 2
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: istio-grpc-server-v2
  labels:
    app: istio-grpc-server
    version: v2
spec:
  replicas: 1
  selector:
    matchLabels:
      app: istio-grpc-server
      version: v2
  template:
    metadata:
      labels:
        app: istio-grpc-server
        version: v2
    spec:
      containers:
        - args:
            - --address=0.0.0.0:8080
          image: registry.cn-hangzhou.aliyuncs.com/aliacs-app-catalog/istio-grpc-server
          imagePullPolicy: Always
          livenessProbe:
            exec:
              command:
                - /bin/grpc_health_probe
                - -addr=:8080
            initialDelaySeconds: 2
          name: istio-grpc-server
          ports:
            - containerPort: 8080
          readinessProbe:
            exec:
              command:
                - /bin/grpc_health_probe
                - -addr=:8080
            initialDelaySeconds: 2
---
apiVersion: v1
kind: Service
metadata:
  name: istio-grpc-server
  labels:
    app: istio-grpc-server
spec:
  ports:
  - name: grpc-backend
```

```
    port: 8080
    protocol: TCP
  selector:
    app: istio-grpc-server
  type: ClusterIP
---
```

步骤二：设置服务网格ASM的路由规则

设置服务网格的服务网关、虚拟服务和目标规则，将流量全部指向istio-grpc-server-v1。

1. 登录ASM控制台。
2. 在左侧导航栏，选择**服务网格 > 网格管理**。
3. 在**网格管理**页面，找到待配置的实例，单击实例的名称或在**操作**列中单击**管理**。
4. 创建网关规则。
   i. 在网格详情页面左侧导航栏选择**流量管理 > 网关规则**，然后在右侧页面单击**使用YAML创建**。
   ii. 在**创建**页面设置**命名空间**为**default**，将下面的YAML模板粘贴到文本框中，单击**确定**。

```
apiVersion: networking.istio.io/v1alpha3
kind: Gateway
metadata:
  name: grpc-gateway
spec:
  selector:
    istio: ingressgateway # use Istio default gateway implementation
  servers:
  - port:
      number: 8080
      name: grpc
      protocol: GRPC
    hosts:
    - "*"
```

5. 创建目标规则。
   i. 在网格详情页面左侧导航栏选择**流量管理 > 目标规则**，然后在右侧页面单击**使用YAML创建**。

ii. 在**创建**页面设置**命名空间**为**default**，将下面的YAML模板粘贴到文本框中，单击**确定**。

```
apiVersion: networking.istio.io/v1alpha3
kind: DestinationRule
metadata:
  name: dr-istio-grpc-server
spec:
  host: istio-grpc-server
  trafficPolicy:
    loadBalancer:
      simple: ROUND_ROBIN
  subsets:
    - name: v1
      labels:
        version: "v1"
    - name: v2
      labels:
        version: "v2"
```

6. 创建虚拟服务。

i. 在网格详情页面左侧导航栏选择**流量管理 > 虚拟服务**，然后在右侧页面单击**使用YAML创建**。

ii. 在**创建**页面设置**命名空间**为**default**，将下面的YAML模板粘贴到文本框中，单击**确定**。

```
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: grpc-vs
spec:
  hosts:
  - "*"
  gateways:
  - grpc-gateway
  http:
    - match:
        - port: 8080
      route:
        - destination:
            host: istio-grpc-server
            subset: v1
          weight: 100
        - destination:
            host: istio-grpc-server
            subset: v2
          weight: 0
```

步骤三：部署入口网关

在入口网关中，添加端口8080，并指向服务网关的8080端口。

1. 登录ASM控制台。
2. 在左侧导航栏，选择**服务网格 > 网格管理**。
3. 在**网格管理**页面，找到待配置的实例，单击实例的名称或在**操作**列中单击**管理**。
4. 在网格详情页面左侧导航栏单击**ASM网关**。

5. 在**ASM网关**页面单击**创建**，在**创建**页面配置参数，其他采用默认值。

参数	配置项
部署集群	选择要部署入口网关的集群。
负载均衡类型	此处指定负载均衡的类型为**公网访问**。
负载均衡	选择负载均衡。 ○ 使用已有负载均衡：从已有负载均衡列表中选择。 ○ 新建负载均衡：单击**新建负载均衡**，从下拉列表中选择所需的负载均衡规格。 **说明** 建议您为每个Kubernetes服务分配一个SLB。如果多个Kubernetes服务复用同一个SLB，存在以下风险和限制： ○ 使用已有的SLB会强制覆盖已有监听，可能会导致您的应用不可访问。 ○ Kubernetes通过Service创建的SLB不能复用，只能复用您手动在控制台（或调用OpenAPI）创建的SLB。 ○ 复用同一个SLB的多个Service不能有相同的前端监听端口，否则会造成端口冲突。 ○ 复用SLB时，监听的名字以及虚拟服务器组的名字被Kubernetes作为唯一标识符。请勿修改监听和虚拟服务器组的名字。 ○ 不支持跨集群复用SLB。
端口映射	单击**添加端口**，设置**名称**为tcp，**服务端口**和**容器端口**为8080。 **说明** 服务端口指的是整个网格对外暴露的端口，是外部访问使用的端口。而容器端口指的是从服务端口进来的流量所指向的服务网关端口。所以容器端口需要与服务网关端口一致。

6. 单击**确定**。

步骤四：运行gRPC客户端

1. 启动gRPC客户端。

```
docker run -d --name grpc-client registry.cn-hangzhou.aliyuncs.com/aliacs-app-catalog/istio-grpc-client 365d
```

2. 登录到容器。

```
docker exec -it grpc-client sh
```

3. 访问网格内的gRPC服务。

```
/bin/greeter-client --insecure=true --address=<入口网关的IP地址>:8080 --repeat=100
```

返回以下结果，可以看到所有的请求都指向了istio-grpc-server-v1。

```
2020/09/11 03:18:51 Hello world from istio-grpc-server-v1-dbdd97cc-n851w
2020/09/11 03:18:51 Hello world from istio-grpc-server-v1-dbdd97cc-n851w
2020/09/11 03:18:51 Hello world from istio-grpc-server-v1-dbdd97cc-n851w
2020/09/11 03:18:51 Hello world from istio-grpc-server-v1-dbdd97cc-n851w
2020/09/11 03:18:51 Hello world from istio-grpc-server-v1-dbdd97cc-n851w
2020/09/11 03:18:51 Hello world from istio-grpc-server-v1-dbdd97cc-n851w
2020/09/11 03:18:51 Hello world from istio-grpc-server-v1-dbdd97cc-n851w
2020/09/11 03:18:51 Hello world from istio-grpc-server-v1-dbdd97cc-n851w
2020/09/11 03:18:51 Hello world from istio-grpc-server-v1-dbdd97cc-n851w
2020/09/11 03:18:51 Hello world from istio-grpc-server-v1-dbdd97cc-n851w
2020/09/11 03:18:51 Hello world from istio-grpc-server-v1-dbdd97cc-n851w
2020/09/11 03:18:51 Hello world from istio-grpc-server-v1-dbdd97cc-n851w
2020/09/11 03:18:51 Hello world from istio-grpc-server-v1-dbdd97cc-n851w
```

步骤五：按比例将流量路由到v2

将40%的流量指向istio-grpc-server-v2，其余60%的流量仍然指向istio-grpc-server-v1。

1. 登录ASM控制台。
2. 在左侧导航栏，选择**服务网格 > 网格管理**。
3. 在**网格管理**页面，找到待配置的实例，单击实例的名称或在**操作**列中单击**管理**。
4. 在网格详情页面左侧导航栏选择**流量管理 > 虚拟服务**。
5. 在**虚拟服务**页面单击grpc-vs**操作**列的**YAML**。
6. 在**编辑**对话框中更新以下YAML内容，单击**确定**。

```
....
      route:
        - destination:
            host: istio-grpc-server
            subset: v1
          weight: 60
        - destination:
            host: istio-grpc-server
            subset: v2
          weight: 40
```

7. 登录到容器，执行以下命令，访问网格内的gRPC服务。

```
/bin/greeter-client --insecure=true --address=<入口网关的IP地址>:8080 --repeat=100
```

返回以下结果，可以看到40%的流量指向了istio-grpc-server-v2。

> **说明** 您的测试结果不一定总是100次中有40次指向istio-grpc-server-v2，但从总体比例来看，一定是接近40% 的。

```
2020/09/11 03:34:51 Hello world from istio-grpc-server-v2-665c4cf57d-h74lw
2020/09/11 03:34:51 Hello world from istio-grpc-server-v1-dbdd97cc-n85lw
2020/09/11 03:34:51 Hello world from istio-grpc-server-v1-dbdd97cc-n85lw
2020/09/11 03:34:51 Hello world from istio-grpc-server-v1-dbdd97cc-n85lw
2020/09/11 03:34:51 Hello world from istio-grpc-server-v1-dbdd97cc-n85lw
2020/09/11 03:34:51 Hello world from istio-grpc-server-v2-665c4cf57d-h74lw
2020/09/11 03:34:51 Hello world from istio-grpc-server-v1-dbdd97cc-n85lw
2020/09/11 03:34:51 Hello world from istio-grpc-server-v1-dbdd97cc-n85lw
2020/09/11 03:34:51 Hello world from istio-grpc-server-v2-665c4cf57d-h74lw
2020/09/11 03:34:51 Hello world from istio-grpc-server-v2-665c4cf57d-h74lw
```

10.架构方案

10.1. 容器跨可用区高可用

本文介绍容器跨可用区高可用的场景描述、解决问题、架构图及操作参考链接。

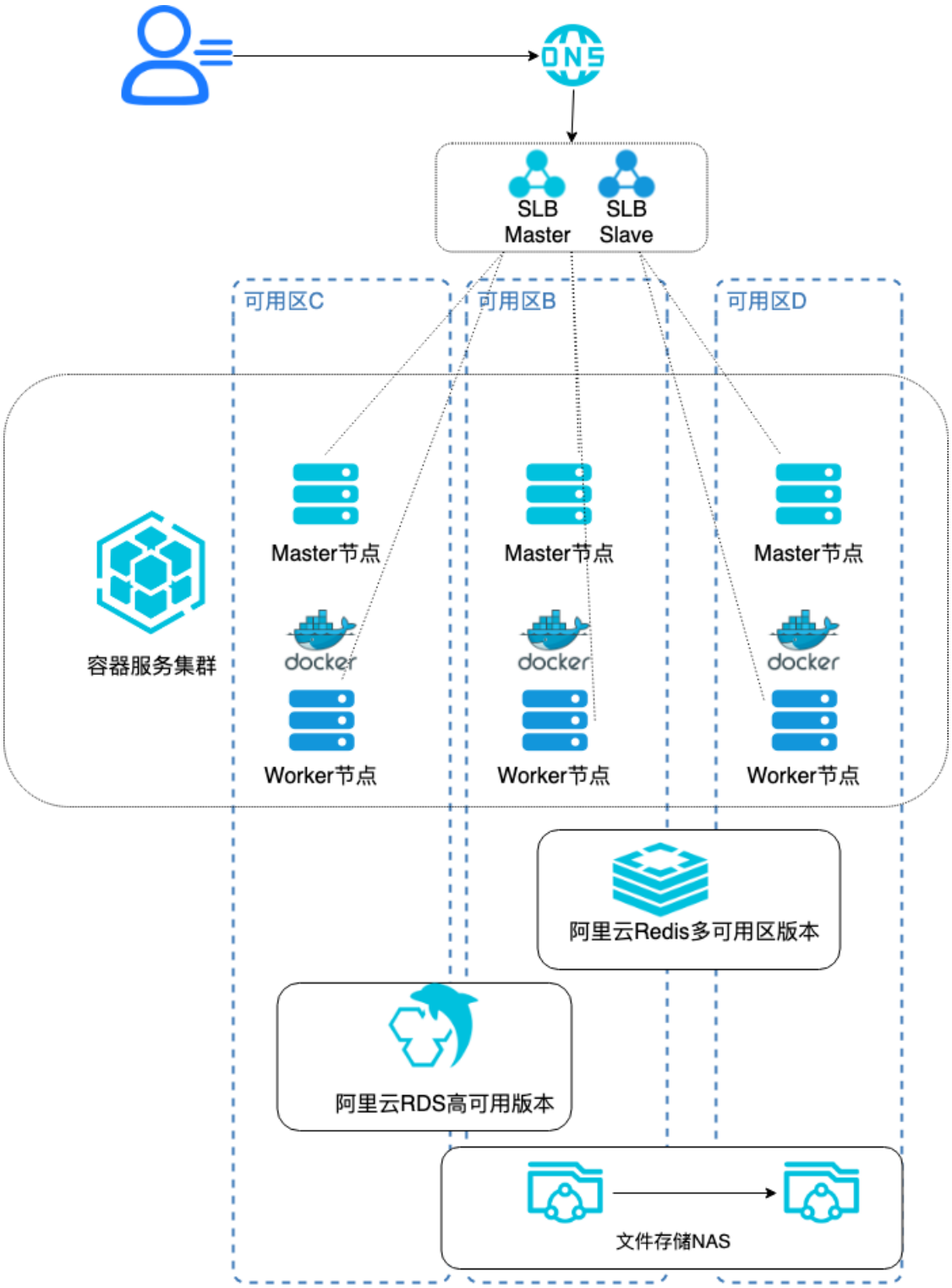
场景描述

本实践适合使用容器服务Kubernetes（ACK）结合阿里云上的其他产品构建跨可用区高可用应用的场景。您在创建ACK时可以把容器集群建成多个可用区的架构，这样的话，如果当某可用区挂掉后，则不影响应用和集群的高可用。阿里云容器服务Kubernetes版（ACK）通常配合高可用阿里云负载均衡SLB、云数据库RDS MySQL版、云数据库Redis版等产品，实现跨可用区高可用。

解决问题

- 利用ACK搭建跨可用区高可用的应用。
- ACK结合负载均衡SLB、云数据库RDS MySQL版、云数据库Redis版等产品构建高可用应用。

架构图



参考链接

有关如何实现容器跨可用区高可用的详情，请参见容器跨可用区高可用最佳实践。

10.2. 基于ECI的ACK集群高弹性架构

本文介绍基于阿里云弹性容器实例ECI的阿里云容器服务Kubernetes版（ACK）集群高弹性架构的场景描述、解决问题、架构图及操作参考链接。

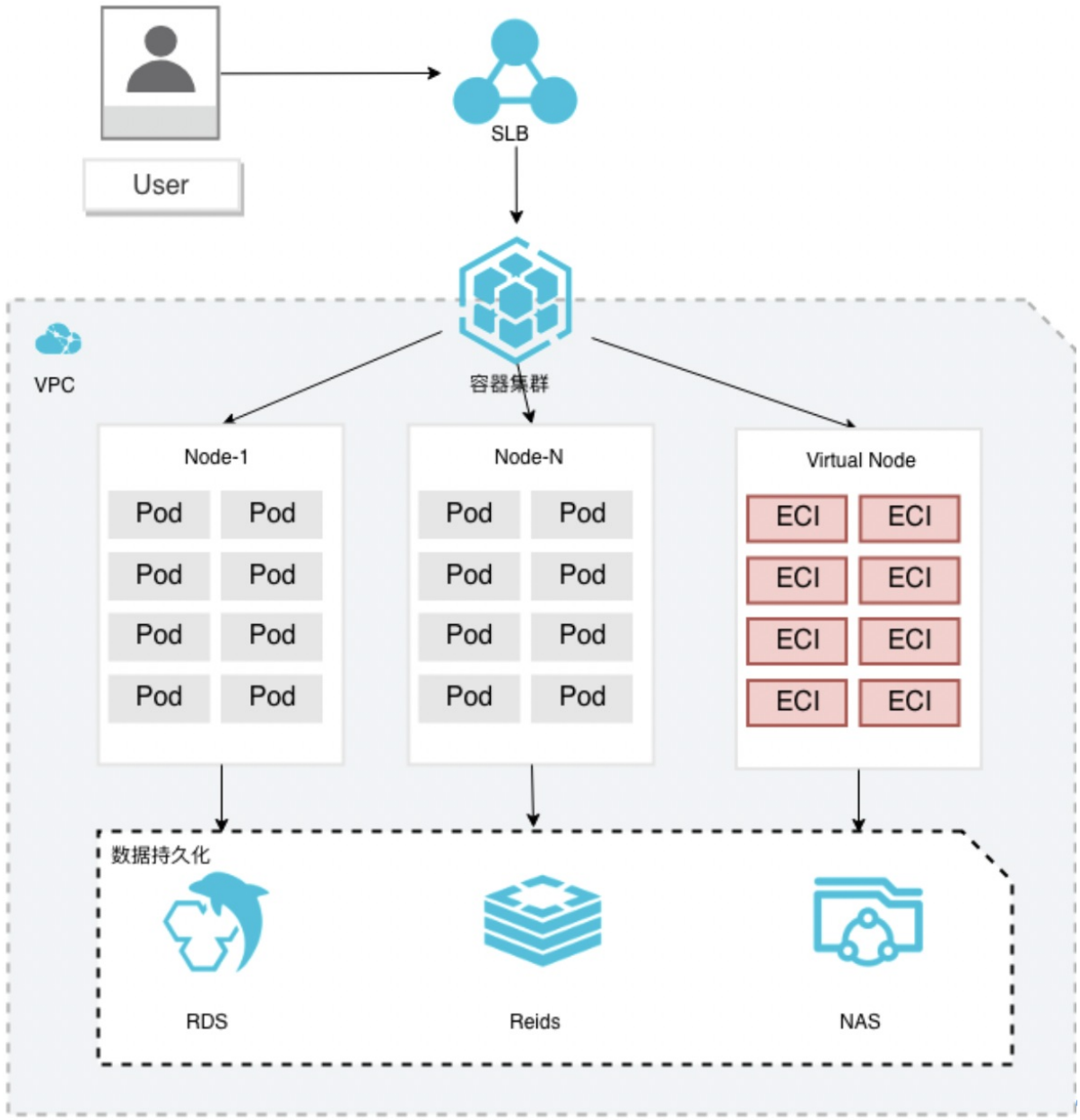
场景描述

您原有的业务部署在ACK集群上并在ACK集群上运行常规业务，当业务突发波动（如秒杀活动）时，让突增的业务运行在ECI实例上，随着业务波动动态，创建或释放ECI实例，达到成本的最优控制。同时，业务方无需管理节点和容量规划，全自动实现容器无限弹性扩容。

解决问题

- 长线类业务，例如，业务生产系统。
- 明显的业务弹性诉求，例如，爬虫、秒杀。
- 离线任务。

架构图



参考链接

有关基于ECI的ACK集群高弹性架构的详情，请参见基于ECI的ACK集群高弹性架构最佳实践。

10.3. ACK集群低成本高弹性架构

本文介绍容器服务Kubernetes版（ACK）集群低成本高弹性架构的场景描述、解决问题、架构图及操作参考链接。

场景描述

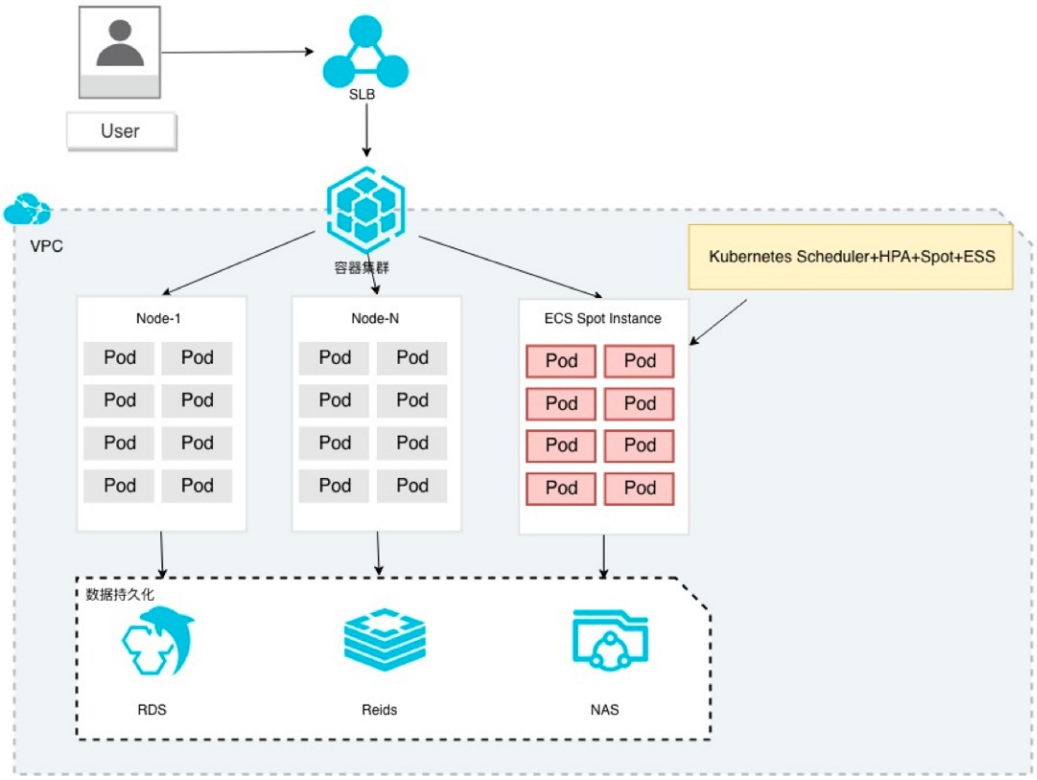
您可以针对以下场景利用云上资源的灵活性和规模化优势，构建云上高弹性、稳定、成本优化的服务。

- 业务应用容器化改造以后，关注应用的高弹性。
- 有明显波峰波谷效应的在线业务。
- 计算类的离线任务，例如机器学习、基因测序等对实时性要求不高，但是对成本敏感的业务应用。
- 定时任务类型，对批量计算、成本节约与调度能力比较关注。
- 一些比较特殊的计算场景，对资源利用和调度能力比较关注。

解决问题

- 应用容器化以后，应对应用负载的波动变化。
- 突发应用负载峰值的弹性应对能力。
- 资源成本和系统可用性的权衡博弈。

架构图



参考链接

有关ACK集群低成本高弹性架构的详情，请参见ACK集群低成本高弹性架构。

10.4. 混合云部署Kubernetes集群

本文介绍混合云部署Kubernetes集群的场景描述、解决问题、架构图及操作参考链接。

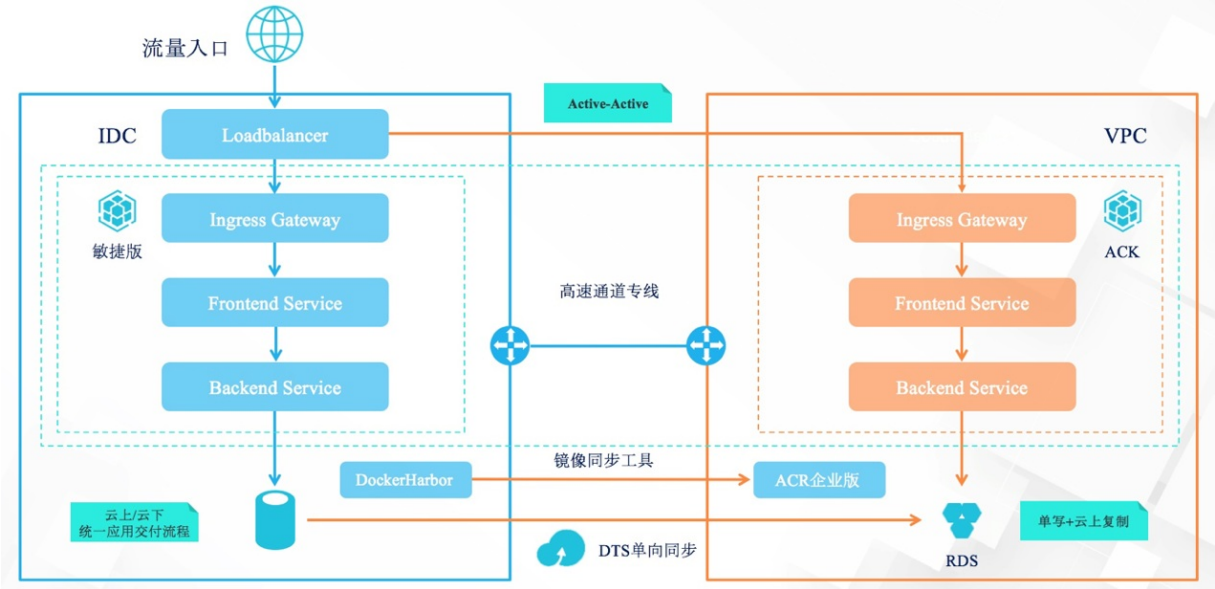
场景描述

本方案演示线下IDC与阿里云以混合云的形式部署Kubernetes集群，实现云下云上服务交叉调用。云上资源可以根据业务请求峰谷值做弹性伸缩。使用数据传输服务DTS实现IDC与云上数据库单向同步，将云上数据库作为混合云场景下的冗余和灾备，可以在云上部署可接受一定读延迟的业务应用，缓解IDC机房业务压力。

解决问题

- 云上冗余部署和快速弹性伸缩，低成本解决IDC弹性伸缩问题。
- 云上应用可作为接受一定读延迟请求分流后端，降低IDC主服务的压力。
- 容器镜像服务ACR企业版支持多地域同步部署，加速镜像分发。
- 数据传输服务DTS单向同步数据灾备，提升IDC机房数据库高可用。

架构图



参考链接

有关混合云部署Kubernetes集群的详情，请参见混合云部署Kubernetes集群最佳实践。

11.DevOps

11.1. 在Eclipse中安装Cloud Toolkit插件并部署应用到容器服务Kubernetes

本文档将向您介绍如何在Eclipse中安装Cloud Toolkit，并使用Cloud Toolkit快速部署一个应用。

背景信息

Alibaba Cloud Toolkit（下文简称Cloud Toolkit）是一个免费的IDE插件，帮助阿里云用户更高效地使用阿里云。

您只需注册或使用一个已有的阿里云账号，即可免费下载Cloud Toolkit。下载完成后，您可以将该插件安装到Eclipse中。

当您在本地完成应用程序的开发、调试及测试后，通过该插件即可轻松将应用程序部署到阿里云。关于Cloud Toolkit的详细信息，请参见Alibaba Cloud Toolkit。

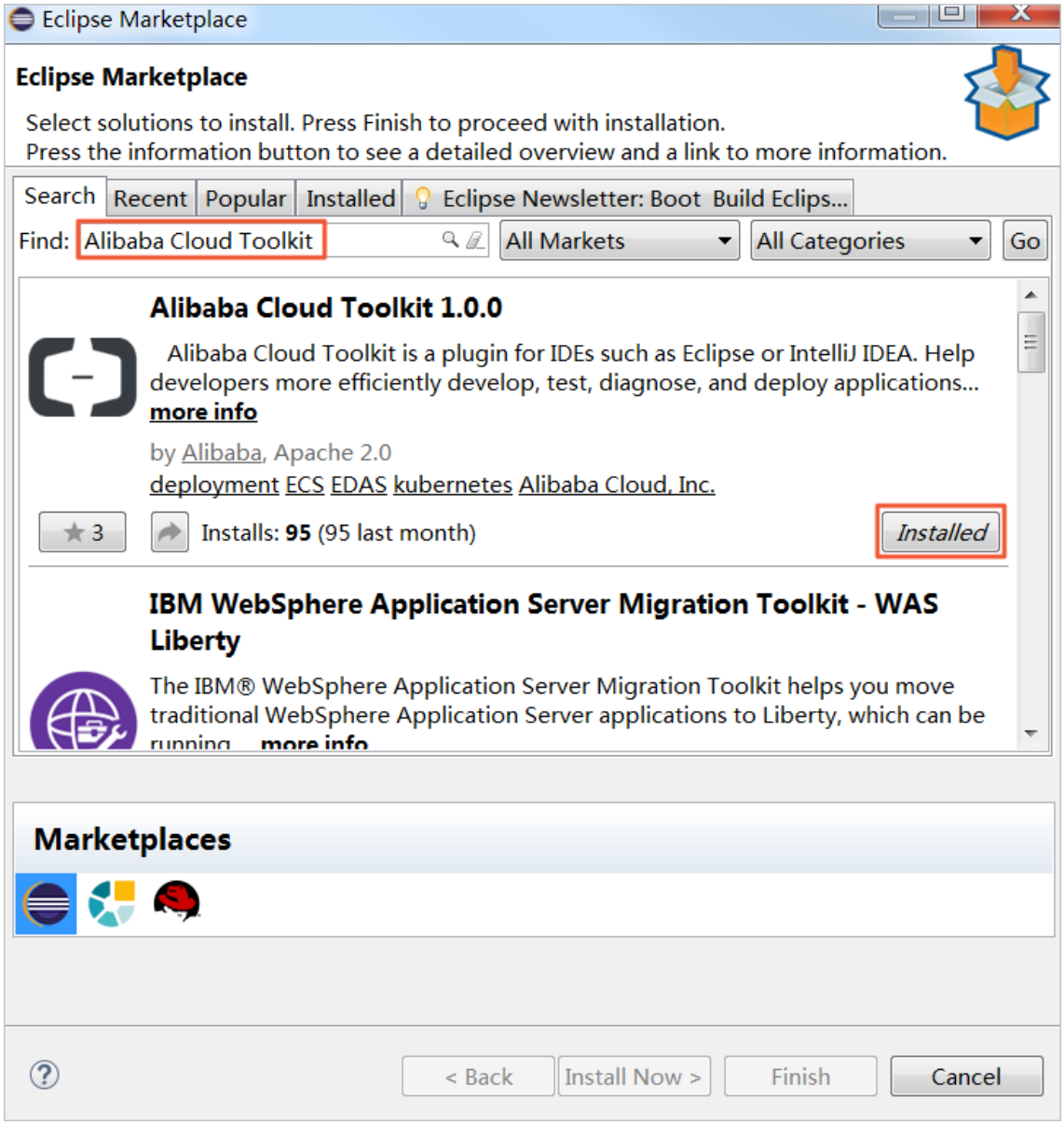
安装Cloud Toolkit

您可以在如下路径下获取软件安装包。

- 下载并安装JDK 1.8 或更高版本。
- 下载并安装适用于Java EE开发人员的Eclipse IDE、4.5.0（代号：Mars）或更高版本。

有两种安装Cloud Toolkit方式：在Eclipse Marketplace中下载安装和直接通过Cloud Toolkit的官方地址安装。

- 在Eclipse Marketplace中下载安装。
  i. 启动Eclipse。
  ii. 在菜单栏中选择**Help > Eclipse Marketplace...**。
  iii. 在**Eclipse Marketplace**对话框中**Find**右侧的文本框中输入**Alibaba Cloud Toolkit**，然后回车。
  iv. 单击搜索结果中**Alibaba Cloud Toolkit**右下角的**Installed**。



v. 按照Eclipse安装页面的提示，完成后续安装步骤。

- 通过Cloud Toolkit的官方地址安装。

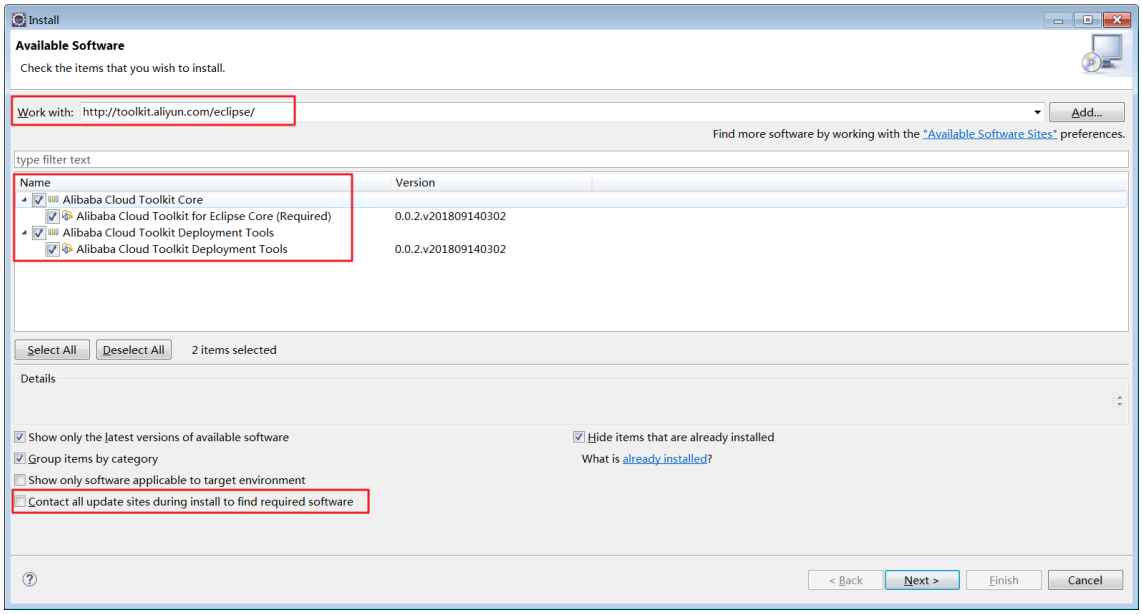
> **说明** 如果您无法连接Eclipse Market服务器，请选择这种安装方式。

i. 启动Eclipse。

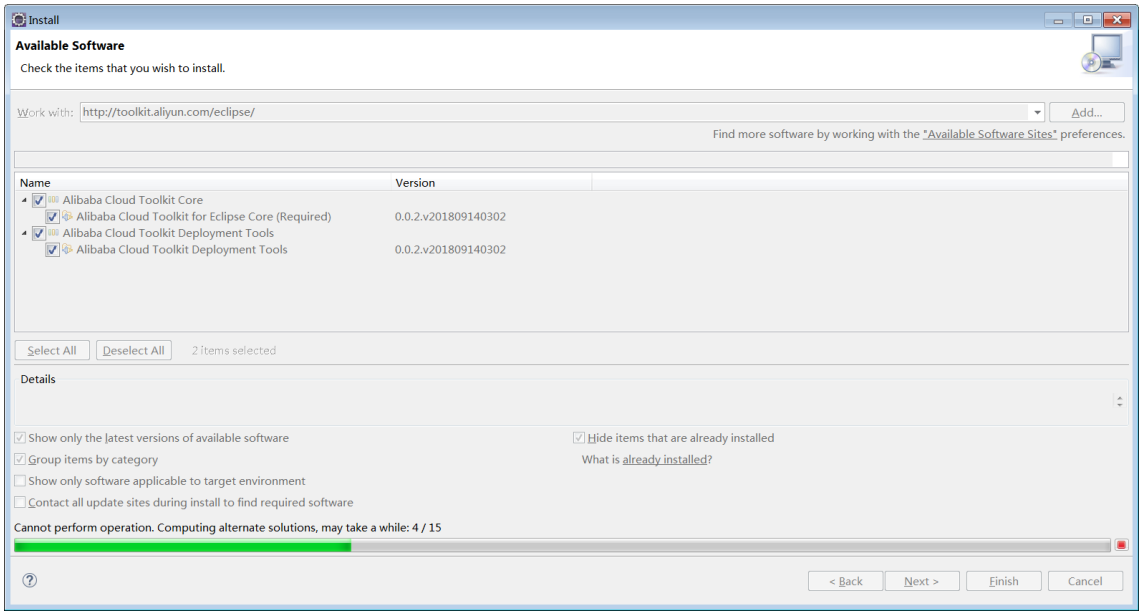
ii. 在菜单栏中选择**Help > Install New Software**。

iii. 在**Available Software**对话框的**Work with**文本框输入Cloud Toolkit for Eclipse的URL：*http://toolkit.aliyun.com/eclipse/*。

iv. 在下面的列表区域中选中需要的组件 `Alibaba Cloud Toolkit Core` 和 `Alibaba Cloud Toolkit Deployment Tools` ，并在下方的**Details**区域中不选中 `Contact all update sites during install to find required software`。



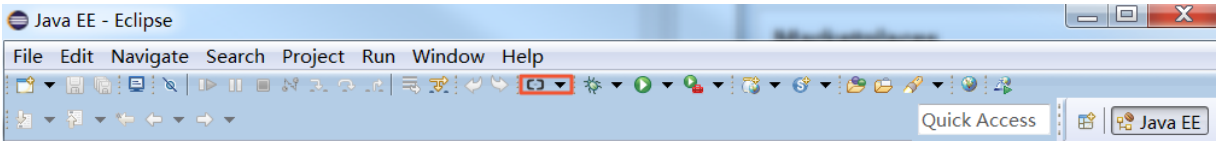
v. 配置完成后，单击**Next**，Eclipse开始安装插件，并显示安装进度。



vi. 按照Eclipse安装页面的提示，完成后续安装步骤。

预期结果：

插件安装成功后，重启Eclipse，您可以在工具栏看到Alibaba Cloud Toolkit的图标。



获取AccessKey ID和AccessKey Secret

您本地的应用部署到云端时，都需要使用阿里云上的资源、应用。所以在部署前，需要设置您的阿里云账号信息，以保证拥有使用、管理、应用相关资源的权限。

- 使用阿里云主账号获取AccessKey ID和AccessKey Secret
  i. 登录容器服务管理控制台。

ii. 将光标滑动（非单击）到控制台页面右上角您的头像上，在弹出的下拉菜单中单击**AccessKey 管理**。



iii. 在**安全提示**对话框中单击**继续使用AccessKey**。



iv. 在**安全信息管理**页面，**用户AccessKey**区域右侧单击**创建AccessKey**，在**手机验证**对话框中单击**点击获取**后输入验证码。

v. 记录该账号的AccessKey ID和AccessKey Secret。

- 使用RAM子账号获取AccessKey ID和AccessKey Secret

i. 登录RAM子账号登录页面，输入您的子账号，单击**下一步**，再输入密码，单击**登录**。

ii. 将光标滑动（非单击）到控制台页面右上角您的头像上，在弹出的下拉菜单中单击**AccessKey管理**。

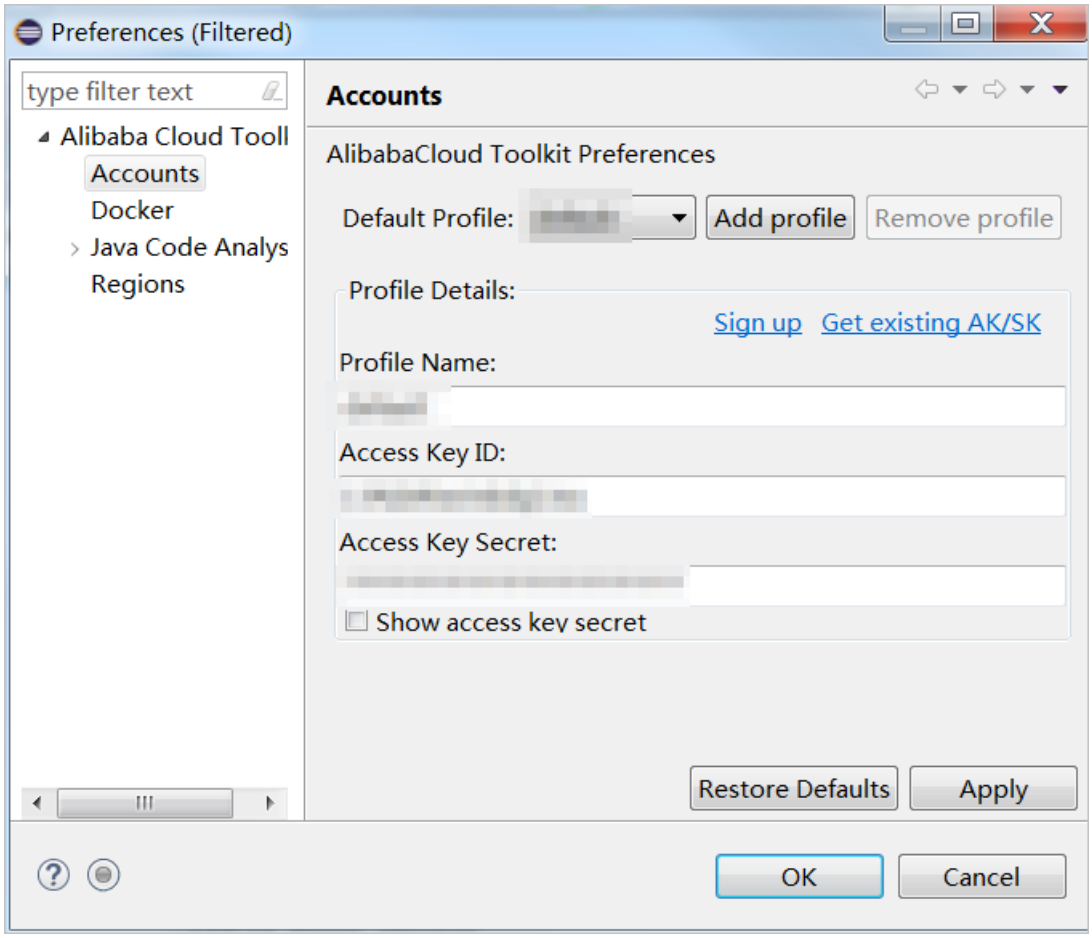


iii. 在**安全信息管理**页面，**用户AccessKey**区域右侧单击**创建AccessKey**。

> **说明**
>
> - 如果您当前子账号的**创建AccessKey**置灰不可用，请使用主账号对该子账号授权，可参见RAM授权。
> - 在**新建用户AccessKey**对话框中，单击**AccessKey详情**右侧下拉箭头，记录当前子账号的AccessKeyID和AccessKeySecret。
> - 这是用户AccessKey可供下载的唯一机会，请及时保存。

设置AccessKey ID和AccessKey Secret

1. 启动Eclipse。
2. 在工具栏单击**AlibabaCloud Toolkit Preferences**图标，在下拉菜单中单击**Preferences...** 。
3. 在**Preferences (Filtered)**对话框的左侧导航栏中单击**Accounts**。
4. 在**Accounts**区域设置**AccessKey ID**和**AccessKey Secret**，然后单击**OK**。



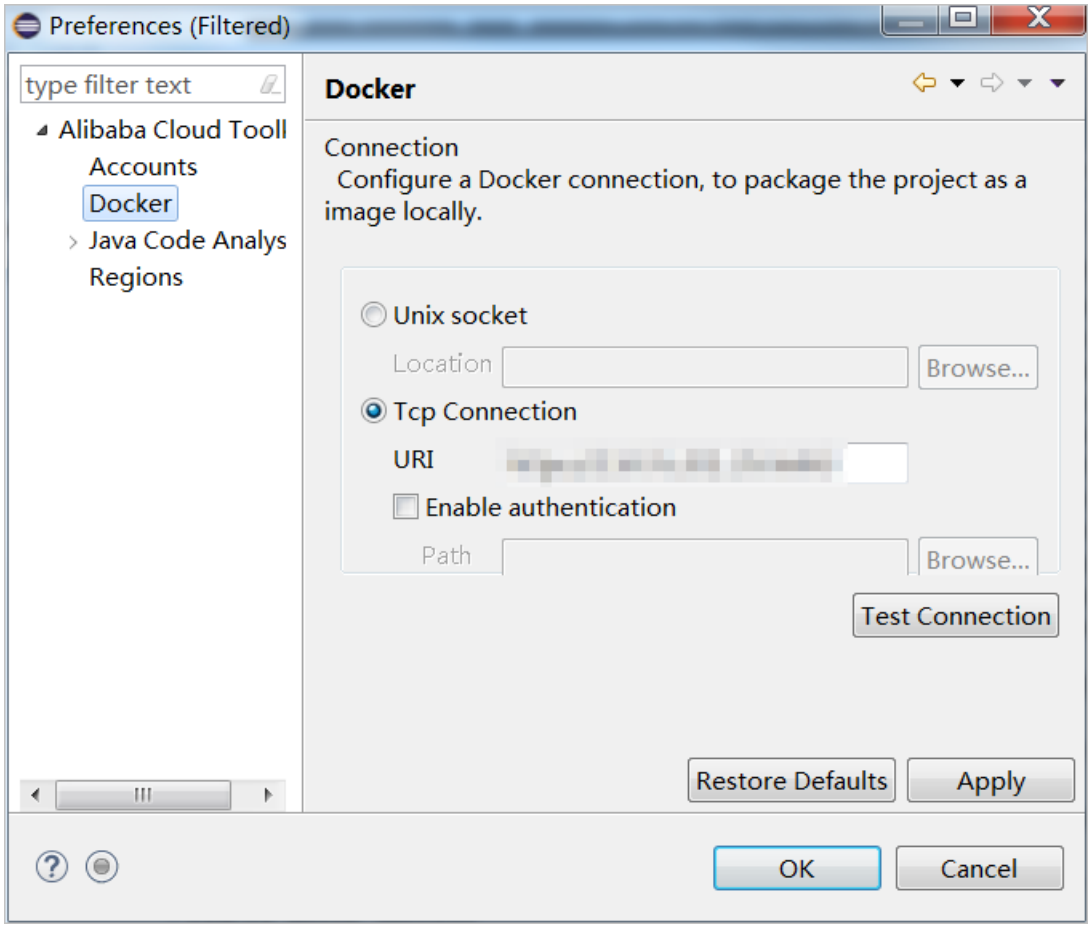
> 说明
>
> - 如果您已有阿里云账号，单击Get existing AK/SK，参见文档获取AccessKey ID和AccessKey Secret。
> - 如果您没有阿里云账号，单击Sign up，进入阿里云账号注册页面，注册账号。注册完成后按照上述方式获取AccessKey ID和AccessKey Secret。

部署应用

1. 您已经完成部署前准备工作。
   - 您已使用设置AccessKey ID和AccessKey Secret的阿里云账号进入容器镜像服务控制台，创建容器镜像仓库，可参见仓库构建。
   - 您已使用设置AccessKey ID和AccessKey Secret的阿里云账号进入容器服务管理控制台，并使用镜像创建应用，可参见创建无状态工作负载Deployment或创建有状态工作负载StatefulSet。
2. 设置用于打包本地镜像的Docker环境。
   i. 在Eclipse工具栏单击AlibabaCloud Toolkit Preferences，在下拉菜单中单击Preferences...。
   ii. 在Preferences (Filtered)对话框的左侧导航栏中单击Docker。

iii. 在Docker界面中设置可连接的Docker环境，包括本地和远程两种方式。

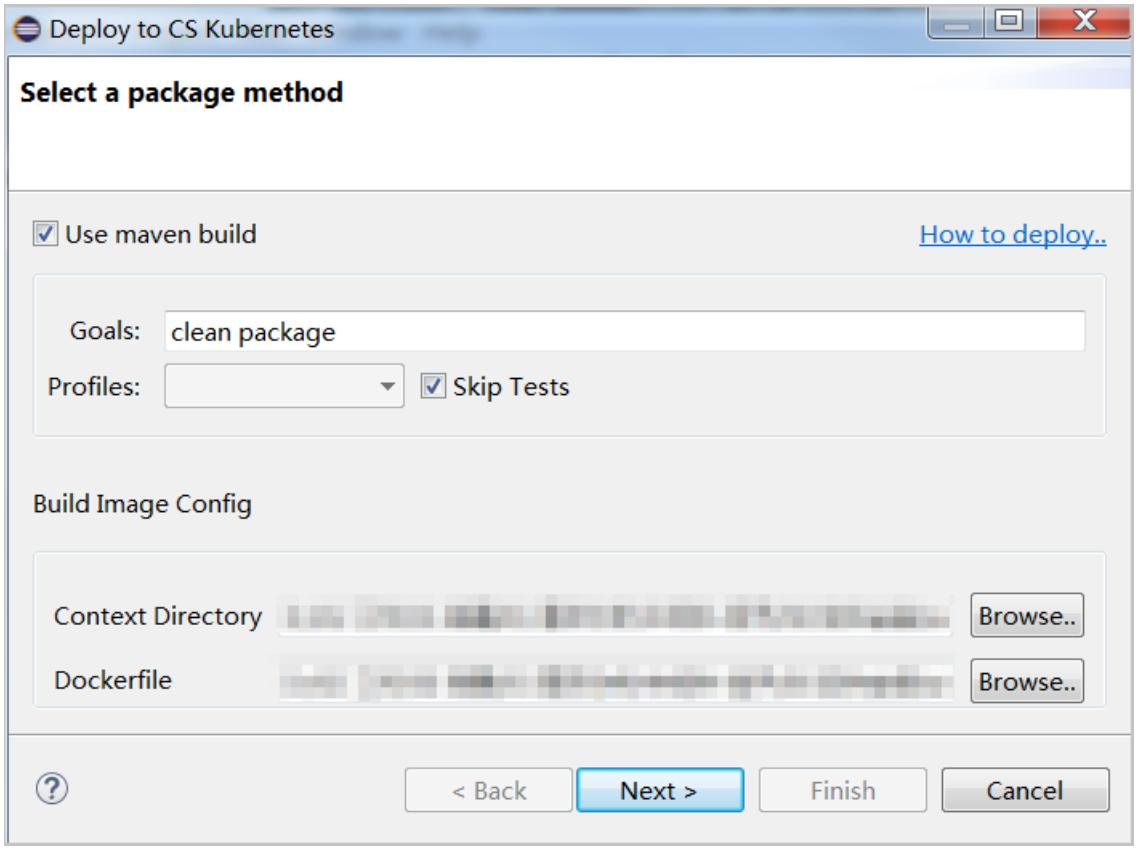
- 本地Docker环境
  - 如果您本地为Mac或Linux操作系统，勾选Unix Socket后单击Browse...，选择本地的Docker安装目录。
  - 如果您本地为Windows操作系统，勾选Tcp Connection后在URI右侧文档框输入本地Docker的URI，如 `http://127.0.0.1:2375` 。
- 远程Docker环境
  选中Tcp Connection后在URI右侧文档框输入远端Docker环境的URI（包括IP地址和端口），如 `http://127.0.0.1:2375` ，并确保远程主机的HTTP服务开启。



iv. 配置然完成后单击OK。

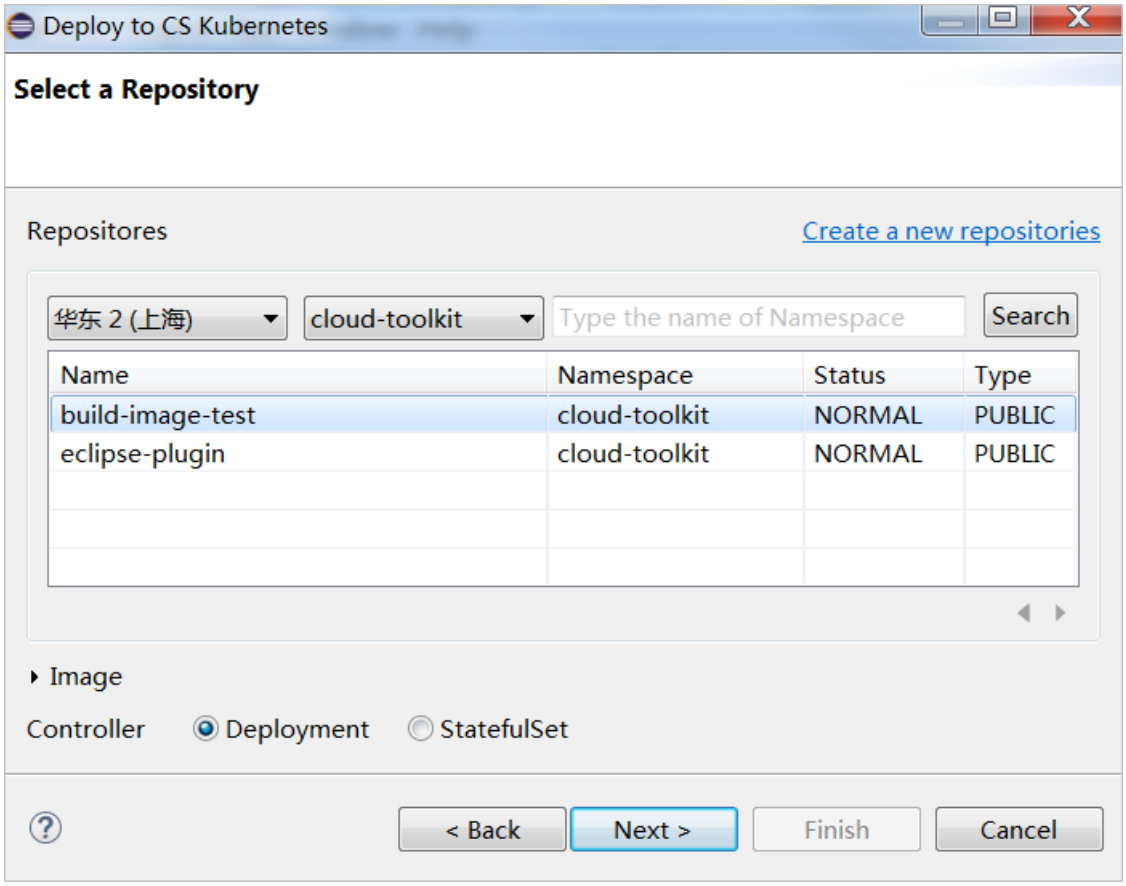
3. 在Eclipse界面左侧的Package Explorer中右键单击您的Docker应用工程名，在弹出的下拉菜单中选择Alibaba Cloud > Deploy to CS Kubernetes...。
4. 在Select a package method对话框选择本地应用程序的Context Directory和Dockerfile（通常会根据您本地的应用工程自动识别并设置），然后单击Next。

> **说明** 您可以根据您的需要决定是否勾选Use maven build使用Maven构建应用工程。



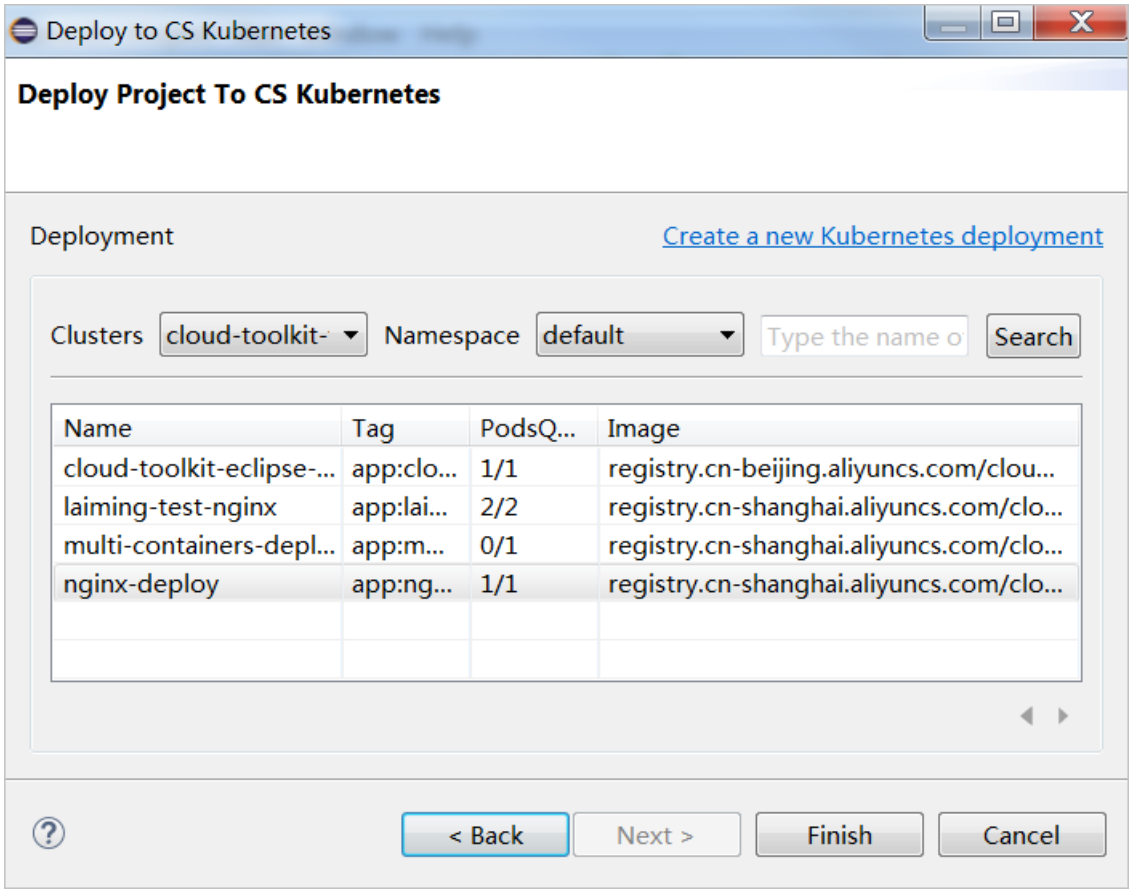
5. 在Select a Repository对话框选择容器镜像服务的地域，命令空间和镜像仓库，然后单击Next。

> 说明 如果您还没有镜像仓库，在对话框右上角单击Create a new repository跳转到容器镜像仓库创建镜像仓库。创建步骤可参见容器镜像服务文档。



6. 在**Deploy Project to CS Kubernetes**对话框选择容器服务Kubernetes的**Cluster**、**Namespace**和**Deployment**，然后单击**Finish**。

> **说明**
>
> - 如果您还没有创建容器服务Kubernetes的Deployment，在对话框右上角单击**Create a new Kubernetes deployment**，跳转到容器服务管理控制台创建Deployment。创建步骤可参见容器服务Kubernetes版文档。
> - Alibaba Cloud Toolkit目前只支持以更新方式部署您的应用，需要您预先在控制台手动创建一个部署（Deployment），然后基于该部署将镜像替换为您的镜像。



预期结果：

部署开始后，Eclipse的**Console**区域会打印部署日志，您可以根据日志信息检查部署结果。 更多信息请参见使用Eclipse部署应用到容器服务Kubernetes。

11.2. 快速搭建Jenkins环境并完成流水线作业

本文主要演示如何在阿里云Kubernetes集群上快速搭建Jenkins持续集成环境，并基于提供的示例快速完成应用源码编译、应用镜像构建和推送以及流水线的应用部署。

前提条件

- 已创建Kubernetes集群。具体操作，请参见创建Kubernetes托管版集群。
- 已通过kubectl连接到Kubernetes集群。具体操作，请参见通过kubectl工具连接集群。

部署Jenkins

建议先按照以下步骤安装部署ack-jenkins应用，成功运行构建任务示例**demo-pipeline**，再依照此构建任务示例改造您的构建任务配置。

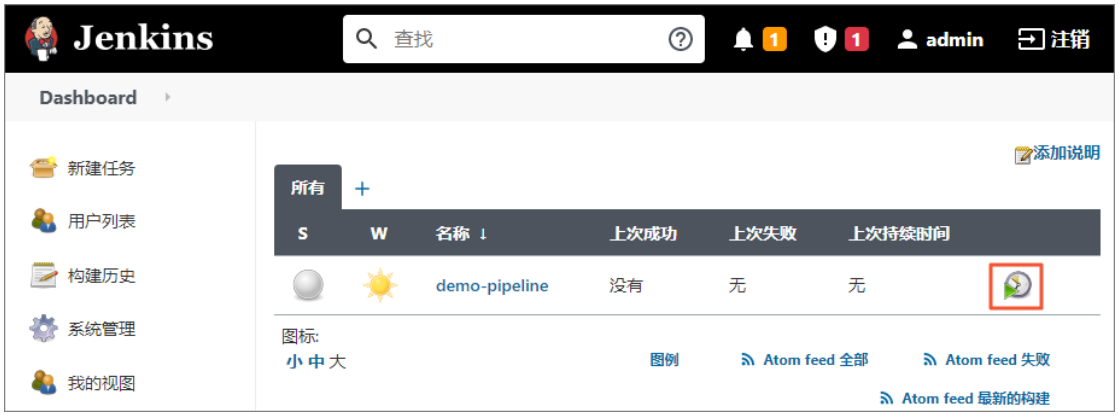
1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，选择**市场 > 应用市场**。
3. 在**应用市场**页面单击**应用目录**页签，然后搜索并选中**ack-jenkins**。
4. 在**ack-jenkins**页面，单击**一键部署**。
5. 在**创建**面板中，选择集群和命名空间，然后单击**下一步**。

> **说明** 建议选择自定义命名空间或default命名空间。本示例中选择自定义命名空间ci。

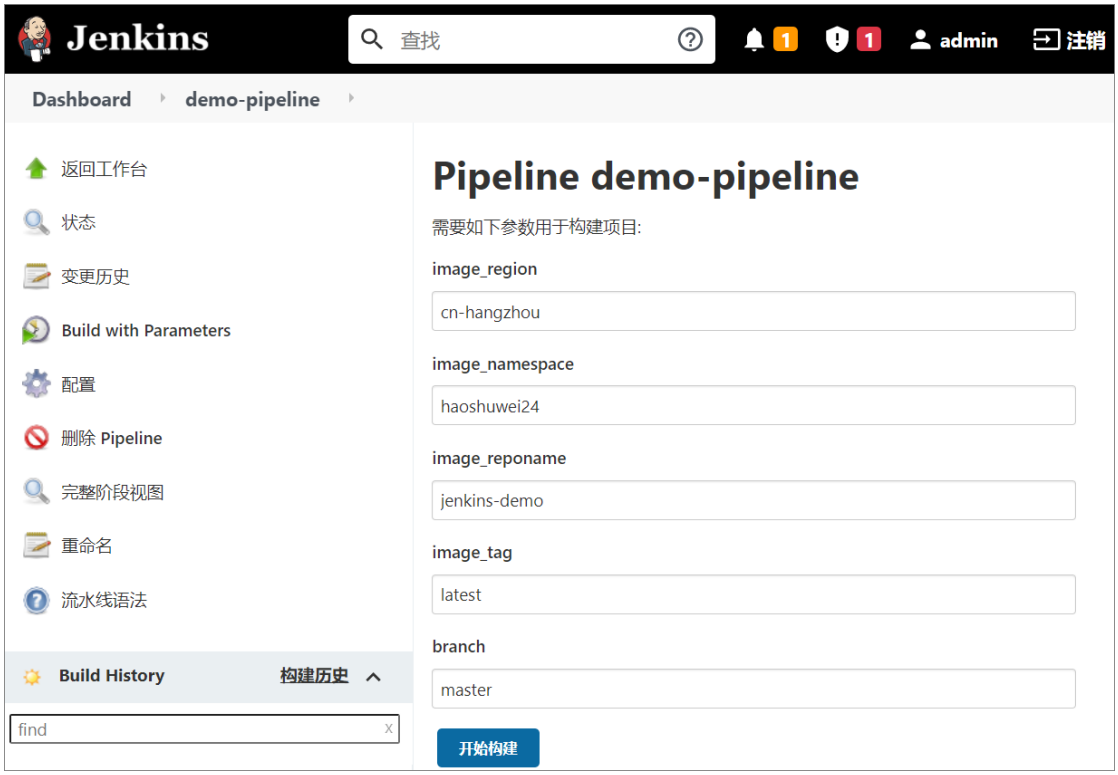
6. 在**参数配置**页面，修改`AdminPassword`字段，然后单击**确定**。

   如果您未设置登录密码，即未修改`AdminPassword`字段，Jenkins部署成功后系统会自动生成密码，默认密码为`admin`。

7. 访问并登录Jenkins。
   i. 在控制台左侧导航栏中，单击**集群**。
   ii. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
   iii. 在集群管理页左侧导航栏中，选择**网络 > 服务**。
   iv. 选择上个步骤设置的命名空间，然后单击**ack-jenkins-default**服务的**外部端点**，登录并访问Jenkins系统。
8. 配置Jenkins的**新手入门**。
   i. 在**新手入门**配置页面，单击**安装推荐的插件**。
   ii. 插件安装完成后，在**新手入门**的**实例配置**页面，单击**保存并完成**。
   iii. 实例配置保存完成后，单击**开始使用Jenkins**。
9. 构建**demo-pipeline**并访问应用服务。
   i. 在Jenkins首页，单击**demo-pipeline**的图标。



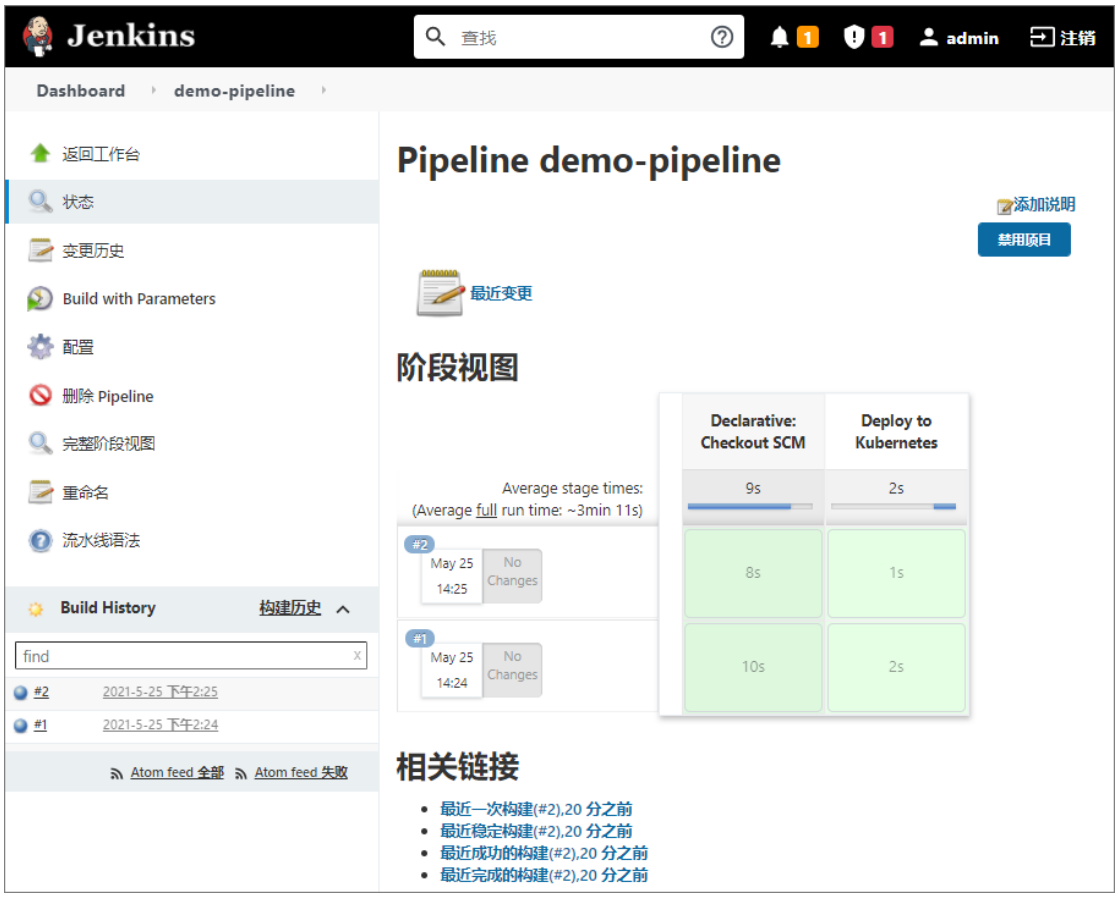
ii. 根据您的镜像仓库信息修改构建参数。本示例中源码仓库分支为 `master` ，镜像为 `registry.cn-beijing.aliyuncs.com/ack-cicd/ack-jenkins-demo:latest` 。



iii. 单击**开始构建**。

测试Kubernetes集群动态分配的Jenkins Slave Pod与Jenkins Master是否连接正常。
执行构建后，Jenkins从Kubernetes集群动态创建一个Slave Pod运行本次构建任务。关于示例应用代码，请参见jenkins-demo-GitHub或jenkins-demo。

iv. 查看**状态**，若构建成功则表示Jenkins on Kubernetes运行正常。

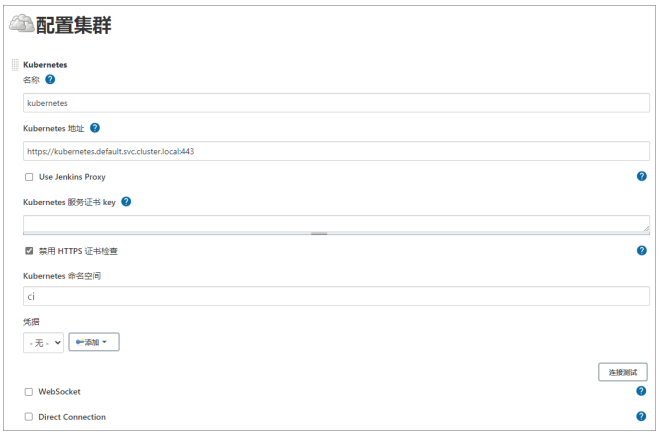


Kubernetes Cloud的配置说明

Jenkins使用Kubernetes Plugin连接Kubernetes集群并动态生成和释放Slave Pod。

1. 在左侧导航栏选择**返回工作台**。
2. 在Jenkins系统左侧导航栏选择**系统管理**。
3. 在**管理Jenkins**页面的**系统配置**下单击**节点管理**。
4. 在**节点列表**页面左侧导航栏选择**Configure Clouds**。
5. 在**配置集群**页面单击**Kubernetes Cloud details...**。
   在部署ack-jenkins时已自动化配置Kubernetes Cloud，以下为各个参数的具体说明：

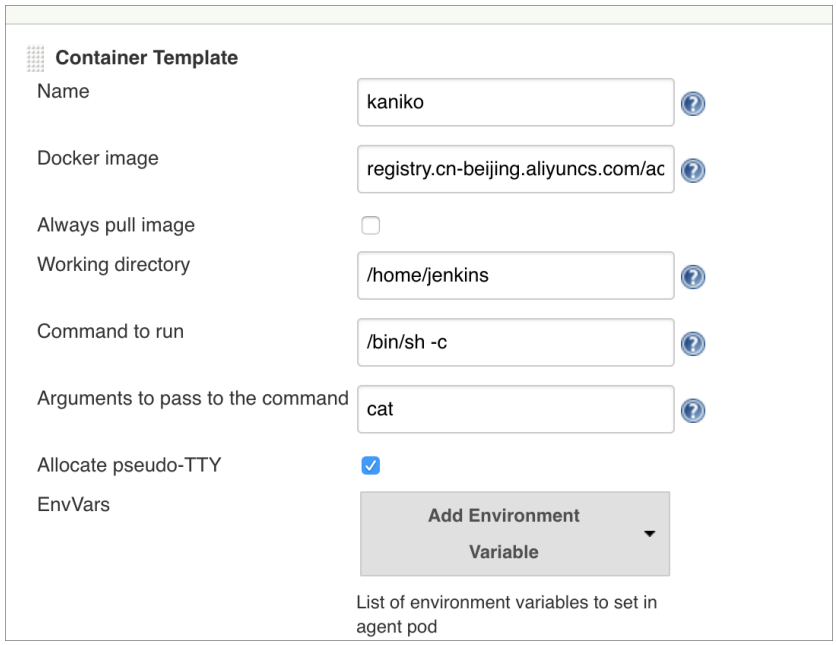
参数	说明

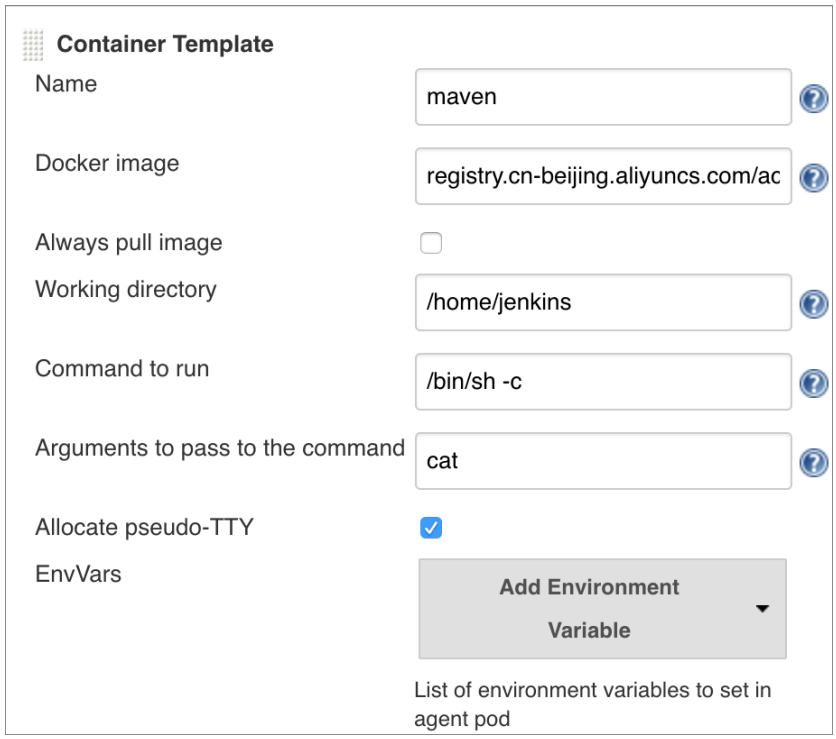
参数	说明
名称	默认为 `kubernetes` 。 
Kubernetes地址	默认为 `https://kubernetes.default.svc.cluster.local:443` ，Jenkins系统安装在当前集群中，可以使用内部服务端点访问集群的API Server。
Kubernetes命名空间	默认为 `default` ，动态Slave Pod会在命名空间 `default` 下生成和销毁。
Jenkins地址	默认为 `http://ack-jenkins-default:8080` ，Slave Pod连接Jenkins Master使用的服务端点。
Jenkins通道	默认为 `ack-jenkins-default-agent:50000` ，Slave Pod使用Jnlp连接Jenkins Master。

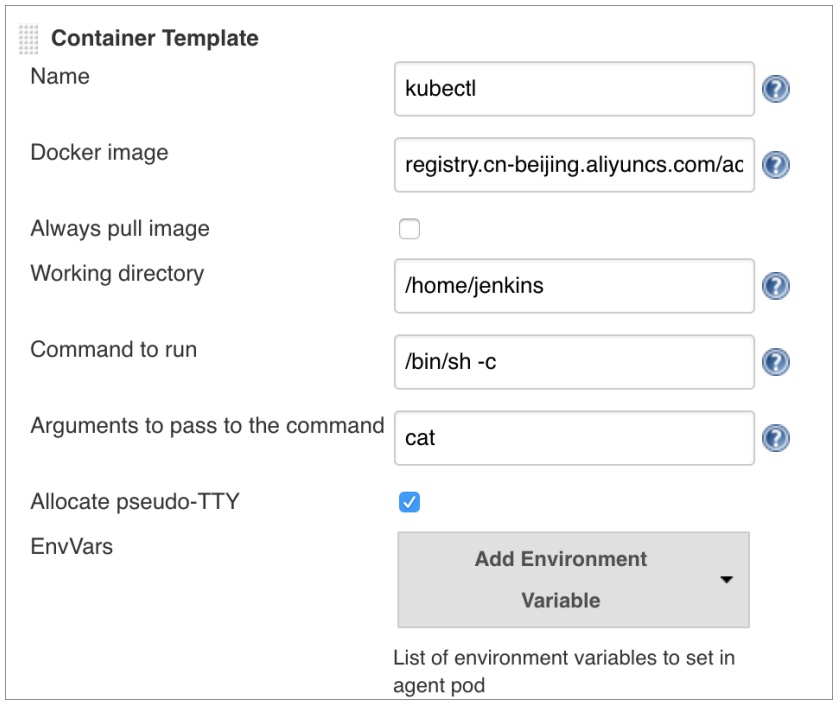
6. 在**配置集群**页面单击**Pod Templates...**，再单击**Pod Templates details...**。
在部署ack-jenkins时已自动化配置Pod Templates，以下为各个参数的具体说明：

参数	说明

参数	说明
名称	默认为 `slave-pipeline`，Slave Pod的名称。 
命名空间	默认为 `jenkins` 。
标签列表	默认为 `slave-pipeline` ，Jenkins构建任务通过此标签选择模板生成Slave Pod。
	`jnlp` 用于连接Jenkins Master。

参数	说明
	`kaniko` 用于构建和推送容器镜像。 

参数	说明
容器列表	`Maven` 用于构建和打包应用。 

参数	说明
	`kubectl` 用于部署应用。 

为Slave Pod配置Maven缓存

由于Slave Pod是在Kubernetes集群中动态生成的，可能运行于集群的任何一个Worker节点，所以要保证每次动态生成Slave Pod时都能使用到**Maven**缓存，就必须使用共享存储持久化存储卷。

1. 创建NAS共享存储持久化存储卷。具体操作，请参见使用NAS动态存储卷。
   执行以下命令查看在**jenkins**命名空间下创建的NAS持久化存储卷。

   ```
   kubectl -n jenkins get pvc
   ```

   预期输出：
   从预期输出可得：存储卷 `ack-jenkins-default` 是Jenkins Master的*/var/jenkins_home*目录的持久化存储， `nas-csi-pvc` 则是为配置Maven缓存所共享的NAS持久化存储卷。

   ```
   NAME                  STATUS   VOLUME                                     CAPACITY   AC
   CESS MODES   STORAGECLASS              AGE
   ack-jenkins-default   Bound    d-2ze8xhgzuw1t278j****                     20Gi       RW
   O            alicloud-disk-efficiency   3h16m
   nas-csi-pvc           Bound    nas-c6c3e703-58a9-484e-8ce5-630486ea****   20Gi       RW
   X            alicloud-nas-subpath       4m17s
   ```

2. （可选）在Jenkins系统的Pod Template中添加Config Map Volume类型挂载。

   若需要为Jenkins Slave Pod挂载自定义Settings文件，则可以先创建Config Map Volume，再配置到Pod Template上。

i. 执行以下命令创建ConfigMap。

若需要更改settings.xml，请先创建ConfigMap。

```
kubectl -n jenkins create configmap maven-config --from-file=settings.xml
```

ii. 在Jenkins系统左侧导航栏选择**系统管理**，在**管理Jenkins**页面的**系统配置**下单击**节点管理**，在**节点列表**页面左侧导航栏选择**Configure Clouds**。

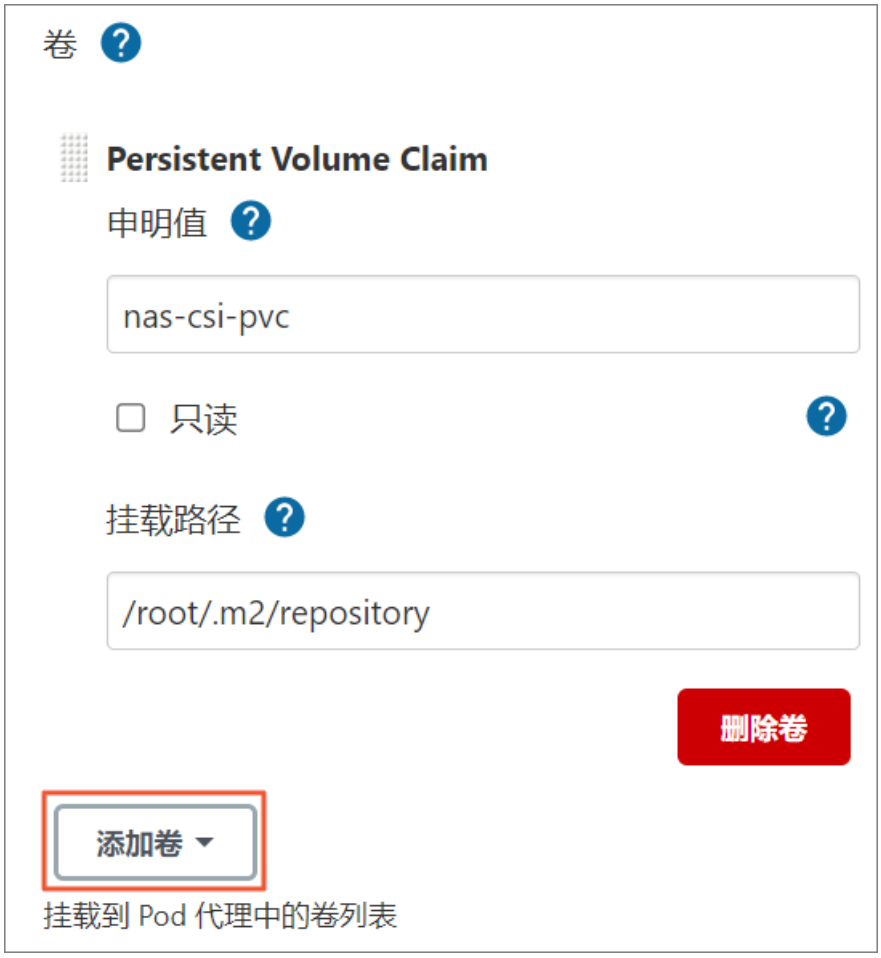
iii. 在**配置集群**页面单击**Pod Template**，再单击**Pod Template details**。

iv. 在**卷**下单击**添加卷**，选择**Config Map Volume**类型卷。



v. 单击**Save**，保存配置。

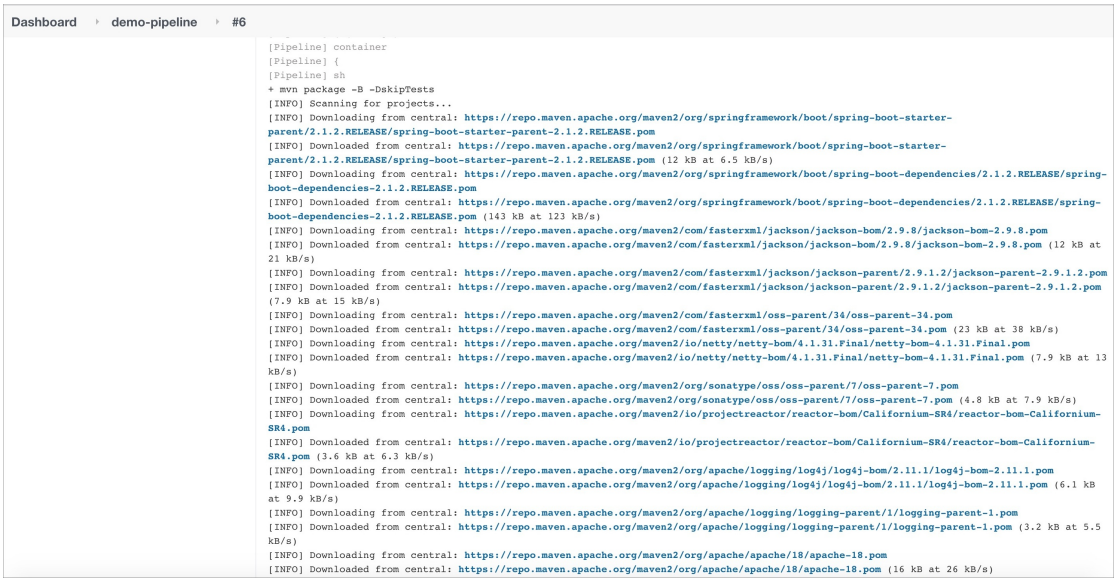
3. 在Jenkins系统的Pod Template中添加Persistent Volume Claim类型挂载卷。具体操作，请参见步骤2。



结果验证

i. 在Jenkins首页，单击**demo-pipeline**名称。
ii. 在Jenkins系统控制台的左侧导航栏选择**配置**。
iii. 单击**流水线**页签。
iv. 在**脚本路径**输入*Jenkinsfile.maven*。
v. 单击**保存**。
vi. 在Jenkins首页，单击**demo-pipeline**的图标，单击**开始构建**。

首次进行Maven构建时，需要花费一段时间下载所有依赖包。



再次进行Maven构建，则会引用缓存的依赖包，快速完成源码打包。



使用kaniko构建和推送容器镜像

使用kaniko推送镜像时，需要设置镜像仓库的访问权限。

在本示例中，需要在Linux环境下（请注意不要在macOS下）生成访问镜像仓库的*config.json*文件。例如，需要构建和推送一个镜像 `registry.cn-hangzhou.aliyuncs.com/haoshuwei24/jenkins-demo:20200428` 。

1. 执行以下命令登录镜像仓库。

   登录镜像仓库的同时会生成*config.json*文件。

   ```
   docker login -u <username> -p <password> registry.cn-hangzhou.aliyuncs.com
   ```

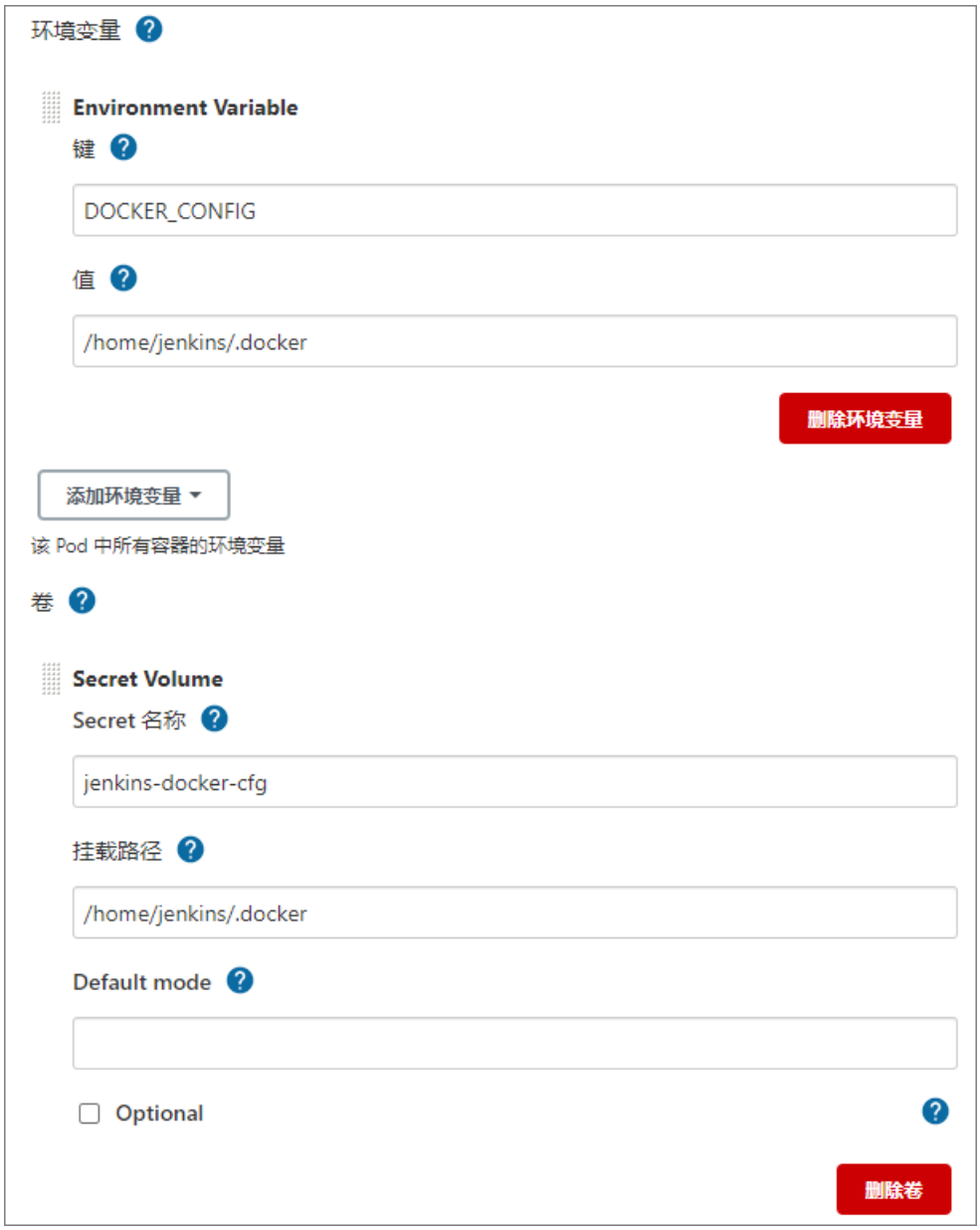
预期输出：

```
Login Succeeded
```

2. 在**jenkins**命名空间下使用生成的*config.json*文件创建名为*jenkins-docker-cfg*的Secret。

```
kubectl create secret generic jenkins-docker-cfg -n jenkins --from-file=/root/.docker/config.json
```

3. 在Jenkins系统的Pod Template中配置挂载卷及环境变量。
   i. 在Jenkins系统左侧导航栏选择**系统管理**，在**管理Jenkins**页面的**系统配置**下单击**节点管理**，在**节点列表**页面左侧导航栏选择**Configure Clouds**。
   ii. 在**配置集群**页面单击**Pod Template**，再单击**Pod Template details**。
   iii. 在**环境变量**下单击**添加环境变量**，选择**Environment Variable**类型卷。



   iv. 在**卷**下单击**添加卷**，选择**Secret Volume**类型卷。

v. 单击**Save**，保存配置。

结果验证

i. 在Jenkins首页，单击**demo-pipeline**名称。

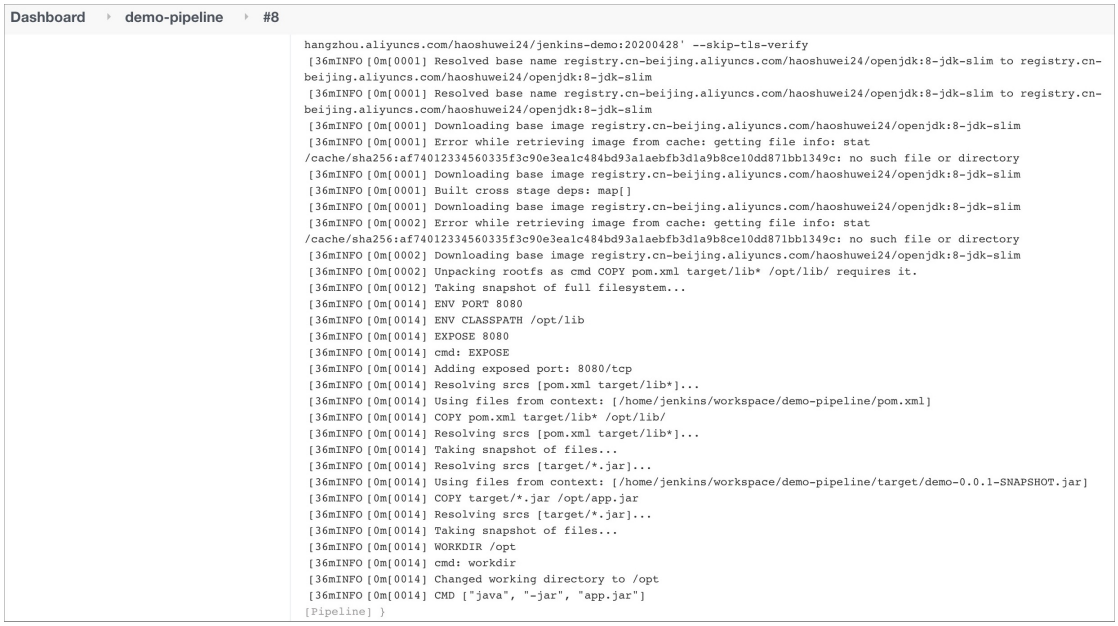
ii. 在Jenkins系统控制台的左侧导航栏选择**配置**。

iii. 单击**流水线**页签。

iv. 在**脚本路径**输入*Jenkinsfile.kaniko*。

v. 单击**保存**。

vi. 在Jenkins首页，单击**demo-pipeline**的图标，请将构建参数修改为您实际的镜像仓库相关信息，然后单击**开始构建**。
查看**kaniko**构建日志。



vii. 查看容器镜像仓库是否已经推送成功。

a. 登录容器镜像服务控制台。

b. 在左侧导航栏，选择**实例列表**。

c. 在个人版实例管理页面选择**仓库管理 > 镜像仓库**。

d. 在**镜像仓库**页面，单击目标仓库右侧操作列的**管理**。

e. 在左侧导航栏，选择**镜像版本**，查看镜像是否已推送成功。



升级Jenkins

若您的Jenkins环境需要更换jenkins-master镜像，由于新版本Jenkins Master的自动化脚本会向*/var/jenkins_home*路径写数据，为了防止覆盖已有数据，请先为*/var/jenkins_home*目录对应的云盘存储卷做快照备份。具体操作，请参见使用云盘存储快照。

相关文档

- 源码仓库
- 容器服务ACK
- kaniko

11.3. 使用GitLab CI运行GitLab Runner并执行Pipeline

本文主要演示如何在Kubernetes集群中安装、注册GitLab Runner，添加Kubernetes类型的Executor来执行构建，并以此为基础完成一个Java源码示例项目从编译构建、镜像打包到应用部署的CICD过程。

背景信息

本文以构建一个Java软件项目并将其部署到阿里云容器服务Kubernetes集群中为例，说明如何使用GitLab CI在阿里云Kubernetes服务上运行GitLab Runner、配置Kubernetes类型的Executor，并执行Pipeline。

创建GitLab源码项目并上传示例代码

1. 创建GitLab源码项目。
   本示例中创建的GitLab源码项目地址为：

   ```
   http://xx.xx.xx.xx/demo/gitlab-java-demo.git
   ```

2. 执行以下命令获取示例代码并上传至GitLab。

   ```
   git clone https://github.com/haoshuwei/gitlab-ci-k8s-demo.git
   git remote add gitlab http://xx.xx.xx.xx/demo/gitlab-java-demo.git
   git push gitlab master
   ```

在Kubernetes集群中安装GitLab Runner

1. 获取GitLab Runner的注册信息。

i. 获取项目专用Runner的注册信息。

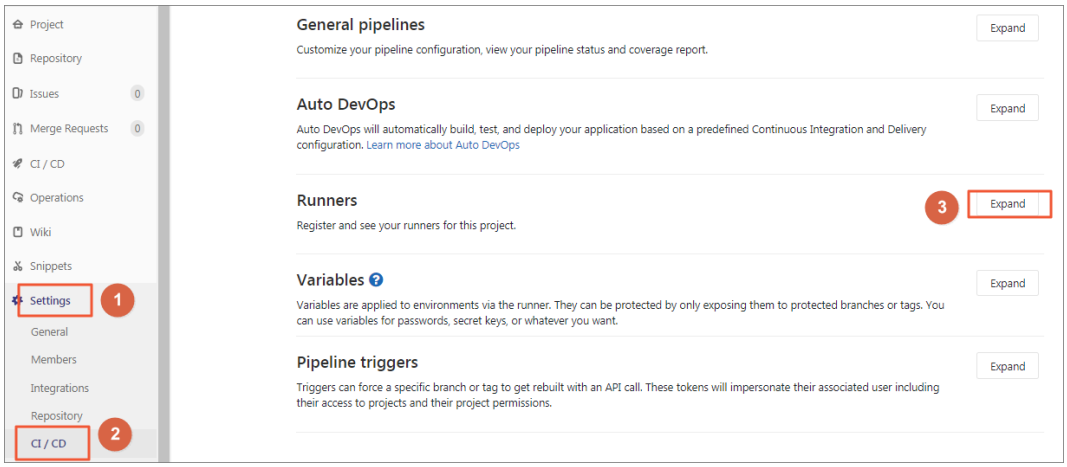
a. 登录GitLab。

b. 在顶部导航栏中，选择**Projects > Your projects**。

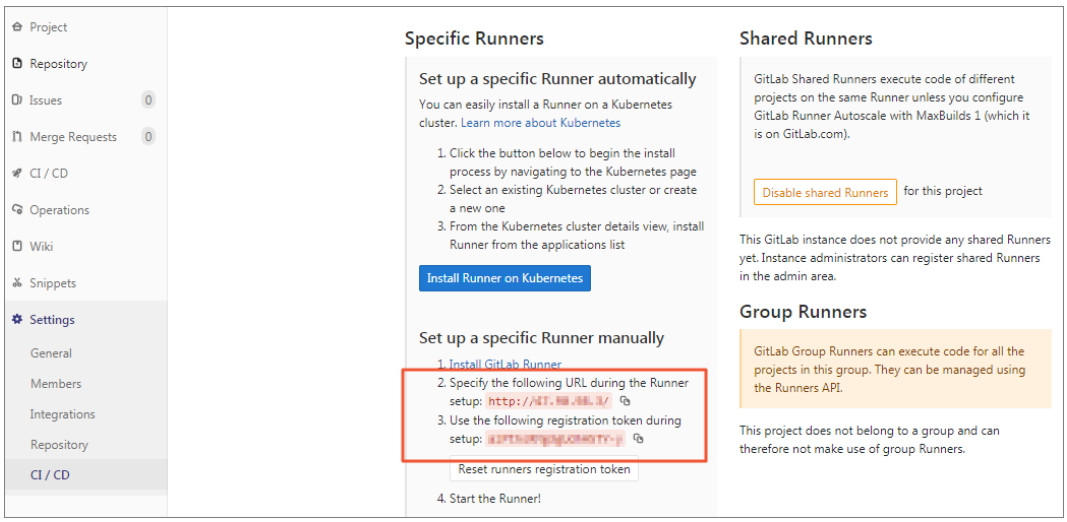
c. 在**Your projects**页签下，选择相应的Project。

d. 在左侧导航栏中，选择**Settings > CI / CD**。

e. 单击**Runners**右侧的**Expand**。



f. 获取URL和registration token信息。



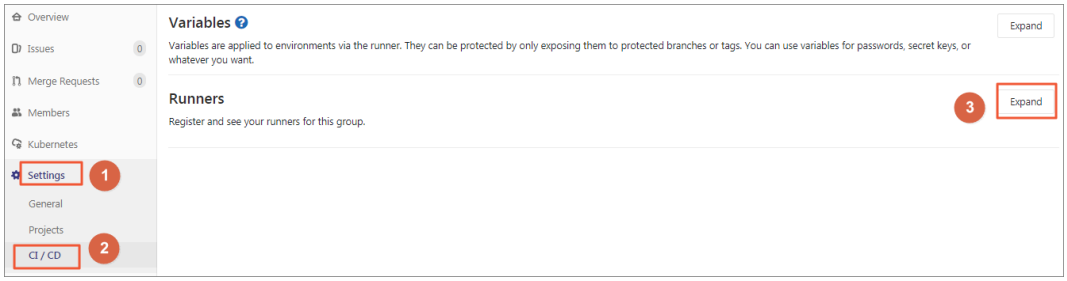
ii. 获取Group Runners的注册信息。

a. 在顶部导航栏中，选择**Groups > Your groups**。

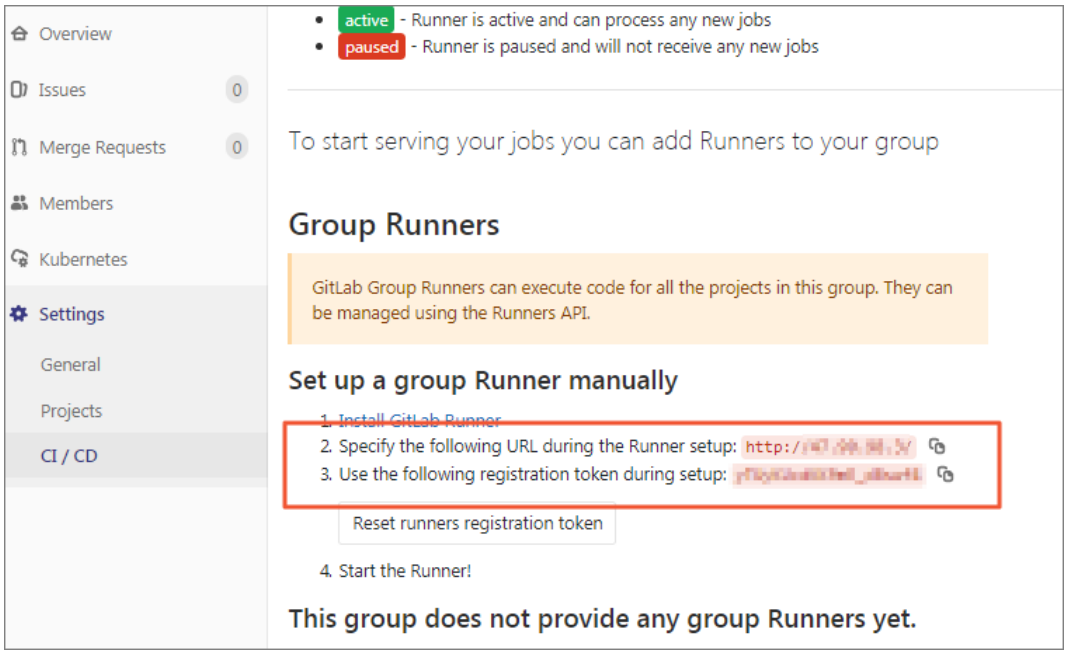
b. 在**Your groups**页签下，选择相应的group。

c. 在左侧导航栏中，选择**Settings > CI / CD**。

d. 单击**Runners**右侧的**Expand**。



e. 获取URL和registration token信息。



iii. 获取Shared Runners的注册信息。

> **说明** 只有管理员有权限执行此步操作。

a. 在顶部导航栏中，单击



进入**Admin Area**页面。

b. 在左侧导航栏中，选择**Overview > Runners**。

c. 获取URL和registration token信息。



2. 执行以下命令获取并修改GitLab Runner的Helm Chart。

```
git clone https://github.com/haoshuwei/gitlab-runner.git
```

替换gitlabUrl和runnerRegistrationToken字段，示例如下：

```
## GitLab Runner Image
##
image: gitlab/gitlab-runner:alpine-v11.4.0
## Specify a imagePullPolicy
##
imagePullPolicy: IfNotPresent
## Default container image to use for initcontainer
init:
  image: busybox
  tag: latest
## The GitLab Server URL (with protocol) that want to register the runner against
##
gitlabUrl: http://xx.xx.xx.xx/
## The Registration Token for adding new Runners to the GitLab Server. This must
## be retreived from your GitLab Instance.
##
runnerRegistrationToken: "AMvEWrBTBu-d8czE****"
## Unregister all runners before termination
##
unregisterRunners: true
## Configure the maximum number of concurrent jobs
##
concurrent: 10
## Defines in seconds how often to check GitLab for a new builds
##
checkInterval: 30
## For RBAC support:
##
rbac:
  create: true
```

```
  create: true
  clusterWideAccess: false
## Configure integrated Prometheus metrics exporter
##
metrics:
  enabled: true
## Configuration for the Pods that that the runner launches for each new job
##
runners:
  ## Default container image to use for builds when none is specified
  ##
  image: ubuntu:16.04
  ## Specify the tags associated with the runner. Comma-separated list of tags.
  ##
  tags: "k8s-runner"
  ## Run all containers with the privileged flag enabled
  ## This will allow the docker:dind image to run if you need to run Docker
  ## commands. Please read the docs before turning this on:
  ##
  privileged: true
  ## Namespace to run Kubernetes jobs in (defaults to the same namespace of this releas
e)
  ##
  namespace: gitlab
  cachePath: "/opt/cache"
  cache: {}
  builds: {}
  services: {}
  helpers: {}
resources: {}
```

3. 执行以下命令安装GitLab Runner。
   - 打包应用

     ```
     helm package .
     ```

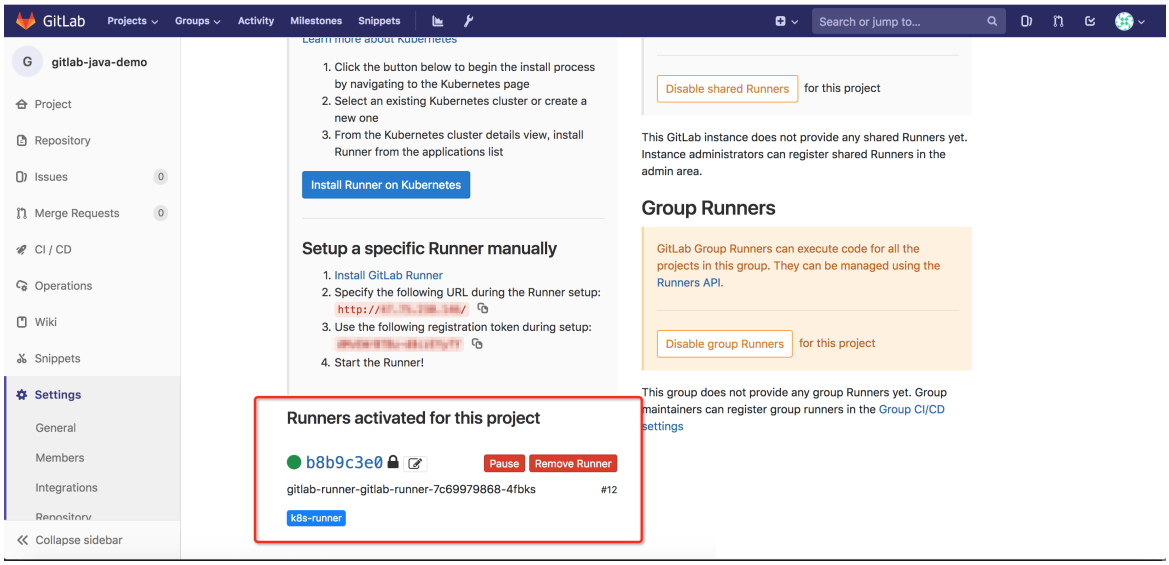
     预期输出：

     ```
     Successfully packaged chart and saved it to: /root/gitlab/gitlab-runner/gitlab-runner
     -0.1.37.tgz
     ```

   - 安装应用

     ```
     helm install --namespace gitlab gitlab-runner *.tgz
     ```

查看相关的deployment/pod是否成功启动。若成功启动，则可在GitLab上看到注册成功的GitLab Runner，如下图所示。



缓存配置

GitLab Runner对缓存方案的支持有限，所以您需要使用挂载Volume的方式做缓存。在上面的示例中，安装GitLab Runner时默认使用*/opt/cache*目录作为缓存空间。您也可以通过修改*values.yaml*文件中的`runners.cachePath`字段修改缓存目录。

例如，如需建立Maven缓存，您可以在`variables`下添加`MAVEN_OPTS`变量并指定本地缓存目录：

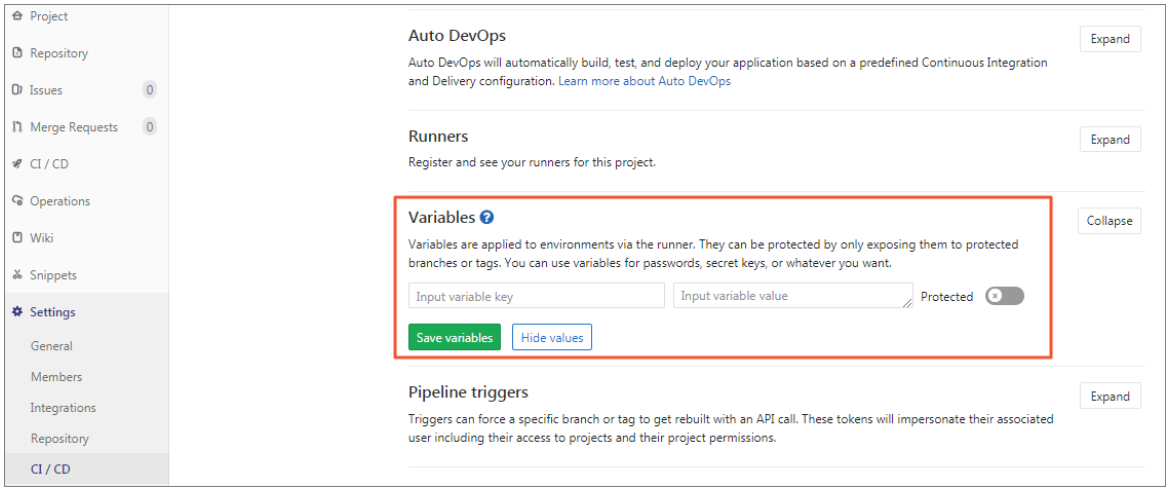
```
variables:
  KUBECONFIG: /etc/deploy/config
  MAVEN_OPTS: "-Dmaven.repo.local=/opt/cache/.m2/repository"
```

修改*templates/configmap.yaml*文件中的以下字段挂载docker.sock和用于cache的volume。

```
cat >>/home/gitlab-runner/.gitlab-runner/config.toml <<EOF
            [[runners.kubernetes.volumes.pvc]]
              name = "gitlab-runner-cache"
              mount_path = "{{ .Values.runners.cachePath }}"
            [[runners.kubernetes.volumes.host_path]]
              name = "docker"
              mount_path = "/var/run/docker.sock"
              read_only = true
              host_path = "/var/run/docker.sock"
    EOF
```

设置全局变量

1. 在GitLab的顶部导航栏中，选择**Projects > Your projects**。
2. 在**Your projects**页签下，选择相应的Project。
3. 在左侧导航栏中，选择**Settings > CI / CD**。
4. 单击**Variables**右侧的**Expand**。添加GitLab Runner可用的环境变量。本示例中，添加以下三个变量。



- REGISTRY_USERNAME：镜像仓库用户名。
- REGISTRY_PASSWORD：镜像仓库密码。
- kube_config：KubeConfig的编码字符串。

执行以下命令生成KubeConfig的编码字符串：

```
echo $(cat ~/.kube/config | base64) | tr -d " "
```

编写.gitlab-ci.yml

编写*.gitlab-ci.yml*文件，完成Java Demo源码项目的编译构建、镜像推送和应用部署（可参考gitlabci-java-demo源码项目中的.gitlab-ci.yml.example）。

*.gitlab-ci.yml*示例如下。

```
image: docker:stable
stages:
  - package
  - docker_build
  - deploy_k8s
variables:
  KUBECONFIG: /etc/deploy/config
  MAVEN_OPTS: "-Dmaven.repo.local=/opt/cache/.m2/repository"
mvn_build_job:
  image: maven:3.6.2-jdk-14
  stage: package
  tags:
    - k8s-runner
  script:
    - mvn package -B -DskipTests
    - cp target/demo.war /opt/cache
docker_build_job:
  image: docker:latest
  stage: docker_build
  tags:
    - k8s-runner
  script:
    - docker login -u $REGISTRY_USERNAME -p $REGISTRY_PASSWORD registry.cn-beijing.aliyuncs
.com
    - mkdir target
    - cp /opt/cache/demo.war target/demo.war
    - docker build -t registry.cn-beijing.aliyuncs.com/haoshuwei24/gitlabci-java-demo:$CI_P
IPELINE_ID .
    - docker push registry.cn-beijing.aliyuncs.com/haoshuwei24/gitlabci-java-demo:$CI_PIPEL
INE_ID
deploy_k8s_job:
  image: registry.cn-hangzhou.aliyuncs.com/haoshuwei24/kubectl:1.16.6
  stage: deploy_k8s
  tags:
    - k8s-runner
  script:
    - mkdir -p /etc/deploy
    - echo $kube_config |base64 -d > $KUBECONFIG
    - sed -i "s/IMAGE_TAG/$CI_PIPELINE_ID/g" deployment.yaml
    - cat deployment.yaml
    - kubectl apply -f deployment.yaml
```

*.gitlab-ci.yml*定义了一个Pipeline， 分三个阶段步骤执行：

```
image: docker:stable  # Pipeline中各个步骤阶段的构建镜像没有指定时， 默认使用docker:stable镜像
stages:
  - package               # 源码打包阶段
  - docker_build         # 镜像构建和打包推送阶段
  - deploy_k8s           # 应用部署阶段
variables:
  KUBECONFIG: /etc/deploy/config   # 定义全局变量KUBECONFIG
```

- maven源码打包阶段。

```
mvn_build_job:     # job名称
  image: maven:3.6.2-jdk-14  # 本阶段构建使用的构建镜像
  stage: package      # 关联的阶段名称
  tags:                     # GitLab Runner的tag
    - k8s-runner
  script:
    - mvn package -B -DskipTests  # 执行构建脚本
    - cp target/demo.war /opt/cache    # 构建物保存至缓存区
```

- 镜像构建和打包推送阶段。

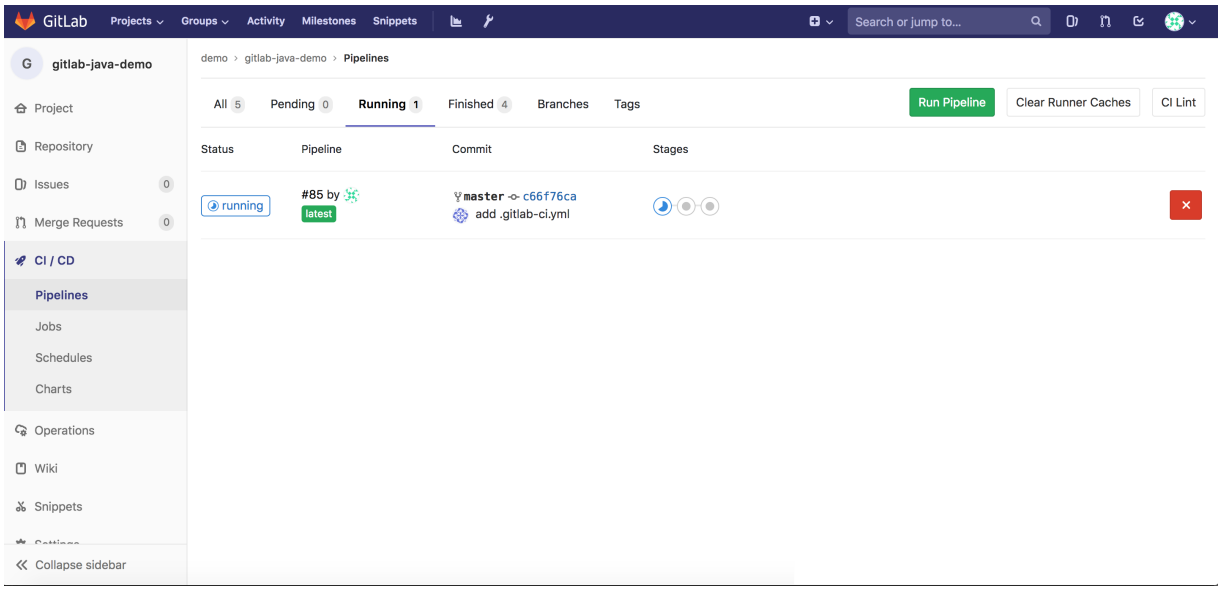
```
docker_build_job:  # job名称
  image: docker:latest # 本阶段构建使用的构建镜像
  stage: docker_build  # 关联的阶段名称
  tags:                      # GitLab Runner的tag
    - k8s-runner
  script:
    - docker login -u $REGISTRY_USERNAME -p $REGISTRY_PASSWORD registry.cn-beijing.aliyun
cs.com   # 登录镜像仓库
    - mkdir target
    - cp /opt/cache/demo.war target/demo.war
    - docker build -t registry.cn-beijing.aliyuncs.com/haoshuwei24/gitlabci-java-demo:$CI
_PIPELINE_ID .     # 打包Docker镜像，使用的tag为本次Pipeline的ID
    - docker push registry.cn-beijing.aliyuncs.com/haoshuwei24/gitlabci-java-demo:$CI_PIP
ELINE_ID      # 推送Docker镜像
```

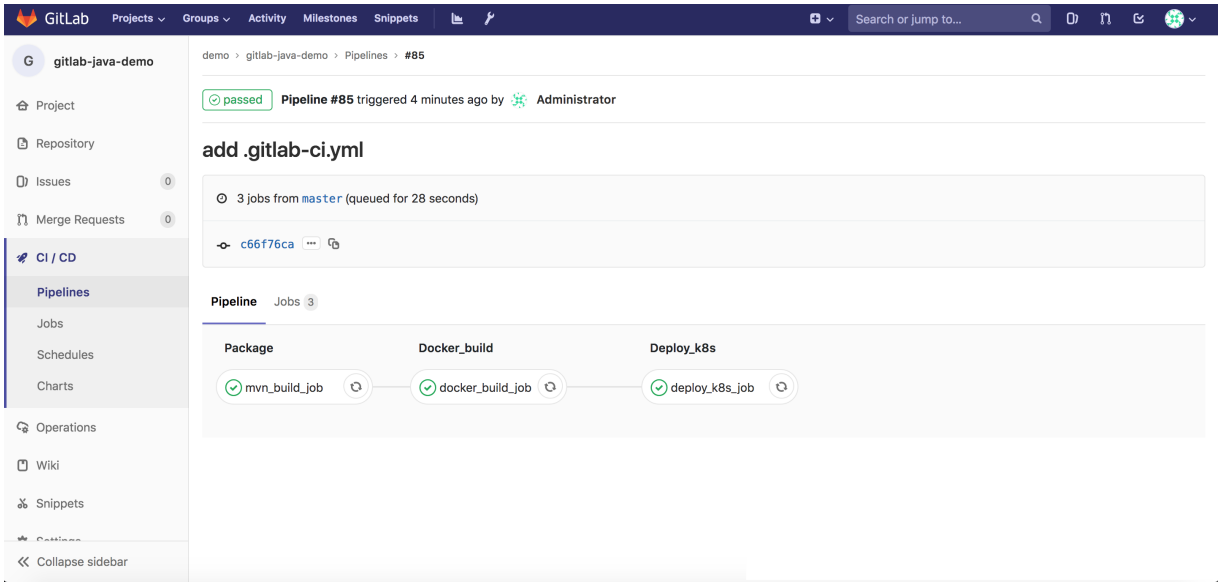
- 应用部署阶段。

```
deploy_k8s_job:   # job名称
  image: registry.cn-hangzhou.aliyuncs.com/haoshuwei24/kubectl:1.16.6   # 本阶段构建使用的
构建镜像
  stage: deploy_k8s   # 关联的阶段名称
  tags:                      # GitLab Runner的tag
    - k8s-runner
  script:
    - mkdir -p /etc/deploy
    - echo $kube_config |base64 -d > $KUBECONFIG   # 配置连接Kubernetes集群的config文件
    - sed -i "s/IMAGE_TAG/$CI_PIPELINE_ID/g" deployment.yaml  # 动态替换部署文件中的镜像tag
    - cat deployment.yaml
    - kubectl apply -f deployment.yaml
```

执行Pipeline

提交.gitlab-ci.yml文件后，Project gitlab-java-demo会自动检测到这个文件并执行Pipeline， 如下图所示。





访问服务

如果部署文件中没有指定Namespace，则默认会部署到GitLab命名空间下：

```
kubectl -n gitlab get svc
```

预期输出：

```
NAME         TYPE           CLUSTER-IP     EXTERNAL-IP      PORT(S)          AGE
java-demo    LoadBalancer   172.19.9.252   xx.xx.xx.xx   80:32349/TCP   1m
```

浏览器访问xx.xx.xx.xx/demo进行验证。

了解更多容器服务的相关内容，请参见容器服务。

了解更多GitLab CI的相关内容，请参见GitLab CI。

11.4. 容器应用DevOps for ACK集群

本文介绍容器应用DevOps for ACK集群的场景描述、解决问题、架构图及操作参考链接。

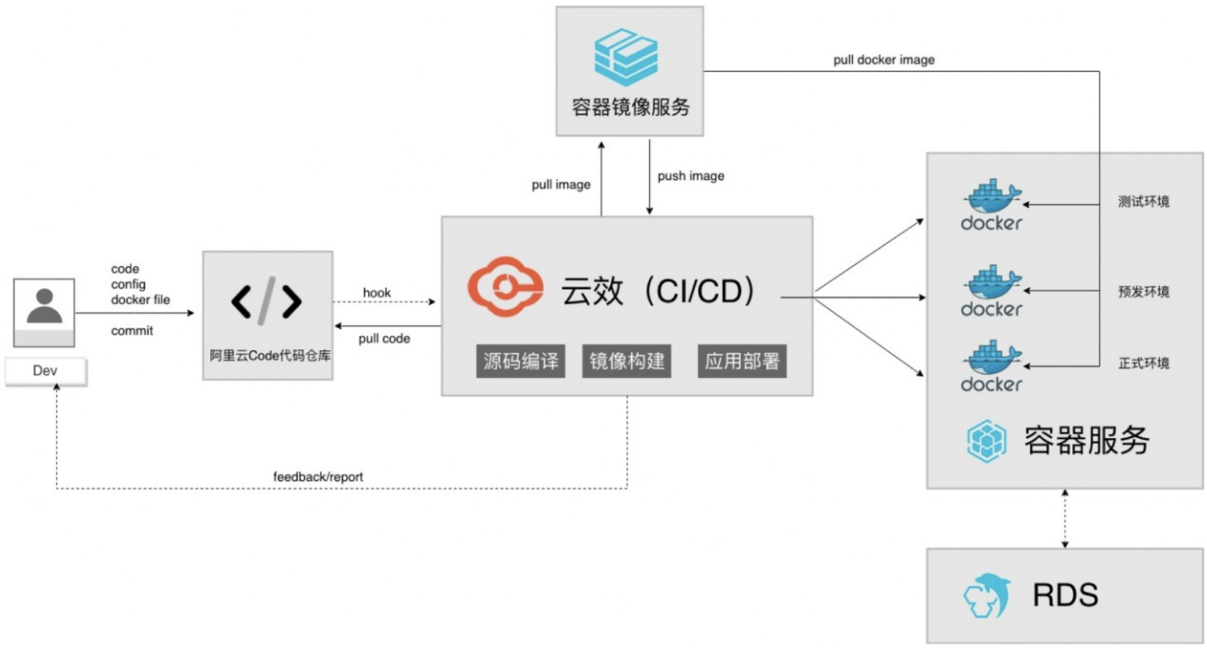
场景描述

DevOps的目的是构建一种文化和环境，使构建、测试、发布软件更加快捷、频繁和可靠 。 而到了容器时代，需要部署的机器不但量更大，变化更剧烈，有的甚至需要根据条件自动升缩，为了满足企业敏捷的需求，持续部署也成了必需。本方案使用云效完成容器应用（小程序后端服务）的自动化构建和持续部署。

解决问题

- 微服务应用的CI/CD（持续集成和持续交付）。
- 线上应用蓝绿部署升级。
- 自动化测试与部署。

架构图



参考链接

有关容器应用DevOps for ACK集群的详情，请参见容器应用DevOps for ACK集群最佳实践。

11.5. ASK弹性低成本CI/CD

本文介绍轻量化阿里云容器服务Kubernetes（ASK）弹性低成本的CI/CD（持续集成和持续交付）的场景描述、方案优势、解决问题、架构图及操作参考链接。

场景描述

基于阿里云弹性容器实例ECI的轻量化阿里云容器服务Kubernetes（ASK）以及文件存储NAS可以帮助用户实现服务高可用、弹性伸缩、资源扩展性好、低成本的自动化CI/CD系统。

方案优势

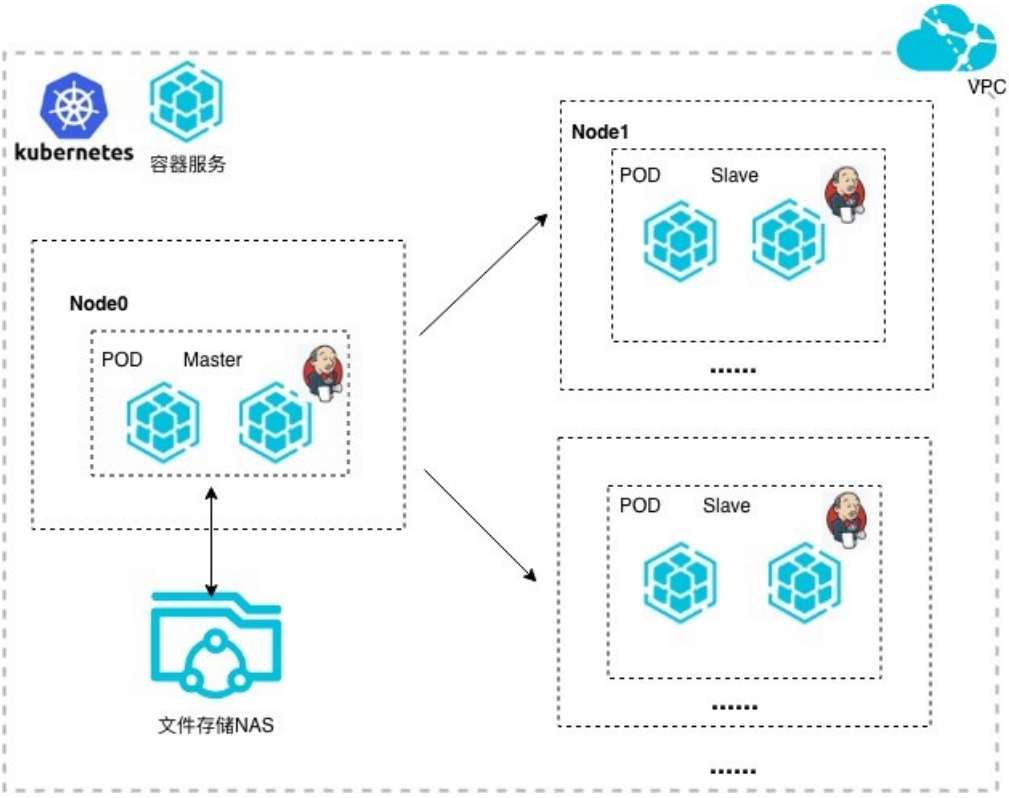
- 高可用服务。

- 资源弹性伸缩、资源合理利用。
- 资源可扩展性好。

解决问题

- 集群Master节点单点故障。
- 集群资源利用率低、资源有浪费。
- 集群资源可扩展性差。

架构图



参考链接

有关轻量化阿里云容器服务Kubernetes弹性低成本的CI/CD的详情，请参见Serverless ACK弹性低成本CI/CD最佳实践。

11.6. 使用Bamboo部署Bamboo Agent并执行构建任务

本文档以构建一个Java软件项目并部署到阿里云容器服务的Kubernetes集群为例，为您介绍如何使用Bamboo在阿里云Kubernetes服务上运行Remote Agents并在agents上运行Build Plans。

前提条件

- 您已经成功创建一个Kubernetes集群，请参见创建Kubernetes托管版集群。
- 您已经创建好一个Bamboo Server。

源码项目

本示例中创建的GitHub源码项目地址为如下：

```
https://github.com/AliyunContainerService/jenkins-demo.git
```

分支为 `bamboo` 。

步骤一、在Kubernetes中部署Remote Agent

1. 创建kaniko-docker-cfg secret。

   在Remote Agent上运行构建任务并且使用kaniko推送容器镜像时，kaniko-docker-cfg secret用于配置镜像仓库访问权限。

   i. 在Linux服务器上使用**root**用户执行如下命令，生成*/root/.docker/config.json*。

   ```
   docker login registry.cn-hangzhou.aliyuncs.com
   ```

   ii. 通过CloudShell连接Kubernetes集群，并执行以下命令创建kaniko-docker-cfg secret。

   ```
   kubectl -n bamboo create secret generic kaniko-docker-cfg  --from-file=/root/.docke
   r/config.json
   ```

2. 创建Bamboo Agent应用。

   创建serviceaccount bamboo以及clusterrolebinding，用于kubectl部署应用到kubernetes集群的权限设置。

   i. 创建并复制以下内容到*bamboo-agent.yaml*，并执行 `kubectl -n bamboo apply -f bamboo-agen t.yaml` 命令，创建Bamboo Agent 应用。

```
apiVersion: v1
kind: ServiceAccount
metadata:
  namespace: bamboo
  name: bamboo
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: bamboo-cluster-admin
subjects:
  - kind: ServiceAccount
    name: bamboo
    namespace: bamboo
roleRef:
  kind: ClusterRole
  name: cluster-admin
  apiGroup: rbac.authorization.k8s.io
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: bamboo-agent
spec:
  replicas: 1
  selector:
    matchLabels:
      app: bamboo-agent
  template:
    metadata:
      labels:
        app: bamboo-agent
    spec:
      serviceAccountName: bamboo
      containers:
      - name: bamboo-agent
        env:
        - name: BAMBOO_SERVER_URL
          value: http://xx.xx.xx.xx:8085
        image: registry.cn-hangzhou.aliyuncs.com/haoshuwei/docker-bamboo-agent:v1
        imagePullPolicy: Always
        volumeMounts:
          - mountPath: /root/.docker/
            name: kaniko-docker-cfg
      volumes:
        - name: kaniko-docker-cfg
          secret:
            secretName: kaniko-docker-cfg
```

ii. 完成后，执行以下命令查看日志。

```
kubectl -n bamboo logs -f <bamboo agent pod name>
```

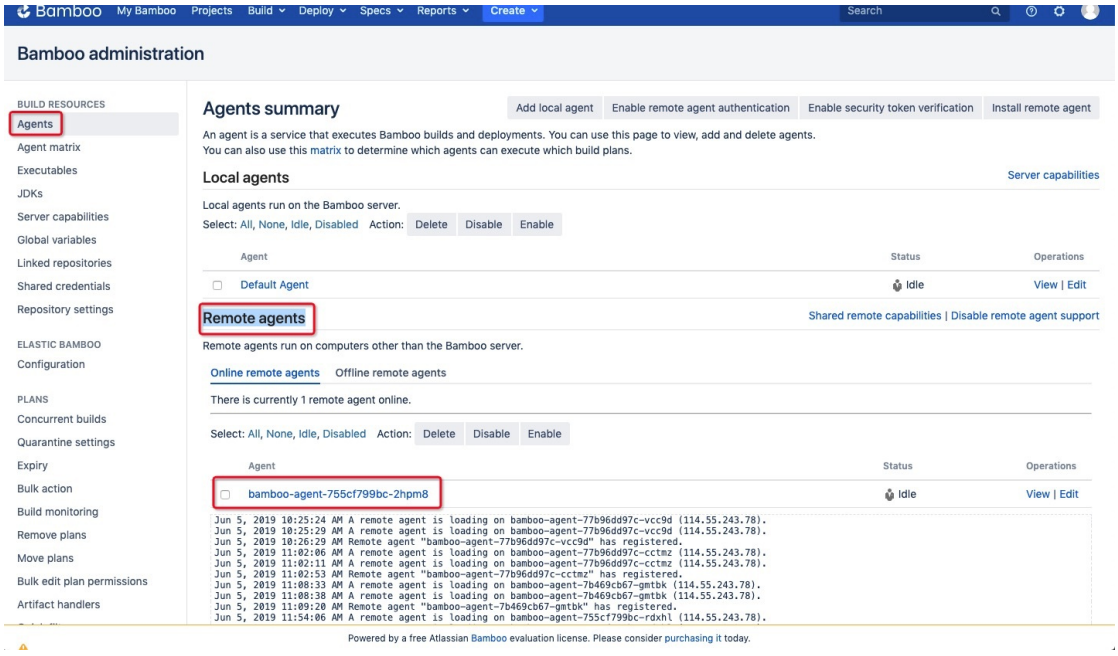
> **说明** 示例中*bamboo agent pod name*需要替换成实际的文件名。

iii. Remote Agent注册成功后，登录Bamboo。单击右上角



，在下拉框中选择**Agent**。

在**Bamboo administration**页面，可以看到部署好的Agent。



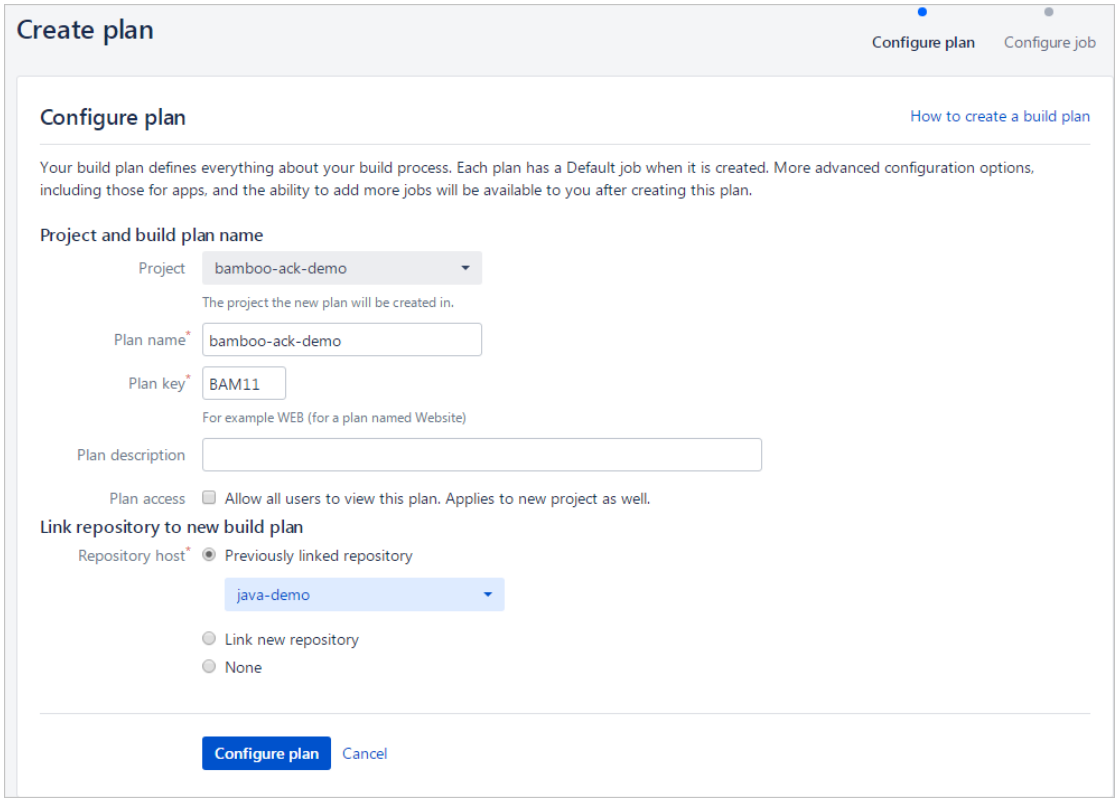
步骤二、配置一个Build plan

完成应用源码拉取、编译打包、容器镜像打包和推送、应用部署的过程。

1. 创建一个Build plan。

   i. 登录Bamboo，选择**Create > Create plan**，进入**Configure plan**页面。

ii. **Project**选择*bamboo-ack-demo*，填写Plan name、Plan key和Plan description，Repository host选择*java-demo*，完成后单击**Configure plan**。

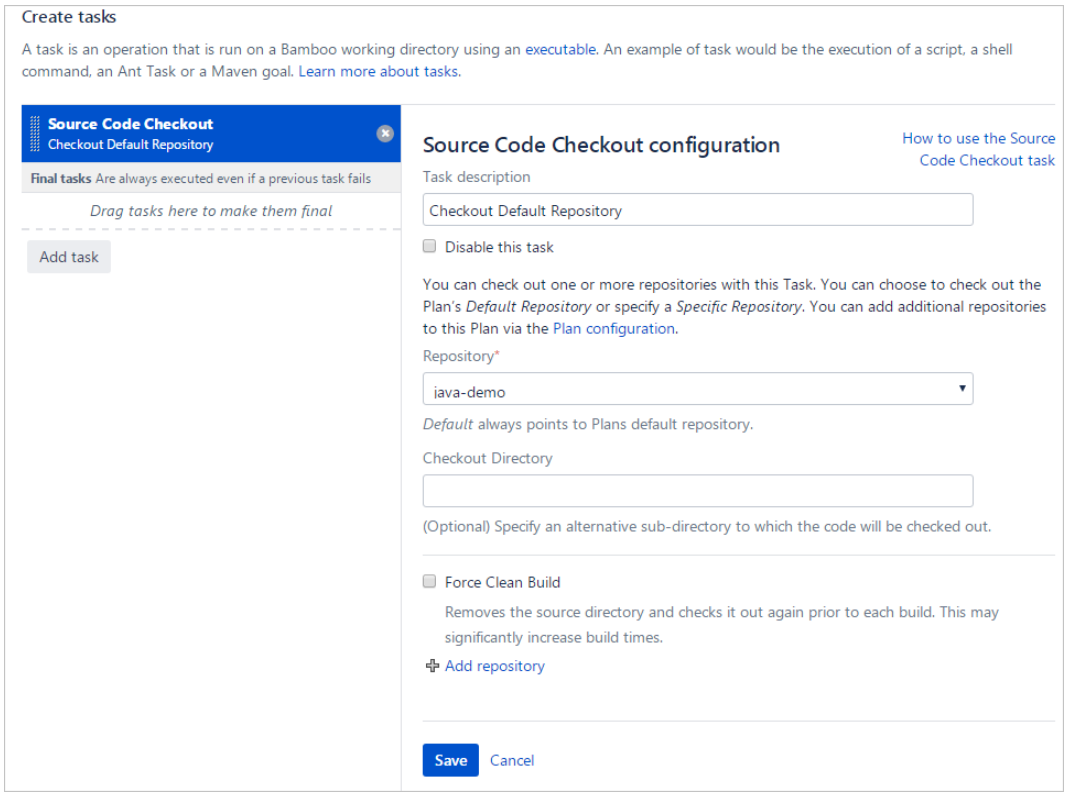


2. 进入**Configure Job**页面，配置Stages并添加Job。

   i. 源码拉取。

   在创建Build plan时，参数Repository host选择*java-demo*，即已完成源码拉取，您可以通过如下操作，修改源码拉取位置。

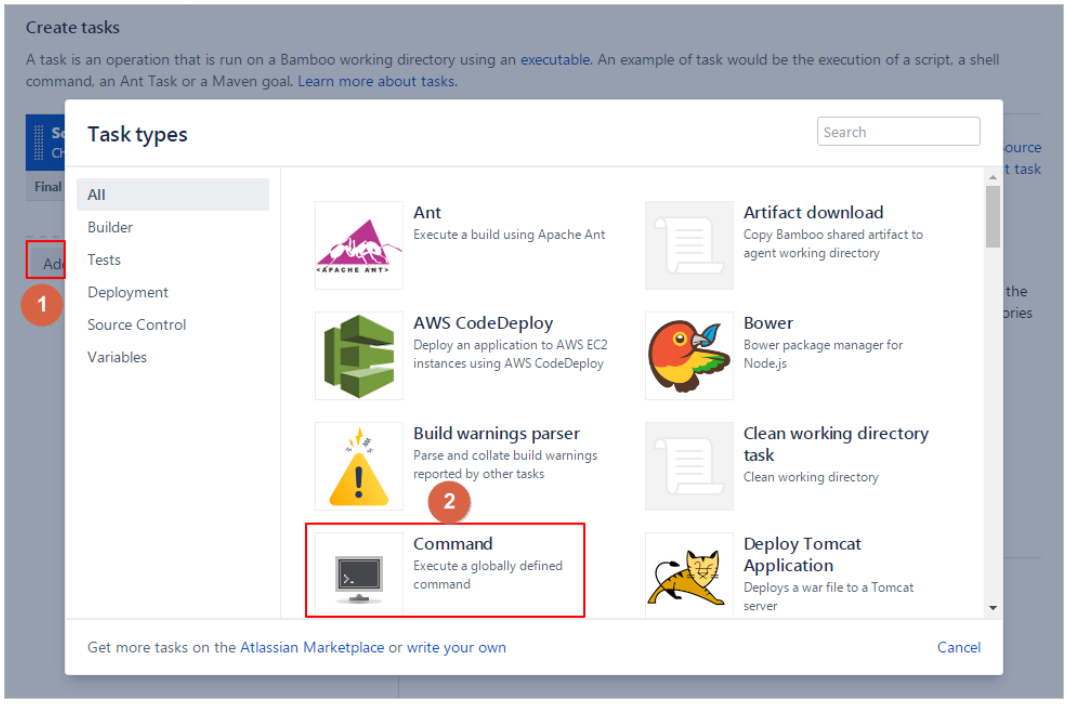
a. 在Create tasks区域，单击Source Code Checkout。



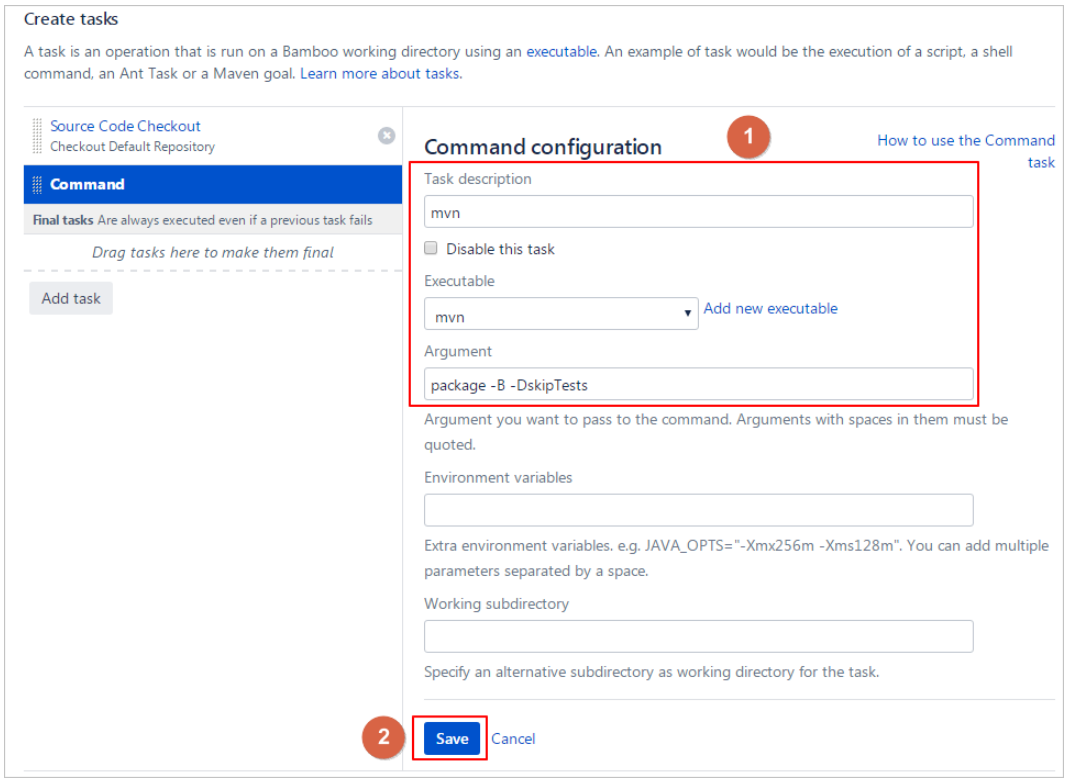
b. 在右侧的Source Code Checkout configuration页面，修改参数Repository host的取值，单击Save。

ii. mvn打包。

a. 在Create tasks区域，单击Add task，在弹出的Task types页面选择Command。

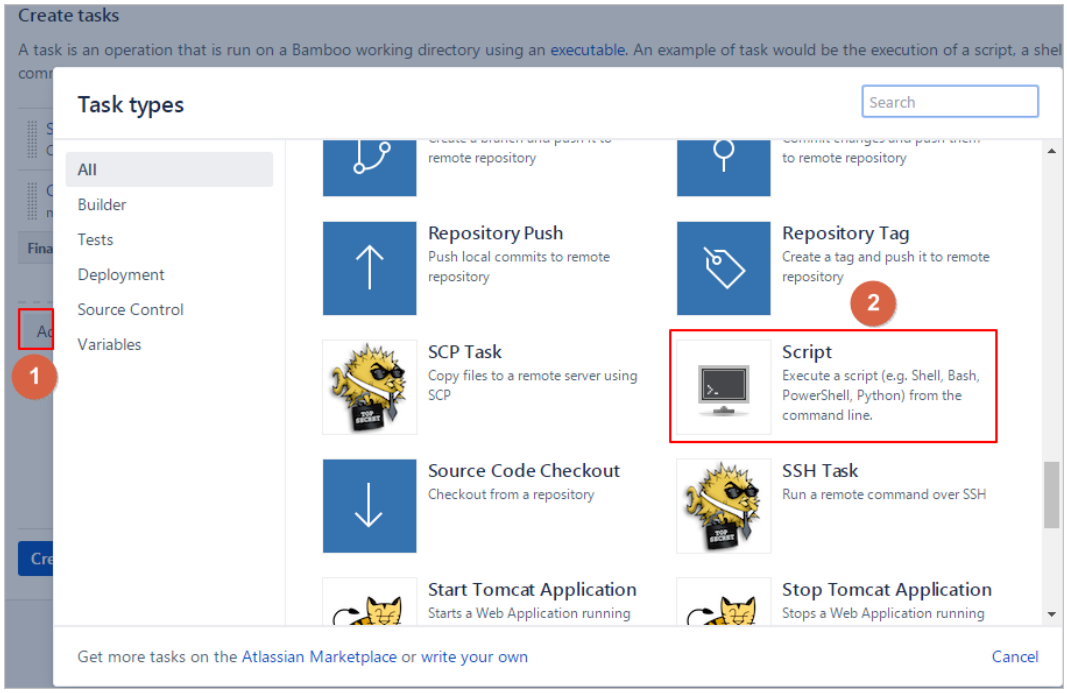


b. 在右侧弹出Command configuration页面，填写Task description、Executable和Argument后，单击Save。

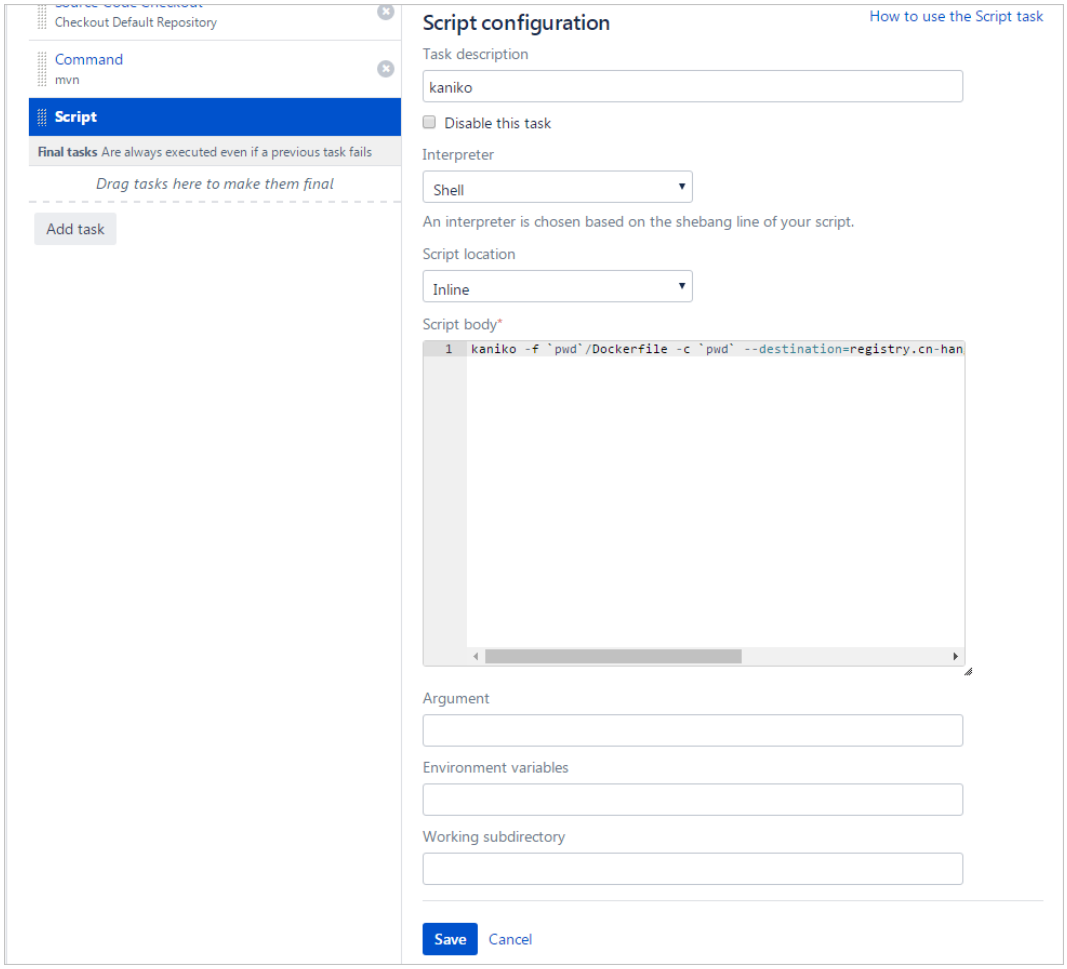


iii. kaniko打包和推送容器镜像。

a. 在Create tasks区域，单击Add task，在弹出的Task types页面选择Script。



b. 在右侧弹出Script configuration页面，填写Task description、Script location后（其他参数保持默认值即可），单击Save。

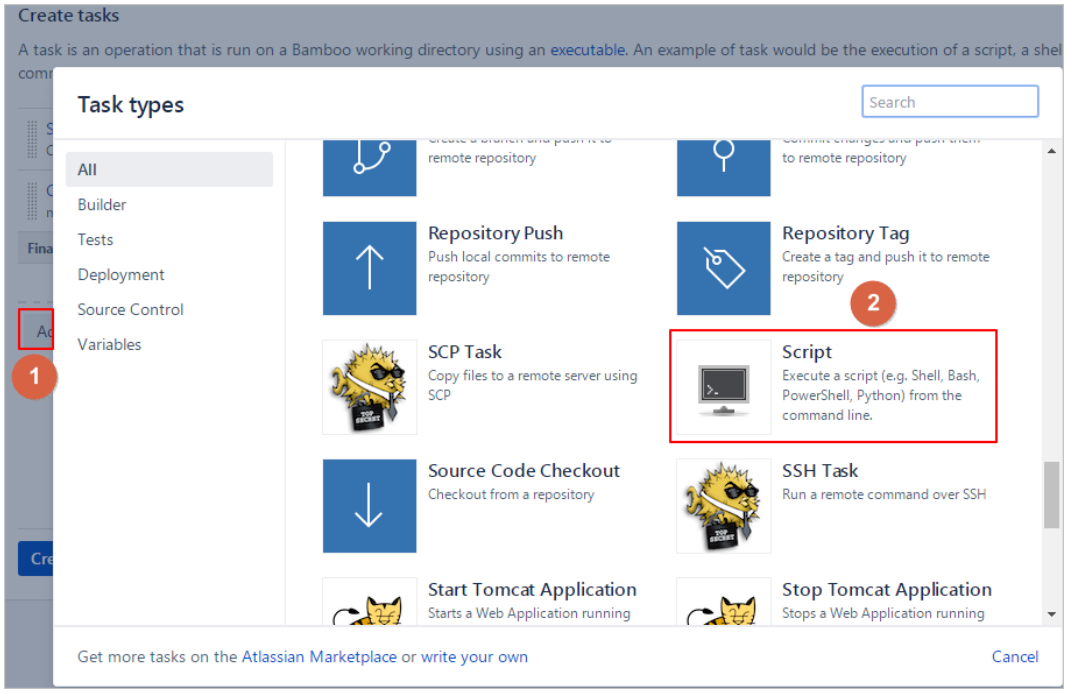


本示例中，Script location如下：

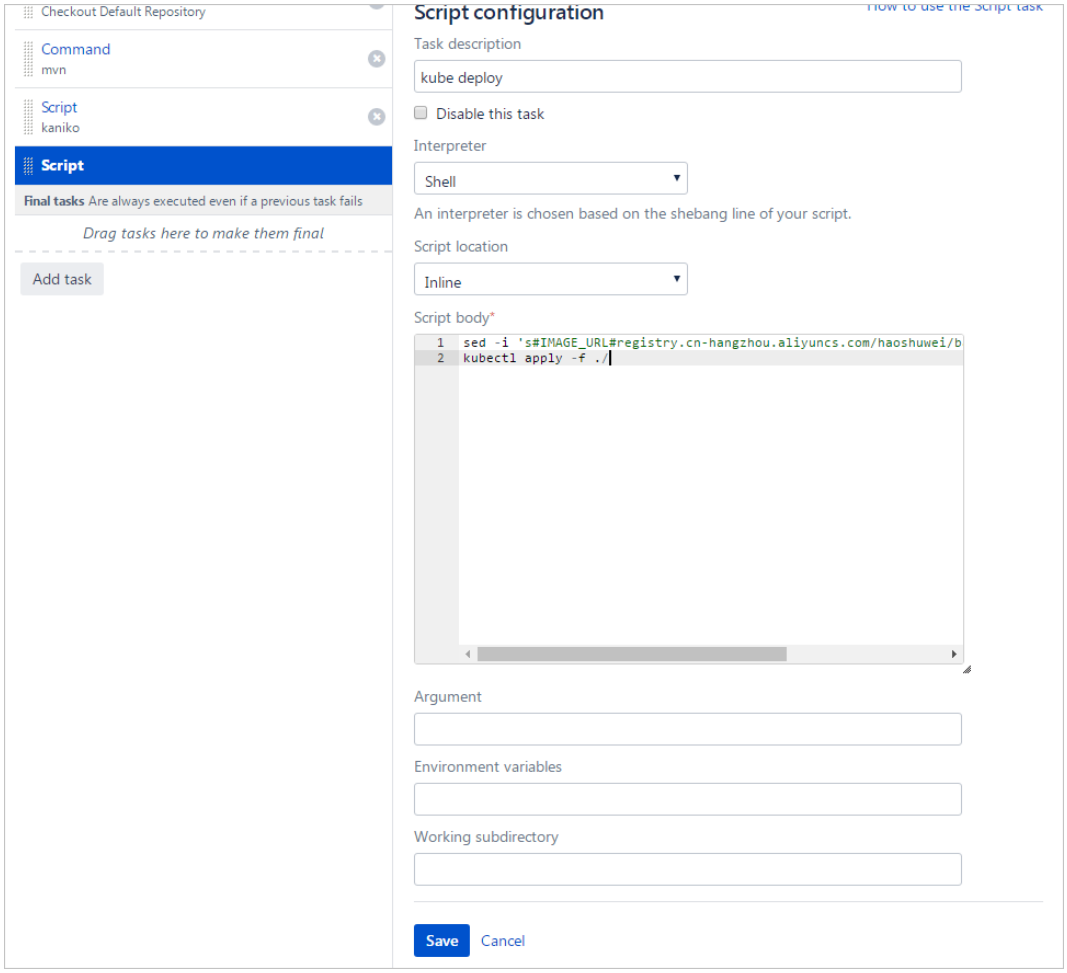
```
kaniko -f `pwd`/Dockerfile -c `pwd` --destination=registry.cn-hangzhou.aliyuncs
.com/haoshuwei/bamboo-java-demo:latest
```

iv. 使用Kubectl命令部署应用到Kubernetes。

a. 在Create tasks区域，单击Add task，在弹出的Task types页面选择Script。



b. 在右侧弹出Script configuration页面，填写Task description、Script location后（其他参数保持默认值即可），单击Save。

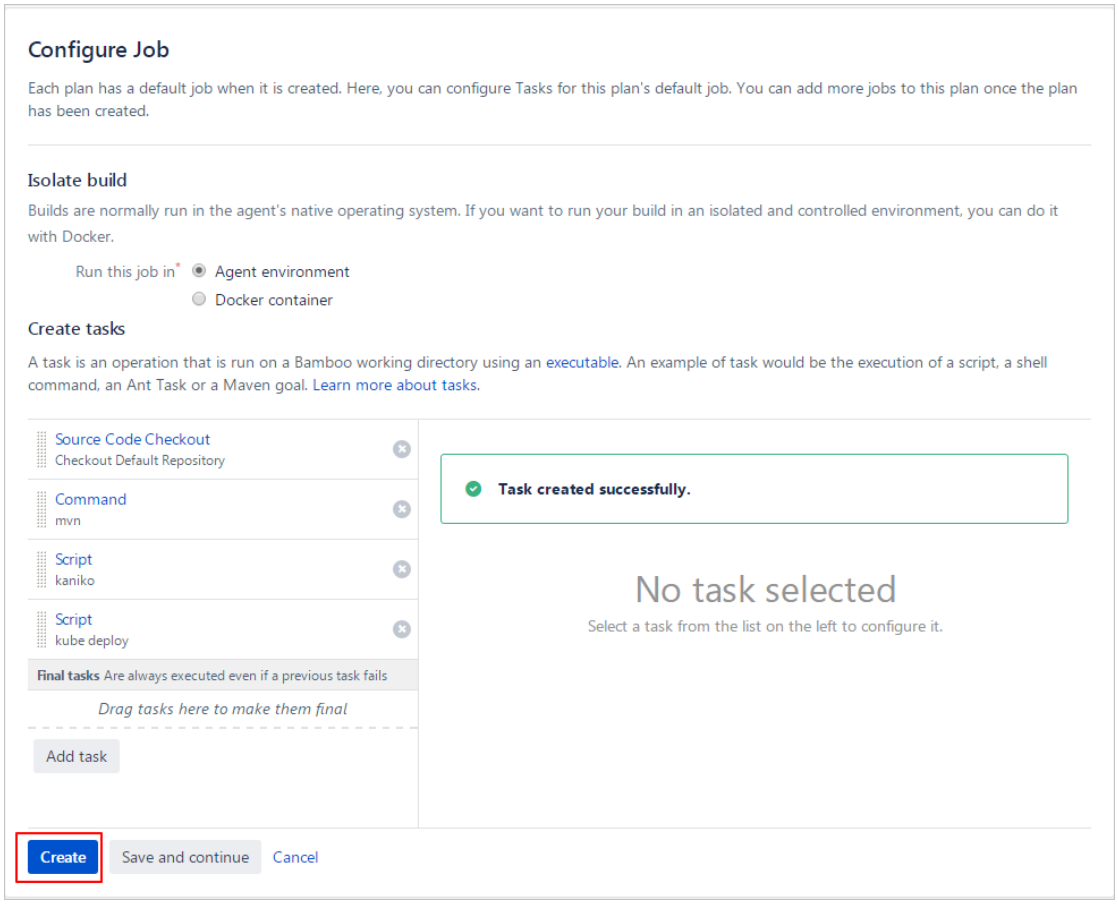


本示例中，Script location如下：

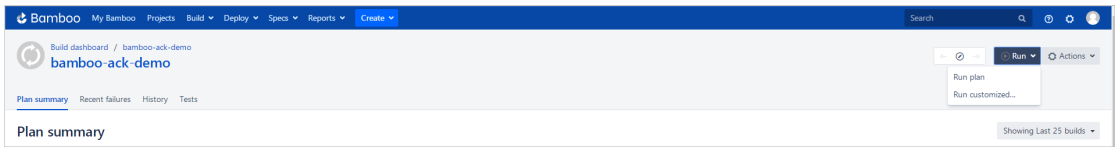
```
sed -i 's#IMAGE_URL#registry.cn-hangzhou.aliyuncs.com/haoshuwei/bamboo-java-dem
o:latest#' ./*.yaml
kubectl apply -f ./
```

3. 运行Build plan。

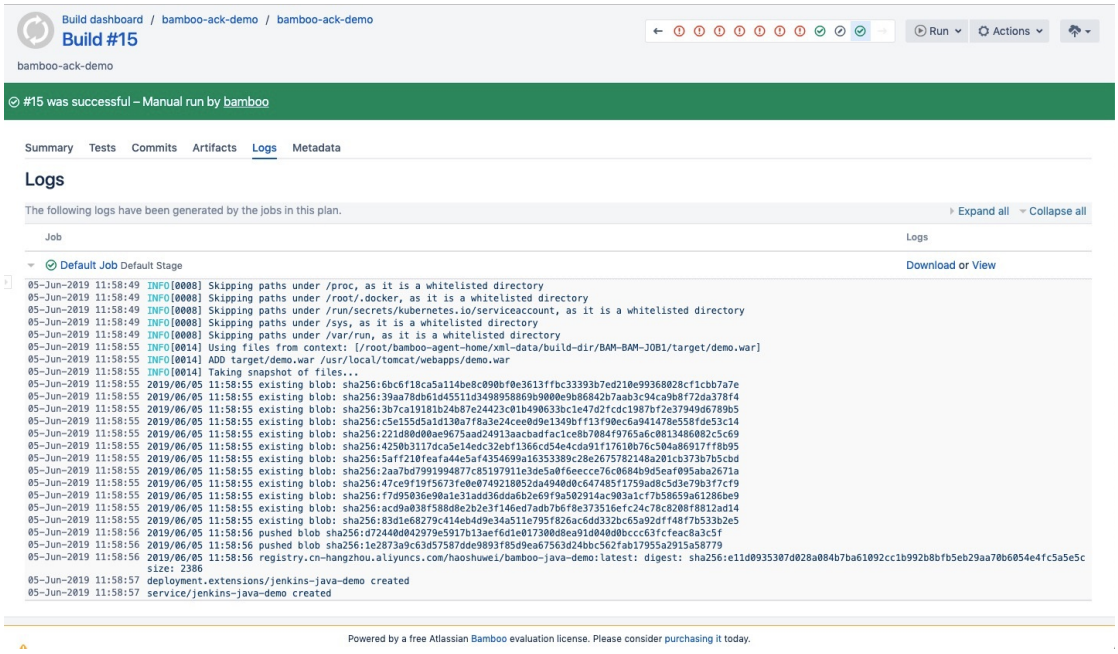
i. 上述配置完成后，单击Create，进入bamboo-ack-demo页面。



ii. 在右上角选择**Run > Run plan**，运行Build plan。



您可以通过单击**Logs**页签查看运行日志。



4. 访问应用服务。

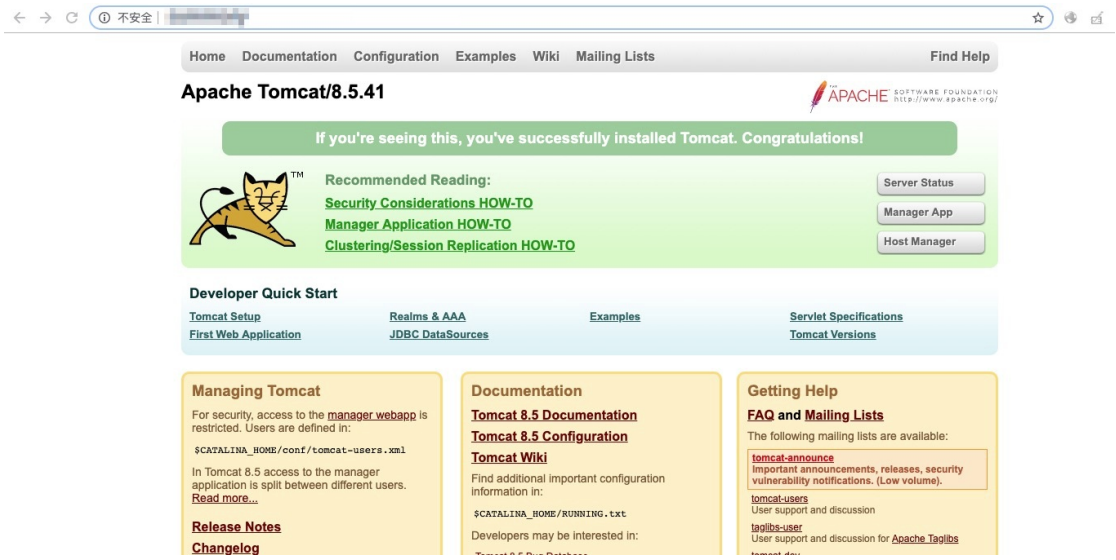
i. 执行 `kubectl -n bamboo get svc` 命令，查看应用服务。

```
[root@iZbp12i73koztp1cz75skaZ bamboo]# kubectl -n bamboo get svc
```

预期输出：

```
NAME                TYPE           CLUSTER-IP      EXTERNAL-IP       PORT(S)
AGE
jenkins-java-demo   LoadBalancer   xx.xx.xx.xx     xx.xx.xx.xx       80:32668/TCP
39m
```

ii. 在浏览器中输入*http://EXTERNAL-IP*，访问应用。



相关参考

查看本示例中镜像registry.cn-hangzhou.aliyuncs.com/haoshuwei/docker-bamboo-agent:v1的制作，请参见docker-bamboo-agent。

了解更多Bamboo的相关内容，请参见Bamboo。

11.7. 基于K8s多集群隔离环境下的DevOps实现

本文介绍基于K8s多集群隔离环境下的DevOps实现的场景描述、方案优势、解决问题、架构图及操作参考链接。

场景描述

DevOps是一组过程、方法与系统的统称，并通过工具实现自动化部署，确保部署任务的可重复性，减少部署出错的可能性。随着微服务、中台架构的兴起，DevOps重要性日益显著。

本方案使用两个独立的Kubernetes集群，模拟测试、生产环境，保证环境的高度隔离，互不影响。通过Gitlab+Jenkins的黄金组合，实现容器应用的自动化构建和持续部署，提供了从源码编译、镜像构建、应用测试、生产上线的全流程示例。

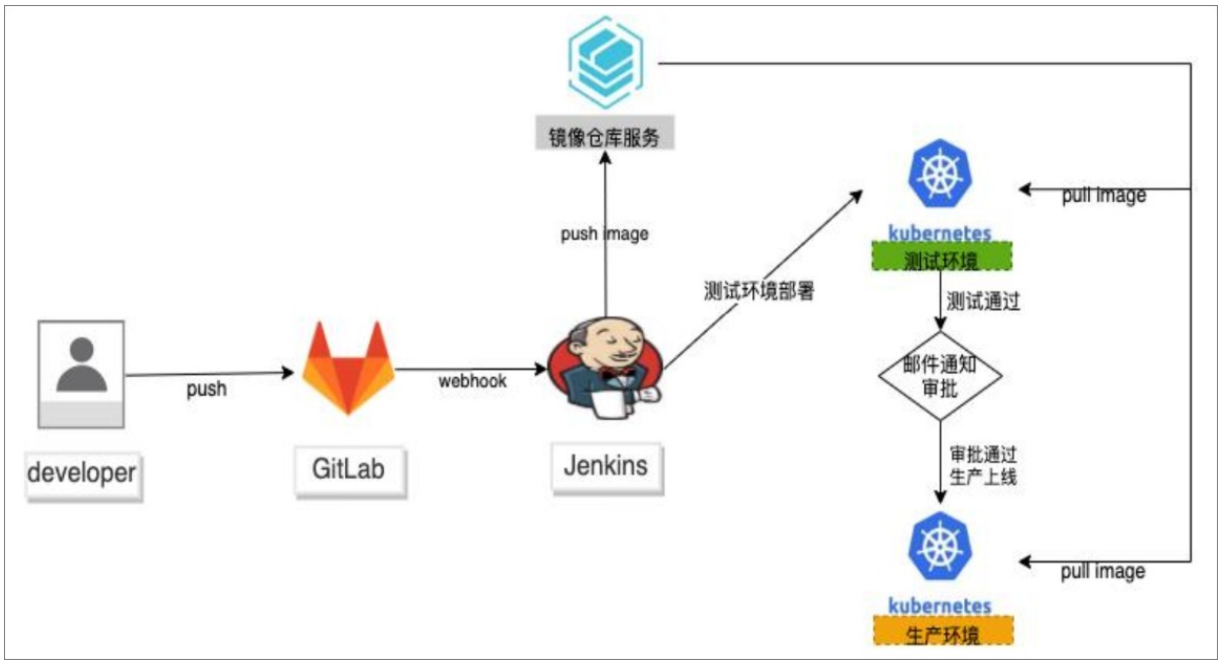
方案优势

- 全自动化构建与部署，提高部署效率与交付效率。
- 应用容器化部署，做到一次构建四处运行。
- 多K8s集群，保证测试、生产环境的完全隔离。

解决问题

- 微服务应用的CI/CD。
- 测试和生产环境的高度隔离。
- 自动化测试与部署。
- 现有CI对接ACK。

架构图



参考链接

基于K8s多集群隔离环境下DevOps实现的详细信息，请参见基于K8s多集群隔离环境下的DevOps实现。

12.自建Kubernetes迁移ACK

12.1. 上云须知

基于合规需求和安全考虑，阿里云Kubernetes与自建的Kubernetes在功能上有部分区别，请在使用阿里云Kubernetes前仔细阅读本文档。

集群规划

阿里云容器服务支持弹性裸金属服务器（神龙），您可以根据业务场景进行选择CPU型或GPU型。

- 弹性裸金属服务器
  是基于阿里云完全自主研发的下一代虚拟化技术而打造的新型计算类服务器产品，兼具虚拟机的弹性和物理机的性能及功能特性。例如，用户独占计算资源、加密计算、兼容多种专有云、异构指令集处理器支持。
- GPU云服务器
  是基于GPU应用的计算服务，多适用于AI深度学习、视频处理、科学计算、图形可视化等应用场景。

网络规划

- 容器集群基础设施（云服务器 ECS）的网络类型选择：专有网络VPC或经典网络。
  - 专有网络VPC：推荐使用。
    采用二层隔离，相对经典网络而言，VPC 具有更高的安全性和灵活性。
  - 经典网络
    采用三层隔离，所有经典网络类型实例都建立在一个共用的基础网络上。
- Kubernetes集群内Pod之间进行通信的网络插件选择：Terway或Flannel。
  - Terway：推荐使用。
    阿里云容器服务自研的网络插件，将阿里云的弹性网卡分配给容器，支持基于Kubernetes标准的NetworkPolicy来定义容器间的访问策略，支持对单个容器做带宽的限流。对于不需要使用Network Policy的用户，可以选择Flannel，其他情况建议选择Terway。
  - Flannel
    使用的是简单稳定的社区的Flannel CNI 插件，配合阿里云的VPC的高速网络，能给集群高性能和稳定的容器网络体验，但功能偏简单，支持的特性少，例如：不支持基于Kubernetes标准的Network Policy。

容量规划

- 基础容量：用户可以根据自己的成本和预算规划一个可满足初期业务正常运行的容量。
- 弹性容量：随后可以配置动态扩缩容随时调整集群规模。
  - horizontal pod autoscaling
    Kubernetes根据指标触发容量扩缩容，自动地伸缩在replication controller、deployment或replica set上Pod的数量。
  - cluster-autoscaler
    通过节点自动伸缩组件实现容量扩缩容，可以按需弹出普通实例、GPU实例、竞价付费实例，支持多可用区、多实例规格、多种伸缩模式，满足不同的节点伸缩场景。

数据规划

- 数据库数据
  阿里云关系型数据库（RDS）是一种稳定可靠、可弹性伸缩的在线数据库服务。基于阿里云分布式文件系统和SSD盘高性能存储，RDS支持MySQL、SQL Server、PostgreSQL、PPAS和MariaDB TX引擎。
- 存储数据

阿里云对象存储服务（OSS），是阿里云提供的海量、安全、低成本、高可靠的云存储服务。

- 容器镜像
  容器镜像服务提供安全的应用镜像托管能力，精确的镜像安全扫描功能，稳定的中国及国外镜像构建服务，便捷的镜像授权功能，方便用户进行镜像全生命周期管理。分为默认实例版和企业版。

安全防护

- 基础架构安全
  设置合理的安全组规则。
- 镜像安全
  使用私有镜像并定义镜像安全扫描。
- K8s应用安全
  设置不同服务间互相访问的网络安全策略等。

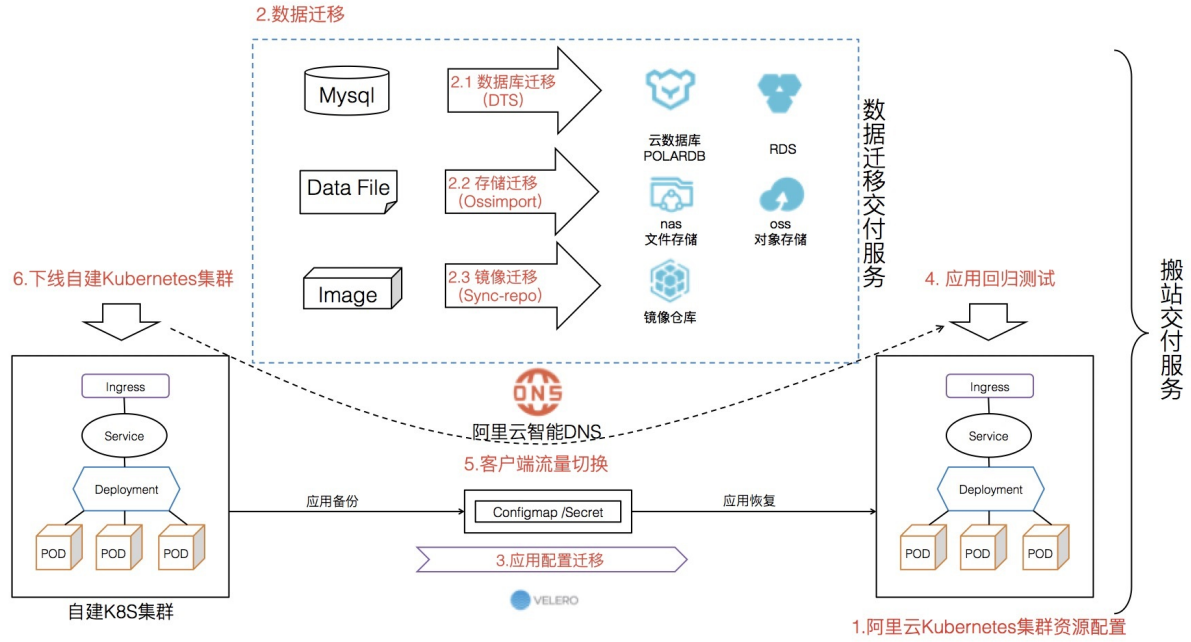
基础运维

- 监控配置：云监控、ARMS可供用户选择，提供全方位的集群及应用监控，并可根据阈值设定触发报警通知。
  - 云监控作为云服务的监控管理入口，能让您快速了解各产品实例的状态和性能。云监控从站点监控、云服务监控、自定义监控三个方面来为您提供服务。通过云监控管理控制台，您可以看到当前服务的监控项数据图表，清晰了解服务运行情况。同时可以通过设置报警规则，管理监控项状态，及时获取异常信息。
  - 应用实时监控服务（ARMS）是一款应用性能管理产品，包含前端监控，应用监控和Prometheus监控三大子产品，涵盖了浏览器，小程序，APP，分布式应用和容器环境等性能管理，能帮助您实现全栈式的性能监控和端到端的全链路追踪诊断， 让应用运维更加高效。
- 日志配置：用户可将自建的日志收集方案切换成阿里云上企业级的日志产品LOG。

12.2. Kubernetes迁移方案概述

本文整体简单介绍下如何通过6个步骤，将自建Kubernetes集群平滑迁移到阿里云Kubernetes集群，并尽量确保迁移期间对业务无影响。

迁移方案



迁移步骤

1. 阿里云Kubernetes集群资源配置。
   由运维人员完成阿里云Kubernetes集群创建及集群维度资源配置，降低开发人员应用迁移复杂度。详情请参见创建Kubernetes专有版集群。
   集群创建和配置过程中，以下配置项为必选。
   - 集群模版：您可以根据业务需要选择以下集群模板。
     - 标准专有集群。
       Kubernetes集群的Master和Worker由用户创建并管理，对集群享有完整的控制能力。您需要承担Master和Worker的资源费用。
     - 标准托管集群。
       Kubernetes集群的Master由容器服务托管，您只需创建并管理Worker，免Master运维。您需要承担Worker的资源费用。
   - **操作系统**：您可以根据需要选择操作系统。
     - 推荐您使用阿里云Kubernetes集群默认的操作系统，CentOS7.6或AliyunLinux 2.1903。
     - 若您对系统内核版本有要求，可使用自定义镜像，详情请参见使用自定义镜像创建ACK集群。
   - **专有网络**：您需要设置VPC和交换机信息。
   - **配置 SNAT**：您需要为专有网络配置SNAT。
   - **公网访问**：您需要使用EIP暴露API Server。
   - **云监控插件**：您需要在ECS节点上安装云监控插件。
   - **日志服务**：您需要安装日志服务插件。
2. 数据迁移。
   - 数据库迁移。
     a. 创建RDS数据库。
     b. 配置数据库白名单。

c. 配置PrivateZone。
通过PrivateZone可以将数据库域名解析到RDS域名，减少应用改造。

d. 迁移MySQL数据。
通过DTS迁移MySQL数据，支持全量、增量、双向同步。详情请参见从自建MySQL迁移至RDS MySQL。

- 存储迁移。
  a. 开通OSS服务。
  b. 创建OSS存储空间（Bucket）。
  c. 迁移存储数据。
  使用ossimport工具把存储在服务器本地、第三方云存储（S3、Azure、腾讯COS等）中的数据批量迁移到OSS。详情请参见说明及配置。
- 镜像迁移。
  a. 创建ACR容器镜像仓库。
  b. 设置镜像服务访问凭证。
  c. 迁移镜像数据。
  使用image-syncer快速将容器镜像批量迁移至ACR容器镜像仓库服务。详情请参见通过image-syncer工具迁移容器镜像。

3. 应用配置迁移。
由运维或者研发人员基于velero工具完成集群或者应用配置迁移。详情请参见Kubernetes应用迁移。
   i. 准备迁移环境。
      a. 安装Velero客户端。
      b. 创建OSS Bucket。
      c. 创建RAM账号并生成AccessKey。
      d. 部署Velero服务端。
   ii. 在自建Kubernetes集群备份应用。
      - 无PV数据的应用备份。
      - 有PV数据的应用备份。
   iii. 在阿里云Kubernetes集群恢复应用。
      a. 创建StorageClass。
      b. 恢复应用。
   iv. 更新应用配置。
      - 更新镜像地址。
      - 优化服务暴露方式。
      - 适配存储盘挂载方式。
   v. 调试并启动应用。
4. 应用回归测试。
在不影响线上流量情况下，由测试人员完成阿里云Kubernetes集群业务功能回归测试。
   i. 配置应用测试域名。
   ii. 测试业务功能。
   iii. 确认应用日志采集。

iv. 确认应用监控。

5. 客户端流量切换。
   由运维人员做DNS切换，将流量切换至阿里云Kubernetes集群。
   i. DNS流量切换：调整DNS解析配置实现流量切换。
   ii. 客户端流量切换：升级客户端代码或配置实现流量切换。
6. 下线自建Kubernetes集群。
   由运维人员确认阿里云Kubernetes集群服务访问正常后，下线自建Kubernetes集群资源。
   i. 确认阿里云Kubernetes集群流量正常。
   ii. 下线自建Kubernetes集群资源。
   iii. 清理OSS Bucket中的备份文件。

12.3. 使用自定义镜像创建ACK集群

在ACK迁移业务场景中，如无特殊需求，我们推荐您使用ACK默认的系统镜像及其他系统服务。

背景信息

迁云业务中，推荐您使用ACK默认的系统镜像（CentOS7.6或AliyunLinux 2.1903）及其他系统服务，例如系统内核、DNS、YUM源等。 如有特殊需求需要制作自定义系统镜像，我们开源了ack-image-builder工具帮助您快速制作符合ACK集群节点要求的自定义镜像。

使用ack-image-builder创建自定义镜像

ack-image-builder项目基于开源工具HashiCorp Packer，提供默认配置模板和校验脚本。

使用ack-image-builder可以降低人工操作的错误率，同时也能记录镜像变更历史，便于故障定位。使用ack-image-builder项目创建ACK集群自定义节点镜像的步骤如下：

1. 安装Packer。

   从官方下载页面选择操作系统对应的软件版本，并按照安装说明文档安装和验证 Packer。
   执行以下命令，显示结果如下时，说明Packer已安装成功。

   ```
   packer version
   ```

   ```
   Packer v1.4.1
   ```

2. 定义Packer 模板。

   使用Packer创建自定义镜像时，需要创建一个JSON格式的模板文件。在该模板文件中，您需要指定创建自定义镜像的Alicloud Image Builder（生成器）和Provisioners（配置器）。

```
{
  "variables": {
    "region": "cn-hangzhou",
    "image_name": "test_image{{timestamp}}",
    "source_image": "centos_7_06_64_20G_alibase_20190711.vhd",
    "instance_type": "ecs.n1.large",
    "access_key": "{{env `ALICLOUD_ACCESS_KEY`}}",
    "secret_key": "{{env `ALICLOUD_SECRET_KEY`}}"
  },
  "builders": [
    {
      "type": "alicloud-ecs",
      "access_key": "{{user `access_key`}}",
      "secret_key": "{{user `secret_key`}}",
      "region": "{{user `region`}}",
      "image_name": "{{user `image_name`}}",
      "source_image": "{{user `source_image`}}",
      "ssh_username": "root",
      "instance_type": "{{user `instance_type`}}",
      "io_optimized": "true"
    }
  ],
  "provisioners": [
    {
      "type": "shell",
      "scripts": [
        "config/default.sh",
        "scripts/updateDNS.sh",
        "scripts/reboot.sh",
        "scripts/verify.sh"
      ],
      "expect_disconnect": true
    }
  ]
}
```

参数	描述
access_key	您的AccessKeyID。
secret_key	您的AccessKeySecret。
region	创建自定义镜像时使用临时资源的地域。
image_name	自定义镜像的名称。
source_image	基础镜像的名称，可以从阿里云公共镜像列表获得。
instance_type	创建自定义镜像时生成的临时实例的类型。
provisioners	创建自定义镜像时使用的 Packer 配置器类型。

3. 创建子账号并生成AccessKey。

制作自定义镜像的权限要求较大，建议您创建子账户并授权Packer需要的对应 RAM Policy 并创建AccessKey。

4. 导入AccessKey信息并制作自定义镜像。
   i. 执行以下命令，导入AccessKey信息。

   ```
   export ALICLOUD_ACCESS_KEY=XXXXXX
   export ALICLOUD_SECRET_KEY=XXXXXX
   ```

   ii. 执行以下命令，制作自定义镜像。

   ```
   packer build alicloud.json
   ```

```
alicloud-ecs output will be in this color.
==> alicloud-ecs: Prevalidating source region and copied regions...
==> alicloud-ecs: Prevalidating image name...
    alicloud-ecs: Found image ID: centos_7_06_64_20G_alibase_20190711.vhd
==> alicloud-ecs: Creating temporary keypair: xxxxxx
==> alicloud-ecs: Creating vpc...
    alicloud-ecs: Created vpc: xxxxxx
==> alicloud-ecs: Creating vswitch...
    alicloud-ecs: Created vswitch: xxxxxx
==> alicloud-ecs: Creating security group...
    alicloud-ecs: Created security group: xxxxxx
==> alicloud-ecs: Creating instance...
    alicloud-ecs: Created instance: xxxxxx
==> alicloud-ecs: Allocating eip...
    alicloud-ecs: Allocated eip: xxxxxx
    alicloud-ecs: Attach keypair xxxxxx to instance: xxxxxx
==> alicloud-ecs: Starting instance: xxxxxx
==> alicloud-ecs: Using ssh communicator to connect: 47.111.127.54
==> alicloud-ecs: Waiting for SSH to become available...
==> alicloud-ecs: Connected to SSH!
==> alicloud-ecs: Provisioning with shell script: scripts/verify.sh
    alicloud-ecs: [20190726 11:04:10]: Check if kernel version >= 3.10.  Verify Pas
sed!
    alicloud-ecs: [20190726 11:04:10]: Check if systemd version >= 219.  Verify Pas
sed!
    alicloud-ecs: [20190726 11:04:10]: Check if sshd is running and listen on port
22.  Verify Passed!
    alicloud-ecs: [20190726 11:04:10]: Check if cloud-init is installed.  Verify Pa
ssed!
    alicloud-ecs: [20190726 11:04:10]: Check if wget is installed.  Verify Passed!
    alicloud-ecs: [20190726 11:04:10]: Check if curl is installed.  Verify Passed!
    alicloud-ecs: [20190726 11:04:10]: Check if kubeadm is cleaned up.  Verify Pass
ed!
    alicloud-ecs: [20190726 11:04:10]: Check if kubelet is cleaned up.  Verify Pass
ed!
    alicloud-ecs: [20190726 11:04:10]: Check if kubectl is cleaned up.  Verify Pass
ed!
    alicloud-ecs: [20190726 11:04:10]: Check if kubernetes-cni is cleaned up.  Veri
fy Passed!
==> alicloud-ecs: Stopping instance: xxxxxx
==> alicloud-ecs: Waiting instance stopped: xxxxxx
==> alicloud-ecs: Creating image: test_image1564110199
    alicloud-ecs: Detach keypair xxxxxx from instance: xxxxxxx
==> alicloud-ecs: Cleaning up 'EIP'
==> alicloud-ecs: Cleaning up 'instance'
==> alicloud-ecs: Cleaning up 'security group'
==> alicloud-ecs: Cleaning up 'vSwitch'
==> alicloud-ecs: Cleaning up 'VPC'
==> alicloud-ecs: Deleting temporary keypair...
Build 'alicloud-ecs' finished.
==> Builds finished. The artifacts of successful builds are:
--> alicloud-ecs: Alicloud images were created:
cn-hangzhou: m-bp1aifbnupnaktj00q7s
```

其中 scripts/verify.sh 为对检查项的校验部分。

使用自定义操作系统内核

ACK要求自定义操作系统内核版本不小于 `3.10` ，请仅更新您需要自定义安装的相关rpm包，并设置正确的内核引导项。

示例如下。

```
cat scripts/updateOSKernel.sh
#!/bin/bash
VERSION_KERNEL="3.10.0-1062.4.3.el7"
yum  localinstall -y  http://xxx.xxx.xxx.xxx/kernel-${VERSION_KERNEL}.x86_64.rpm   http://x
xx.xxx.xxx.xxx/kernel-devel-${VERSION_KERNEL}.x86_64.rpm   http://xxx.xxx.xxx.xxx/kernel-he
aders-${VERSION_KERNEL}.x86_64.rpm
grub_num=$(cat /etc/grub2.cfg |awk -F\' '$1=="menuentry " {print i++ " : " $2}' |grep $VERS
ION_KERNEL |awk -F ':' '{print $1}')
grub2-set-default $grub_num
```

> **说明** 不推荐使用 `yum update -y` 等全局更新的命令。

使用自定义内核参数

如果您需设置自定义内核参数，请注意不要覆盖以下参数项。

```
["vm.max_map_count"]="262144"
["kernel.softlockup_panic"]="1"
["kernel.softlockup_all_cpu_backtrace"]="1"
["net.core.somaxconn"]="32768"
["net.core.rmem_max"]="16777216"
["net.core.wmem_max"]="16777216"
["net.ipv4.tcp_wmem"]="4096 12582912 16777216"
["net.ipv4.tcp_rmem"]="4096 12582912 16777216"
["net.ipv4.tcp_max_syn_backlog"]="8096"
["net.ipv4.tcp_slow_start_after_idle"]="0"
["net.core.netdev_max_backlog"]="16384"
["fs.file-max"]="2097152"
["fs.inotify.max_user_instances"]="8192"
["fs.inotify.max_user_watches"]="524288"
["fs.inotify.max_queued_events"]="16384"
["net.ipv4.ip_forward"]="1"
["net.bridge.bridge-nf-call-iptables"]="1"
["fs.may_detach_mounts"]="1"
["net.ipv4.conf.default.rp_filter"]="0"
["net.ipv4.tcp_tw_reuse"]="0"
["net.ipv4.tcp_tw_recycle"]="0"
```

> **说明** 如若以上某些参数必须覆盖，请先申请工单请求ACK的同学评估影响。然后在控制台创建集群或扩容集群页面的**高级选项** > **实例自定义数据**处填写对应的脚本。

使用自定义DNS服务

使用自定义DNS服务需要关注以下几点：

- 自定义DNS服务的upstream nameserver必须添加阿里云的nameserver 。

```
cat /etc/resolv.conf
options timeout:2 attempts:3 rotate single-request-reopen
; generated by /usr/sbin/dhclient-script
nameserver 100.XX.XX.136
nameserver 100.XX.XX.138
```

- 修改*/etc/resolve.conf*后必须锁定文件，不然ECS实例重启后cloud-init会刷新文件恢复默认设置，示例如下：

```
cat scripts/updateDNS.sh
#!/bin/bash
# unlock DNS file in case it was locked
chattr -i /etc/resolv.conf
# Using your custom nameserver to replace xxx.xxx.xxx.xxx
echo -e "nameserver xxx.xxx.xxx.xxx\nnameserver xxx.xxx.xxx.xxx" > /etc/resolv.conf
# Keep resolv locked to prevent overwriting by cloudinit/NetworkManager
chattr +i /etc/resolv.conf
```

- 确保自定义DNS服务的性能。
  如果ACK容器集群规模较大，请确保自定义DNS服务的性能能够满足您的业务需求。

使用自定义YUM源

使用自定义YUM源需要关注以下两点：

- 避免RPM包的全量更新 。
  只更新需要安装部署的rpm包，避免使用 `yum update -y` 命令。
- 自定义YUM源的性能 。
  如果ACK集群单次扩容规模较大且依赖YUM源的话，请确保YUM源的性能能够满足您的业务需求， 示例如下。

```
cat scripts/add-yum-repo.sh
#!/bin/bash
cat << EOF > /etc/yum.repos.d/my.repo
[base]
name=CentOS-\$releasever
enabled=1
failovermethod=priority
baseurl=http://mirrors.cloud.aliyuncs.com/centos/\$releasever/os/\$basearch/
gpgcheck=1
gpgkey=http://mirrors.cloud.aliyuncs.com/centos/RPM-GPG-KEY-CentOS-7
EOF
```

ACK系统组件容器镜像预加载

当ACK集群单次扩容规模大于1000台时，推荐您在制作自定义系统镜像时预加载ACK的一些DaemonSet资源类型的系统组件容器镜像，这样可以减少节点启动时拉取容器镜像的操作，优化扩容效率。

1. 将相关系统组件容器镜像打tar包存储在镜像中。

   本例中，假设您创建的集群使用Terway网络、CSI存储插件，镜像用于cn-hangzhou地域的集群，则预加载脚本示例如下：

```
cat scripts/prepare-images.sh
#!/bin/bash
set -x -e
EXPORT_PATH=/preheated
# 安装 Docker
yum install -y docker
systemctl start docker
# 批量拉取、保存镜像
images=(
registry-vpc.cn-hangzhou.aliyuncs.com/acs/csi-plugin:v1.14.5.60-5318afe-aliyun
registry-vpc.cn-hangzhou.aliyuncs.com/acs/terway:v1.0.10.78-g97729ee-aliyun
)
mkdir -p ${EXPORT_PATH}
for image in "${images[@]}"; do
    echo "preheating ${image}"
    docker pull ${image}
    docker save -o ${EXPORT_PATH}/$(echo ${image}| md5sum | cut -f1 -d" ").tar ${image}
done
# 卸载 Docker
yum erase -y docker
rm -rf /var/lib/docker
```

2. 登录容器服务管理控制台，在创建集群的**高级选项 > 实例自定义数据**处填写以下脚本。

```
ls /preheated/ | xargs -n 1 -i docker load -i /preheated/{}
rm -rf /preheated
```

自定义镜像的配置文件

在 `alicloud.json` 文件中的 `provisioners` 添加以下内容，生成自定义系统镜像。

```
"provisioners": [
  {
    "type": "shell",
    "scripts": [
      "config/default.sh",
      "scripts/updateOSKernel.sh",
      "scripts/updateDNS.sh",
      "scripts/add-yum-repo.sh",
      "scripts/prepare-images.sh",
      "scripts/reboot.sh",
      "scripts/verify.sh"
    ],
    "expect_disconnect": true
  }
]
```

说明 `config/default.sh` 、 `scripts/reboot.sh` 和 `scripts/verify.sh` 为默认必须执行的脚本，其他为用户自定义的脚本。

`config/default.sh` 中设置时区并关闭swap分区。

`scripts/verify.sh` 中检查自定义镜像是否符合ACK集群节点要求。

此时，您可以使用自定义系统镜像扩容ACK集群并进行验证。

创建ACK集群

为了节约创建集群的时间，以及降低出错概率，您可以先创建一个无Worker节点的专有版集群或者选择2个Worker节点的托管版集群，然后选择自定义镜像进行集群扩容 ，并进行验证。

1. 创建一个基于ACK默认系统镜像的3Masters+0Worker或者5Masters+0Worker的集群。请参见创建Kubernetes专有版集群。

   > **说明** 如果您创建的是托管版Kubernetes集群，需要至少选择2个Worker节点。请参见创建Kubernetes托管版集群。

2. 选择自定义镜像进行集群扩容，请参见扩容ACK集群的节点。

   如若需要在自定义镜像节点添加到ACK集群后运行其他初始化脚本，可以选择填写实例自定义数据。

   > **说明** 您可以通过提交工单申请开通**自定义镜像选择**和**实例自定义数据**功能。

12.4. 自建Kubernetes迁移至ACK弹性裸金属集群

本文介绍将自建Kubernetes集群迁移至ACK弹性裸金属集群的场景描述、解决问题、架构图及操作参考链接。

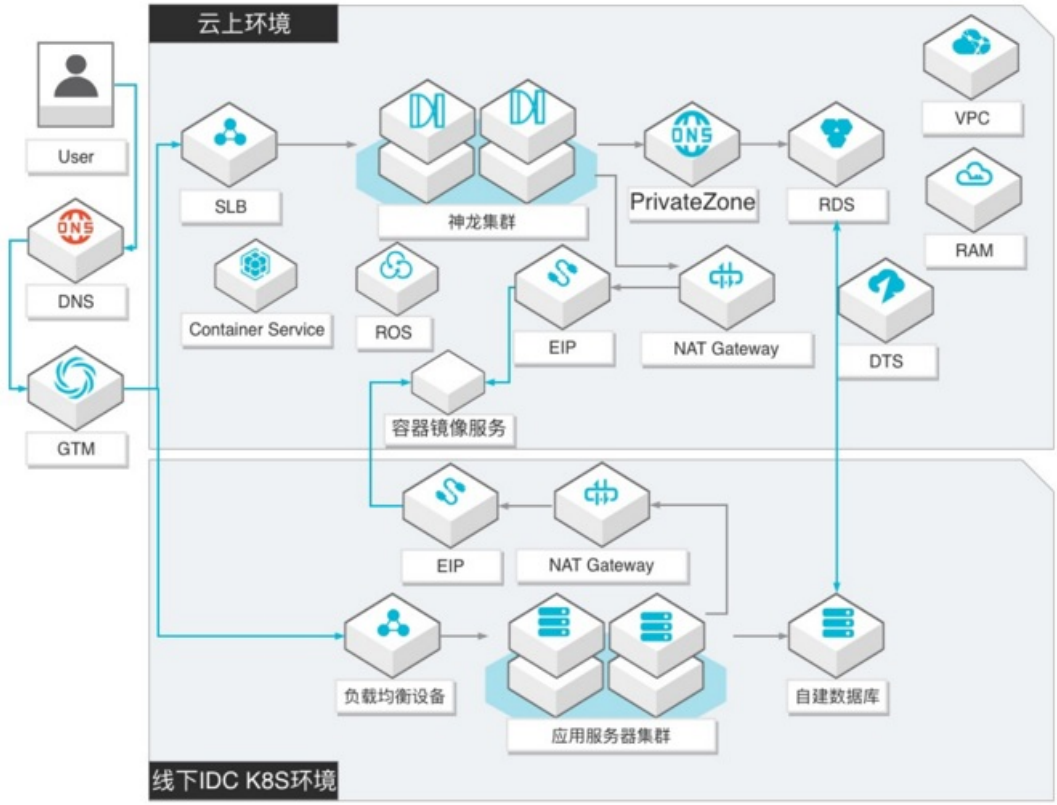
场景描述

在微服务化改造之后，企业在享受Kubernetes带来应用管理的便利的同时，存在硬件性能不足、本地扩展性差、容器容灾难、Kubernetes管理复杂等问题。

解决问题

- 增强Kubernetes扩展性：如何通过弹性裸金属服务器的强劲硬件性能，实现云端资源急速扩展，从容应对应用访问压力大的问题。
- 如何简化云端Kubernetes运维，通过ACK实现敏捷开发和部署落地，加速企业业务迭代。
- 如何综合考虑迁移和容灾，如何整合云上和云下容器资源实现遇到故障时可以通过健康检查实现自动容灾。
- 如何不改应用上云，实现应用上云数据库连接零修改。
- 数据库上云及回退，如何实现上云回退。

架构图



参考链接

有关如何将自建Kubernetes集群迁移至ACK弹性裸金属集群，请参见自建Kubernetes集群迁移ACK弹性裸金属集群最佳实践。

12.5. 自建Kubernetes集群迁移镜像及应用至ACK

本文介绍将自建Kubernetes集群迁移镜像及应用至阿里云容器服务ACK集群的场景描述、方案优势、解决问题、架构图及操作参考链接。

场景描述

本最佳实践构建以下场景：

- 以河源ECS构建Harbor仓库，模拟IDC的镜像仓库服务。
- 以河源ECS构建Registry仓库，模拟IDC的镜像仓库服务。
- 以河源地域模的ECS搭建K8s集群，模拟线下IDC的K8s环境。
- 使用velero对云上的K8s应用进行定期备份，并存至OSS上，确保应用数据不丢失。

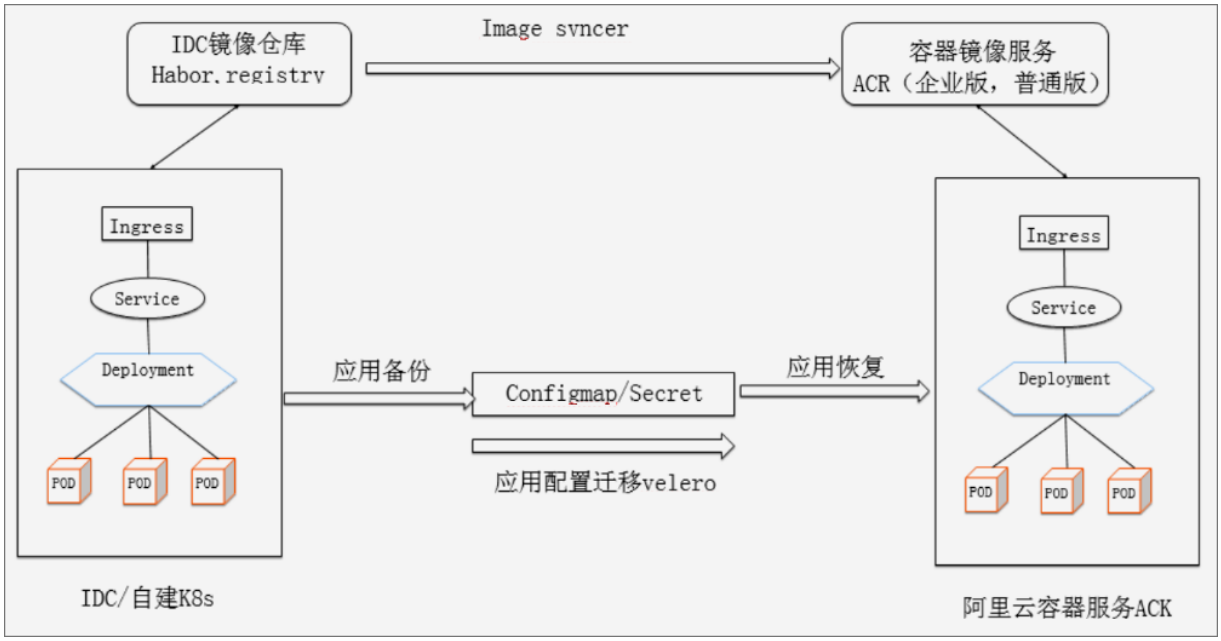
方案优势

随着技术的更新迭代，自建的K8s相关的容器镜像服务、集群管理、稳定性保障让企业IT人员感觉到压力，所以上云可以将底层的IaaS基础设施和K8s的基础PaaS能力交给阿里云来管理。

解决问题

平滑、便捷的迁移自建的K8s镜像和应用至阿里云容器服务ACK。

架构图



参考链接

关于如何将自建Kubernetes迁移镜像及应用至阿里云ACK，请参见自建Kubernetes迁移镜像及应用至阿里云ACK。

12.6. 容器Swarm集群向ACK集群灰度迁移

本文介绍将容器Swarm集群向ACK集群灰度迁移的场景描述、解决问题、架构图及操作参考链接。

场景描述

随着K8s生态的完善，越来越多的客户需要从Swarm集群迁移向ACK集群。

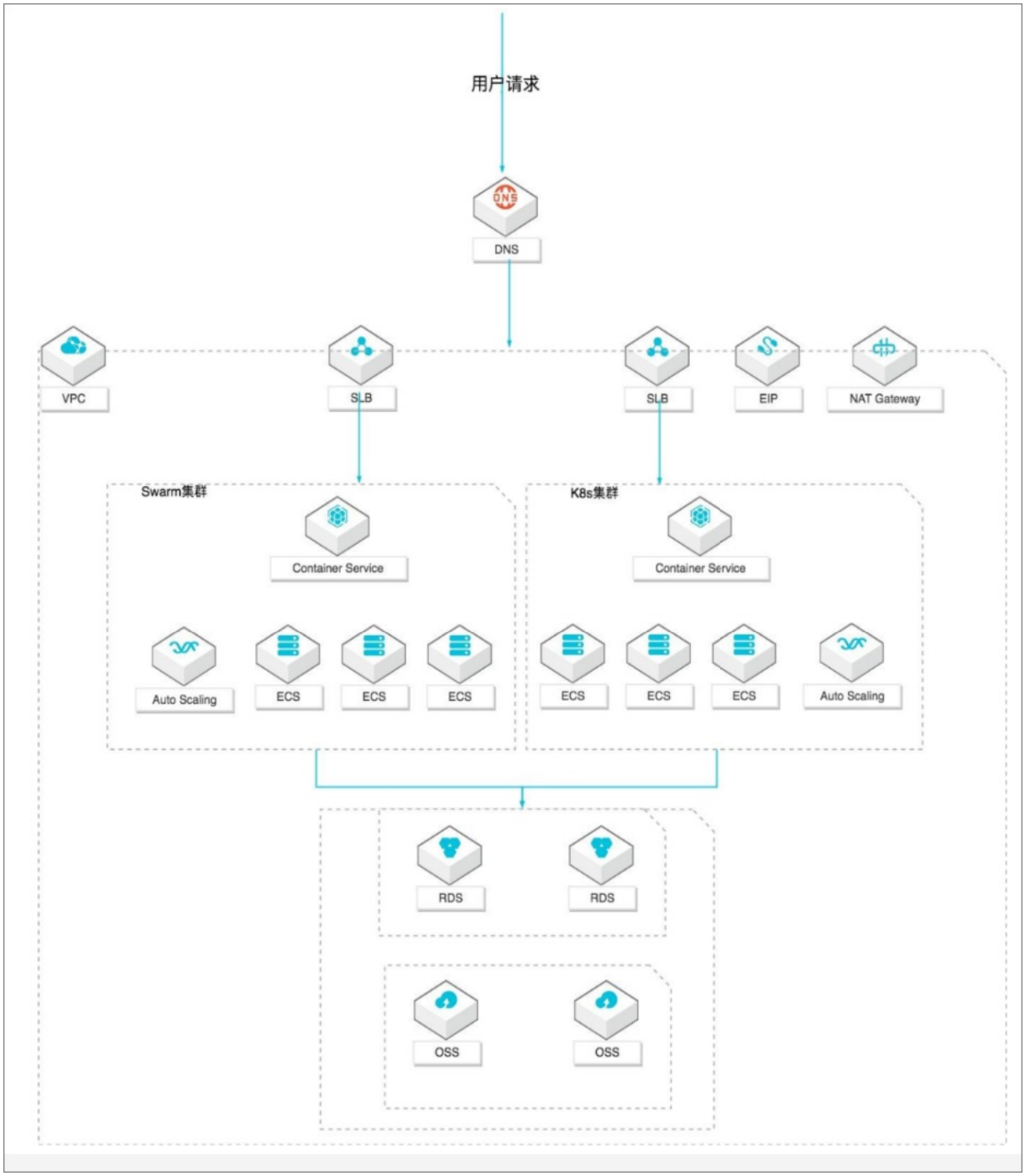
解决问题

- 迁移过程中维持业务的延续性
- 迁移过程业务高可用
- 迁移过程可灰度
- 迁移过程可回滚
- 迁移进度可把控

架构图

方案说明：

- 容器集群部署，左侧Swarm集群，右侧K8s集群。
- SLB做应用高可用和应用扩展。
- DNS结合全局流量控制，权重和后端健康检查。



参考链接

关于如何将容器Swarm集群向ACK集群灰度迁移，请参见容器Swarm集群向ACK集群灰度迁移。

12.7. 源服务器迁移至容器镜像

SMC支持将源服务器迁移到容器镜像服务，实现低成本容器化应用迁移。容器的优势在于提高了资源利用率，降低了计算成本，自动化管理调度及低风险的快速部署。本文介绍源服务器迁移至容器镜像的操作步骤。

前提条件

- 已确认迁移源的操作系统，Windows操作系统不支持迁移至容器镜像。

- 已开通容器镜像服务并创建镜像仓库。更多信息，请参见构建仓库与镜像。
- 已创建SMC中转实例所需的RAM角色，配置信息如下所示。具体操作，请参见创建可信实体为阿里云服务的RAM角色。
  - 可信实体类型选择**阿里云服务**。
  - 角色类型选择**普通服务角色**。
  - 受信服务选择**云服务器**。
- 已创建满足容器镜像迁移的自定义策略。自定义策略如下所示，并且已为RAM角色授权该策略。具体操作，请参见创建自定义权限策略及为RAM角色授权。

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cr:GetAuthorizationToken",
                "cr:PushRepository",
                "cr:PullRepository",
                "cr:CreateRepository"
            ],
            "Resource": [
                "*"
            ]
        }
    ],
    "Version": "1"
}
```

- 已将源服务器信息导入SMC控制台。具体操作，请参见步骤一：导入迁移源。

> **注意**
>
> - SMC客户端从2.3.0版本开始支持迁移至容器镜像，因此请使用2.3.0及以上版本客户端导入迁移源。点击下载新版本客户端。
> - 迁移任务执行过程中请保持SMC客户端处于运行状态。如果数据传输中断，重新运行客户端并重新启动迁移任务即可继续迁移。

背景信息

- Docker容器镜像服务的基础知识请参见基本概念。
- 迁移任务运行期间会创建中转实例辅助迁移。中转实例会产生少量的费用，计费详情请参见按量付费。
- 当迁移任务为**已完成**（Finished）状态、**已过期**（Expired）状态或迁移任务被删除时，中转实例会自动清理释放。

（可选）步骤一：过滤动态数据目录

为确保迁移更加稳定，建议您在迁移前，先排除动态数据目录（如大型数据库的数据目录等），等到业务暂停后再迁移。若无需过滤动态数据目录，可跳过本节步骤。

在源服务器系统业务不暂停的情况下，过滤掉源服务器系统的动态数据目录。具体步骤如下：

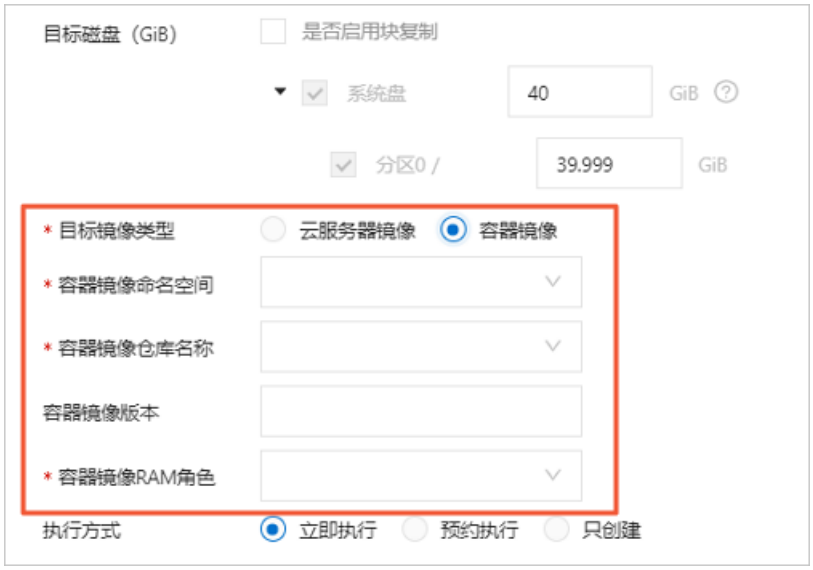
1. 登录源服务器。
2. 配置SMC客户端，排除动态数据目录。

具体操作，请参见排除不迁移的文件或目录。

步骤二：创建并启动迁移任务

在源服务器系统业务不暂停的情况下，通过SMC控制台创建并启动迁移任务。具体步骤如下：

1. 登录SMC控制台。
2. 在左侧导航栏，单击**迁移源**。
3.
4. 在**操作**列，单击**创建迁移任务**。
5. 在**创建迁移任务**页面，设置容器镜像相关配置项。



容器镜像相关配置项说明如下。 其他配置项的设置，请参见步骤二：创建并启动迁移任务中的配置项说明。

- **目标镜像类型**：选择**容器镜像**。
- **容器镜像命名空间**：表示存放迁移生成的容器镜像仓库的命名空间。
- **容器镜像仓库名称**：表示存放迁移生成的容器镜像的仓库地址。
- **容器镜像版本**：表示存放迁移生成的容器镜像的版本信息。
- **容器镜像RAM角色**：表示绑定中转实例的实例角色。

迁移任务创建后立即开始执行。执行结果如下：

- 当迁移任务状态为已完成（Finished），表示任务完成并得到最终的容器镜像。
- 当任务状态为**出错**（InError），表示任务失败。您需要查看日志修复问题后，再次重启迁移任务。常见错误及修复方案，请参见SMC FAQ。

步骤三：验证容器镜像

迁移成功获取到最终容器镜像后，您可以进行如下操作验证容器镜像。本操作以部署了Nginx环境的容器镜像为例。

1. 创建容器服务集群，详情请参见创建Kubernetes专有版集群。
2. 登录容器服务管理控制台。
3. 在左侧导航栏，单击**集群**，查看您已经创建好的容器服务集群。
4. 在相应容器服务集群的**操作**列，单击**应用管理**。

5. 在**无状态**页签内，单击**使用镜像创建**进行无状态应用的创建。

   本文中的配置如下所示。未说明的配置项及具体操作请参见创建无状态工作负载Deployment。

   - 在**应用基本信息**页面。配置如下基本信息。

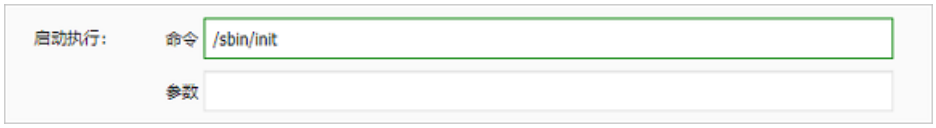


     - **应用名称**：示例值nginx。
     - **副本数量**：1。
     - **类型**：选择**无状态**。

   - 在**容器配置**页面。配置如下信息。
     - **镜像名称**：单击**选择镜像**。选择迁移生成的容器镜像（容器镜像仓库和容器服务集群在同一地域时，可以使用容器镜像的VPC地址拉取镜像）。
     - **镜像Tag**：单击**选择镜像Tag**。选择迁移生成的容器镜像Tag。
     - **设置镜像密钥**：如果您的容器镜像为私有镜像，则需要设置镜像密钥。也可以配置免密拉取，详情请参见使用免密组件拉取容器镜像。
     - **端口**：新增80端口。

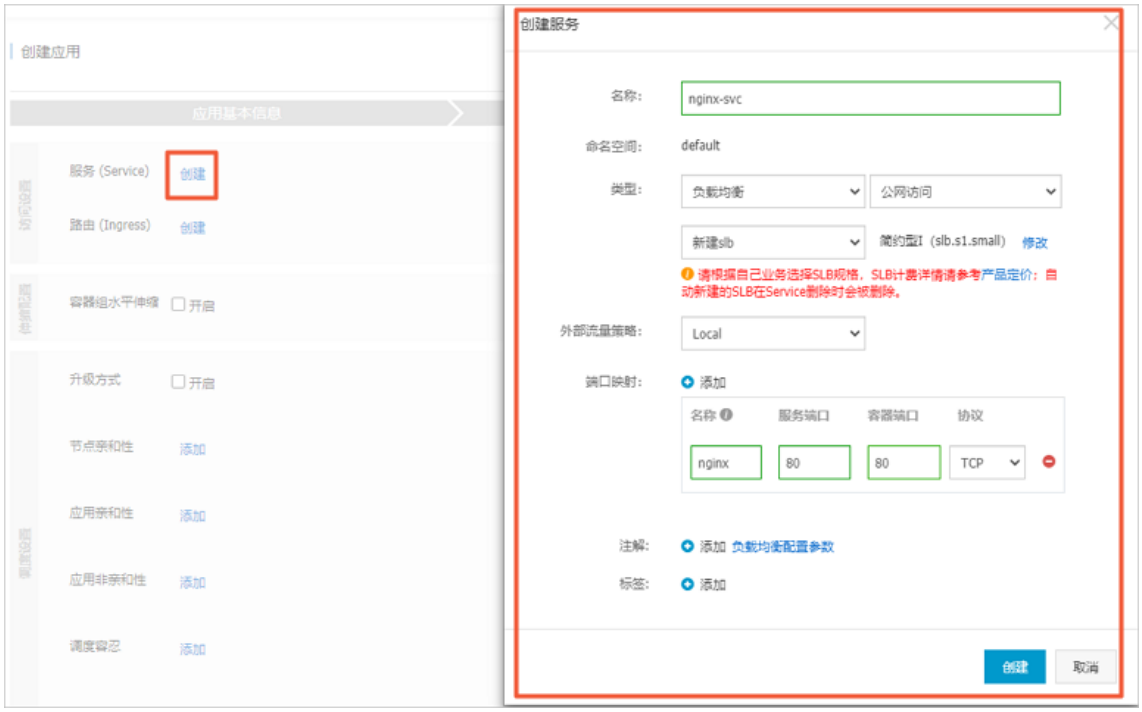


     - **启动执行**：设置命令/sbin/init。

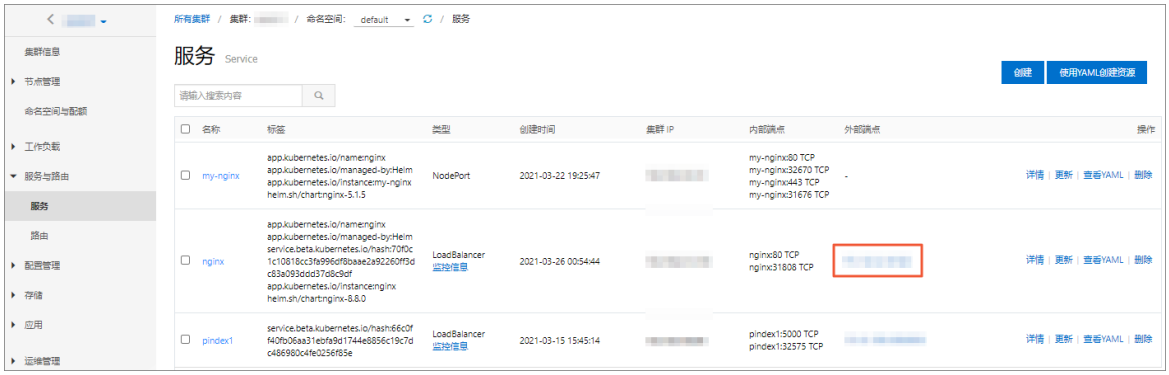


   - 在**高级配置**页面，创建服务用于访问应用。

本文配置信息如下图所示。



6. 创建完成后，单击**查看应用详情**。查看应用运行情况。
7. 在左侧导航栏，单击**服务**，查看服务列表的**外部端点**地址。



8. 通过本地浏览器访问外部端点地址。



12.8. 容器镜像迁移

12.8.1. 通过image-syncer工具迁移容器镜像

在Kubernetes集群迁移场景中，镜像仓库之间进行镜像迁移和同步是基本需求，image-syncer工具可以解决通用的容器镜像批量迁移和镜像同步复制的问题，将已有的容器镜像平滑的迁移到阿里云镜像服务ACR上。本文主要为您介绍如何通过image-syncer工具迁移容器镜像。

背景信息

阿里云上的容器服务ACK在使用成本、运维成本、方便性、长期稳定性上大大超过云厂商自建自维护Kubernetes集群，有不少云厂商纷纷想把之前自己维护Kubernetes负载迁移到阿里云ACK服务上。在迁移过程中，当镜像个数较少时，可以通过**docker pull**或**docker push**命令完成镜像迁移，如果涉及到成千上百个镜像，甚至几T的镜像仓库数据时，迁移过程就变得非常漫长，并且可能丢失数据。此时，用户在各种容器镜像仓库之间迁移时，期望有镜像同步复制的能力。阿里云开源image-syncer工具具备此能力并已经帮助多家云厂商成功迁移镜像，其中最大镜像仓库的总量达到3TB以上。同步任务时能跑满机器带宽，并且对进行同步任务的机器磁盘容量没有要求。

image-syncer简介

在Kubernetes集群迁移场景中，镜像仓库之间进行镜像迁移和镜像同步复制是基本需求，而使用**docker pull**或**docker push**结合脚本的传统方式进行镜像同步，有如下几个局限性：

- 依赖磁盘存储，需要及时进行本地镜像的清理，并且落盘造成多余的时间开销，难以胜任生产场景中大量镜像的迁移。
- 依赖Docker程序，Docker Daemon对Pull和Push的并发数进行了严格的限制，无法进行高并发同步。
- 一些功能只能通过HTTP API进行操作，仅使用Docker CLI无法做到，使脚本变得复杂。

image-syncer的定位是一个简单、易用的批量镜像迁移和镜像同步复制工具，支持几乎所有目前主流的基于Docker Registry V2搭建的镜像存储服务，例如ACR、Docker、Hub、Quay、自建Harbor等，目前已经初步经过了TB级别的生产环境镜像迁移验证，详情请参见image-syncer。

工具特点

image-syncer提供了以下几项支持：

- 支持多对多镜像仓库同步。
- 支持基于Docker Registry V2搭建的Docker镜像仓库服务
  例如，Docker Hub、Quay、 阿里云镜像服务ACR、Harbor等。
- 镜像同步复制只经过内存和网络，不依赖磁盘存储，同步速度快。
- 支持增量同步。
  通过对同步过的镜像blob信息落盘，不会对已同步的镜像进行重复同步。
- 支持并发同步。
  可以通过配置文件调整并发数。
- 支持自动重试失败的同步任务，解决大部分镜像同步中的网络抖动问题。
- 不依赖Docker以及其他程序。

通过使用image-syncer，只需要保证image-syncer的运行环境与需要同步的registry网络连通，您可以快速地完成从镜像仓库的迁移、复制以及增量同步，并且image-syncer对硬件资源几乎没有要求（因为image-syncer严格控制网络连接数目=并发数，所以只有在单个镜像层过大的情况下，并发数目过大可能会打满内存，内存占用 <= 并发数x最大镜像层大小）。除了使用重传机制规避同步过程中可能出现的偶发问题之外，image-syncer会在运行结束时统计最后同步失败的镜像个数，并且打印出详细的日志，帮助使用者定位同步过程中出现的问题。

准备工作

使用image-syncer时，您只需要提供一个配置文件，内容示例如下：

```
{
    "auth": {                   // 认证字段，其中每个对象为一个registry的一个账号和
                                // 密码；通常，同步源需要具有pull以及访问tags权限，
                                // 同步目标需要拥有push以及创建仓库权限，如果没有提供，则默认匿名访
问
        "quay.io": {            // registry的url，需要和下面images中对应registry的url相同
            "username": "xxx",              // 用户名，可选
            "password": "xxxxxxxxx",        // 密码，可选
            "insecure": true                // registry是否是http服务，如果是，insecure字段需
要为true，默认是false，可选，支持这个选项需要image-syncer版本 > v1.0.1
        },
        "registry.cn-beijing.aliyuncs.com": {
            "username": "xxx",
            "password": "xxxxxxxxx"
        },
        "registry.hub.docker.com": {
            "username": "xxx",
            "password": "xxxxxxxxxx"
        }
    },
    "images": {
        // 同步镜像规则字段，其中一条规则包括一个源仓库（键）和一个目标仓库（值）
        // 同步的最大单位是仓库（repo），不支持通过一条规则同步整个namespace以及registry
        // 源仓库和目标仓库的格式与docker pull/push命令使用的镜像url类似（registry/namespace/repo
sitory:tag）
        // 源仓库和目标仓库（如果目标仓库不为空字符串）都至少包含registry/namespace/repository
        // 源仓库字段不能为空，如果需要将一个源仓库同步到多个目标仓库需要配置多条规则
        // 目标仓库名可以和源仓库名不同（tag也可以不同），此时同步功能类似于：docker pull + docker
tag + docker push
        "quay.io/coreos/kube-rbac-proxy": "quay.io/ruohe/kube-rbac-proxy",
        "xxxx":"xxxxx",
        "xxx/xxx/xx:tag1,tag2,tag3":"xxx/xxx/xx"
        // 当源仓库字段中不包含tag时，表示将该仓库所有tag同步到目标仓库，此时目标仓库不能包含tag
        // 当源仓库字段中包含tag时，表示只同步源仓库中的一个tag到目标仓库，如果目标仓库中不包含tag，
则默认使用源tag
        // 源仓库字段中的tag可以同时包含多个（比如"a/b/c:1,2,3"），tag之间通过","隔开，此时目标仓库
不能包含tag，并且默认使用原来的tag
        // 当目标仓库为空字符串时，会将源镜像同步到默认registry的默认namespace下，并且repo以及tag与
源仓库相同，默认registry和默认namespace可以通过命令行参数以及环境变量配置，参考下面的描述
    }
}
```

使用示例

- 从自建Harbor同步镜像到ACR默认实例版
- 从自建Harbor同步镜像到ACR企业版

12.8.2. 从自建Harbor同步镜像到ACR默认实例版

ACR（Alibaba Cloud Container Registry）是阿里云提供的容器镜像托管服务，支持全球20个地域的镜像全生命周期管理，联合容器服务等云产品，打造云原生应用的一站式体验。本文将为您介绍将自建Harbor上的镜像同步到ACR默认实例版，提供image-synce的基本使用示例。

前提条件

您需要开通容器镜像服务。

登录容器镜像服务控制台开通相应的服务。

创建命名空间

命名空间可以有效管理其名下的仓库集合，包括仓库权限和仓库属性。如果需要同步的仓库不存在，需要在命名空间下打开**自动创建仓库**，并通过**docker push**自动创建仓库。

> **说明** 当目标仓库不存在时，docker push可以根据默认仓库类型自动创建的仓库类型是公有的还是私有。

1. 登录容器镜像服务控制台。
2. 单击左侧导航栏的**默认实例 > 命名空间**。
3. 在**命名空间**页面，单击右上角的**创建命名空间**。
4. 在**创建命名空间**页面中自定义命名空间，并单击**确定**。

创建完成后，您可以在**命名空间**列表中看到新创建的命名空间。同时可以对命名空间进行管理，请参见命名空间的基本操作。

授权子账号

如果您使用子账号进行后续操作，需要先创建RAM子账号并授予授权。如果您使用主账号进行后续操作，请跳过此步骤。

1. 创建RAM子账号。请参见创建RAM用户。
2. 授予子账号相应的权限。请参见自定义RAM授权策略。

   本示例中仅使用到创建、更新镜像仓库相关权限，访问控制的资源粒度为image-syncer命名空间，最小权限设置如下。

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cr:CreateRepository",
                "cr:UpdateRepository",
                "cr:PushRepository",
                "cr:PullRepository"
            ],
            "Resource": [
                "acs:cr:*:*:repository/image-syncer/*"
            ]
        }
    ],
    "Version": "1"
}
```

获取访问凭证

在拉取私有镜像或者上传镜像前，需要 `docker login` 输入您的凭证信息。请按照以下操作获取访问凭证。

1. 单击左侧导航栏的**默认实例 > 访问凭证**。
2. 在**访问凭证**页面单击**设置固定密码**。
3. 在**设置固定密码**页面填写**密码**和**确认密码**，并单击**确认**。

您可以通过API获取临时密码访问镜像服务实例，请参见GetAuthorizationToken。

image-syncer的同步配置

本例中，假设将本地搭建的Harbor中的*library/nginx*仓库同步到华北2的image-syncer命名空间下，并且保持仓库名称为nginx。配置文件示例如下。

```
{
    "auth": {
        "harbor.myk8s.paas.com:32080": {
            "username": "admin",
            "password": "xxxxxxxxx",
            "insecure": true
        },
        "registry.cn-beijing.aliyuncs.com": {
            "username": "acr_pusher@1938562138124787",
            "password": "xxxxxxxx"
        }
    },
    "images": {
        "harbor.myk8s.paas.com:32080/library/nginx": ""
    }
}
```

- `harbor.myk8s.paas.com:32080`：本地搭建的harbor访问地址。需要替换成实际值。
  - `username`：Harbor的用户名，本例中为admin。
  - `password`：Harbor的密码。
  - `insecure`：需要设置为true。
- `registry.cn-beijing.aliyuncs.com`：目标仓库访问地址。本例中假设为镜像所在的地域为华北2。
  - `username`：获取访问凭证的登录用户名。
  - `password`：获取访问凭证的创建的密码。
- `"harbor.myk8s.paas.com:32080/library/nginx": ""`：通过harbor.myk8s.paas.com:32080访问*library/nginx*仓库。

通过image-syncer同步镜像

1. 从image-syncer下载最新的安装包。

   > **说明** 目前只支持Linux amd64版本。

2. 安装和配置image-syncer工具。

   有关详细信息，请参见安装和配置image-syncer。

3. 执行如下命令同步镜像。

```
# 设置默认目标registry为registry.cn-beijing.aliyuncs.com，默认目标namespace为image-syncer
# 并发数为10，重试次数为10
# 日志输出到./log文件下，不存在会自动创建，不指定的话默认会将日志打印到Stderr
# 指定配置文件为harbor-to-acr.json，内容如上所述
./image-syncer --proc=10 --config=./harbor-to-acr.json --registry=registry.cn-beijing.aliyuncs.com --namespace=image-syncer --retries=10 --log=./log
```

执行结果

一次镜像同步会经历三个阶段：生成同步任务、执行同步任务以及重试失败任务。其中，每个同步任务都代表了一个需要同步的镜像（tag），如果配置文件中某条规则没有指定tag，在**生成同步任务**阶段会自动列出源仓库所有tag，并生成对应的同步任务，如果生成同步任务失败，也会在重试阶段进行重试。

- 同步镜像正常结果输出如下所示。

```
hhy@IT-C02YV1YJLVDL ~/go/src/github.com/AliyunContainerService/image-syncer master ./image-syncer --proc=10 --config=./harbor-to-acr.json --registry=registry.cn-beijing.aliyuncs.com --namespace=image-syncer --retries=10 --log=./log
Start to generate sync tasks, please wait ...
Start to handle sync tasks, please wait ...
Finished, 0 sync tasks failed, 0 tasks generate failed
```

- 同步镜像异常结果（例如，账号或密码错误）输出如下所示。

```
hhy@IT-C02YV1YJLVDL ~/go/src/github.com/AliyunContainerService/image-syncer master ./image-syncer --proc=10 --config=./harbor-to-acr.json --registry=registry.cn-beijing.aliyuncs.com --namespace=image-syncer --retries=10 --log=./log
Start to generate sync tasks, please wait ...
Start to handle sync tasks, please wait ...
Start to retry sync tasks, please wait ...
Start to retry sync tasks, please wait ...
Start to retry sync tasks, please wait ...
Start to retry sync tasks, please wait ...
Start to retry sync tasks, please wait ...
Start to retry sync tasks, please wait ...
Start to retry sync tasks, please wait ...
Start to retry sync tasks, please wait ...
Start to retry sync tasks, please wait ...
Finished, 54 sync tasks failed, 0 tasks generate failed
```

- 在运行时，image-syncer会打印出如下的日志信息。

```
INFO[0360] Put blob sha256:60807a6...0f68(21524425) to ruohe-test-registry.cn-shanghai.cr.aliyuncs.com/image-syncer/nginx:1.12.0 success
INFO[0360] Get a blob sha256:2c...36c4(31447638) from harbor.myk8s.paas.com:32080/library/nginx:1.13.0-perl success
INFO[0360] Get a blob sha256:55c9...7330b(194) from harbor.myk8s...library/nginx:1.12.0 success
INFO[0362] Put blob sha256:ff3d52...68(23275620) to ruohe-test-registry...ghai.cr.aliyuncs.com/image-syncer/nginx:1.13.0 success
INFO[0362] Put blob sha256:325b62...99(196) to ruohe-test-registry.cn-shanghai.cr.aliyuncs.com/image-syncer/nginx:1.11.7 success
INFO[0362] Put blob sha256:15ea7f...ee(31445058) to ruohe-test-registry.cn-shanghai.cr.aliyuncs.com/image-syncer/nginx:1.12.0-perl success
INFO[0362] Put blob sha256:abf312...a1(3703) to ruohe-test-registry.cn-shanghai.cr.aliyuncs.com/image-syncer/nginx:1.11.6 success
INFO[0362] Get a blob sha256:b0543...bcac2ec(21531803) from harbor.myk8s.paas.com:32080/library/nginx:1.13.0 success
INFO[0362] Put blob sha256:6973f...fc8(9010) to ruohe-test-registry.cn-shanghai.cr.aliyuncs.com/image-syncer/nginx:1.13.1-alpine-perl success
INFO[0362] Get a blob sha256:1914...0e9b5d(3702) from harbor.myk8s.paas.com:32080/library/nginx:1.11.7 success
INFO[0362] Get a blob sha256:292a...c03723(194) from harbor.myk8s.paas.com:32080/library/nginx:1.12.0-perl success
INFO[0362] Put blob sha256:a60696...453a(8762) to ruohe-test-registry.cn-shanghai.cr.aliyuncs.com/image-syncer/nginx:1.13.1-alpine success
INFO[0363] Put manifest to ruohe-test-registry.cn-shanghai.cr.image-syncer/ngiqe-syncer/nginx:1.13.1-alpine-perl
INFO[0363] Synchronization successfully from harbor.myk8s.paas.com:32080/library/nginx:1.13.1-alpine-perl to ruohe-test-registry.cn-shanghai.cr.aliyuncs.com/image-syncer/nginx:1.13.1-alpine-perl
```

12.8.3. 从自建Harbor同步镜像到ACR企业版

容器镜像服务企业版提供镜像极速导入和自定义域名等功能，可以短时间内迁移自建Harbor至容器镜像服务企业版。本文将为您介绍将自建Harbor上的镜像同步到ACR（Alibaba Cloud Container Registry）企业版。

背景信息

ACR企业版提供企业级容器镜像、Helm Chart安全托管能力，拥有企业级安全独享特性，具备千节点镜像分发、全球多地域同步能力。提供云原生应用交付链，实现一次应用变更，全球化多场景自动交付。主要适用于安全需求高、业务多地域部署、拥有大规模集群节点的企业级客户使用。

步骤一：迁移Harbor的后端存储数据

- 如果Harbor的后端存储为NAS，则需要先将NAS服务器上的数据迁移到阿里云OSS的Bucket中，详细介绍请参见NAS迁移至OSS教程。
- 如果Harbor的后端存储为阿里云OSS，请跳过此步骤。

步骤二：自定义OSS Bucket

容器镜像服务企业版创建实例时，支持自定义阿里云OSS Bucket作为企业版实例的后端存储。

1. 为账号添加RAM角色，并为该RAM角色授予对OSS Bucket的操作权限，详细介绍请参见配置使用自定义OSS Bucket时的RAM访问控制。
2. 创建企业版实例。

   创建企业版实例时，设置**实例存储**为**自定义**，并选择Bucket。详细介绍请参见创建企业版实例。

步骤三：导入镜像

1. 登录容器镜像服务控制台。
2. 在顶部菜单栏，选择所需地域。
3.
4. 在**实例列表**页面单击目标企业版实例。
5. 在企业版实例管理页左侧导航栏中选中**实例管理 > 镜像导入**。
6. 在镜像导入页面单击**触发导入**。
7. 在**提示**对话框中选中**确认以上信息，触发导入任务**，然后单击**确定**。

> **说明** 在镜像导入页面，单击对应的镜像导入任务**操作**列中的**详情**，可查看任务进度。

步骤四：自定义域名

容器镜像服务企业版支持自定义域名功能，该功能允许为容器镜像服务企业版实例添加自定义域名和相应的SSL证书，从而使用自定义域名并通过HTTPS协议来访问实例。

自定义容器镜像服务企业版实例的域名为Harbor使用的域名，详细介绍请参见通过自定义域名访问容器镜像服务企业版实例。

12.9. Kubernetes应用迁移

本文记录使用备份中心快速完成云原生应用及PV数据从自建Kubernetes迁移到阿里云Kubernetes集群的实践过程。 此过程也同样适用于其他云厂商Kubernetes集群内的应用及PV数据迁移至阿里云Kubernetes集群。

前提条件

- 自建Kubernetes集群版本要大于1.20。
- 自建Kubernetes集群可以访问公网，且需要开通HBR服务。更多信息，请参见混合云备份HBR。
- 自建Kubernetes集群通常位于用户自己的IDC中，容器镜像的存储也会使用自建镜像仓库，在自建Kubernetes应用迁移上云之前，您需要先将容器镜像迁移上云到ACR。具体操作，请参见容器镜像迁移。

操作步骤

1. 步骤一：准备迁移环境
2. 步骤二：在自建Kubernetes集群备份应用
3. 步骤三：在阿里云ACK集群恢复应用
4. 步骤四：更新应用配置

步骤一：准备迁移环境

请按照以下步骤，分别在自建Kubernetes集群和阿里云Kubernetes集群中部署备份中心。

1. 在自建Kubernetes集群中安装备份中心。

i. 使用以下内容，创建*csdr-controller.yaml*文件。

> **注意** 请替换YAML文件中的如下字段：
>
> - `{{.ClusterId}}`：替换为自建Kubernetes集群的ID，如果有多个集群需要保证每个集群的该值均不同。
> - `{{.Region}}`：替换为迁移目标ACK集群所在的地域，例如北京需要替换为`"cn-beijing"`，杭州需要替换成`"cn-hangzhou"`。
> - `{{.ACCESSKEY}}`和`{{.ACEESSKEYSECRET}}`替换为RAM用户账号的AK信息，该AK要拥有访问OSS及HBR服务的权限。更多信息，请参见（可选）步骤二：为专有版集群配置OSS权限。

查看csdr-controller的YAML文件

```
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.4.1
  creationTimestamp: null
  name: applicationbackups.csdr.alibabacloud.com
spec:
  group: csdr.alibabacloud.com
  names:
    kind: ApplicationBackup
    listKind: ApplicationBackupList
    plural: applicationbackups
    singular: applicationbackup
  scope: Namespaced
  versions:
  - name: v1beta1
    schema:
      openAPIV3Schema:
        description: ApplicationBackup is the Schema for the applicationbackups API
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation of an object. Servers should convert recognized schemas to the latest interna
l value, and may reject unrecognized values. More info: https://git.k8s.io/communit
y/contributors/devel/sig-architecture/api-conventions.md#resources'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s object represents. Servers may infer this from the endpoint the client submits re
quests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community
/contributors/devel/sig-architecture/api-conventions.md#types-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: ApplicationBackupSpec defines the desired state of Applica
tionBackup
            properties:
```

```
              properties:
                backupType:
                  description: BackupType means what the input backups such as OnlyAp
pBackup, AppAndPvBackup and OnlyPvBackup default value is OnlyAppBackup
                  type: string
                excludedNamespaces:
                  description: ExcludedNamespaces contains a list of namespaces that
are not included in the backup.
                  items:
                    type: string
                  nullable: true
                  type: array
                excludedResources:
                  description: ExcludedResources is a slice of resource names that ar
e not included in the backup.
                  items:
                    type: string
                  nullable: true
                  type: array
                includeClusterResources:
                  description: IncludeClusterResources specifies whether cluster-scop
ed resources should be included for consideration in the backup.
                  nullable: true
                  type: boolean
                includedNamespaces:
                  description: IncludedNamespaces is a slice of namespace names to in
clude objects from. If empty, all namespaces are included.
                  items:
                    type: string
                  nullable: true
                  type: array
                includedResources:
                  description: IncludedResources is a slice of resource names to incl
ude in the backup. If empty, all resources are included.
                  items:
                    type: string
                  nullable: true
                  type: array
                labelSelector:
                  description: LabelSelector is a metav1.LabelSelector to filter with
when adding individual objects to the backup. If empty or nil, all objects are incl
uded. Optional.
                  nullable: true
                  properties:
                    matchExpressions:
                      description: matchExpressions is a list of label selector requi
rements. The requirements are ANDed.
                      items:
                        description: A label selector requirement is a selector that
contains values, a key, and an operator that relates the key and values.
                        properties:
                          key:
                            description: key is the label key that the selector appli
es to.
                            type: string
```

```
                        operator:
                          description: operator represents a key's relationship to
a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
                          type: string
                        values:
                          description: values is an array of string values. If the
operator is In or NotIn, the values array must be non-empty. If the operator is Exi
sts or DoesNotExist, the values array must be empty. This array is replaced during
a strategic merge patch.
                          items:
                            type: string
                          type: array
                      required:
                      - key
                      - operator
                      type: object
                    type: array
                  matchLabels:
                    additionalProperties:
                      type: string
                    description: matchLabels is a map of {key,value} pairs. A singl
e {key,value} in the matchLabels map is equivalent to an element of matchExpression
s, whose key field is "key", the operator is "In", and the values array contains on
ly "value". The requirements are ANDed.
                    type: object
                type: object
              orderedResources:
                additionalProperties:
                  type: string
                description: OrderedResources specifies the backup order of resourc
es of specific Kind. The map key is the Kind name and value is a list of resource n
ames separeted by commas. Each resource name has format "namespace/resourcename".
For cluster resources, simply use "resourcename".
                nullable: true
                type: object
              pvBackup:
                description: PvBackupSpec includes whether backup volume data
                properties:
                  defaultPvBackup:
                    description: whether backup pod's pv, default value is false if
the value is true, then csdr-controller will select pv automatically which pod is u
sing
                    type: boolean
                  pvcList:
                    description: 'NamespacedPvcList means will backup pvc with snap
shot Notice: only disk snapshot support if pvc type is not disk, it will do nothing
'
                    items:
                      properties:
                        name:
                          type: string
                        namespace:
                          type: string
                      type: object
```

```
                    type: array
                  pvcSelector:
                    additionalProperties:
                      type: string
                    description: backup pv use label selector.If empty or nil, all
objects are included. Optional.
                    type: object
                  snapshotPostRule:
                    description: after executing pre-rule, it should unfreeze appli
cation. so user should config post-rule to unfreeze application.
                    type: string
                  snapshotPreRule:
                    description: when enabled application consistency, user should
config snapshot pre-rule to application, so it will take application consistent bac
kups of volume data. the rule will be executed before snapshot, and it will call ap
p freeze interface.
                    type: string
                  snapshotToRegion:
                    type: string
                  targetRegions:
                    description: with group snapshot, if this sets, the disk snapsh
ot will be synced to destination region
                    items:
                      type: string
                    type: array
                type: object
              storageLocation:
                description: StorageLocation is a string containing the name of a B
ackupStorageLocation where the backup should be stored.
                type: string
              ttl:
                description: TTL is a time.Duration-parseable string describing how
long the Backup should be retained for.
                type: string
            type: object
          status:
            description: ApplicationBackupStatus defines the observed state of Appl
icationBackup
            properties:
              completionTimestamp:
                description: CompletionTimestamp records the time a backup was comp
leted. Completion time is recorded even on failed backups. Completion time is recor
ded before uploading the backup object. The server's time is used for CompletionTim
estamps
                format: date-time
                nullable: true
                type: string
              errors:
                description: Errors is a count of all error messages that were gene
rated during execution of the backup.  The actual errors are in the backup's log fi
le in object storage.
                type: integer
              expiration:
                description: 'INSERT ADDITIONAL STATUS FIELD - define observed stat
```

```
e of cluster Important: Run "make" to regenerate code after modifying this file Exp
iration is when this Backup is eligible for garbage-collection.'
                format: date-time
                nullable: true
                type: string
              message:
                description: message records backup message info such as failed rea
son
                type: string
              phase:
                description: Phase is the current state of the Backup.
                enum:
                - New
                - FailedValidation
                - InProgress
                - Completed
                - PartiallyFailed
                - Failed
                - Deleting
                type: string
              resourceList:
                properties:
                  applicationResource:
                    properties:
                      completionTimestamp:
                        format: date-time
                        type: string
                      phase:
                        type: string
                      progress:
                        properties:
                          itemsBackedUp:
                            description: ItemsBackedUp is the number of items that
have actually been written to the backup tarball so far.
                            type: integer
                          totalItems:
                            type: integer
                        type: object
                      startTimestamp:
                        format: date-time
                        type: string
                    type: object
                  dataResource:
                    properties:
                      completionTimestamp:
                        format: date-time
                        type: string
                      phase:
                        type: string
                      pvcBackupInfo:
                        items:
                          properties:
                            backupId:
                              type: string
```

```
                              convertToStorageClassType:
                                type: string
                              dataType:
                                type: string
                              nameSpace:
                                type: string
                              pvcName:
                                type: string
                              backupInfo:
                                type: string
                            type: object
                          type: array
                        advancedVolumeSnapshotsInfo:
                          type: array
                          items:
                            properties:
                              namespace:
                                type: string
                              name:
                                type: string
                            type: object
                        startTimestamp:
                          format: date-time
                          type: string
                        status:
                          items:
                            properties:
                              progress:
                                type: string
                              pvcName:
                                type: string
                            type: object
                          type: array
                      type: object
                  type: object
                startTimestamp:
                  description: StartTimestamp records the time a backup was started.
Separate from CreationTimestamp, since that value changes on restores. The server's
time is used for StartTimestamps
                  format: date-time
                  nullable: true
                  type: string
                warnings:
                  description: Warnings is a count of all warning messages that were
generated during execution of the backup. The actual warnings are in the backup's l
og file in object storage.
                  type: integer
              type: object
          type: object
      served: true
      storage: true
      subresources:
        status: {}
status:
  acceptedNames:
```

```
    kind: ""
    plural: ""
  conditions: []
  storedVersions: []
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.4.1
  creationTimestamp: null
  name: applicationrestores.csdr.alibabacloud.com
spec:
  group: csdr.alibabacloud.com
  names:
    kind: ApplicationRestore
    listKind: ApplicationRestoreList
    plural: applicationrestores
    singular: applicationrestore
  scope: Namespaced
  versions:
  - name: v1beta1
    schema:
      openAPIV3Schema:
        description: ApplicationRestore is the Schema for the applicationrestores A
PI
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation of an object. Servers should convert recognized schemas to the latest interna
l value, and may reject unrecognized values. More info: https://git.k8s.io/communit
y/contributors/devel/sig-architecture/api-conventions.md#resources'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s object represents. Servers may infer this from the endpoint the client submits re
quests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community
/contributors/devel/sig-architecture/api-conventions.md#types-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: ApplicationRestoreSpec defines the desired state of Applic
ationRestore
            properties:
              backupName:
                description: BackupName is the unique name of the Velero backup to
restore from.
                type: string
              excludedNamespaces:
                description: ExcludedNamespaces contains a list of namespaces that
are not included in the restore.
                items:
                  type: string
```

```
                nullable: true
                type: array
              excludedResources:
                description: ExcludedResources is a slice of resource names that ar
e not included in the restore.
                items:
                  type: string
                nullable: true
                type: array
              includeClusterResources:
                description: IncludeClusterResources specifies whether cluster-scop
ed resources should be included for consideration in the restore. If null, defaults
to true.
                nullable: true
                type: boolean
              includedNamespaces:
                description: IncludedNamespaces is a slice of namespace names to in
clude objects from. If empty, all namespaces are included.
                items:
                  type: string
                nullable: true
                type: array
              includedResources:
                description: IncludedResources is a slice of resource names to incl
ude in the restore. If empty, all resources in the backup are included.
                items:
                  type: string
                nullable: true
                type: array
              labelSelector:
                description: LabelSelector is a metav1.LabelSelector to filter with
when restoring individual objects from the backup. If empty or nil, all objects are
included. Optional.
                nullable: true
                properties:
                  matchExpressions:
                    description: matchExpressions is a list of label selector requi
rements. The requirements are ANDed.
                    items:
                      description: A label selector requirement is a selector that
contains values, a key, and an operator that relates the key and values.
                      properties:
                        key:
                          description: key is the label key that the selector appli
es to.
                          type: string
                        operator:
                          description: operator represents a key's relationship to
a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
                          type: string
                        values:
                          description: values is an array of string values. If the
operator is In or NotIn, the values array must be non-empty. If the operator is Exi
sts or DoesNotExist, the values array must be empty. This array is replaced during
```

```
a strategic merge patch.
                          items:
                            type: string
                          type: array
                      required:
                      - key
                      - operator
                      type: object
                    type: array
                  matchLabels:
                    additionalProperties:
                      type: string
                    description: matchLabels is a map of {key,value} pairs. A singl
e {key,value} in the matchLabels map is equivalent to an element of matchExpression
s, whose key field is "key", the operator is "In", and the values array contains on
ly "value". The requirements are ANDed.
                    type: object
                type: object
              namespaceMapping:
                additionalProperties:
                  type: string
                description: NamespaceMapping is a map of source namespace names to
target namespace names to restore into. Any source namespaces not included in the m
ap will be restored into namespaces of the same name.
                type: object
              storageclassMapping:
                additionalProperties:
                  type: string
                description: storageclassMapping is a map of source storageclass na
me to target storageclass name.
                type: object
              restorePVs:
                description: RestorePVs specifies whether to restore all included P
Vs from snapshot (via the cloudprovider).
                nullable: true
                type: boolean
            required:
            - backupName
            type: object
          status:
            description: ApplicationRestoreStatus defines the observed state of App
licationRestore
            properties:
              completionTimestamp:
                format: date-time
                type: string
              message:
                description: message records restore message info such as failed re
ason
                type: string
              phase:
                description: 'INSERT ADDITIONAL STATUS FIELD - define observed stat
e of cluster Important: Run "make" to regenerate code after modifying this file'
                enum:
```

```
                - New
                - FailedValidation
                - InProgress
                - Completed
                - PartiallyFailed
                - Failed
                type: string
              startTimestamp:
                format: date-time
                type: string
              templateResource:
                properties:
                  completionTimestamp:
                    description: CompletionTimestamp records the time the restore o
peration was completed. Completion time is recorded even on failed restore. The ser
ver's time is used for StartTimestamps
                    format: date-time
                    nullable: true
                    type: string
                  errors:
                    description: Errors is a count of all error messages that were
generated during execution of the restore. The actual errors are stored in object s
torage.
                    type: integer
                  failureReason:
                    description: FailureReason is an error that caused the entire r
estore to fail.
                    type: string
                  phase:
                    description: 'INSERT ADDITIONAL STATUS FIELD - define observed
state of cluster Important: Run "make" to regenerate code after modifying this file
Phase is the current state of the Restore'
                    enum:
                    - New
                    - FailedValidation
                    - InProgress
                    - Completed
                    - PartiallyFailed
                    - Failed
                    type: string
                  startTimestamp:
                    description: StartTimestamp records the time the restore operat
ion was started. The server's time is used for StartTimestamps
                    format: date-time
                    nullable: true
                    type: string
                  validationErrors:
                    description: ValidationErrors is a slice of all validation erro
rs (if applicable)
                    items:
                      type: string
                    nullable: true
                    type: array
                  warnings:
```

```
                    description: Warnings is a count of all warning messages that w
ere generated during execution of the restore. The actual warnings are stored in ob
ject storage.
                    type: integer
                type: object
            type: object
        type: object
    served: true
    storage: true
    subresources:
      status: {}
status:
  acceptedNames:
    kind: ""
    plural: ""
  conditions: []
  storedVersions: []
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.4.1
  creationTimestamp: null
  name: backuplocations.csdr.alibabacloud.com
spec:
  group: csdr.alibabacloud.com
  names:
    kind: BackupLocation
    listKind: BackupLocationList
    plural: backuplocations
    singular: backuplocation
  scope: Namespaced
  versions:
  - name: v1beta1
    schema:
      openAPIV3Schema:
        description: BackupLocation is the Schema for the backuplocations API
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation of an object. Servers should convert recognized schemas to the latest interna
l value, and may reject unrecognized values. More info: https://git.k8s.io/communit
y/contributors/devel/sig-architecture/api-conventions.md#resources'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s object represents. Servers may infer this from the endpoint the client submits re
quests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community
/contributors/devel/sig-architecture/api-conventions.md#types-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: BackupLocationSpec defines the desired state of BackupLoca
```

```
            description: BackupLocationSpec defines the desired state of BackupLoca
tion
            properties:
              accessMode:
                description: AccessMode defines the permissions for the backup stor
age location.
                enum:
                - ReadOnly
                - ReadWrite
                type: string
              backupSyncPeriod:
                description: BackupSyncPeriod defines how frequently to sync backup
API objects from object storage. A value of 0 disables sync.
                nullable: true
                type: string
              config:
                additionalProperties:
                  type: string
                description: Config is for provider-specific configuration fields.
                type: object
              objectStorage:
                description: ObjectStorageLocation specifies the settings necessary
to connect to a provider's object storage.
                properties:
                  bucket:
                    description: Bucket is the bucket to use for object storage.
                    type: string
                  caCert:
                    description: CACert defines a CA bundle to use when verifying T
LS connections to the provider.
                    format: byte
                    type: string
                  prefix:
                    description: Prefix is the path inside a bucket to use for Vele
ro storage. Optional.
                    type: string
                required:
                - bucket
                type: object
              provider:
                description: 'INSERT ADDITIONAL SPEC FIELDS - desired state of clus
ter Important: Run "make" to regenerate code after modifying this file Provider is
the provider of the backup storage.'
                type: string
              validationFrequency:
                description: ValidationFrequency defines how frequently to validate
the corresponding object storage. A value of 0 disables validation.
                nullable: true
                type: string
            required:
            - objectStorage
            - provider
            type: object
          status:
            description: BackupLocationStatus defines the observed state of BackupL
```

```
ocation
            properties:
              accessMode:
                description: "AccessMode is an unused field. \n Deprecated: there i
s now an AccessMode field on the Spec and this field will be removed entirely as of
v2.0."
                enum:
                - ReadOnly
                - ReadWrite
                type: string
              lastSyncedRevision:
                description: "LastSyncedRevision is the value of the `metadata/revi
sion` file in the backup storage location the last time the BSL's contents were syn
ced into the cluster. \n Deprecated: this field is no longer updated or used for de
tecting changes to the location's contents and will be removed entirely in v2.0."
                type: string
              lastSyncedTime:
                description: LastSyncedTime is the last time the contents of the lo
cation were synced into the cluster.
                format: date-time
                nullable: true
                type: string
              lastValidationTime:
                description: LastValidationTime is the last time the backup store l
ocation was validated the cluster.
                format: date-time
                nullable: true
                type: string
              message:
                description: 'INSERT ADDITIONAL STATUS FIELD - define observed stat
e of cluster Important: Run "make" to regenerate code after modifying this file'
                type: string
              phase:
                description: 'INSERT ADDITIONAL STATUS FIELD - define observed stat
e of cluster Important: Run "make" to regenerate code after modifying this file Pha
se is the current state of the BackupStorageLocation.'
                enum:
                - Available
                - Unavailable
                - Inprogress
                type: string
            type: object
        type: object
    served: true
    storage: true
    subresources:
      status: {}
status:
  acceptedNames:
    kind: ""
    plural: ""
  conditions: []
  storedVersions: []
---
```

```
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.4.1
  creationTimestamp: null
  name: backupschedules.csdr.alibabacloud.com
spec:
  group: csdr.alibabacloud.com
  names:
    kind: BackupSchedule
    listKind: BackupScheduleList
    plural: backupschedules
    singular: backupschedule
  scope: Namespaced
  versions:
  - name: v1beta1
    schema:
      openAPIV3Schema:
        description: BackupSchedule is the Schema for the backupschedules API
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation of an object. Servers should convert recognized schemas to the latest interna
l value, and may reject unrecognized values. More info: https://git.k8s.io/communit
y/contributors/devel/sig-architecture/api-conventions.md#resources'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s object represents. Servers may infer this from the endpoint the client submits re
quests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community
/contributors/devel/sig-architecture/api-conventions.md#types-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: BackupScheduleSpec defines the desired state of BackupSche
dule
            properties:
              schedule:
                description: Schedule is a Cron expression defining when to run the
Backup.
                type: string
              template:
                description: 'INSERT ADDITIONAL SPEC FIELDS - desired state of clus
ter Important: Run "make" to regenerate code after modifying this file'
                properties:
                  backupType:
                    description: BackupType means what the input backups such as On
lyAppBackup, AppAndPvBackup and OnlyPvBackup default value is OnlyAppBackup
                    type: string
                  excludedNamespaces:
                    description: ExcludedNamespaces contains a list of namespaces t
hat are not included in the backup.
```

```
                    items:
                      type: string
                    nullable: true
                    type: array
                  excludedResources:
                    description: ExcludedResources is a slice of resource names tha
t are not included in the backup.
                    items:
                      type: string
                    nullable: true
                    type: array
                  includeClusterResources:
                    description: IncludeClusterResources specifies whether cluster-
scoped resources should be included for consideration in the backup.
                    nullable: true
                    type: boolean
                  includedNamespaces:
                    description: IncludedNamespaces is a slice of namespace names t
o include objects from. If empty, all namespaces are included.
                    items:
                      type: string
                    nullable: true
                    type: array
                  includedResources:
                    description: IncludedResources is a slice of resource names to
include in the backup. If empty, all resources are included.
                    items:
                      type: string
                    nullable: true
                    type: array
                  labelSelector:
                    description: LabelSelector is a metav1.LabelSelector to filter
with when adding individual objects to the backup. If empty or nil, all objects are
included. Optional.
                    nullable: true
                    properties:
                      matchExpressions:
                        description: matchExpressions is a list of label selector r
equirements. The requirements are ANDed.
                        items:
                          description: A label selector requirement is a selector t
hat contains values, a key, and an operator that relates the key and values.
                          properties:
                            key:
                              description: key is the label key that the selector a
pplies to.
                              type: string
                            operator:
                              description: operator represents a key's relationship
to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
                              type: string
                            values:
                              description: values is an array of string values. If
the operator is In or NotIn, the values array must be non-empty. If the operator is
```

```
Exists or DoesNotExist, the values array must be empty. This array is replaced duri
ng a strategic merge patch.
                              items:
                                type: string
                              type: array
                            required:
                            - key
                            - operator
                            type: object
                          type: array
                        matchLabels:
                          additionalProperties:
                            type: string
                          description: matchLabels is a map of {key,value} pairs. A s
ingle {key,value} in the matchLabels map is equivalent to an element of matchExpres
sions, whose key field is "key", the operator is "In", and the values array contain
s only "value". The requirements are ANDed.
                          type: object
                      type: object
                    orderedResources:
                      additionalProperties:
                        type: string
                      description: OrderedResources specifies the backup order of res
ources of specific Kind. The map key is the Kind name and value is a list of resour
ce names separeted by commas. Each resource name has format "namespace/resourcename
".  For cluster resources, simply use "resourcename".
                      nullable: true
                      type: object
                    pvBackup:
                      description: PvBackupSpec includes whether backup volume data
                      properties:
                        defaultPvBackup:
                          description: whether backup pod's pv, default value is fals
e if the value is true, then csdr-controller will select pv automatically which pod
is using
                          type: boolean
                        pvcList:
                          description: 'NamespacedPvcList means will backup pvc with
snapshot Notice: only disk snapshot support if pvc type is not disk, it will do not
hing'
                          items:
                            properties:
                              name:
                                type: string
                              namespace:
                                type: string
                            type: object
                          type: array
                        pvcSelector:
                          additionalProperties:
                            type: string
                          description: backup pv use label selector.If empty or nil,
all objects are included. Optional.
                          type: object
                        snapshotPostPule:
```

```
                      snapshotPostRule:
                        description: after executing pre-rule, it should unfreeze a
pplication. so user should config post-rule to unfreeze application.
                        type: string
                      snapshotPreRule:
                        description: when enabled application consistency, user sho
uld config snapshot pre-rule to application, so it will take application consistent
backups of volume data. the rule will be executed before snapshot, and it will call
app freeze interface.
                        type: string
                      snapshotToRegion:
                        type: string
                      targetRegions:
                        description: with group snapshot, if this sets, the disk sn
apshot will be synced to destination region
                        items:
                          type: string
                        type: array
                    type: object
                  storageLocation:
                    description: StorageLocation is a string containing the name of
a BackupStorageLocation where the backup should be stored.
                    type: string
                  ttl:
                    description: TTL is a time.Duration-parseable string describing
how long the Backup should be retained for.
                    type: string
                type: object
              required:
              - schedule
              - template
              type: object
            status:
              description: BackupScheduleStatus defines the observed state of BackupS
chedule
              properties:
                lastBackup:
                  description: LastBackup is the last time a Backup was run for this
Schedule schedule
                  format: date-time
                  nullable: true
                  type: string
                lastProcessedTime:
                  format: date-time
                  type: string
                phase:
                  description: 'INSERT ADDITIONAL STATUS FIELD - define observed stat
e of cluster Important: Run "make" to regenerate code after modifying this file Pha
se is the current phase of the Schedule'
                  type: string
                validationErrors:
                  description: ValidationErrors is a slice of all validation errors (
if applicable)
                  items:
                    type: string
```

```
                type: array
            type: object
        type: object
    served: true
    storage: true
    subresources:
      status: {}
status:
  acceptedNames:
    kind: ""
    plural: ""
  conditions: []
  storedVersions: []
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.4.1
  creationTimestamp: null
  name: converttosnapshots.csdr.alibabacloud.com
spec:
  group: csdr.alibabacloud.com
  names:
    kind: ConvertToSnapshot
    listKind: ConvertToSnapshotList
    plural: converttosnapshots
    singular: converttosnapshot
  scope: Namespaced
  versions:
  - name: v1beta1
    schema:
      openAPIV3Schema:
        description: ConvertToSnapshot is the Schema for the converttosnapshots API
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation of an object. Servers should convert recognized schemas to the latest interna
l value, and may reject unrecognized values. More info: https://git.k8s.io/communit
y/contributors/devel/sig-architecture/api-conventions.md#resources'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s object represents. Servers may infer this from the endpoint the client submits re
quests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community
/contributors/devel/sig-architecture/api-conventions.md#types-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: ConvertToSnapshotSpec defines the desired state of Convert
ToSnapshot
            properties:
              backupName:
```

```
                type: string
              convertedarg:
                items:
                  properties:
                    convertToStorageClassType:
                      type: string
                    namespace:
                      type: string
                    persistentVolumeClaim:
                      type: string
                  required:
                  - namespace
                  - persistentVolumeClaim
                  type: object
                type: array
            required:
            - backupName
            - convertedarg
            type: object
          status:
            description: ConvertToSnapshotStatus defines the observed state of Conv
ertToSnapshot
            properties:
              completionTimestamp:
                format: date-time
                type: string
              message:
                type: string
              phase:
                description: ConvertPhase is a string representation of the lifecyc
le phase of a Velero backup.
                enum:
                - ConvertionNew
                - ConvertionInProgress
                - ConvertionCompleted
                - ConvertionFailed
                type: string
              pvcConvertStatus:
                additionalProperties:
                  type: string
                type: object
              startTimestamp:
                format: date-time
                type: string
            type: object
        type: object
    served: true
    storage: true
    subresources:
      status: {}
status:
  acceptedNames:
    kind: ""
    plural: ""
```

```
  conditions: []
  storedVersions: []
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.4.1
  creationTimestamp: null
  name: deleterequests.csdr.alibabacloud.com
spec:
  group: csdr.alibabacloud.com
  names:
    kind: DeleteRequest
    listKind: DeleteRequestList
    plural: deleterequests
    singular: deleterequest
  scope: Namespaced
  versions:
  - name: v1beta1
    schema:
      openAPIV3Schema:
        description: DeleteRequest is the Schema for the deleterequests API
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation of an object. Servers should convert recognized schemas to the latest interna
l value, and may reject unrecognized values. More info: https://git.k8s.io/communit
y/contributors/devel/sig-architecture/api-conventions.md#resources'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s object represents. Servers may infer this from the endpoint the client submits re
quests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community
/contributors/devel/sig-architecture/api-conventions.md#types-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: DeleteRequestSpec defines the desired state of DeleteReque
st
            properties:
              backupName:
                description: Foo is an example field of DeleteRequest. Edit DeleteR
equest_types.go to remove/update
                type: string
            required:
            - backupName
            type: object
          status:
            description: DeleteRequestStatus defines the observed state of DeleteRe
quest
            properties:
              completionTimestamp:
```

```
                format: date-time
                type: string
              phase:
                description: DeleteBackupRequestPhase represents the lifecycle phas
e of a DeleteBackupRequest.
                enum:
                - New
                - InProgress
                - Processed
                type: string
              startTimestamp:
                description: 'INSERT ADDITIONAL STATUS FIELD - define observed stat
e of cluster Important: Run "make" to regenerate code after modifying this file'
                format: date-time
                type: string
            type: object
        type: object
    served: true
    storage: true
    subresources:
      status: {}
status:
  acceptedNames:
    kind: ""
    plural: ""
  conditions: []
  storedVersions: []
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  creationTimestamp: null
  name: manager-role
rules:
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - applicationbackups
  verbs:
  - create
  - delete
  - get
  - list
  - patch
  - update
  - watch
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - applicationbackups/finalizers
  verbs:
  - update
- apiGroups:
  - csdr.alibabacloud.com
  resources:
```

```
  resources:
  - applicationbackups/status
  verbs:
  - get
  - patch
  - update
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - applicationrestores
  verbs:
  - create
  - delete
  - get
  - list
  - patch
  - update
  - watch
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - applicationrestores/finalizers
  verbs:
  - update
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - applicationrestores/status
  verbs:
  - get
  - patch
  - update
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - backuplocations
  verbs:
  - create
  - delete
  - get
  - list
  - patch
  - update
  - watch
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - backuplocations/finalizers
  verbs:
  - update
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - backuplocations/status
  verbs:
```

```
  - get
  - patch
  - update
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - backupschedules
  verbs:
  - create
  - delete
  - get
  - list
  - patch
  - update
  - watch
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - backupschedules/finalizers
  verbs:
  - update
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - backupschedules/status
  verbs:
  - get
  - patch
  - update
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - converttosnapshots
  verbs:
  - create
  - delete
  - get
  - list
  - patch
  - update
  - watch
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - converttosnapshots/finalizers
  verbs:
  - update
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - converttosnapshots/status
  verbs:
  - get
  - patch
```

```
  - update
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - deleterequests
  verbs:
  - create
  - delete
  - get
  - list
  - patch
  - update
  - watch
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - deleterequests/finalizers
  verbs:
  - update
- apiGroups:
  - csdr.alibabacloud.com
  resources:
  - deleterequests/status
  verbs:
  - get
  - patch
  - update
---
apiVersion: v1
kind: ServiceAccount
metadata:
  namespace: csdr
  name: csdr
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  labels:
    component: csdr
  name: csdr-rolebinding
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
  - kind: ServiceAccount
    name: csdr
    namespace: csdr
---
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    control-plane: csdr-controller
```

```
  name: csdr-controller
  namespace: csdr
spec:
  replicas: 1
  selector:
    matchLabels:
      control-plane: csdr-controller
  template:
    metadata:
      labels:
        control-plane: csdr-controller
    spec:
      containers:
      - env:
        - name: IS_HYBRID
          value: "true"
        - name: USE_ADDON_TOKEN
          value: "false"
        - name: CLUSTER_ID
          value: {{.ClusterId}}
        - name: REGION_ID
          value: {{.Region}}
        - name: ALIBABA_CLOUD_ACCESS_KEY_ID
          value: {{.ACCESSKEY}}
        - name: ALIBABA_CLOUD_ACCESS_KEY_SECRET
          value: {{.ACEESSKEYSECRET}}
        image: registry.{{.Region}}.aliyuncs.com/acs/csdr-controller:v1.2.7-543cf40
-aliyun
        imagePullPolicy: Always
        livenessProbe:
          httpGet:
            path: /healthz
            port: 8191
          initialDelaySeconds: 15
          periodSeconds: 20
        name: manager
        readinessProbe:
          httpGet:
            path: /readyz
            port: 8191
          initialDelaySeconds: 5
          periodSeconds: 10
        resources:
          limits:
            cpu: 1000m
            memory: 1000Mi
          requests:
            cpu: 500m
            memory: 200Mi
        securityContext:
          allowPrivilegeEscalation: false
          readOnlyRootFilesystem: true
          runAsNonRoot: true
        volumeMounts:
```

```
        - mountPath: /var/csdr/config
          name: csdr-config-mnt
        - mountPath: /tmp/
          name: writeable
      serviceAccountName: csdr
      terminationGracePeriodSeconds: 10
      volumes:
      - configMap:
          name: csdr-config
        name: csdr-config-mnt
      - emptyDir: {}
        name: writeable
```

ii. 执行以下命令，部署csdr-controller。

```
kubectl apply -f csdr-controller.yaml
```

iii. 使用以下内容，创建*velero-deploy.yaml*文件。

查看velero-deploy的YAML文件

```
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.7.0
  creationTimestamp: null
  name: backups.velero.io
spec:
  group: velero.io
  names:
    kind: Backup
    listKind: BackupList
    plural: backups
    singular: backup
  scope: Namespaced
  versions:
  - name: v1
    schema:
      openAPIV3Schema:
        description: Backup is a Velero resource that represents the capture of Kub
ernetes
          cluster state at a point in time (API objects and associated volume state
).
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation
              of an object. Servers should convert recognized schemas to the latest
              internal value, and may reject unrecognized values. More info: https:
//git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resou
rces'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
```

```
s
              object represents. Servers may infer this from the endpoint the clien
t
              submits requests to. Cannot be updated. In CamelCase. More info: http
s://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#typ
es-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: BackupSpec defines the specification for a Velero backup.
            properties:
              defaultVolumesToRestic:
                description: DefaultVolumesToRestic specifies whether restic should
                  be used to take a backup of all pod volumes by default.
                type: boolean
              excludedNamespaces:
                description: ExcludedNamespaces contains a list of namespaces that
                  are not included in the backup.
                items:
                  type: string
                nullable: true
                type: array
              excludedResources:
                description: ExcludedResources is a slice of resource names that ar
e
                  not included in the backup.
                items:
                  type: string
                nullable: true
                type: array
              hooks:
                description: Hooks represent custom behaviors that should be execut
ed
                  at different phases of the backup.
                properties:
                  resources:
                    description: Resources are hooks that should be executed when
                      backing up individual instances of a resource.
                    items:
                      description: BackupResourceHookSpec defines one or more Backu
pResourceHooks
                        that should be executed based on the rules defined for name
spaces,
                        resources, and label selector.
                      properties:
                        excludedNamespaces:
                          description: ExcludedNamespaces specifies the namespaces
                            to which this hook spec does not apply.
                          items:
                            type: string
                          nullable: true
                          type: array
                        excludedResources:
```

```
                        description: ExcludedResources specifies the resources to
                          which this hook spec does not apply.
                        items:
                          type: string
                        nullable: true
                        type: array
                      includedNamespaces:
                        description: IncludedNamespaces specifies the namespaces
                          to which this hook spec applies. If empty, it applies
                          to all namespaces.
                        items:
                          type: string
                        nullable: true
                        type: array
                      includedResources:
                        description: IncludedResources specifies the resources to
                          which this hook spec applies. If empty, it applies to
                          all resources.
                        items:
                          type: string
                        nullable: true
                        type: array
                      labelSelector:
                        description: LabelSelector, if specified, filters the res
ources
                          to which this hook spec applies.
                        nullable: true
                        properties:
                          matchExpressions:
                            description: matchExpressions is a list of label sele
ctor
                              requirements. The requirements are ANDed.
                            items:
                              description: A label selector requirement is a sele
ctor
                                that contains values, a key, and an operator that
                                relates the key and values.
                              properties:
                                key:
                                  description: key is the label key that the sele
ctor
                                    applies to.
                                  type: string
                                operator:
                                  description: operator represents a key's relati
onship
                                    to a set of values. Valid operators are In,
                                    NotIn, Exists and DoesNotExist.
                                  type: string
                                values:
                                  description: values is an array of string value
s.
                                    If the operator is In or NotIn, the values ar
ray
```

```
                                    must be non-empty. If the operator is Exists
                                    or DoesNotExist, the values array must be emp
ty.
                                    This array is replaced during a strategic mer
ge
                                    patch.
                                  items:
                                    type: string
                                  type: array
                                required:
                                - key
                                - operator
                                type: object
                              type: array
                            matchLabels:
                              additionalProperties:
                                type: string
                              description: matchLabels is a map of {key,value} pair
s.
                                A single {key,value} in the matchLabels map is equi
valent
                                to an element of matchExpressions, whose key field
                                is "key", the operator is "In", and the values arra
y
                                contains only "value". The requirements are ANDed.
                              type: object
                          type: object
                        name:
                          description: Name is the name of this hook.
                          type: string
                        post:
                          description: PostHooks is a list of BackupResourceHooks
                            to execute after storing the item in the backup. These
                            are executed after all "additional items" from item act
ions
                            are processed.
                          items:
                            description: BackupResourceHook defines a hook for a re
source.
                            properties:
                              exec:
                                description: Exec defines an exec hook.
                                properties:
                                  command:
                                    description: Command is the command and argumen
ts
                                      to execute.
                                    items:
                                      type: string
                                    minItems: 1
                                    type: array
                                  container:
                                    description: Container is the container in the
                                      pod where the command should be executed. If
                                      not specified, the pod's first container is
```

```
                                    not specified, the pod's first container is
                                      used.
                                    type: string
                                  onError:
                                    description: OnError specifies how Velero shoul
d
                                      behave if it encounters an error executing th
is
                                      hook.
                                    enum:
                                    - Continue
                                    - Fail
                                    type: string
                                  timeout:
                                    description: Timeout defines the maximum amount
                                      of time Velero should wait for the hook to co
mplete
                                      before considering the execution a failure.
                                    type: string
                                required:
                                - command
                                type: object
                              required:
                              - exec
                              type: object
                            type: array
                          pre:
                            description: PreHooks is a list of BackupResourceHooks to
                              execute prior to storing the item in the backup. These
                              are executed before any "additional items" from item ac
tions
                              are processed.
                            items:
                              description: BackupResourceHook defines a hook for a re
source.
                              properties:
                                exec:
                                  description: Exec defines an exec hook.
                                  properties:
                                    command:
                                      description: Command is the command and argumen
ts
                                        to execute.
                                      items:
                                        type: string
                                      minItems: 1
                                      type: array
                                    container:
                                      description: Container is the container in the
                                        pod where the command should be executed. If
                                        not specified, the pod's first container is
                                        used.
                                      type: string
                                    onError:
                                      description: OnError specifies how Velero shoul
```

```
                                    description: OnError specifies how Velero should
d
                                      behave if it encounters an error executing th
is
                                      hook.
                                    enum:
                                    - Continue
                                    - Fail
                                    type: string
                                  timeout:
                                    description: Timeout defines the maximum amount
                                      of time Velero should wait for the hook to co
mplete
                                      before considering the execution a failure.
                                    type: string
                                required:
                                - command
                                type: object
                              required:
                              - exec
                              type: object
                            type: array
                        required:
                        - name
                        type: object
                      nullable: true
                      type: array
                  type: object
                includeClusterResources:
                  description: IncludeClusterResources specifies whether cluster-scop
ed
                    resources should be included for consideration in the backup.
                  nullable: true
                  type: boolean
                includedNamespaces:
                  description: IncludedNamespaces is a slice of namespace names to in
clude
                    objects from. If empty, all namespaces are included.
                  items:
                    type: string
                  nullable: true
                  type: array
                includedResources:
                  description: IncludedResources is a slice of resource names to incl
ude
                    in the backup. If empty, all resources are included.
                  items:
                    type: string
                  nullable: true
                  type: array
                labelSelector:
                  description: LabelSelector is a metav1.LabelSelector to filter with
                    when adding individual objects to the backup. If empty or nil, al
l
                    objects are included. Optional.
```

```
                nullable: true
                properties:
                  matchExpressions:
                    description: matchExpressions is a list of label selector requi
rements.
                      The requirements are ANDed.
                    items:
                      description: A label selector requirement is a selector that
                        contains values, a key, and an operator that relates the ke
y
                        and values.
                      properties:
                        key:
                          description: key is the label key that the selector appli
es
                            to.
                          type: string
                        operator:
                          description: operator represents a key's relationship to
                            a set of values. Valid operators are In, NotIn, Exists
                            and DoesNotExist.
                          type: string
                        values:
                          description: values is an array of string values. If the
                            operator is In or NotIn, the values array must be non-e
mpty.
                            If the operator is Exists or DoesNotExist, the values
                            array must be empty. This array is replaced during a st
rategic
                            merge patch.
                          items:
                            type: string
                          type: array
                      required:
                      - key
                      - operator
                      type: object
                    type: array
                  matchLabels:
                    additionalProperties:
                      type: string
                    description: matchLabels is a map of {key,value} pairs. A singl
e
                      {key,value} in the matchLabels map is equivalent to an elemen
t
                      of matchExpressions, whose key field is "key", the operator
                      is "In", and the values array contains only "value". The requ
irements
                      are ANDed.
                    type: object
                type: object
              metadata:
                properties:
                  labels:
```

```
                    additionalProperties:
                      type: string
                    type: object
                type: object
              orderedResources:
                additionalProperties:
                  type: string
                description: OrderedResources specifies the backup order of resourc
es
                  of specific Kind. The map key is the Kind name and value is a lis
t
                  of resource names separated by commas. Each resource name has for
mat
                  "namespace/resourcename".  For cluster resources, simply use "res
ourcename".
                nullable: true
                type: object
              snapshotVolumes:
                description: SnapshotVolumes specifies whether to take cloud snapsh
ots
                  of any PV's referenced in the set of objects included in the Back
up.
                nullable: true
                type: boolean
              storageLocation:
                description: StorageLocation is a string containing the name of a
                  BackupStorageLocation where the backup should be stored.
                type: string
              ttl:
                description: TTL is a time.Duration-parseable string describing how
                  long the Backup should be retained for.
                type: string
              volumeSnapshotLocations:
                description: VolumeSnapshotLocations is a list containing names of
                  VolumeSnapshotLocations associated with this backup.
                items:
                  type: string
                type: array
            type: object
          status:
            description: BackupStatus captures the current status of a Velero backu
p.
            properties:
              completionTimestamp:
                description: CompletionTimestamp records the time a backup was comp
leted.
                  Completion time is recorded even on failed backups. Completion ti
me
                  is recorded before uploading the backup object. The server's time
                  is used for CompletionTimestamps
                format: date-time
                nullable: true
                type: string
              errors:
```

```
                  description: Errors is a count of all error messages that were gene
rated
                    during execution of the backup.  The actual errors are in the bac
kup's
                    log file in object storage.
                  type: integer
                expiration:
                  description: Expiration is when this Backup is eligible for garbage
-collection.
                  format: date-time
                  nullable: true
                  type: string
                formatVersion:
                  description: FormatVersion is the backup format version, including
                    major, minor, and patch version.
                  type: string
                phase:
                  description: Phase is the current state of the Backup.
                  enum:
                  - New
                  - FailedValidation
                  - InProgress
                  - Completed
                  - PartiallyFailed
                  - Failed
                  - Deleting
                  type: string
                progress:
                  description: Progress contains information about the backup's execu
tion
                    progress. Note that this information is best-effort only -- if Ve
lero
                    fails to update it during a backup for any reason, it may be inac
curate/stale.
                  nullable: true
                  properties:
                    itemsBackedUp:
                      description: ItemsBackedUp is the number of items that have act
ually
                        been written to the backup tarball so far.
                      type: integer
                    totalItems:
                      description: TotalItems is the total number of items to be back
ed
                        up. This number may change throughout the execution of the ba
ckup
                        due to plugins that return additional related items to back
                        up, the velero.io/exclude-from-backup label, and various othe
r
                        filters that happen as items are processed.
                      type: integer
                  type: object
                startTimestamp:
                  description: StartTimestamp records the time a backup was started.
```

```
                  Separate from CreationTimestamp, since that value changes on rest
ores.
                  The server's time is used for StartTimestamps
                format: date-time
                nullable: true
                type: string
              validationErrors:
                description: ValidationErrors is a slice of all validation errors
                  (if applicable).
                items:
                  type: string
                nullable: true
                type: array
              version:
                description: 'Version is the backup format major version. Deprecate
d:
                  Please see FormatVersion'
                type: integer
              volumeSnapshotsAttempted:
                description: VolumeSnapshotsAttempted is the total number of attemp
ted
                  volume snapshots for this backup.
                type: integer
              volumeSnapshotsCompleted:
                description: VolumeSnapshotsCompleted is the total number of succes
sfully
                  completed volume snapshots for this backup.
                type: integer
              warnings:
                description: Warnings is a count of all warning messages that were
                  generated during execution of the backup. The actual warnings are
                  in the backup's log file in object storage.
                type: integer
            type: object
        type: object
    served: true
    storage: true
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.7.0
  creationTimestamp: null
  name: backupstoragelocations.velero.io
spec:
  group: velero.io
  names:
    kind: BackupStorageLocation
    listKind: BackupStorageLocationList
    plural: backupstoragelocations
    shortNames:
    - bsl
    singular: backupstoragelocation
  scope: Namespaced
```

```
  scope: Namespaced
  versions:
  - additionalPrinterColumns:
    - description: Backup Storage Location status such as Available/Unavailable
      jsonPath: .status.phase
      name: Phase
      type: string
    - description: LastValidationTime is the last time the backup store location wa
s
        validated
      jsonPath: .status.lastValidationTime
      name: Last Validated
      type: date
    - jsonPath: .metadata.creationTimestamp
      name: Age
      type: date
    - description: Default backup storage location
      jsonPath: .spec.default
      name: Default
      type: boolean
    name: v1
    schema:
      openAPIV3Schema:
        description: BackupStorageLocation is a location where Velero stores backup
          objects
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation
              of an object. Servers should convert recognized schemas to the latest
              internal value, and may reject unrecognized values. More info: https:
//git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resou
rces'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s
              object represents. Servers may infer this from the endpoint the clien
t
              submits requests to. Cannot be updated. In CamelCase. More info: http
s://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#typ
es-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: BackupStorageLocationSpec defines the desired state of a
              Velero BackupStorageLocation
            properties:
              accessMode:
                description: AccessMode defines the permissions for the backup stor
age
                  location.
                enum:
                - ReadOnly
```

```
                - ReadWrite
                type: string
              backupSyncPeriod:
                description: BackupSyncPeriod defines how frequently to sync backup
                  API objects from object storage. A value of 0 disables sync.
                nullable: true
                type: string
              config:
                additionalProperties:
                  type: string
                description: Config is for provider-specific configuration fields.
                type: object
              credential:
                description: Credential contains the credential information intende
d
                  to be used with this location
                properties:
                  key:
                    description: The key of the secret to select from.  Must be a
                      valid secret key.
                    type: string
                  name:
                    description: 'Name of the referent. More info: https://kubernet
es.io/docs/concepts/overview/working-with-objects/names/#names
                      TODO: Add other useful fields. apiVersion, kind, uid?'
                    type: string
                  optional:
                    description: Specify whether the Secret or its key must be defi
ned
                    type: boolean
                required:
                - key
                type: object
              default:
                description: Default indicates this location is the default backup
                  storage location.
                type: boolean
              objectStorage:
                description: ObjectStorageLocation specifies the settings necessary
                  to connect to a provider's object storage.
                properties:
                  bucket:
                    description: Bucket is the bucket to use for object storage.
                    type: string
                  caCert:
                    description: CACert defines a CA bundle to use when verifying
                      TLS connections to the provider.
                    format: byte
                    type: string
                  prefix:
                    description: Prefix is the path inside a bucket to use for Vele
ro
                      storage. Optional.
                    type: string
```

```
                required:
                - bucket
                type: object
              provider:
                description: Provider is the provider of the backup storage.
                type: string
              validationFrequency:
                description: ValidationFrequency defines how frequently to validate
                  the corresponding object storage. A value of 0 disables validatio
n.
                nullable: true
                type: string
            required:
            - objectStorage
            - provider
            type: object
          status:
            description: BackupStorageLocationStatus defines the observed state of
              BackupStorageLocation
            properties:
              accessMode:
                description: "AccessMode is an unused field. \n Deprecated: there
                  is now an AccessMode field on the Spec and this field will be rem
oved
                  entirely as of v2.0."
                enum:
                - ReadOnly
                - ReadWrite
                type: string
              lastSyncedRevision:
                description: "LastSyncedRevision is the value of the `metadata/revi
sion`
                  file in the backup storage location the last time the BSL's conte
nts
                  were synced into the cluster. \n Deprecated: this field is no lon
ger
                  updated or used for detecting changes to the location's contents
                  and will be removed entirely in v2.0."
                type: string
              lastSyncedTime:
                description: LastSyncedTime is the last time the contents of the lo
cation
                  were synced into the cluster.
                format: date-time
                nullable: true
                type: string
              lastValidationTime:
                description: LastValidationTime is the last time the backup store
                  location was validated the cluster.
                format: date-time
                nullable: true
                type: string
              phase:
                description: Phase is the current state of the BackupStorageLocatio
```

```
n.
                enum:
                - Available
                - Unavailable
                type: string
            type: object
        type: object
    served: true
    storage: true
    subresources:
      status: {}
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.7.0
  creationTimestamp: null
  name: deletebackuprequests.velero.io
spec:
  group: velero.io
  names:
    kind: DeleteBackupRequest
    listKind: DeleteBackupRequestList
    plural: deletebackuprequests
    singular: deletebackuprequest
  scope: Namespaced
  versions:
  - name: v1
    schema:
      openAPIV3Schema:
        description: DeleteBackupRequest is a request to delete one or more backups
.
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation
              of an object. Servers should convert recognized schemas to the latest
              internal value, and may reject unrecognized values. More info: https:
//git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resou
rces'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s
              object represents. Servers may infer this from the endpoint the clien
t
              submits requests to. Cannot be updated. In CamelCase. More info: http
s://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#typ
es-kinds'
            type: string
          metadata:
            type: object
          spec:
```

```
            description: DeleteBackupRequestSpec is the specification for which bac
kups
              to delete.
            properties:
              backupName:
                type: string
            required:
            - backupName
            type: object
          status:
            description: DeleteBackupRequestStatus is the current status of a Delet
eBackupRequest.
            properties:
              errors:
                description: Errors contains any errors that were encountered durin
g
                  the deletion process.
                items:
                  type: string
                nullable: true
                type: array
              phase:
                description: Phase is the current state of the DeleteBackupRequest.
                enum:
                - New
                - InProgress
                - Processed
                type: string
            type: object
        type: object
    served: true
    storage: true
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.7.0
  creationTimestamp: null
  name: downloadrequests.velero.io
spec:
  group: velero.io
  names:
    kind: DownloadRequest
    listKind: DownloadRequestList
    plural: downloadrequests
    singular: downloadrequest
  scope: Namespaced
  versions:
  - name: v1
    schema:
      openAPIV3Schema:
        description: DownloadRequest is a request to download an artifact from back
up
          object storage, such as a backup log file.
```

```
          object storage, such as a backup log file.
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation
              of an object. Servers should convert recognized schemas to the latest
              internal value, and may reject unrecognized values. More info: https:
//git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resou
rces'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s
              object represents. Servers may infer this from the endpoint the clien
t
              submits requests to. Cannot be updated. In CamelCase. More info: http
s://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#typ
es-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: DownloadRequestSpec is the specification for a download re
quest.
            properties:
              target:
                description: Target is what to download (e.g. logs for a backup).
                properties:
                  kind:
                    description: Kind is the type of file to download.
                    enum:
                    - BackupLog
                    - BackupContents
                    - BackupVolumeSnapshots
                    - BackupItemSnapshots
                    - BackupResourceList
                    - RestoreLog
                    - RestoreResults
                    type: string
                  name:
                    description: Name is the name of the kubernetes resource with
                      which the file is associated.
                    type: string
                required:
                - kind
                - name
                type: object
            required:
            - target
            type: object
          status:
            description: DownloadRequestStatus is the current status of a DownloadR
equest.
            properties:
              downloadURL:
```

```
                description: DownloadURL contains the pre-signed URL for the target
                  file.
                type: string
              expiration:
                description: Expiration is when this DownloadRequest expires and ca
n
                  be deleted by the system.
                format: date-time
                nullable: true
                type: string
              phase:
                description: Phase is the current state of the DownloadRequest.
                enum:
                - New
                - Processed
                type: string
            type: object
        type: object
    served: true
    storage: true
    subresources:
      status: {}
---
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.7.0
  creationTimestamp: null
  name: schedules.velero.io
spec:
  group: velero.io
  names:
    kind: Schedule
    listKind: ScheduleList
    plural: schedules
    singular: schedule
  scope: Namespaced
  versions:
  - name: v1
    schema:
      openAPIV3Schema:
        description: Schedule is a Velero resource that represents a pre-scheduled
          or periodic Backup that should be run.
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation
              of an object. Servers should convert recognized schemas to the latest
              internal value, and may reject unrecognized values. More info: https:
//git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resou
rces'
            type: string
```

```
          kind:
            description: 'Kind is a string value representing the REST resource thi
s
              object represents. Servers may infer this from the endpoint the clien
t
              submits requests to. Cannot be updated. In CamelCase. More info: http
s://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#typ
es-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: ScheduleSpec defines the specification for a Velero schedu
le
            properties:
              schedule:
                description: Schedule is a Cron expression defining when to run the
                  Backup.
                type: string
              template:
                description: Template is the definition of the Backup to be run on
                  the provided schedule
                properties:
                  defaultVolumesToRestic:
                    description: DefaultVolumesToRestic specifies whether restic sh
ould
                      be used to take a backup of all pod volumes by default.
                    type: boolean
                  excludedNamespaces:
                    description: ExcludedNamespaces contains a list of namespaces
                      that are not included in the backup.
                    items:
                      type: string
                    nullable: true
                    type: array
                  excludedResources:
                    description: ExcludedResources is a slice of resource names tha
t
                      are not included in the backup.
                    items:
                      type: string
                    nullable: true
                    type: array
                  hooks:
                    description: Hooks represent custom behaviors that should be ex
ecuted
                      at different phases of the backup.
                    properties:
                      resources:
                        description: Resources are hooks that should be executed wh
en
                          backing up individual instances of a resource.
                        items:
                          description: BackupResourceHookSpec defines one or more
```

```
                            BackupResourceHooks that should be executed based on th
e
                            rules defined for namespaces, resources, and label sele
ctor.
                          properties:
                            excludedNamespaces:
                              description: ExcludedNamespaces specifies the namespa
ces
                                to which this hook spec does not apply.
                              items:
                                type: string
                              nullable: true
                              type: array
                            excludedResources:
                              description: ExcludedResources specifies the resource
s
                                to which this hook spec does not apply.
                              items:
                                type: string
                              nullable: true
                              type: array
                            includedNamespaces:
                              description: IncludedNamespaces specifies the namespa
ces
                                to which this hook spec applies. If empty, it appli
es
                                to all namespaces.
                              items:
                                type: string
                              nullable: true
                              type: array
                            includedResources:
                              description: IncludedResources specifies the resource
s
                                to which this hook spec applies. If empty, it appli
es
                                to all resources.
                              items:
                                type: string
                              nullable: true
                              type: array
                            labelSelector:
                              description: LabelSelector, if specified, filters the
                                resources to which this hook spec applies.
                              nullable: true
                              properties:
                                matchExpressions:
                                  description: matchExpressions is a list of label
                                    selector requirements. The requirements are AND
ed.
                                  items:
                                    description: A label selector requirement is a
                                      selector that contains values, a key, and an
                                      operator that relates the key and values.
```

```
                                    properties:
                                      key:
                                        description: key is the label key that the
                                          selector applies to.
                                        type: string
                                      operator:
                                        description: operator represents a key's re
lationship
                                          to a set of values. Valid operators are
                                          In, NotIn, Exists and DoesNotExist.
                                        type: string
                                      values:
                                        description: values is an array of string
                                          values. If the operator is In or NotIn,
                                          the values array must be non-empty. If th
e
                                          operator is Exists or DoesNotExist, the
                                          values array must be empty. This array is
                                          replaced during a strategic merge patch.
                                        items:
                                          type: string
                                        type: array
                                    required:
                                    - key
                                    - operator
                                    type: object
                                  type: array
                                matchLabels:
                                  additionalProperties:
                                    type: string
                                  description: matchLabels is a map of {key,value}
                                    pairs. A single {key,value} in the matchLabels
                                    map is equivalent to an element of matchExpress
ions,
                                    whose key field is "key", the operator is "In",
                                    and the values array contains only "value". The
                                    requirements are ANDed.
                                  type: object
                              type: object
                            name:
                              description: Name is the name of this hook.
                              type: string
                            post:
                              description: PostHooks is a list of BackupResourceHoo
ks
                                to execute after storing the item in the backup. Th
ese
                                are executed after all "additional items" from item
                                actions are processed.
                              items:
                                description: BackupResourceHook defines a hook for
                                  a resource.
                                properties:
                                  exec:
                                    description: Exec defines an exec hook
```

```
                                    description: Exec defines an exec hook.
                                    properties:
                                      command:
                                        description: Command is the command and arg
uments
                                          to execute.
                                        items:
                                          type: string
                                        minItems: 1
                                        type: array
                                      container:
                                        description: Container is the container in
                                          the pod where the command should be execu
ted.
                                          If not specified, the pod's first contain
er
                                          is used.
                                        type: string
                                      onError:
                                        description: OnError specifies how Velero
                                          should behave if it encounters an error
                                          executing this hook.
                                        enum:
                                        - Continue
                                        - Fail
                                        type: string
                                      timeout:
                                        description: Timeout defines the maximum am
ount
                                          of time Velero should wait for the hook
                                          to complete before considering the execut
ion
                                          a failure.
                                        type: string
                                    required:
                                    - command
                                    type: object
                                  required:
                                  - exec
                                  type: object
                                type: array
                              pre:
                                description: PreHooks is a list of BackupResourceHook
s
                                  to execute prior to storing the item in the backup.
                                  These are executed before any "additional items" fr
om
                                  item actions are processed.
                                items:
                                  description: BackupResourceHook defines a hook for
                                    a resource.
                                  properties:
                                    exec:
                                      description: Exec defines an exec hook.
                                      properties:
```

```
                                    properties:
                                      command:
                                        description: Command is the command and arg
uments
                                          to execute.
                                        items:
                                          type: string
                                        minItems: 1
                                        type: array
                                      container:
                                        description: Container is the container in
                                          the pod where the command should be execu
ted.
                                          If not specified, the pod's first contain
er
                                          is used.
                                        type: string
                                      onError:
                                        description: OnError specifies how Velero
                                          should behave if it encounters an error
                                          executing this hook.
                                        enum:
                                        - Continue
                                        - Fail
                                        type: string
                                      timeout:
                                        description: Timeout defines the maximum am
ount
                                          of time Velero should wait for the hook
                                          to complete before considering the execut
ion
                                          a failure.
                                        type: string
                                    required:
                                    - command
                                    type: object
                                required:
                                - exec
                                type: object
                              type: array
                          required:
                          - name
                          type: object
                        nullable: true
                        type: array
                    type: object
                  includeClusterResources:
                    description: IncludeClusterResources specifies whether cluster-
scoped
                      resources should be included for consideration in the backup.
                    nullable: true
                    type: boolean
                  includedNamespaces:
                    description: IncludedNamespaces is a slice of namespace names
                      to include objects from. If empty, all namespaces are include
```

```
d.
                    items:
                      type: string
                    nullable: true
                    type: array
                  includedResources:
                    description: IncludedResources is a slice of resource names to
                      include in the backup. If empty, all resources are included.
                    items:
                      type: string
                    nullable: true
                    type: array
                  labelSelector:
                    description: LabelSelector is a metav1.LabelSelector to filter
                      with when adding individual objects to the backup. If empty
                      or nil, all objects are included. Optional.
                    nullable: true
                    properties:
                      matchExpressions:
                        description: matchExpressions is a list of label selector
                          requirements. The requirements are ANDed.
                        items:
                          description: A label selector requirement is a selector
                            that contains values, a key, and an operator that relat
es
                            the key and values.
                          properties:
                            key:
                              description: key is the label key that the selector
                                applies to.
                              type: string
                            operator:
                              description: operator represents a key's relationship
                                to a set of values. Valid operators are In, NotIn,
                                Exists and DoesNotExist.
                              type: string
                            values:
                              description: values is an array of string values. If
                                the operator is In or NotIn, the values array must
                                be non-empty. If the operator is Exists or DoesNotE
xist,
                                the values array must be empty. This array is repla
ced
                                during a strategic merge patch.
                              items:
                                type: string
                              type: array
                          required:
                          - key
                          - operator
                          type: object
                        type: array
                      matchLabels:
                        additionalProperties:
```

```
                        type: string
                      description: matchLabels is a map of {key,value} pairs. A
                        single {key,value} in the matchLabels map is equivalent
                        to an element of matchExpressions, whose key field is "ke
y",
                        the operator is "In", and the values array contains only
                        "value". The requirements are ANDed.
                      type: object
                  type: object
                metadata:
                  properties:
                    labels:
                      additionalProperties:
                        type: string
                      type: object
                  type: object
                orderedResources:
                  additionalProperties:
                    type: string
                  description: OrderedResources specifies the backup order of res
ources
                    of specific Kind. The map key is the Kind name and value is
                    a list of resource names separated by commas. Each resource
                    name has format "namespace/resourcename".  For cluster resour
ces,
                    simply use "resourcename".
                  nullable: true
                  type: object
                snapshotVolumes:
                  description: SnapshotVolumes specifies whether to take cloud sn
apshots
                    of any PV's referenced in the set of objects included in the
                    Backup.
                  nullable: true
                  type: boolean
                storageLocation:
                  description: StorageLocation is a string containing the name of
                    a BackupStorageLocation where the backup should be stored.
                  type: string
                ttl:
                  description: TTL is a time.Duration-parseable string describing
                    how long the Backup should be retained for.
                  type: string
                volumeSnapshotLocations:
                  description: VolumeSnapshotLocations is a list containing names
                    of VolumeSnapshotLocations associated with this backup.
                  items:
                    type: string
                  type: array
              type: object
            useOwnerReferencesInBackup:
              description: UseOwnerReferencesBackup specifies whether to use Owne
rReferences
                on backups created by this Schedule.
```

```
                nullable: true
                type: boolean
            required:
            - schedule
            - template
            type: object
          status:
            description: ScheduleStatus captures the current state of a Velero sche
dule
            properties:
              lastBackup:
                description: LastBackup is the last time a Backup was run for this
                  Schedule schedule
                format: date-time
                nullable: true
                type: string
              phase:
                description: Phase is the current phase of the Schedule
                enum:
                - New
                - Enabled
                - FailedValidation
                type: string
              validationErrors:
                description: ValidationErrors is a slice of all validation errors
                  (if applicable)
                items:
                  type: string
                type: array
            type: object
        type: object
    served: true
    storage: true
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.7.0
  creationTimestamp: null
  name: restores.velero.io
spec:
  group: velero.io
  names:
    kind: Restore
    listKind: RestoreList
    plural: restores
    singular: restore
  scope: Namespaced
  versions:
  - name: v1
    schema:
      openAPIV3Schema:
        description: Restore is a Velero resource that represents the application
```

```
          of resources from a Velero backup to a target Kubernetes cluster.
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation
              of an object. Servers should convert recognized schemas to the latest
              internal value, and may reject unrecognized values. More info: https:
//git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resou
rces'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s
              object represents. Servers may infer this from the endpoint the clien
t
              submits requests to. Cannot be updated. In CamelCase. More info: http
s://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#typ
es-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: RestoreSpec defines the specification for a Velero restore
.
            properties:
              backupName:
                description: BackupName is the unique name of the Velero backup to
                  restore from.
                type: string
              excludedNamespaces:
                description: ExcludedNamespaces contains a list of namespaces that
                  are not included in the restore.
                items:
                  type: string
                nullable: true
                type: array
              excludedResources:
                description: ExcludedResources is a slice of resource names that ar
e
                  not included in the restore.
                items:
                  type: string
                nullable: true
                type: array
              hooks:
                description: Hooks represent custom behaviors that should be execut
ed
                  during or post restore.
                properties:
                  resources:
                    items:
                      description: RestoreResourceHookSpec defines one or more Rest
oreResrouceHooks
                        that should be executed based on the rules defined for name
spaces.
```

```
spaces,
                        resources, and label selector.
                      properties:
                        excludedNamespaces:
                          description: ExcludedNamespaces specifies the namespaces
                            to which this hook spec does not apply.
                          items:
                            type: string
                          nullable: true
                          type: array
                        excludedResources:
                          description: ExcludedResources specifies the resources to
                            which this hook spec does not apply.
                          items:
                            type: string
                          nullable: true
                          type: array
                        includedNamespaces:
                          description: IncludedNamespaces specifies the namespaces
                            to which this hook spec applies. If empty, it applies
                            to all namespaces.
                          items:
                            type: string
                          nullable: true
                          type: array
                        includedResources:
                          description: IncludedResources specifies the resources to
                            which this hook spec applies. If empty, it applies to
                            all resources.
                          items:
                            type: string
                          nullable: true
                          type: array
                        labelSelector:
                          description: LabelSelector, if specified, filters the res
ources
                            to which this hook spec applies.
                          nullable: true
                          properties:
                            matchExpressions:
                              description: matchExpressions is a list of label sele
ctor
                                requirements. The requirements are ANDed.
                              items:
                                description: A label selector requirement is a sele
ctor
                                  that contains values, a key, and an operator that
                                  relates the key and values.
                                properties:
                                  key:
                                    description: key is the label key that the sele
ctor
                                      applies to.
                                    type: string
                                  operator:
```

```
                                    description: operator represents a key's relati
onship
                                      to a set of values. Valid operators are In,
                                      NotIn, Exists and DoesNotExist.
                                    type: string
                                  values:
                                    description: values is an array of string value
s.
                                      If the operator is In or NotIn, the values ar
ray
                                      must be non-empty. If the operator is Exists
                                      or DoesNotExist, the values array must be emp
ty.
                                      This array is replaced during a strategic mer
ge
                                      patch.
                                    items:
                                      type: string
                                    type: array
                                required:
                                - key
                                - operator
                                type: object
                              type: array
                            matchLabels:
                              additionalProperties:
                                type: string
                              description: matchLabels is a map of {key,value} pair
s.
                                A single {key,value} in the matchLabels map is equi
valent
                                to an element of matchExpressions, whose key field
                                is "key", the operator is "In", and the values arra
y
                                contains only "value". The requirements are ANDed.
                              type: object
                          type: object
                        name:
                          description: Name is the name of this hook.
                          type: string
                        postHooks:
                          description: PostHooks is a list of RestoreResourceHooks
                            to execute during and after restoring a resource.
                          items:
                            description: RestoreResourceHook defines a restore hook
                              for a resource.
                            properties:
                              exec:
                                description: Exec defines an exec restore hook.
                                properties:
                                  command:
                                    description: Command is the command and argumen
ts
                                      to execute from within a container after a po
```

```
d
                                    has been restored.
                                  items:
                                    type: string
                                  minItems: 1
                                  type: array
                                container:
                                  description: Container is the container in the
                                    pod where the command should be executed. If
                                    not specified, the pod's first container is
                                    used.
                                  type: string
                                execTimeout:
                                  description: ExecTimeout defines the maximum am
ount
                                    of time Velero should wait for the hook to co
mplete
                                    before considering the execution a failure.
                                  type: string
                                onError:
                                  description: OnError specifies how Velero shoul
d
                                    behave if it encounters an error executing th
is
                                    hook.
                                  enum:
                                  - Continue
                                  - Fail
                                  type: string
                                waitTimeout:
                                  description: WaitTimeout defines the maximum am
ount
                                    of time Velero should wait for the container
                                    to be Ready before attempting to run the comm
and.
                                  type: string
                              required:
                              - command
                              type: object
                            init:
                              description: Init defines an init restore hook.
                              properties:
                                initContainers:
                                  description: InitContainers is list of init con
tainers
                                    to be added to a pod during its restore.
                                  items:
                                    description: A single application container
                                      that you want to run within a pod.
                                    properties:
                                      args:
                                        description: 'Arguments to the entrypoint
.
                                          The docker image''s CMD is used if this
```

```
                                                  is not provided. Variable references $(
VAR_NAME)
                                                  are expanded using the container''s env
ironment.
                                                  If a variable cannot be resolved, the
                                                  reference in the input string will be
                                                  unchanged. Double $$ are reduced to a
                                                  single $, which allows for escaping the
                                                  $(VAR_NAME) syntax: i.e. "$$(VAR_NAME)"
                                                  will produce the string literal "$(VAR_
NAME)".
                                                  Escaped references will never be expand
ed,
                                                  regardless of whether the variable exis
ts
                                                  or not. Cannot be updated. More info:
                                                  https://kubernetes.io/docs/tasks/inject
-data-application/define-command-argument-container/#running-a-command-in-a-shell'
                                              items:
                                                type: string
                                              type: array
                                            command:
                                              description: 'Entrypoint array. Not execu
ted
                                                within a shell. The docker image''s ENT
RYPOINT
                                                is used if this is not provided. Variab
le
                                                references $(VAR_NAME) are expanded usi
ng
                                                the container''s environment. If a vari
able
                                                cannot be resolved, the reference in th
e
                                                input string will be unchanged. Double
                                                $$ are reduced to a single $, which all
ows
                                                for escaping the $(VAR_NAME) syntax: i.
e.
                                                "$$(VAR_NAME)" will produce the string
                                                literal "$(VAR_NAME)". Escaped referenc
es
                                                will never be expanded, regardless of
                                                whether the variable exists or not. Can
not
                                                be updated. More info: https://kubernet
es.io/docs/tasks/inject-data-application/define-command-argument-container/#running
-a-command-in-a-shell'
                                              items:
                                                type: string
                                              type: array
                                            env:
                                              description: List of environment variable
s
```

```
                                        to set in the container. Cannot be upda
ted.
                                      items:
                                        description: EnvVar represents an envir
onment
                                          variable present in a Container.
                                        properties:
                                          name:
                                            description: Name of the environmen
t
                                              variable. Must be a C_IDENTIFIER.
                                            type: string
                                          value:
                                            description: 'Variable references
                                              $(VAR_NAME) are expanded using th
e
                                              previously defined environment va
riables
                                              in the container and any service
                                              environment variables. If a varia
ble
                                              cannot be resolved, the reference
                                              in the input string will be uncha
nged.
                                              Double $$ are reduced to a single
                                              $, which allows for escaping the
                                              $(VAR_NAME) syntax: i.e. "$$(VAR_
NAME)"
                                              will produce the string literal
                                              "$(VAR_NAME)". Escaped references
                                              will never be expanded, regardles
s
                                              of whether the variable exists or
                                              not. Defaults to "".'
                                            type: string
                                          valueFrom:
                                            description: Source for the environ
ment
                                              variable's value. Cannot be used
                                              if value is not empty.
                                            properties:
                                              configMapKeyRef:
                                                description: Selects a key of
                                                  a ConfigMap.
                                                properties:
                                                  key:
                                                    description: The key to sel
ect.
                                                      type: string
                                                  name:
                                                    description: 'Name of the
                                                      referent. More info: http
s://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
                                                      TODO: Add other useful fi
elds
```

```
elds.
                                                        apiVersion, kind, uid?'
                                                      type: string
                                                    optional:
                                                      description: Specify whethe
r
                                                        the ConfigMap or its key
                                                        must be defined
                                                      type: boolean
                                                  required:
                                                  - key
                                                  type: object
                                                fieldRef:
                                                  description: 'Selects a field
                                                    of the pod: supports metadata
.name,
                                                    metadata.namespace, `metadata
.labels[''<KEY>'']`,
                                                    `metadata.annotations[''<KEY>
'']`,
                                                    spec.nodeName, spec.serviceAc
countName,
                                                    status.hostIP, status.podIP,
                                                    status.podIPs.'
                                                  properties:
                                                    apiVersion:
                                                      description: Version of the
                                                        schema the FieldPath is
                                                        written in terms of, defa
ults
                                                        to "v1".
                                                      type: string
                                                    fieldPath:
                                                      description: Path of the fi
eld
                                                        to select in the specifie
d
                                                        API version.
                                                      type: string
                                                  required:
                                                  - fieldPath
                                                  type: object
                                                resourceFieldRef:
                                                  description: 'Selects a resourc
e
                                                    of the container: only resour
ces
                                                    limits and requests (limits.c
pu,
                                                    limits.memory, limits.ephemer
al-storage,
                                                    requests.cpu, requests.memory
                                                    and requests.ephemeral-storag
e)
                                                    are currently supported.'
```

```
                                                  properties:
                                                    containerName:
                                                      description: 'Container nam
e:
                                                        required for volumes, opt
ional
                                                        for env vars'
                                                      type: string
                                                    divisor:
                                                      anyOf:
                                                      - type: integer
                                                      - type: string
                                                      description: Specifies the
                                                        output format of the expo
sed
                                                        resources, defaults to "1
"
                                                        pattern: ^(\+|-)?(([0-9]+(\
.[0-9]*)?)|(\.[0-9]+))(([KMGTPE]i)|[numkMGTPE]|([eE](\+|-)?(([0-9]+(\.[0-9]*)?)|(\.
[0-9]+))))?$
                                                      x-kubernetes-int-or-string:
true
                                                    resource:
                                                      description: 'Required: res
ource
                                                        to select'
                                                      type: string
                                                  required:
                                                  - resource
                                                  type: object
                                                secretKeyRef:
                                                  description: Selects a key of
                                                    a secret in the pod's namespa
ce
                                                  properties:
                                                    key:
                                                      description: The key of the
                                                        secret to select from.  M
ust
                                                        be a valid secret key.
                                                      type: string
                                                    name:
                                                      description: 'Name of the
                                                        referent. More info: http
s://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
                                                        TODO: Add other useful fi
elds.
                                                        apiVersion, kind, uid?'
                                                      type: string
                                                    optional:
                                                      description: Specify whethe
r
                                                        the Secret or its key mus
t
```

```
                                                    be defined
                                                  type: boolean
                                              required:
                                              - key
                                              type: object
                                          type: object
                                      required:
                                      - name
                                      type: object
                                  type: array
                              envFrom:
                                description: List of sources to populate
                                  environment variables in the container.
                                  The keys defined within a source must
                                  be a C_IDENTIFIER. All invalid keys wil
l
                                  be reported as an event when the contai
ner
                                  is starting. When a key exists in multi
ple
                                  sources, the value associated with the
                                  last source will take precedence. Value
s
                                  defined by an Env with a duplicate key
                                  will take precedence. Cannot be updated
.
                                items:
                                  description: EnvFromSource represents
                                    the source of a set of ConfigMaps
                                  properties:
                                    configMapRef:
                                      description: The ConfigMap to selec
t
                                        from
                                      properties:
                                        name:
                                          description: 'Name of the refer
ent.
                                            More info: https://kubernetes
.io/docs/concepts/overview/working-with-objects/names/#names
                                            TODO: Add other useful fields
.
                                            apiVersion, kind, uid?'
                                          type: string
                                        optional:
                                          description: Specify whether th
e
                                            ConfigMap must be defined
                                          type: boolean
                                      type: object
                                    prefix:
                                      description: An optional identifier
                                        to prepend to each key in the Con
figMap.
```

```
                                                  Must be a C_IDENTIFIER.
                                                type: string
                                              secretRef:
                                                description: The Secret to select
                                                  from
                                                properties:
                                                  name:
                                                    description: 'Name of the refer
ent.
                                                      More info: https://kubernetes
.io/docs/concepts/overview/working-with-objects/names/#names
                                                      TODO: Add other useful fields
.
                                                      apiVersion, kind, uid?'
                                                    type: string
                                                  optional:
                                                    description: Specify whether th
e
                                                      Secret must be defined
                                                    type: boolean
                                                type: object
                                            type: object
                                          type: array
                                        image:
                                          description: 'Docker image name. More inf
o:
                                            https://kubernetes.io/docs/concepts/con
tainers/images
                                            This field is optional to allow higher
                                            level config management to default or
                                            override container images in workload
                                            controllers like Deployments and Statef
ulSets.'
                                          type: string
                                        imagePullPolicy:
                                          description: 'Image pull policy. One of
                                            Always, Never, IfNotPresent. Defaults
                                            to Always if :latest tag is specified,
                                            or IfNotPresent otherwise. Cannot be up
dated.
                                            More info: https://kubernetes.io/docs/c
oncepts/containers/images#updating-images'
                                          type: string
                                        lifecycle:
                                          description: Actions that the management
                                            system should take in response to conta
iner
                                            lifecycle events. Cannot be updated.
                                          properties:
                                            postStart:
                                              description: 'PostStart is called imm
ediately
                                                after a container is created. If th
e
```

```
                                                handler fails, the container is ter
minated
                                                and restarted according to its rest
art
                                                policy. Other management of the con
tainer
                                                blocks until the hook completes. Mo
re
                                                info: https://kubernetes.io/docs/co
ncepts/containers/container-lifecycle-hooks/#container-hooks'
                                              properties:
                                                exec:
                                                  description: One and only one of
                                                    the following should be specifi
ed.
                                                    Exec specifies the action to ta
ke.
                                                  properties:
                                                    command:
                                                      description: Command is the
                                                        command line to execute ins
ide
                                                        the container, the working
                                                        directory for the command
is
                                                        root ('/') in the container
's
                                                        filesystem. The command is
                                                        simply exec'd, it is not ru
n
                                                        inside a shell, so traditio
nal
                                                        shell instructions ('|', et
c)
                                                        won't work. To use a shell,
                                                        you need to explicitly call
                                                        out to that shell. Exit sta
tus
                                                        of 0 is treated as live/hea
lthy
                                                        and non-zero is unhealthy.
                                                      items:
                                                        type: string
                                                      type: array
                                                  type: object
                                                httpGet:
                                                  description: HTTPGet specifies th
e
                                                    http request to perform.
                                                  properties:
                                                    host:
                                                      description: Host name to con
nect
                                                        to, defaults to the pod IP.
                                                        You probably want to set "H
```

```
                                                    You probably want to set "H
ost"
                                                      in httpHeaders instead.
                                                    type: string
                                                  httpHeaders:
                                                    description: Custom headers
                                                      to set in the request. HTTP
                                                      allows repeated headers.
                                                    items:
                                                      description: HTTPHeader des
cribes
                                                        a custom header to be use
d
                                                        in HTTP probes
                                                      properties:
                                                        name:
                                                          description: The header
                                                            field name
                                                          type: string
                                                        value:
                                                          description: The header
                                                            field value
                                                          type: string
                                                      required:
                                                      - name
                                                      - value
                                                      type: object
                                                    type: array
                                                  path:
                                                    description: Path to access
                                                      on the HTTP server.
                                                    type: string
                                                  port:
                                                    anyOf:
                                                    - type: integer
                                                    - type: string
                                                    description: Name or number
                                                      of the port to access on th
e
                                                      container. Number must be
                                                      in the range 1 to 65535. Na
me
                                                      must be an IANA_SVC_NAME.
                                                    x-kubernetes-int-or-string: t
rue
                                                  scheme:
                                                    description: Scheme to use fo
r
                                                      connecting to the host. Def
aults
                                                      to HTTP.
                                                    type: string
                                                required:
                                                - port
                                                type: object
```

```
                                                  tcpSocket:
                                                    description: 'TCPSocket specifies
                                                      an action involving a TCP port.
                                                      TCP hooks not yet supported TOD
O:
                                                      implement a realistic TCP lifec
ycle
                                                      hook'
                                                    properties:
                                                      host:
                                                        description: 'Optional: Host
                                                          name to connect to, default
s
                                                          to the pod IP.'
                                                        type: string
                                                      port:
                                                        anyOf:
                                                        - type: integer
                                                        - type: string
                                                        description: Number or name
                                                          of the port to access on th
e
                                                          container. Number must be
                                                          in the range 1 to 65535. Na
me
                                                          must be an IANA_SVC_NAME.
                                                        x-kubernetes-int-or-string: t
rue
                                                      required:
                                                      - port
                                                      type: object
                                                  type: object
                                                preStop:
                                                  description: 'PreStop is called immed
iately
                                                    before a container is terminated du
e
                                                    to an API request or management eve
nt
                                                    such as liveness/startup probe fail
ure,
                                                    preemption, resource contention, et
c.
                                                    The handler is not called if the co
ntainer
                                                    crashes or exits. The reason for te
rmination
                                                    is passed to the handler. The Pod''
s
                                                    termination grace period countdown
                                                    begins before the PreStop hooked is
                                                    executed. Regardless of the outcome
                                                    of the handler, the container will
                                                    eventually terminate within the Pod
```

```
''s
                                                termination grace period. Other man
agement
                                                of the container blocks until the
                                                hook completes or until the termina
tion
                                                grace period is reached. More info:
                                                https://kubernetes.io/docs/concepts
/containers/container-lifecycle-hooks/#container-hooks'
                                              properties:
                                                exec:
                                                  description: One and only one of
                                                    the following should be specifi
ed.
                                                    Exec specifies the action to ta
ke.
                                                  properties:
                                                    command:
                                                      description: Command is the
                                                        command line to execute ins
ide
                                                        the container, the working
                                                        directory for the command
is
                                                        root ('/') in the container
's
                                                        filesystem. The command is
                                                        simply exec'd, it is not ru
n
                                                        inside a shell, so traditio
nal
                                                        shell instructions ('|', et
c)
                                                        won't work. To use a shell,
                                                        you need to explicitly call
                                                        out to that shell. Exit sta
tus
                                                        of 0 is treated as live/hea
lthy
                                                        and non-zero is unhealthy.
                                                      items:
                                                        type: string
                                                      type: array
                                                  type: object
                                                httpGet:
                                                  description: HTTPGet specifies th
e
                                                    http request to perform.
                                                  properties:
                                                    host:
                                                      description: Host name to con
nect
                                                        to, defaults to the pod IP.
                                                        You probably want to set "H
```

```
ost"
                                                    in httpHeaders instead.
                                                  type: string
                                                httpHeaders:
                                                  description: Custom headers
                                                    to set in the request. HTTP
                                                    allows repeated headers.
                                                  items:
                                                    description: HTTPHeader des
cribes
                                                      a custom header to be use
d
                                                      in HTTP probes
                                                    properties:
                                                      name:
                                                        description: The header
                                                          field name
                                                        type: string
                                                      value:
                                                        description: The header
                                                          field value
                                                        type: string
                                                    required:
                                                    - name
                                                    - value
                                                    type: object
                                                  type: array
                                                path:
                                                  description: Path to access
                                                    on the HTTP server.
                                                  type: string
                                                port:
                                                  anyOf:
                                                  - type: integer
                                                  - type: string
                                                  description: Name or number
                                                    of the port to access on th
e
                                                    container. Number must be
                                                    in the range 1 to 65535. Na
me
                                                    must be an IANA_SVC_NAME.
                                                  x-kubernetes-int-or-string: t
rue
                                                scheme:
                                                  description: Scheme to use fo
r
                                                    connecting to the host. Def
aults
                                                    to HTTP.
                                                  type: string
                                              required:
                                              - port
                                              type: object
```

```
                                                tcpSocket:
                                                  description: 'TCPSocket specifies
                                                    an action involving a TCP port.
                                                    TCP hooks not yet supported TOD
O:
                                                    implement a realistic TCP lifec
ycle
                                                    hook'
                                                  properties:
                                                    host:
                                                      description: 'Optional: Host
                                                        name to connect to, default
s
                                                        to the pod IP.'
                                                      type: string
                                                    port:
                                                      anyOf:
                                                      - type: integer
                                                      - type: string
                                                      description: Number or name
                                                        of the port to access on th
e
                                                        container. Number must be
                                                        in the range 1 to 65535. Na
me
                                                        must be an IANA_SVC_NAME.
                                                      x-kubernetes-int-or-string: t
rue
                                                  required:
                                                  - port
                                                  type: object
                                              type: object
                                            type: object
                                          livenessProbe:
                                            description: 'Periodic probe of container
                                              liveness. Container will be restarted
                                              if the probe fails. Cannot be updated.
                                              More info: https://kubernetes.io/docs/c
oncepts/workloads/pods/pod-lifecycle#container-probes'
                                            properties:
                                              exec:
                                                description: One and only one of the
                                                  following should be specified. Exec
                                                  specifies the action to take.
                                                properties:
                                                  command:
                                                    description: Command is the comma
nd
                                                      line to execute inside the cont
ainer,
                                                      the working directory for the
                                                      command  is root ('/') in the
                                                      container's filesystem. The com
mand
                                                      is simply exec'd, it is not run
```

```
                                                  is simply exec'd, it is not run
                                                  inside a shell, so traditional
                                                  shell instructions ('|', etc)
                                                  won't work. To use a shell, you
                                                  need to explicitly call out to
                                                  that shell. Exit status of 0 is
                                                  treated as live/healthy and non
-zero
                                                  is unhealthy.
                                                items:
                                                  type: string
                                                type: array
                                              type: object
                                            failureThreshold:
                                              description: Minimum consecutive fail
ures
                                                for the probe to be considered fail
ed
                                                after having succeeded. Defaults to
                                                3. Minimum value is 1.
                                              format: int32
                                              type: integer
                                            httpGet:
                                              description: HTTPGet specifies the ht
tp
                                                request to perform.
                                              properties:
                                                host:
                                                  description: Host name to connect
                                                    to, defaults to the pod IP. You
                                                    probably want to set "Host" in
                                                    httpHeaders instead.
                                                  type: string
                                                httpHeaders:
                                                  description: Custom headers to se
t
                                                    in the request. HTTP allows rep
eated
                                                    headers.
                                                  items:
                                                    description: HTTPHeader describ
es
                                                      a custom header to be used in
                                                      HTTP probes
                                                    properties:
                                                      name:
                                                        description: The header fie
ld
                                                          name
                                                        type: string
                                                      value:
                                                        description: The header fie
ld
                                                          value
                                                        type: string
```

```
                                                    type: string
                                                  required:
                                                  - name
                                                  - value
                                                  type: object
                                                type: array
                                              path:
                                                description: Path to access on th
e
                                                  HTTP server.
                                                type: string
                                              port:
                                                anyOf:
                                                - type: integer
                                                - type: string
                                                description: Name or number of th
e
                                                  port to access on the container
.
                                                  Number must be in the range 1
                                                  to 65535. Name must be an IANA_
SVC_NAME.
                                                x-kubernetes-int-or-string: true
                                              scheme:
                                                description: Scheme to use for co
nnecting
                                                  to the host. Defaults to HTTP.
                                                type: string
                                            required:
                                            - port
                                            type: object
                                          initialDelaySeconds:
                                            description: 'Number of seconds after
                                              the container has started before li
veness
                                              probes are initiated. More info: ht
tps://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes'
                                            format: int32
                                            type: integer
                                          periodSeconds:
                                            description: How often (in seconds)
                                              to perform the probe. Default to 10
                                              seconds. Minimum value is 1.
                                            format: int32
                                            type: integer
                                          successThreshold:
                                            description: Minimum consecutive succ
esses
                                              for the probe to be considered succ
essful
                                              after having failed. Defaults to 1.
                                              Must be 1 for liveness and startup.
                                              Minimum value is 1.
                                            format: int32
                                            type: integer
```

```
                                        tcpSocket:
                                          description: 'TCPSocket specifies an
                                            action involving a TCP port. TCP ho
oks
                                            not yet supported TODO: implement
                                            a realistic TCP lifecycle hook'
                                          properties:
                                            host:
                                              description: 'Optional: Host name
                                                to connect to, defaults to the
                                                pod IP.'
                                              type: string
                                            port:
                                              anyOf:
                                              - type: integer
                                              - type: string
                                              description: Number or name of th
e
                                                port to access on the container
.
                                                Number must be in the range 1
                                                to 65535. Name must be an IANA_
SVC_NAME.
                                              x-kubernetes-int-or-string: true
                                          required:
                                          - port
                                          type: object
                                        terminationGracePeriodSeconds:
                                          description: Optional duration in sec
onds
                                            the pod needs to terminate graceful
ly
                                            upon probe failure. The grace perio
d
                                            is the duration in seconds after th
e
                                            processes running in the pod are se
nt
                                            a termination signal and the time
                                            when the processes are forcibly hal
ted
                                            with a kill signal. Set this value
                                            longer than the expected cleanup ti
me
                                            for your process. If this value is
                                            nil, the pod's terminationGracePeri
odSeconds
                                            will be used. Otherwise, this value
                                            overrides the value provided by the
                                            pod spec. Value must be non-negativ
e
                                            integer. The value zero indicates
                                            stop immediately via the kill signa
l
```

```
                                                (no opportunity to shut down). This
                                                is a beta field and requires enabli
ng
                                                ProbeTerminationGracePeriod feature
                                                gate. Minimum value is 1. spec.term
inationGracePeriodSeconds
                                                is used if unset.
                                              format: int64
                                              type: integer
                                            timeoutSeconds:
                                              description: 'Number of seconds after
                                                which the probe times out. Defaults
                                                to 1 second. Minimum value is 1. Mo
re
                                                info: https://kubernetes.io/docs/co
ncepts/workloads/pods/pod-lifecycle#container-probes'
                                              format: int32
                                              type: integer
                                          type: object
                                        name:
                                          description: Name of the container specif
ied
                                            as a DNS_LABEL. Each container in a pod
                                            must have a unique name (DNS_LABEL). Ca
nnot
                                            be updated.
                                          type: string
                                        ports:
                                          description: List of ports to expose from
                                            the container. Exposing a port here giv
es
                                            the system additional information about
                                            the network connections a container use
s,
                                            but is primarily informational. Not spe
cifying
                                            a port here DOES NOT prevent that port
                                            from being exposed. Any port which is
                                            listening on the default "0.0.0.0" addr
ess
                                            inside a container will be accessible
                                            from the network. Cannot be updated.
                                          items:
                                            description: ContainerPort represents
                                              a network port in a single container.
                                            properties:
                                              containerPort:
                                                description: Number of port to expo
se
                                                  on the pod's IP address. This mus
t
                                                  be a valid port number, 0 < x <
                                                  65536.
                                                format: int32
```

```
                                        type: integer
                                      hostIP:
                                        description: What host IP to bind
                                          the external port to.
                                        type: string
                                      hostPort:
                                        description: Number of port to expo
se
                                          on the host. If specified, this
                                          must be a valid port number, 0 <
                                          x < 65536. If HostNetwork is spec
ified,
                                          this must match ContainerPort. Mo
st
                                          containers do not need this.
                                        format: int32
                                        type: integer
                                      name:
                                        description: If specified, this mus
t
                                          be an IANA_SVC_NAME and unique wi
thin
                                          the pod. Each named port in a pod
                                          must have a unique name. Name for
                                          the port that can be referred to
                                          by services.
                                        type: string
                                      protocol:
                                        default: TCP
                                        description: Protocol for port. Mus
t
                                          be UDP, TCP, or SCTP. Defaults to
                                          "TCP".
                                        type: string
                                    required:
                                    - containerPort
                                    - protocol
                                    type: object
                                  type: array
                                  x-kubernetes-list-map-keys:
                                  - containerPort
                                  - protocol
                                  x-kubernetes-list-type: map
                                readinessProbe:
                                  description: 'Periodic probe of container
                                    service readiness. Container will be re
moved
                                    from service endpoints if the probe fai
ls.
                                    Cannot be updated. More info: https://k
ubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes'
                                  properties:
                                    exec:
                                      description: One and only one of the
                                        following should be specified. Exec
```

```
                                                  following should be specified. Exec
                                                  specifies the action to take.
                                                properties:
                                                  command:
                                                    description: Command is the comma
nd
                                                      line to execute inside the cont
ainer,
                                                      the working directory for the
                                                      command  is root ('/') in the
                                                      container's filesystem. The com
mand
                                                      is simply exec'd, it is not run
                                                      inside a shell, so traditional
                                                      shell instructions ('|', etc)
                                                      won't work. To use a shell, you
                                                      need to explicitly call out to
                                                      that shell. Exit status of 0 is
                                                      treated as live/healthy and non
-zero
                                                      is unhealthy.
                                                    items:
                                                      type: string
                                                    type: array
                                                type: object
                                              failureThreshold:
                                                description: Minimum consecutive fail
ures
                                                  for the probe to be considered fail
ed
                                                  after having succeeded. Defaults to
                                                  3. Minimum value is 1.
                                                format: int32
                                                type: integer
                                              httpGet:
                                                description: HTTPGet specifies the ht
tp
                                                  request to perform.
                                                properties:
                                                  host:
                                                    description: Host name to connect
                                                      to, defaults to the pod IP. You
                                                      probably want to set "Host" in
                                                      httpHeaders instead.
                                                    type: string
                                                  httpHeaders:
                                                    description: Custom headers to se
t
                                                      in the request. HTTP allows rep
eated
                                                      headers.
                                                    items:
                                                      description: HTTPHeader describ
es
                                                        a custom header to be used in
```

```
                                                    a custom header to be used in
                                                    HTTP probes
                                                  properties:
                                                    name:
                                                      description: The header fie
ld
                                                        name
                                                      type: string
                                                    value:
                                                      description: The header fie
ld
                                                        value
                                                      type: string
                                                  required:
                                                  - name
                                                  - value
                                                  type: object
                                                type: array
                                              path:
                                                description: Path to access on th
e
                                                  HTTP server.
                                                type: string
                                              port:
                                                anyOf:
                                                - type: integer
                                                - type: string
                                                description: Name or number of th
e
                                                  port to access on the container
.
                                                  Number must be in the range 1
                                                  to 65535. Name must be an IANA_
SVC_NAME.
                                                x-kubernetes-int-or-string: true
                                              scheme:
                                                description: Scheme to use for co
nnecting
                                                  to the host. Defaults to HTTP.
                                                type: string
                                            required:
                                            - port
                                            type: object
                                          initialDelaySeconds:
                                            description: 'Number of seconds after
                                              the container has started before li
veness
                                              probes are initiated. More info: ht
tps://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes'
                                            format: int32
                                            type: integer
                                          periodSeconds:
                                            description: How often (in seconds)
                                              to perform the probe. Default to 10
                                              seconds. Minimum value is 1.
```

```
                                        format: int32
                                        type: integer
                                      successThreshold:
                                        description: Minimum consecutive succ
esses
                                          for the probe to be considered succ
essful
                                          after having failed. Defaults to 1.
                                          Must be 1 for liveness and startup.
                                          Minimum value is 1.
                                        format: int32
                                        type: integer
                                      tcpSocket:
                                        description: 'TCPSocket specifies an
                                          action involving a TCP port. TCP ho
oks
                                          not yet supported TODO: implement
                                          a realistic TCP lifecycle hook'
                                        properties:
                                          host:
                                            description: 'Optional: Host name
                                              to connect to, defaults to the
                                              pod IP.'
                                            type: string
                                          port:
                                            anyOf:
                                            - type: integer
                                            - type: string
                                            description: Number or name of th
e
                                              port to access on the container
.
                                              Number must be in the range 1
                                              to 65535. Name must be an IANA_
SVC_NAME.
                                            x-kubernetes-int-or-string: true
                                        required:
                                        - port
                                        type: object
                                      terminationGracePeriodSeconds:
                                        description: Optional duration in sec
onds
                                          the pod needs to terminate graceful
ly
                                          upon probe failure. The grace perio
d
                                          is the duration in seconds after th
e
                                          processes running in the pod are se
nt
                                          a termination signal and the time
                                          when the processes are forcibly hal
ted
                                          with a kill signal. Set this value
```

```
                                                longer than the expected cleanup ti
me
                                                for your process. If this value is
                                                nil, the pod's terminationGracePeri
odSeconds
                                                will be used. Otherwise, this value
                                                overrides the value provided by the
                                                pod spec. Value must be non-negativ
e
                                                integer. The value zero indicates
                                                stop immediately via the kill signa
l
                                                (no opportunity to shut down). This
                                                is a beta field and requires enabli
ng
                                                ProbeTerminationGracePeriod feature
                                                gate. Minimum value is 1. spec.term
inationGracePeriodSeconds
                                                is used if unset.
                                              format: int64
                                              type: integer
                                            timeoutSeconds:
                                              description: 'Number of seconds after
                                                which the probe times out. Defaults
                                                to 1 second. Minimum value is 1. Mo
re
                                                info: https://kubernetes.io/docs/co
ncepts/workloads/pods/pod-lifecycle#container-probes'
                                              format: int32
                                              type: integer
                                          type: object
                                        resources:
                                          description: 'Compute Resources required
                                            by this container. Cannot be updated.
                                            More info: https://kubernetes.io/docs/c
oncepts/configuration/manage-resources-containers/'
                                          properties:
                                            limits:
                                              additionalProperties:
                                                anyOf:
                                                - type: integer
                                                - type: string
                                                pattern: ^(\+|-)?(([0-9]+(\.[0-9]*)
?)|(\.[0-9]+))(([KMGTPE]i)|[numkMGTPE]|([eE](\+|-)?(([0-9]+(\.[0-9]*)?)|(\.[0-9]+))
))?$
                                                x-kubernetes-int-or-string: true
                                              description: 'Limits describes the ma
ximum
                                                amount of compute resources allowed
.
                                                More info: https://kubernetes.io/do
cs/concepts/configuration/manage-resources-containers/'
                                              type: object
                                            requests:
```

```
                                                  additionalProperties:
                                                    anyOf:
                                                    - type: integer
                                                    - type: string
                                                    pattern: ^(\+|-)?(([0-9]+(\.[0-9]*)
?)|(\.[0-9]+))(([KMGTPE]i)|[numkMGTPE]|([eE](\+|-)?(([0-9]+(\.[0-9]*)?)|(\.[0-9]+))
))?$
                                                    x-kubernetes-int-or-string: true
                                                  description: 'Requests describes the
                                                    minimum amount of compute resources
                                                    required. If Requests is omitted fo
r
                                                    a container, it defaults to Limits
                                                    if that is explicitly specified, ot
herwise
                                                    to an implementation-defined value.
                                                    More info: https://kubernetes.io/do
cs/concepts/configuration/manage-resources-containers/'
                                                  type: object
                                                type: object
                                              securityContext:
                                                description: 'SecurityContext defines the
                                                  security options the container should
                                                  be run with. If set, the fields of Secu
rityContext
                                                  override the equivalent fields of PodSe
curityContext.
                                                  More info: https://kubernetes.io/docs/t
asks/configure-pod-container/security-context/'
                                                properties:
                                                  allowPrivilegeEscalation:
                                                    description: 'AllowPrivilegeEscalatio
n
                                                      controls whether a process can gain
                                                      more privileges than its parent pro
cess.
                                                      This bool directly controls if the
                                                      no_new_privs flag will be set on th
e
                                                      container process. AllowPrivilegeEs
calation
                                                      is true always when the container
                                                      is: 1) run as Privileged 2) has CAP
_SYS_ADMIN'
                                                    type: boolean
                                                  capabilities:
                                                    description: The capabilities to add/
drop
                                                      when running containers. Defaults
                                                      to the default set of capabilities
                                                      granted by the container runtime.
                                                    properties:
                                                      add:
                                                        description: Added capabilities
```

```
                                                items:
                                                  description: Capability represe
nt
                                                    POSIX capabilities type
                                                  type: string
                                                type: array
                                              drop:
                                                description: Removed capabilities
                                                items:
                                                  description: Capability represe
nt
                                                    POSIX capabilities type
                                                  type: string
                                                type: array
                                            type: object
                                          privileged:
                                            description: Run container in privile
ged
                                              mode. Processes in privileged conta
iners
                                              are essentially equivalent to root
                                              on the host. Defaults to false.
                                            type: boolean
                                          procMount:
                                            description: procMount denotes the ty
pe
                                              of proc mount to use for the contai
ners.
                                              The default is DefaultProcMount whi
ch
                                              uses the container runtime defaults
                                              for readonly paths and masked paths
.
                                              This requires the ProcMountType fea
ture
                                              flag to be enabled.
                                            type: string
                                          readOnlyRootFilesystem:
                                            description: Whether this container
                                              has a read-only root filesystem. De
fault
                                              is false.
                                            type: boolean
                                          runAsGroup:
                                            description: The GID to run the entry
point
                                              of the container process. Uses runt
ime
                                              default if unset. May also be set
                                              in PodSecurityContext.  If set in
                                              both SecurityContext and PodSecurit
yContext,
                                              the value specified in SecurityCont
ext
                                              takes precedence
```

```
                                                takes precedence.
                                              format: int64
                                              type: integer
                                            runAsNonRoot:
                                              description: Indicates that the conta
iner
                                                must run as a non-root user. If tru
e,
                                                the Kubelet will validate the image
                                                at runtime to ensure that it does
                                                not run as UID 0 (root) and fail to
                                                start the container if it does. If
                                                unset or false, no such validation
                                                will be performed. May also be set
                                                in PodSecurityContext.  If set in
                                                both SecurityContext and PodSecurit
yContext,
                                                the value specified in SecurityCont
ext
                                                takes precedence.
                                              type: boolean
                                            runAsUser:
                                              description: The UID to run the entry
point
                                                of the container process. Defaults
                                                to user specified in image metadata
                                                if unspecified. May also be set in
                                                PodSecurityContext.  If set in both
                                                SecurityContext and PodSecurityCont
ext,
                                                the value specified in SecurityCont
ext
                                                takes precedence.
                                              format: int64
                                              type: integer
                                            seLinuxOptions:
                                              description: The SELinux context to
                                                be applied to the container. If uns
pecified,
                                                the container runtime will allocate
                                                a random SELinux context for each
                                                container.  May also be set in PodS
ecurityContext.  If
                                                set in both SecurityContext and Pod
SecurityContext,
                                                the value specified in SecurityCont
ext
                                                takes precedence.
                                              properties:
                                                level:
                                                  description: Level is SELinux lev
el
                                                    label that applies to the conta
iner.
                                                  type: string
```

```
                                                role:
                                                  description: Role is a SELinux ro
le
                                                    label that applies to the conta
iner.
                                                  type: string
                                                type:
                                                  description: Type is a SELinux ty
pe
                                                    label that applies to the conta
iner.
                                                  type: string
                                                user:
                                                  description: User is a SELinux us
er
                                                    label that applies to the conta
iner.
                                                  type: string
                                              type: object
                                            seccompProfile:
                                              description: The seccomp options to
                                                use by this container. If seccomp
                                                options are provided at both the po
d
                                                & container level, the container op
tions
                                                override the pod options.
                                              properties:
                                                localhostProfile:
                                                  description: localhostProfile ind
icates
                                                    a profile defined in a file on
                                                    the node should be used. The pr
ofile
                                                    must be preconfigured on the no
de
                                                    to work. Must be a descending
                                                    path, relative to the kubelet's
                                                    configured seccomp profile loca
tion.
                                                    Must only be set if type is "Lo
calhost".
                                                  type: string
                                                type:
                                                  description: "type indicates whic
h
                                                    kind of seccomp profile will be
                                                    applied. Valid options are: \n
                                                    Localhost - a profile defined
                                                    in a file on the node should be
                                                    used. RuntimeDefault - the cont
ainer
                                                    runtime default profile should
                                                    be used. Unconfined - no profil
```

```
e
                                                  should be applied."
                                                type: string
                                            required:
                                            - type
                                            type: object
                                          windowsOptions:
                                            description: The Windows specific set
tings
                                              applied to all containers. If unspe
cified,
                                              the options from the PodSecurityCon
text
                                              will be used. If set in both Securi
tyContext
                                              and PodSecurityContext, the value
                                              specified in SecurityContext takes
                                              precedence.
                                            properties:
                                              gmsaCredentialSpec:
                                                description: GMSACredentialSpec
                                                  is where the GMSA admission web
hook
                                                  (https://github.com/kubernetes-
sigs/windows-gmsa)
                                                  inlines the contents of the GMS
A
                                                  credential spec named by the GM
SACredentialSpecName
                                                  field.
                                                type: string
                                              gmsaCredentialSpecName:
                                                description: GMSACredentialSpecNa
me
                                                  is the name of the GMSA credent
ial
                                                  spec to use.
                                                type: string
                                              hostProcess:
                                                description: HostProcess determin
es
                                                  if a container should be run as
                                                  a 'Host Process' container. Thi
s
                                                  field is alpha-level and will
                                                  only be honored by components
                                                  that enable the WindowsHostProc
essContainers
                                                  feature flag. Setting this fiel
d
                                                  without the feature flag will
                                                  result in errors when validatin
g
                                                  the Pod. All of a Pod's contain
```

```
ers
                                                must have the same effective Ho
stProcess
                                                value (it is not allowed to hav
e
                                                a mix of HostProcess containers
                                                and non-HostProcess containers)
.  In
                                                addition, if HostProcess is tru
e
                                                then HostNetwork must also be
                                                set to true.
                                              type: boolean
                                            runAsUserName:
                                              description: The UserName in Wind
ows
                                                to run the entrypoint of the co
ntainer
                                                process. Defaults to the user
                                                specified in image metadata if
                                                unspecified. May also be set in
                                                PodSecurityContext. If set in
                                                both SecurityContext and PodSec
urityContext,
                                                the value specified in Security
Context
                                                takes precedence.
                                              type: string
                                          type: object
                                        type: object
                                      startupProbe:
                                        description: 'StartupProbe indicates that
                                          the Pod has successfully initialized.
                                          If specified, no other probes are execu
ted
                                          until this completes successfully. If
                                          this probe fails, the Pod will be resta
rted,
                                          just as if the livenessProbe failed. Th
is
                                          can be used to provide different probe
                                          parameters at the beginning of a Pod''s
                                          lifecycle, when it might take a long ti
me
                                          to load data or warm a cache, than duri
ng
                                          steady-state operation. This cannot be
                                          updated. More info: https://kubernetes.
io/docs/concepts/workloads/pods/pod-lifecycle#container-probes'
                                        properties:
                                          exec:
                                            description: One and only one of the
                                              following should be specified. Exec
                                              specifies the action to take.
```

```
                                        properties:
                                          command:
                                            description: Command is the comma
nd
                                              line to execute inside the cont
ainer,
                                              the working directory for the
                                              command  is root ('/') in the
                                              container's filesystem. The com
mand
                                              is simply exec'd, it is not run
                                              inside a shell, so traditional
                                              shell instructions ('|', etc)
                                              won't work. To use a shell, you
                                              need to explicitly call out to
                                              that shell. Exit status of 0 is
                                              treated as live/healthy and non
-zero
                                              is unhealthy.
                                            items:
                                              type: string
                                            type: array
                                          type: object
                                        failureThreshold:
                                          description: Minimum consecutive fail
ures
                                            for the probe to be considered fail
ed
                                            after having succeeded. Defaults to
                                            3. Minimum value is 1.
                                          format: int32
                                          type: integer
                                        httpGet:
                                          description: HTTPGet specifies the ht
tp
                                            request to perform.
                                          properties:
                                            host:
                                              description: Host name to connect
                                                to, defaults to the pod IP. You
                                                probably want to set "Host" in
                                                httpHeaders instead.
                                              type: string
                                            httpHeaders:
                                              description: Custom headers to se
t
                                                in the request. HTTP allows rep
eated
                                                headers.
                                              items:
                                                description: HTTPHeader describ
es
                                                  a custom header to be used in
                                                  HTTP probes
                                                properties:
```

```
                                                  properties:
                                                    name:
                                                      description: The header fie
ld
                                                        name
                                                      type: string
                                                    value:
                                                      description: The header fie
ld
                                                        value
                                                      type: string
                                                  required:
                                                  - name
                                                  - value
                                                  type: object
                                                type: array
                                              path:
                                                description: Path to access on th
e
                                                  HTTP server.
                                                type: string
                                              port:
                                                anyOf:
                                                - type: integer
                                                - type: string
                                                description: Name or number of th
e
                                                  port to access on the container
.
                                                  Number must be in the range 1
                                                  to 65535. Name must be an IANA_
SVC_NAME.
                                                x-kubernetes-int-or-string: true
                                              scheme:
                                                description: Scheme to use for co
nnecting
                                                  to the host. Defaults to HTTP.
                                                type: string
                                            required:
                                            - port
                                            type: object
                                          initialDelaySeconds:
                                            description: 'Number of seconds after
                                              the container has started before li
veness
                                              probes are initiated. More info: ht
tps://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes'
                                            format: int32
                                            type: integer
                                          periodSeconds:
                                            description: How often (in seconds)
                                              to perform the probe. Default to 10
                                              seconds. Minimum value is 1.
                                            format: int32
                                            type: integer
```

```
                                                successThreshold:
                                                  description: Minimum consecutive succ
esses
                                                    for the probe to be considered succ
essful
                                                    after having failed. Defaults to 1.
                                                    Must be 1 for liveness and startup.
                                                    Minimum value is 1.
                                                  format: int32
                                                  type: integer
                                                tcpSocket:
                                                  description: 'TCPSocket specifies an
                                                    action involving a TCP port. TCP ho
oks
                                                    not yet supported TODO: implement
                                                    a realistic TCP lifecycle hook'
                                                  properties:
                                                    host:
                                                      description: 'Optional: Host name
                                                        to connect to, defaults to the
                                                        pod IP.'
                                                      type: string
                                                    port:
                                                      anyOf:
                                                      - type: integer
                                                      - type: string
                                                      description: Number or name of th
e
                                                        port to access on the container
.
                                                        Number must be in the range 1
                                                        to 65535. Name must be an IANA_
SVC_NAME.
                                                      x-kubernetes-int-or-string: true
                                                  required:
                                                  - port
                                                  type: object
                                                terminationGracePeriodSeconds:
                                                  description: Optional duration in sec
onds
                                                    the pod needs to terminate graceful
ly
                                                    upon probe failure. The grace perio
d
                                                    is the duration in seconds after th
e
                                                    processes running in the pod are se
nt
                                                    a termination signal and the time
                                                    when the processes are forcibly hal
ted
                                                    with a kill signal. Set this value
                                                    longer than the expected cleanup ti
me
```

```
                                                for your process. If this value is
                                                nil, the pod's terminationGracePeri
odSeconds
                                                will be used. Otherwise, this value
                                                overrides the value provided by the
                                                pod spec. Value must be non-negativ
e
                                                integer. The value zero indicates
                                                stop immediately via the kill signa
l
                                                (no opportunity to shut down). This
                                                is a beta field and requires enabli
ng
                                                ProbeTerminationGracePeriod feature
                                                gate. Minimum value is 1. spec.term
inationGracePeriodSeconds
                                                is used if unset.
                                              format: int64
                                              type: integer
                                            timeoutSeconds:
                                              description: 'Number of seconds after
                                                which the probe times out. Defaults
                                                to 1 second. Minimum value is 1. Mo
re
                                                info: https://kubernetes.io/docs/co
ncepts/workloads/pods/pod-lifecycle#container-probes'
                                              format: int32
                                              type: integer
                                          type: object
                                        stdin:
                                          description: Whether this container shoul
d
                                            allocate a buffer for stdin in the cont
ainer
                                            runtime. If this is not set, reads from
                                            stdin in the container will always resu
lt
                                            in EOF. Default is false.
                                          type: boolean
                                        stdinOnce:
                                          description: Whether the container runtim
e
                                            should close the stdin channel after it
                                            has been opened by a single attach. Whe
n
                                            stdin is true the stdin stream will rem
ain
                                            open across multiple attach sessions.
                                            If stdinOnce is set to true, stdin is
                                            opened on container start, is empty unt
il
                                            the first client attaches to stdin, and
                                            then remains open and accepts data unti
l
```

```
                                    the client disconnects, at which time
                                    stdin is closed and remains closed unti
l
                                    the container is restarted. If this fla
g
                                    is false, a container processes that re
ads
                                    from stdin will never receive an EOF.
                                    Default is false
                                  type: boolean
                                terminationMessagePath:
                                  description: 'Optional: Path at which the
                                    file to which the container''s terminat
ion
                                    message will be written is mounted into
                                    the container''s filesystem. Message wr
itten
                                    is intended to be brief final status,
                                    such as an assertion failure message.
                                    Will be truncated by the node if greate
r
                                    than 4096 bytes. The total message leng
th
                                    across all containers will be limited
                                    to 12kb. Defaults to /dev/termination-l
og.
                                    Cannot be updated.'
                                  type: string
                                terminationMessagePolicy:
                                  description: Indicate how the termination
                                    message should be populated. File will
                                    use the contents of terminationMessageP
ath
                                    to populate the container status messag
e
                                    on both success and failure. FallbackTo
LogsOnError
                                    will use the last chunk of container lo
g
                                    output if the termination message file
                                    is empty and the container exited with
                                    an error. The log output is limited to
                                    2048 bytes or 80 lines, whichever is sm
aller.
                                    Defaults to File. Cannot be updated.
                                  type: string
                                tty:
                                  description: Whether this container shoul
d
                                    allocate a TTY for itself, also require
s
                                    'stdin' to be true. Default is false.
                                  type: boolean
                                volumeDevices:
```

```
                                        description: volumeDevices is the list of
                                          block devices to be used by the contain
er.
                                        items:
                                          description: volumeDevice describes a
                                            mapping of a raw block device within
                                            a container.
                                          properties:
                                            devicePath:
                                              description: devicePath is the path
                                                inside of the container that the
                                                device will be mapped to.
                                              type: string
                                            name:
                                              description: name must match the na
me
                                                of a persistentVolumeClaim in the
                                                pod
                                              type: string
                                          required:
                                          - devicePath
                                          - name
                                          type: object
                                        type: array
                                      volumeMounts:
                                        description: Pod volumes to mount into th
e
                                          container's filesystem. Cannot be updat
ed.
                                        items:
                                          description: VolumeMount describes a mo
unting
                                            of a Volume within a container.
                                          properties:
                                            mountPath:
                                              description: Path within the contai
ner
                                                at which the volume should be mou
nted.  Must
                                                not contain ':'.
                                              type: string
                                            mountPropagation:
                                              description: mountPropagation deter
mines
                                                how mounts are propagated from th
e
                                                host to container and the other
                                                way around. When not set, MountPr
opagationNone
                                                is used. This field is beta in 1.
10.
                                              type: string
                                            name:
                                              description: This must match the Na
```

```
me
                                                of a Volume.
                                              type: string
                                            readOnly:
                                              description: Mounted read-only if
                                                true, read-write otherwise (false
                                                or unspecified). Defaults to fals
e.
                                              type: boolean
                                            subPath:
                                              description: Path within the volume
                                                from which the container's volume
                                                should be mounted. Defaults to ""
                                                (volume's root).
                                              type: string
                                            subPathExpr:
                                              description: Expanded path within
                                                the volume from which the contain
er's
                                                volume should be mounted. Behaves
                                                similarly to SubPath but environm
ent
                                                variable references $(VAR_NAME)
                                                are expanded using the container'
s
                                                environment. Defaults to "" (volu
me's
                                                root). SubPathExpr and SubPath ar
e
                                                mutually exclusive.
                                              type: string
                                          required:
                                          - mountPath
                                          - name
                                          type: object
                                        type: array
                                      workingDir:
                                        description: Container's working director
y.
                                          If not specified, the container runtime
's
                                          default will be used, which might be co
nfigured
                                          in the container image. Cannot be updat
ed.
                                        type: string
                                    required:
                                    - name
                                    type: object
                                  type: array
                                timeout:
                                  description: Timeout defines the maximum amount
                                    of time Velero should wait for the initContai
ners
                                    to complete.
```

```
                                    type: string
                                type: object
                            type: object
                        type: array
                    required:
                    - name
                    type: object
                type: array
            type: object
          includeClusterResources:
            description: IncludeClusterResources specifies whether cluster-scop
ed
              resources should be included for consideration in the restore. If
              null, defaults to true.
            nullable: true
            type: boolean
          includedNamespaces:
            description: IncludedNamespaces is a slice of namespace names to in
clude
              objects from. If empty, all namespaces are included.
            items:
              type: string
            nullable: true
            type: array
          includedResources:
            description: IncludedResources is a slice of resource names to incl
ude
              in the restore. If empty, all resources in the backup are include
d.
            items:
              type: string
            nullable: true
            type: array
          labelSelector:
            description: LabelSelector is a metav1.LabelSelector to filter with
              when restoring individual objects from the backup. If empty or ni
l,
              all objects are included. Optional.
            nullable: true
            properties:
              matchExpressions:
                description: matchExpressions is a list of label selector requi
rements.
                  The requirements are ANDed.
                items:
                  description: A label selector requirement is a selector that
                    contains values, a key, and an operator that relates the ke
y
                    and values.
                  properties:
                    key:
                      description: key is the label key that the selector appli
es
                        to.
```

```
                        type: string
                      operator:
                        description: operator represents a key's relationship to
                          a set of values. Valid operators are In, NotIn, Exists
                          and DoesNotExist.
                        type: string
                      values:
                        description: values is an array of string values. If the
                          operator is In or NotIn, the values array must be non-e
mpty.
                          If the operator is Exists or DoesNotExist, the values
                          array must be empty. This array is replaced during a st
rategic
                          merge patch.
                        items:
                          type: string
                        type: array
                    required:
                    - key
                    - operator
                    type: object
                  type: array
                matchLabels:
                  additionalProperties:
                    type: string
                  description: matchLabels is a map of {key,value} pairs. A singl
e
                    {key,value} in the matchLabels map is equivalent to an elemen
t
                    of matchExpressions, whose key field is "key", the operator
                    is "In", and the values array contains only "value". The requ
irements
                    are ANDed.
                  type: object
              type: object
            namespaceMapping:
              additionalProperties:
                type: string
              description: NamespaceMapping is a map of source namespace names to
                target namespace names to restore into. Any source namespaces not
                included in the map will be restored into namespaces of the same
                name.
              type: object
            preserveNodePorts:
              description: PreserveNodePorts specifies whether to restore old nod
ePorts
                from backup.
              nullable: true
              type: boolean
            restorePVs:
              description: RestorePVs specifies whether to restore all included
                PVs from snapshot (via the cloudprovider).
              nullable: true
              type: boolean
```

```
              scheduleName:
                description: ScheduleName is the unique name of the Velero schedule
                  to restore from. If specified, and BackupName is empty, Velero wi
ll
                  restore from the most recent successful backup created from this
                  schedule.
                type: string
            required:
            - backupName
            type: object
          status:
            description: RestoreStatus captures the current status of a Velero rest
ore
            properties:
              completionTimestamp:
                description: CompletionTimestamp records the time the restore opera
tion
                  was completed. Completion time is recorded even on failed restore
.
                  The server's time is used for StartTimestamps
                format: date-time
                nullable: true
                type: string
              errors:
                description: Errors is a count of all error messages that were gene
rated
                  during execution of the restore. The actual errors are stored in
                  object storage.
                type: integer
              failureReason:
                description: FailureReason is an error that caused the entire resto
re
                  to fail.
                type: string
              phase:
                description: Phase is the current state of the Restore
                enum:
                - New
                - FailedValidation
                - InProgress
                - Completed
                - PartiallyFailed
                - Failed
                type: string
              progress:
                description: Progress contains information about the restore's exec
ution
                  progress. Note that this information is best-effort only -- if Ve
lero
                  fails to update it during a restore for any reason, it may be ina
ccurate/stale.
                nullable: true
                properties:
                  itemsRestored:
```

```
                    description: ItemsRestored is the number of items that have act
ually
                      been restored so far
                    type: integer
                  totalItems:
                    description: TotalItems is the total number of items to be rest
ored.
                      This number may change throughout the execution of the restor
e
                      due to plugins that return additional related items to restor
e
                    type: integer
                type: object
              startTimestamp:
                description: StartTimestamp records the time the restore operation
                  was started. The server's time is used for StartTimestamps
                format: date-time
                nullable: true
                type: string
              validationErrors:
                description: ValidationErrors is a slice of all validation errors
                  (if applicable)
                items:
                  type: string
                nullable: true
                type: array
              warnings:
                description: Warnings is a count of all warning messages that were
                  generated during execution of the restore. The actual warnings ar
e
                  stored in object storage.
                type: integer
            type: object
        type: object
    served: true
    storage: true
---
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.7.0
  creationTimestamp: null
  name: podvolumebackups.velero.io
spec:
  group: velero.io
  names:
    kind: PodVolumeBackup
    listKind: PodVolumeBackupList
    plural: podvolumebackups
    singular: podvolumebackup
  scope: Namespaced
  versions:
```

```
  - name: v1
    schema:
      openAPIV3Schema:
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation
              of an object. Servers should convert recognized schemas to the latest
              internal value, and may reject unrecognized values. More info: https:
//git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resou
rces'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s
              object represents. Servers may infer this from the endpoint the clien
t
              submits requests to. Cannot be updated. In CamelCase. More info: http
s://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#typ
es-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: PodVolumeBackupSpec is the specification for a PodVolumeBa
ckup.
            properties:
              backupStorageLocation:
                description: BackupStorageLocation is the name of the backup storag
e
                  location where the restic repository is stored.
                type: string
              node:
                description: Node is the name of the node that the Pod is running
                  on.
                type: string
              pod:
                description: Pod is a reference to the pod containing the volume to
                  be backed up.
                properties:
                  apiVersion:
                    description: API version of the referent.
                    type: string
                  fieldPath:
                    description: 'If referring to a piece of an object instead of
                      an entire object, this string should contain a valid JSON/Go
                      field access statement, such as desiredState.manifest.contain
ers[2].
                      For example, if the object reference is to a container within
                      a pod, this would take on a value like: "spec.containers{name
}"
                      (where "name" refers to the name of the container that trigge
red
                      the event) or if no container name is specified "spec.contain
ers[2]"
```

```
                      (container with index 2 in this pod). This syntax is chosen
                      only to have some well-defined way of referencing a part of
                      an object. TODO: this design is not final and this field is
                      subject to change in the future.'
                    type: string
                  kind:
                    description: 'Kind of the referent. More info: https://git.k8s.
io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds'
                    type: string
                  name:
                    description: 'Name of the referent. More info: https://kubernet
es.io/docs/concepts/overview/working-with-objects/names/#names'
                    type: string
                  namespace:
                    description: 'Namespace of the referent. More info: https://kub
ernetes.io/docs/concepts/overview/working-with-objects/namespaces/'
                    type: string
                  resourceVersion:
                    description: 'Specific resourceVersion to which this reference
                      is made, if any. More info: https://git.k8s.io/community/cont
ributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consiste
ncy'
                    type: string
                  uid:
                    description: 'UID of the referent. More info: https://kubernete
s.io/docs/concepts/overview/working-with-objects/names/#uids'
                    type: string
                type: object
              repoIdentifier:
                description: RepoIdentifier is the restic repository identifier.
                type: string
              tags:
                additionalProperties:
                  type: string
                description: Tags are a map of key-value pairs that should be appli
ed
                  to the volume backup as tags.
                type: object
              volume:
                description: Volume is the name of the volume within the Pod to be
                  backed up.
                type: string
            required:
            - backupStorageLocation
            - node
            - pod
            - repoIdentifier
            - volume
            type: object
          status:
            description: PodVolumeBackupStatus is the current status of a PodVolume
Backup.
            properties:
              completionTimestamp:
```

```
                description: CompletionTimestamp records the time a backup was comp
leted.
                  Completion time is recorded even on failed backups. Completion ti
me
                  is recorded before uploading the backup object. The server's time
                  is used for CompletionTimestamps
                format: date-time
                nullable: true
                type: string
              message:
                description: Message is a message about the pod volume backup's sta
tus.
                type: string
              path:
                description: Path is the full path within the controller pod being
                  backed up.
                type: string
              phase:
                description: Phase is the current state of the PodVolumeBackup.
                enum:
                - New
                - InProgress
                - Completed
                - Failed
                type: string
              progress:
                description: Progress holds the total number of bytes of the volume
                  and the current number of backed up bytes. This can be used to di
splay
                  progress information about the backup operation.
                properties:
                  bytesDone:
                    format: int64
                    type: integer
                  totalBytes:
                    format: int64
                    type: integer
                type: object
              snapshotID:
                description: SnapshotID is the identifier for the snapshot of the
                  pod volume.
                type: string
              startTimestamp:
                description: StartTimestamp records the time a backup was started.
                  Separate from CreationTimestamp, since that value changes on rest
ores.
                  The server's time is used for StartTimestamps
                format: date-time
                nullable: true
                type: string
            type: object
        type: object
    served: true
    storage: true
```

```
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.7.0
  creationTimestamp: null
  name: podvolumerestores.velero.io
spec:
  group: velero.io
  names:
    kind: PodVolumeRestore
    listKind: PodVolumeRestoreList
    plural: podvolumerestores
    singular: podvolumerestore
  scope: Namespaced
  versions:
  - name: v1
    schema:
      openAPIV3Schema:
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation
              of an object. Servers should convert recognized schemas to the latest
              internal value, and may reject unrecognized values. More info: https:
//git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resou
rces'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s
              object represents. Servers may infer this from the endpoint the clien
t
              submits requests to. Cannot be updated. In CamelCase. More info: http
s://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#typ
es-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: PodVolumeRestoreSpec is the specification for a PodVolumeR
estore.
            properties:
              backupStorageLocation:
                description: BackupStorageLocation is the name of the backup storag
e
                  location where the restic repository is stored.
                type: string
              pod:
                description: Pod is a reference to the pod containing the volume to
                  be restored.
                properties:
                  apiVersion:
```

```
                    description: API version of the referent.
                    type: string
                  fieldPath:
                    description: 'If referring to a piece of an object instead of
                      an entire object, this string should contain a valid JSON/Go
                      field access statement, such as desiredState.manifest.contain
ers[2].
                      For example, if the object reference is to a container within
                      a pod, this would take on a value like: "spec.containers{name
}"
                      (where "name" refers to the name of the container that trigge
red
                      the event) or if no container name is specified "spec.contain
ers[2]"
                      (container with index 2 in this pod). This syntax is chosen
                      only to have some well-defined way of referencing a part of
                      an object. TODO: this design is not final and this field is
                      subject to change in the future.'
                    type: string
                  kind:
                    description: 'Kind of the referent. More info: https://git.k8s.
io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds'
                    type: string
                  name:
                    description: 'Name of the referent. More info: https://kubernet
es.io/docs/concepts/overview/working-with-objects/names/#names'
                    type: string
                  namespace:
                    description: 'Namespace of the referent. More info: https://kub
ernetes.io/docs/concepts/overview/working-with-objects/namespaces/'
                    type: string
                  resourceVersion:
                    description: 'Specific resourceVersion to which this reference
                      is made, if any. More info: https://git.k8s.io/community/cont
ributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consiste
ncy'
                    type: string
                  uid:
                    description: 'UID of the referent. More info: https://kubernete
s.io/docs/concepts/overview/working-with-objects/names/#uids'
                    type: string
                type: object
              repoIdentifier:
                description: RepoIdentifier is the restic repository identifier.
                type: string
              snapshotID:
                description: SnapshotID is the ID of the volume snapshot to be rest
ored.
                type: string
              volume:
                description: Volume is the name of the volume within the Pod to be
                  restored.
                type: string
            required:
```

```
            - backupStorageLocation
            - pod
            - repoIdentifier
            - snapshotID
            - volume
            type: object
          status:
            description: PodVolumeRestoreStatus is the current status of a PodVolum
eRestore.
            properties:
              completionTimestamp:
                description: CompletionTimestamp records the time a restore was com
pleted.
                  Completion time is recorded even on failed restores. The server's
                  time is used for CompletionTimestamps
                format: date-time
                nullable: true
                type: string
              message:
                description: Message is a message about the pod volume restore's st
atus.
                type: string
              phase:
                description: Phase is the current state of the PodVolumeRestore.
                enum:
                - New
                - InProgress
                - Completed
                - Failed
                type: string
              progress:
                description: Progress holds the total number of bytes of the snapsh
ot
                  and the current number of restored bytes. This can be used to dis
play
                  progress information about the restore operation.
                properties:
                  bytesDone:
                    format: int64
                    type: integer
                  totalBytes:
                    format: int64
                    type: integer
                type: object
              startTimestamp:
                description: StartTimestamp records the time a restore was started.
                  The server's time is used for StartTimestamps
                format: date-time
                nullable: true
                type: string
            type: object
        type: object
    served: true
    storage: true
---
```

```
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.7.0
  creationTimestamp: null
  name: serverstatusrequests.velero.io
spec:
  group: velero.io
  names:
    kind: ServerStatusRequest
    listKind: ServerStatusRequestList
    plural: serverstatusrequests
    shortNames:
    - ssr
    singular: serverstatusrequest
  scope: Namespaced
  versions:
  - name: v1
    schema:
      openAPIV3Schema:
        description: ServerStatusRequest is a request to access current status info
rmation
          about the Velero server.
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation
              of an object. Servers should convert recognized schemas to the latest
              internal value, and may reject unrecognized values. More info: https:
//git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resou
rces'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s
              object represents. Servers may infer this from the endpoint the clien
t
              submits requests to. Cannot be updated. In CamelCase. More info: http
s://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#typ
es-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: ServerStatusRequestSpec is the specification for a ServerS
tatusRequest.
            type: object
          status:
            description: ServerStatusRequestStatus is the current status of a Serve
rStatusRequest.
            properties:
              phase:
                description: Phase is the current lifecycle phase of the ServerStat
```

```
usRequest.
                enum:
                - New
                - Processed
                type: string
              plugins:
                description: Plugins list information about the plugins running on
                  the Velero server
                items:
                  description: PluginInfo contains attributes of a Velero plugin
                  properties:
                    kind:
                      type: string
                    name:
                      type: string
                  required:
                  - kind
                  - name
                  type: object
                nullable: true
                type: array
              processedTimestamp:
                description: ProcessedTimestamp is when the ServerStatusRequest was
                  processed by the ServerStatusRequestController.
                format: date-time
                nullable: true
                type: string
              serverVersion:
                description: ServerVersion is the Velero server version.
                type: string
            type: object
        type: object
    served: true
    storage: true
    subresources:
      status: {}
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.7.0
  creationTimestamp: null
  name: resticrepositories.velero.io
spec:
  group: velero.io
  names:
    kind: ResticRepository
    listKind: ResticRepositoryList
    plural: resticrepositories
    singular: resticrepository
  scope: Namespaced
  versions:
  - name: v1
```

```
    schema:
      openAPIV3Schema:
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation
              of an object. Servers should convert recognized schemas to the latest
              internal value, and may reject unrecognized values. More info: https:
//git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resou
rces'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource thi
s
              object represents. Servers may infer this from the endpoint the clien
t
              submits requests to. Cannot be updated. In CamelCase. More info: http
s://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#typ
es-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: ResticRepositorySpec is the specification for a ResticRepo
sitory.
            properties:
              backupStorageLocation:
                description: BackupStorageLocation is the name of the BackupStorage
Location
                  that should contain this repository.
                type: string
              maintenanceFrequency:
                description: MaintenanceFrequency is how often maintenance should
                  be run.
                type: string
              resticIdentifier:
                description: ResticIdentifier is the full restic-compatible string
                  for identifying this repository.
                type: string
              volumeNamespace:
                description: VolumeNamespace is the namespace this restic repositor
y
                  contains pod volume backups for.
                type: string
            required:
            - backupStorageLocation
            - maintenanceFrequency
            - resticIdentifier
            - volumeNamespace
            type: object
          status:
            description: ResticRepositoryStatus is the current status of a ResticRe
pository.
            properties:
```

```
              lastMaintenanceTime:
                description: LastMaintenanceTime is the last time maintenance was
                  run.
                format: date-time
                nullable: true
                type: string
              message:
                description: Message is a message about the current status of the
                  ResticRepository.
                type: string
              phase:
                description: Phase is the current state of the ResticRepository.
                enum:
                - New
                - Ready
                - NotReady
                type: string
            type: object
        type: object
    served: true
    storage: true
---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  annotations:
    controller-gen.kubebuilder.io/version: v0.7.0
  creationTimestamp: null
  name: volumesnapshotlocations.velero.io
spec:
  group: velero.io
  names:
    kind: VolumeSnapshotLocation
    listKind: VolumeSnapshotLocationList
    plural: volumesnapshotlocations
    singular: volumesnapshotlocation
  scope: Namespaced
  versions:
  - name: v1
    schema:
      openAPIV3Schema:
        description: VolumeSnapshotLocation is a location where Velero stores volum
e
          snapshots.
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this represent
ation
              of an object. Servers should convert recognized schemas to the latest
              internal value, and may reject unrecognized values. More info: https:
//git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resou
rces'
            type: string
          kind:
```

```
            description: 'Kind is a string value representing the REST resource thi
s
              object represents. Servers may infer this from the endpoint the clien
t
              submits requests to. Cannot be updated. In CamelCase. More info: http
s://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#typ
es-kinds'
            type: string
          metadata:
            type: object
          spec:
            description: VolumeSnapshotLocationSpec defines the specification for
              a Velero VolumeSnapshotLocation.
            properties:
              config:
                additionalProperties:
                  type: string
                description: Config is for provider-specific configuration fields.
                type: object
              provider:
                description: Provider is the provider of the volume storage.
                type: string
            required:
            - provider
            type: object
          status:
            description: VolumeSnapshotLocationStatus describes the current status
              of a Velero VolumeSnapshotLocation.
            properties:
              phase:
                description: VolumeSnapshotLocationPhase is the lifecycle phase of
                  a Velero VolumeSnapshotLocation.
                enum:
                - Available
                - Unavailable
                type: string
            type: object
        type: object
    served: true
    storage: true
---
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    component: csdr
  name: csdr-velero
  namespace: csdr
spec:
  progressDeadlineSeconds: 600
  replicas: 1
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      deploy: csdr
```

```
    deploy: csdr
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      annotations:
        prometheus.io/path: /metrics
        prometheus.io/port: "8085"
        prometheus.io/scrape: "true"
      labels:
        component: csdr
        deploy: csdr
    spec:
      containers:
      - args:
        - server
        - --features=
        command:
        - /velero
        env:
        - name: VELERO_SCRATCH_DIR
          value: /scratch
        - name: VELERO_NAMESPACE
          value: "csdr"
        - name: LD_LIBRARY_PATH
          value: /plugins
        - name: VELERO_FOR_ACK
          value: "true"
        - name: IS_HYBRID
          value: "true"
        - name: USE_ADDON_TOKEN
          value: "false"
        - name: ALIBABA_CLOUD_ACCESS_KEY_ID
          value: {{.ACCESSKEYID}}
        - name: ALIBABA_CLOUD_ACCESS_KEY_SECRET
          value: {{.ACEESSKEYSECRET}}
        image: registry.{{.Region}}.aliyuncs.com/acs/velero:1.6.1-ca01808b-aliyun
        imagePullPolicy: Always
        livenessProbe:
          exec:
            command:
            - /bin/sh
            - -c
            - ps -ef|grep velero
          failureThreshold: 3
          initialDelaySeconds: 5
          periodSeconds: 10
          successThreshold: 1
          timeoutSeconds: 1
        name: velero
        ports:
        - containerPort: 8085
```

```
          name: metrics
          protocol: TCP
        readinessProbe:
          exec:
            command:
            - /bin/sh
            - -c
            - ps -ef|grep velero
          failureThreshold: 3
          initialDelaySeconds: 5
          periodSeconds: 10
          successThreshold: 1
          timeoutSeconds: 1
        resources:
          limits:
            cpu: "1"
            memory: 512Mi
          requests:
            cpu: 500m
            memory: 128Mi
        securityContext:
          allowPrivilegeEscalation: false
          readOnlyRootFilesystem: true
          runAsNonRoot: true
        volumeMounts:
        - mountPath: /plugins
          name: plugins
        - mountPath: /scratch
          name: scratch
        - mountPath: /tmp
          name: writeable
      dnsPolicy: ClusterFirst
      initContainers:
      - image: registry.{{.Region}}.aliyuncs.com/acs/velero-plugin-alibabacloud:v1.
6.1-b7e1657-aliyun
        imagePullPolicy: Always
        name: velero-plugin-alibabacloud
        resources: {}
        securityContext:
          allowPrivilegeEscalation: false
          readOnlyRootFilesystem: true
          runAsNonRoot: true
        volumeMounts:
        - mountPath: /target
          name: plugins
      serviceAccount: csdr
      serviceAccountName: csdr
      volumes:
      - emptyDir: {}
        name: plugins
      - emptyDir: {}
        name: scratch
      - emptyDir: {}
        name: writeable
```

```
```

iv. 执行以下命令，部署velero-deploy。

```
kubectl apply -f velero-deploy.yaml
```

创建完成后，在**csdr**命名空间中检查 `csdr-controller` 和 `velero-deploy` 的Pod状态正常，说明备份中心安装成功。

2. 创建OSS Bucket。请参见创建存储空间。

   您需要先创建一个OSS Bucket，用于存储Kubernetes应用数据及其PV数据， 推荐每个Kubernetes集群单独使用各自的OSS Bucket。

   i. 登录OSS管理控制台。

   ii. 您可以在**Bucket列表**页，单击右侧的**创建Bucket**。

   iii. 在**创建Bucket列表**对话框配置Bucket参数。

   本示例中创建的OSS Bucket名称为**cnfs-oss-backup-test**，创建的地域为**华东1（杭州）**。

3. 在ACK集群中安装备份中心。具体操作，请参见步骤一：安装备份服务组件。

步骤二：在自建Kubernetes集群备份应用

1. 使用以下内容，创建*backuplocation.yaml*文件。

   用来声明自建Kubernetes集群中资源备份的位置，您需要将步骤2中创建的OSS Bucket基本信息填入。

```
apiVersion: csdr.alibabacloud.com/v1beta1
kind: BackupLocation
metadata:
  name: cnfs-test
  namespace: csdr    # 固定命名空间。
spec:
  config:
    region: cn-hangzhou
  objectStorage:
    bucket: cnfs-oss-backup-test
    prefix: test1
  provider: alibabacloud
```

2. 执行以下命令，部署backuplocation对象。

```
kubectl apply -f backuplocation.yaml
```

3. 执行以下命令，查看backuplocation的状态。

```
kubectl get backuplocation -n csdr -oyaml
```

预期输出：

```
status:
  lastSyncedTime: "2022-05-23T09:55:21Z"
  lastValidationTime: "2022-05-23T09:55:21Z"
  message: backup location is ok
  phase: Available
```

由预期输出可看到backuplocation的状态为 `Available` ，说明当前集群有访问OSS的权限并可以直接访问。

4. 使用以下内容，创建*applicationbackups.yaml*文件。

   用来声明自建Kubernetes集群中的备份任务。

   > 注意 参数说明如下：
   > - `includedNamespaces` ：需要备份的Namespace范围。
   > - `pvBackup.defaultPvBackup` ：是否需要备份数据卷，如果设置为 `false` ，则只备份YAML数据，不备份数据卷。
   > - `storageLocation` ：本次备份关联的备份中心。
   > - `ttl` ：备份超时时间。

   ```
   apiVersion: csdr.alibabacloud.com/v1beta1
   kind: ApplicationBackup
   metadata:
     name: backup-all
     namespace: csdr
   spec:
     includedNamespaces:
     - default
     pvBackup:
       defaultPvBackup: true
     storageLocation: cnfs-test
     ttl: 720h0m0s
   ```

5. 执行以下命令，部署applicationbackups对象。

   ```
   kubectl apply -f applicationbackups.yaml
   ```

6. 执行以下命令，查看applicationbackups的状态。

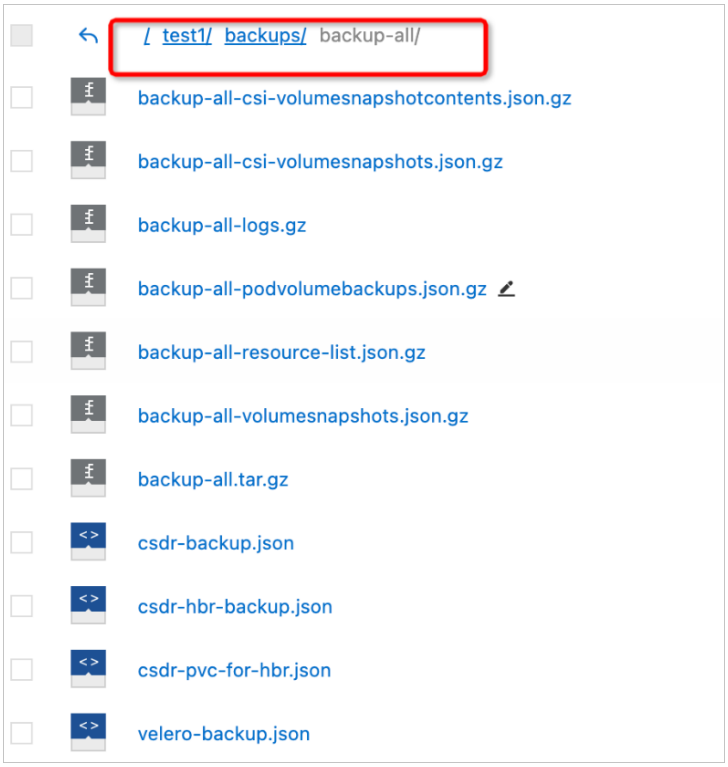
   ```
   kubectl get applicationbackups -n csdr -oyaml
   ```

   预期输出：

   ```
   status:
     completionTimestamp: "2022-05-23T09:55:13Z"
     expiration: "2022-06-22T09:54:53Z"
     message: success
     phase: Completed
     resourceList:
       applicationResource:
         completionTimestamp: "2022-05-23T09:54:55Z"
         phase: Completed
         progress:
           itemsBackedUp: 14
           totalItems: 14
         startTimestamp: "2022-05-23T09:54:53Z"
   ```

   由预期输出可看到applicationbackups的状态为 `success` ，说明任务备份已完成。

备份完成后，您可以登录OSS管理控制台，查看OSS Bucket，可以看到备份的文件。



步骤三：在阿里云ACK集群恢复应用

1. 使用以下内容，在ACK集群中创建和自建Kubernetes集群相同的*backuplocation.yaml*文件。

```
apiVersion: csdr.alibabacloud.com/v1beta1
kind: BackupLocation
metadata:
  name: cnfs-test       # 名字一定要与自建Kubernetes集群的一致。
  namespace: csdr       # 固定命名空间。
spec:
  config:
    region: cn-hangzhou
  objectStorage:        # bucket和prefix必须与自建Kubernetes集群的一致。
    bucket: cnfs-oss-backup-test
    prefix: test1
  provider: alibabacloud
```

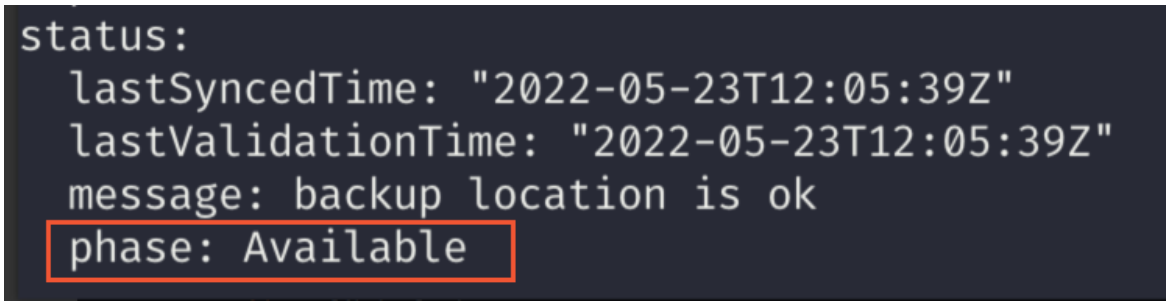
2. 执行以下命令，在ACK集群中部署backuplocation资源（备份仓库）。

```
kubectl apply -f backuplocation.yaml
```

3. 执行以下命令，查看backuplocation的状态。

```
kubectl get backuplocation -n csdr -oyaml
```

预期输出：



由预期输出可看到backuplocation的状态为`Available`，备份中心会自动同步以上备份仓库中创建过的所有备份。

4. 执行以下命令，查看applicationbackups备份任务的状态。

```
kubectl get applicationbackups -n csdr -oyaml
```

预期输出：

```
NAME                AGE
backup-all          35s
backup-all-crds     7d5h
hastest             10d
```

5. 使用以下内容，创建*restore.yaml*文件。

```
apiVersion: csdr.alibabacloud.com/v1beta1
kind: ApplicationRestore
metadata:
  name: idc-backup-all-restore
  namespace: csdr
spec:
  backupName: backup-all    #待恢复的备份任务的名称。
  namespaceMapping:
    default: default1
```

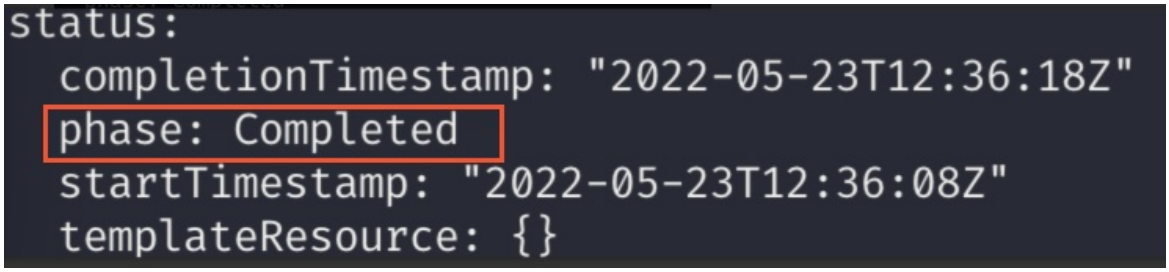
6. 执行以下命令，在ACK集群中创建恢复任务。

```
kubectl apply -f restore.yaml
```

7. 执行以下命令，查看备份任务恢复状态。

```
kubectl get applicationrestore -n csdr -oyaml
```

预期输出：



由预期输出可看到applicationrestore的状态为`Completed`，说明任务恢复完成；同时查看ACK集群中已存在自建Kubernetes集群中需要恢复的命名空间。

步骤四：更新应用配置

应用配置项主要包含镜像地址、服务暴露方式及存储盘挂载等。本例仅涉及更新镜像地址。

1.
2.
3. 在控制台左侧导航栏中，单击**集群**。
4. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
5. 在集群管理页左侧导航栏中，选择**工作负载 > 无状态**。
6. 在顶部选择目标命名空间，找到目标应用后，单击目标应用右侧**操作**列下的**更多 > 查看Yaml**。
7. 在**编辑 YAML**页面把 `image` 字段替换成迁移后的镜像地址后，单击**更新**。

13.Swarm迁移Kubernetes

13.1. 容器服务swarm集群与Kubernetes集群的主要功能比对

13.1.1. 概述

本文将介绍容器服务swarm集群与Kubernetes集群主要功能比对的前提条件及使用限制。

前提条件

您已经成功创建一个 Kubernetes 集群，参见创建Kubernetes专有版集群。

> 说明 。
>
> - 目前容器服务Kubernetes版支持四种集群：经典集群、Kubernetes托管版、多可用区Kubernetes及Serverless Kubernetes（公测）。
> - 本文以创建Kubernetes集群为例，进行容器服务Swarm集群与Kubernetes集群的功能比对。

使用限制

本文主要介绍以下两种场景的功能比对：

- 应用均为无状态应用。
- 应用的数据存在数据库或存储中。

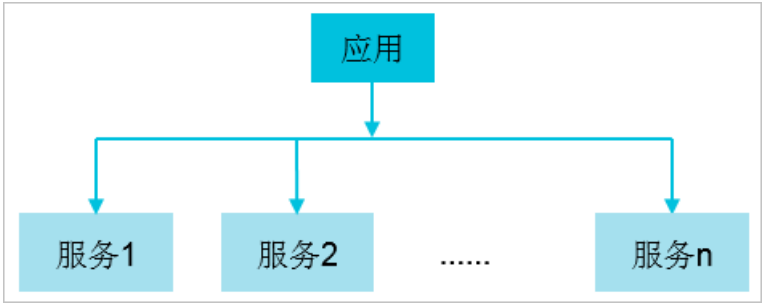
13.1.2. 概念比对

本文主要介绍容器服务Swarm集群与Kubernetes集群主要概念的比对。

应用

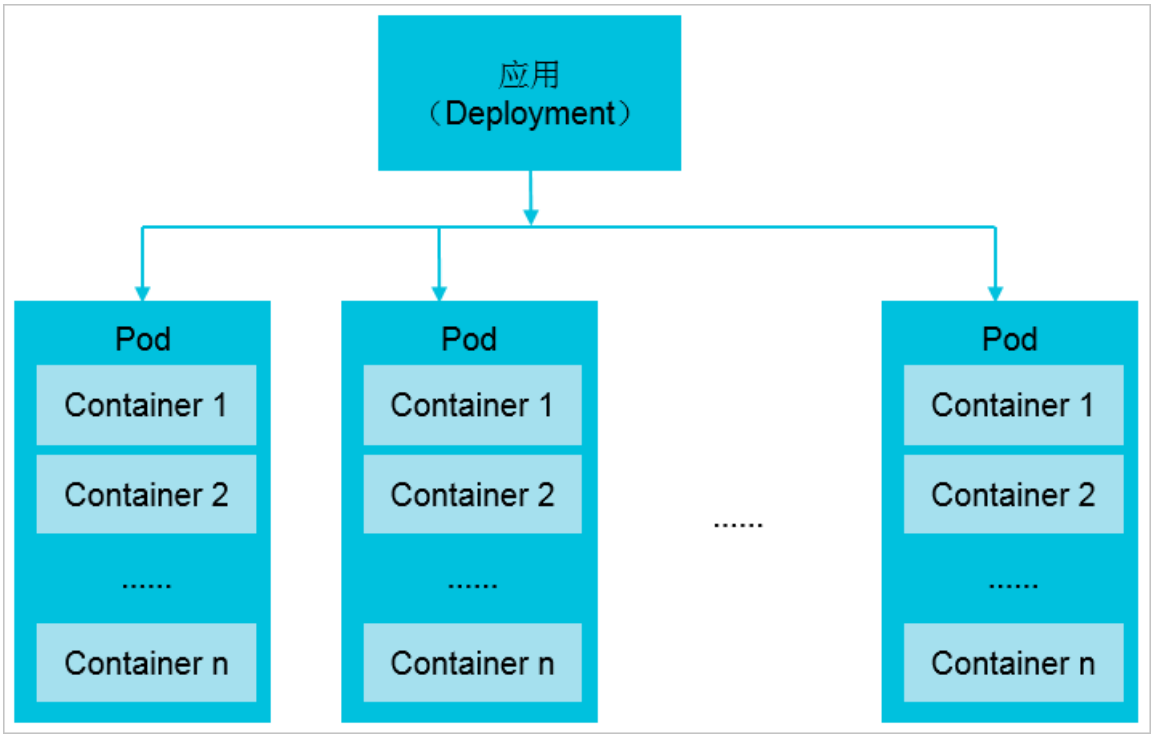
容器服务Swarm集群

容器服务Swarm中，应用类似于项目，一个应用下面可以有多个服务。服务是具体提供应用功能的实例。服务可以水平扩展。



容器服务Kubernetes集群

容器服务Kubernetes中，应用是指部署（deployment），能够提供应用对外暴露的功能。部署中会有Pod和container，但Kubernetes中最小的调度单位是一个pod，其中Pod可包含多个container。一个Pod可以认为是应用的一个实例，多实例（多Pod）可以调度到不同的节点上，也就是说Pod可以水平扩展。



说明 虽然上图中一个Pod中有多个container，但是在实际使用时，建议一个Pod对应一个container，这里对应多个container是为了说明Pod的能力。

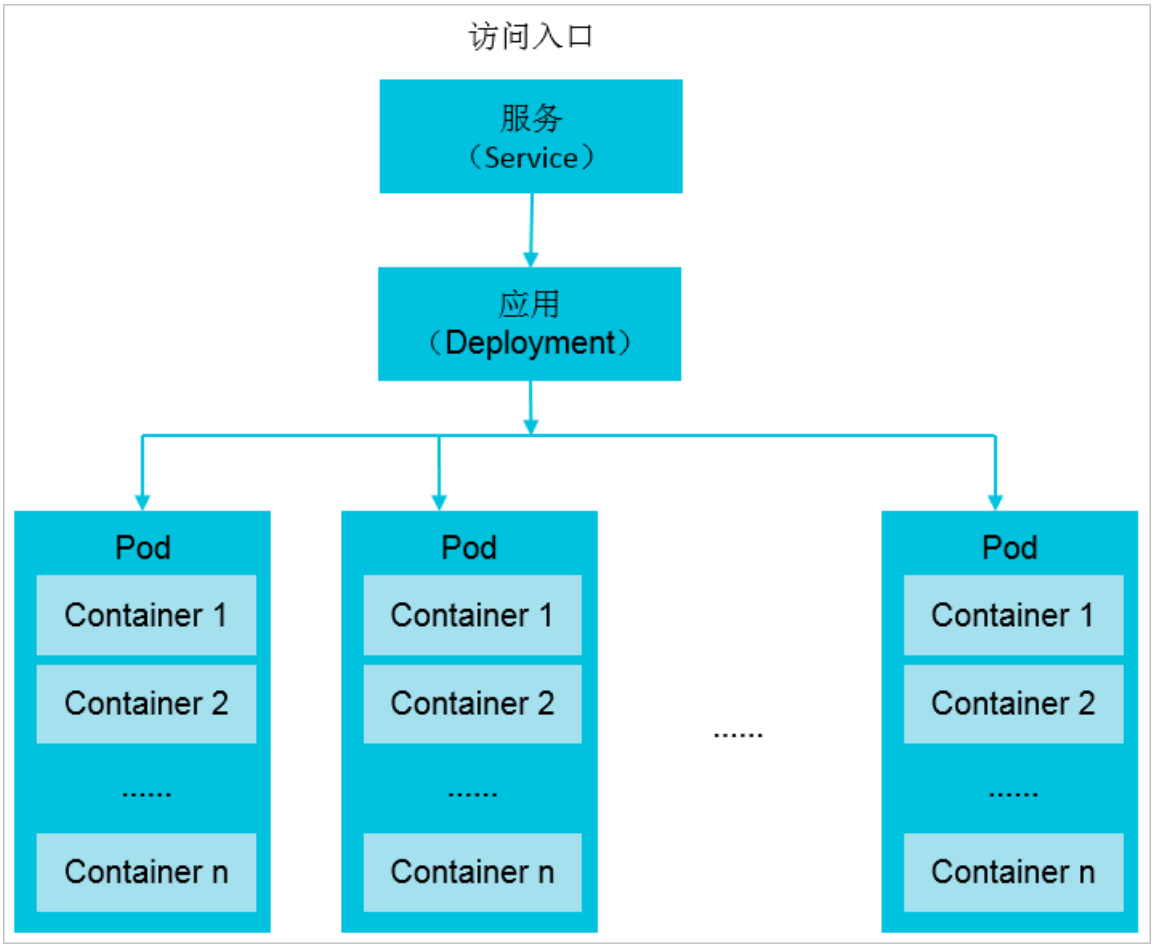
服务

容器服务Swarm集群

容器服务Swarm中的服务即提供应用功能的具体实例。当在Swarm集群中创建一个应用的时候，服务的访问方式会直接暴露给外部。

容器服务Kubernetes集群

容器服务Kubernetes中的服务是一个抽象概念，通过服务（Service）可以将应用（Deployment）的访问方式暴露给外部。



应用访问

容器服务Swarm集群

容器服务Swarm在部署应用的时候，可以选择三种应用访问方式，无论哪一种方式都可以直接暴露应用访问，不需要额外的操作：

- <HostIP>:<port>
- 简单路由
- 负载均衡

容器服务Kubernetes集群

容器服务Kubernetes在创建应用（即部署deployment）后，不能直接暴露应用的访问方式，需要通过创建服务（Service）进行应用访问的暴露。在容器服务Kubernetes集群内部应用之间可以通过服务名（Service Name）进行访问，服务名也只能用于集群内部访问。若要在集群外部访问应用，需要创建NodePort类型和LoadBalancer类型的服务进行对外暴露：

- ClusterIP（集群内部访问使用，也可以使用服务名）
- NodePort（类似于Swarm集群的<HostIP>:<port>）
- LoadBalancer（类似于Swarm集群的负载均衡）
- 域名，通过创建路由（ingress）来实现（类似于Swarm集群的简单路由）

13.1.3. 基本配置比对（使用镜像创建应用）

本文介绍容器服务Swarm集群与Kubernetes集群使用镜像创建应用时，基本配置的比对。

使用镜像创建应用

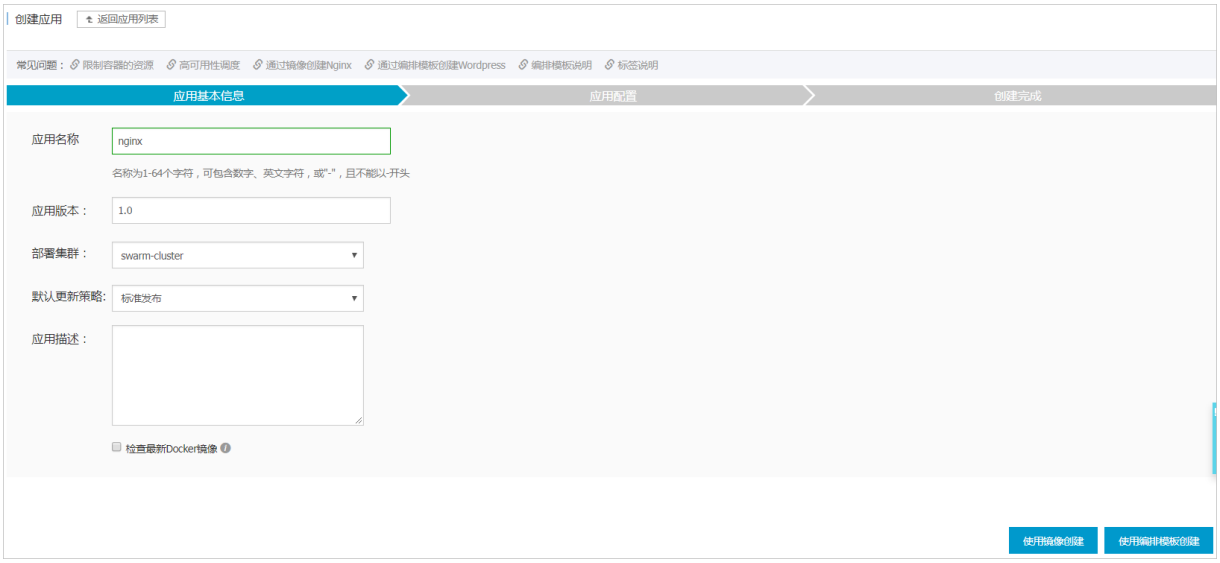
容器服务Swarm与Kubernetes集群在使用镜像创建应用时，部署界面差异较大。

- 容器服务Swarm集群，请参见创建应用。
- 容器服务Kubernetes集群，请参见创建无状态工作负载Deployment。

应用基本信息

容器服务Swarm集群

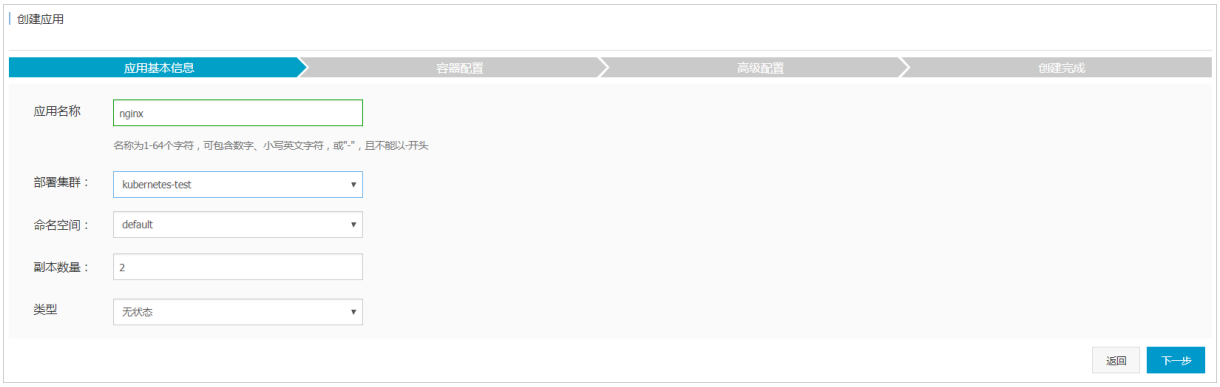
在应用基本信息中，部署的内容包括应用名称、应用版本、部署集群、默认更新策略及应用描述。



容器服务Kubernetes集群

与Swarm集群不同的是：需要配置命名空间、副本数量及类型。

命名空间是容器服务Kubernetes集群特有的概念，Kubernetes通过命名空间（namespace）进行资源的隔离，如CPU等。同时也可以区分不同的使用环境，例如测试环境、开发环境。如果涉及生产环境，建议通过集群隔离。Kubernetes的命名空间，可参见基本概念。

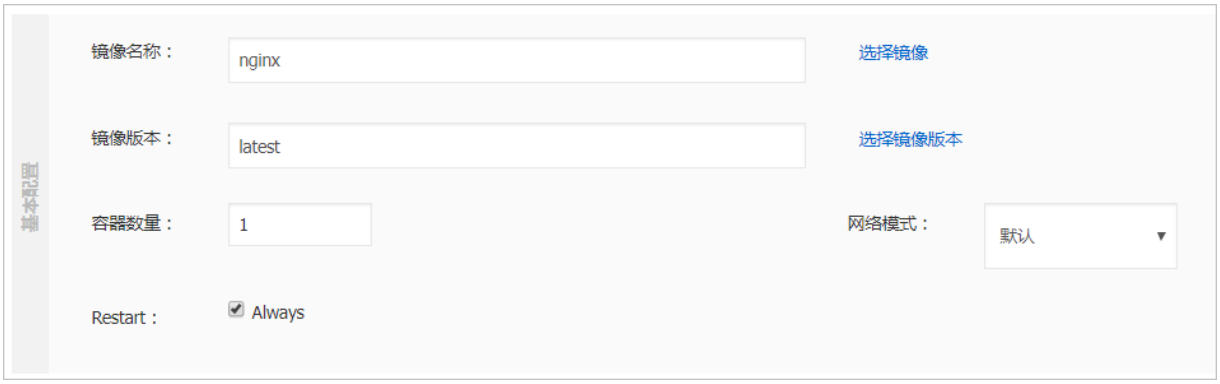


基本配置

基本配置，主要是选择镜像和镜像版本。

容器服务Swarm集群

网络模式目前支持**默认**和**host**两种。



容器服务Kubernetes集群

- 网络模式，在创建集群时已经选定，可选择两种网络插件**Flannel**和**Terway**，请参见使用Terway网络插件。
- 所需资源是声明需要的CPU和内存资源，资源限制是实际使用中不能超过的资源上限。类似于容器服务Swarm集群的**容器配置**部分的**CPU限制**和**内存限制**。



13.1.4. 网络配置比对（使用镜像创建应用）

本文介绍容器服务Swarm集群与Kubernetes集群使用镜像创建应用时，网络配置的比对。

使用镜像创建应用

容器服务Swarm与Kubernetes集群在使用镜像创建应用时，部署界面差异较大。

- 容器服务Swarm集群，请参见创建应用。
- 容器服务Kubernetes集群，请参见创建无状态工作负载Deployment。

网络配置

容器服务Swarm集群的**网络配置**主要完成应用对外访问方式的暴露。

端口映射

容器服务Swarm集群

容器服务Swarm集群的**端口映射**是将应用端口映射到宿主机，在每个宿主机上都会启用相同的端口，这样访问应用的时候只需要 `<HostIP>:<Port>` 即可访问。

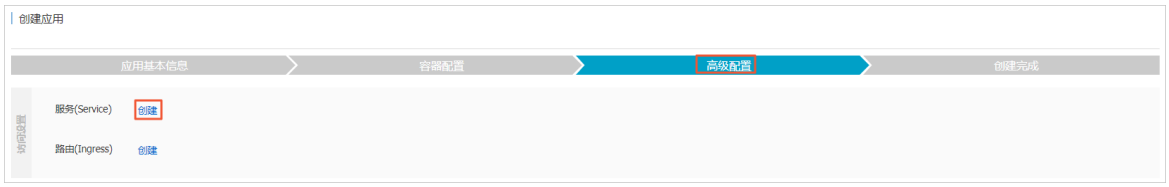


容器服务Kubernetes集群

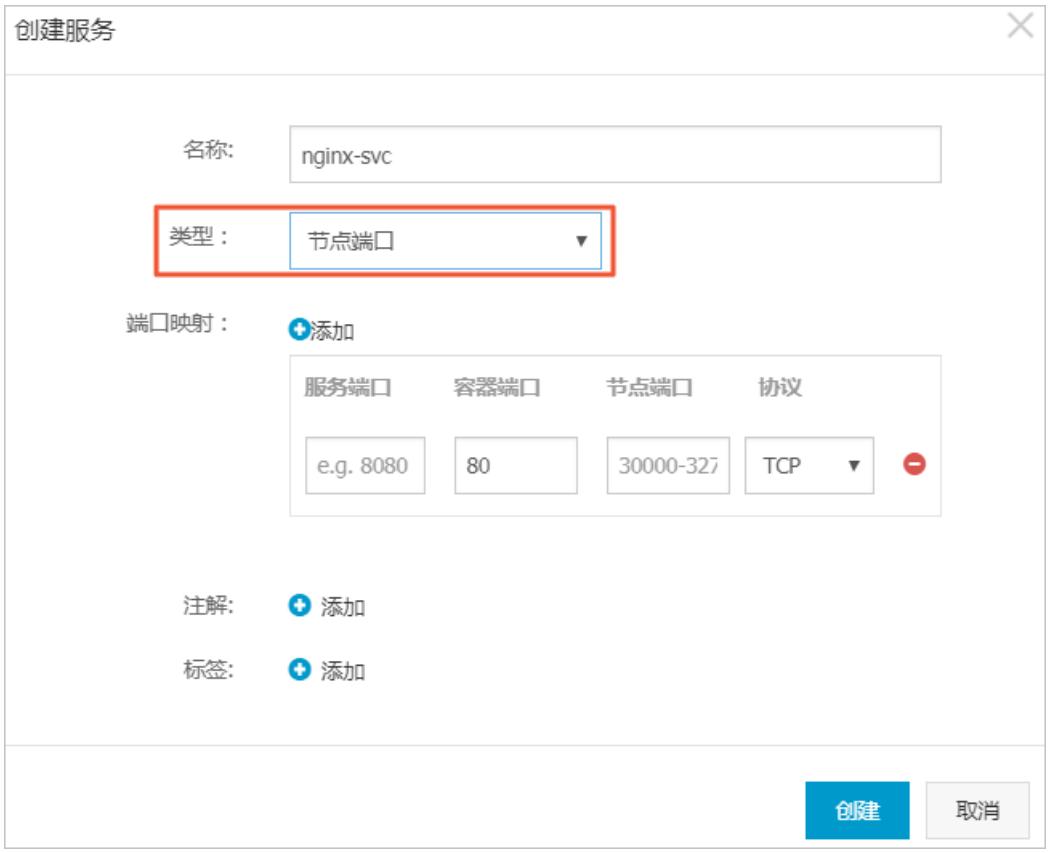
在容器服务Kubernetes集群可以通过**NodePort**类型的Service来实现，有以下两种方法：

方法一：创建应用时配置

1. 部署完**容器配置**后，进行**高级配置**时，在**访问设置区域**，单击**服务（Service）**右侧的**创建**。

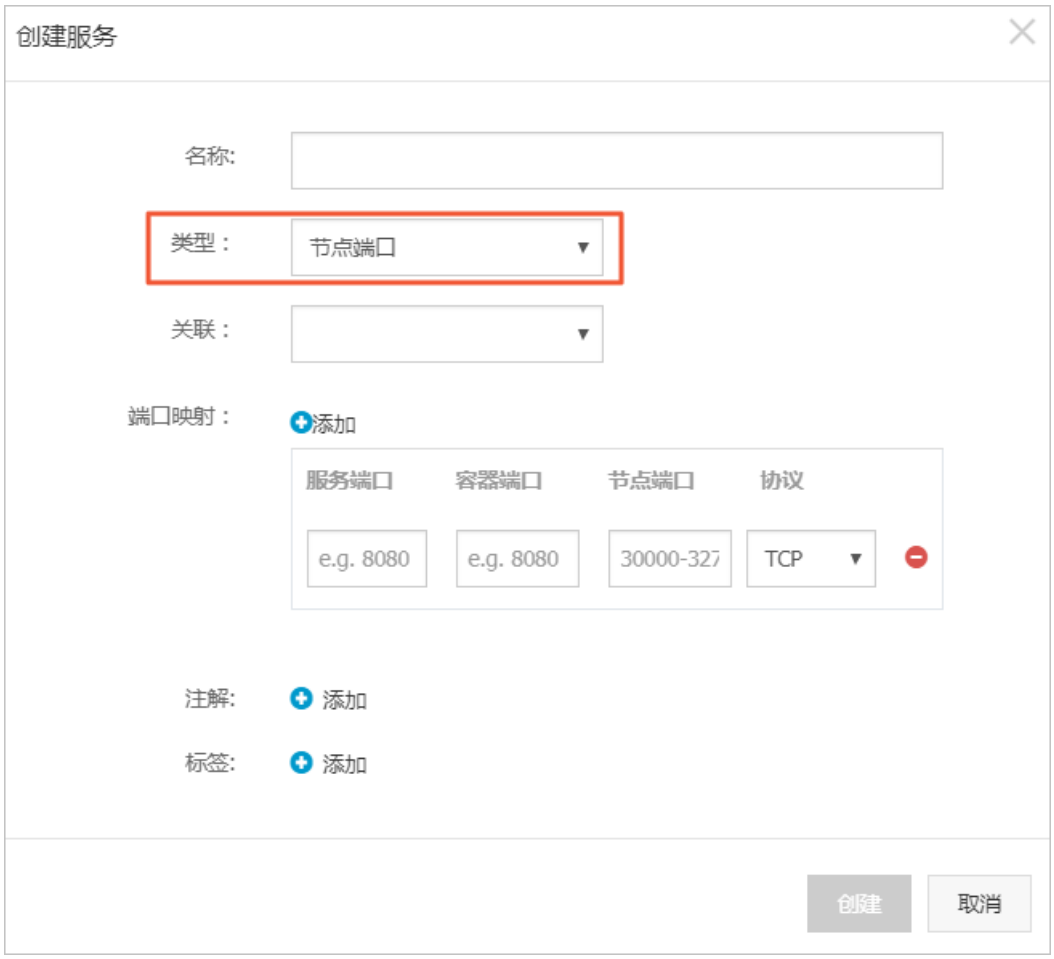


2. **类型**请选择**节点端口**。具体操作，请参见创建无状态工作负载Deployment。



方法二：通过创建服务配置

1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，单击**集群**。
3. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
4. 在集群管理页左侧导航栏中，选择**网络 > 服务**。
5. 选择命名空间，单击**创建**，在**创建服务**页面，**类型**选**节点端口**。具体操作，请参见管理服务。



简单路由配置

容器服务Swarm集群

容器服务Swarm集群的**简单路由配置**为用户提供了通过域名访问应用的方式，用户可以选择使用容器服务提供的域名也可以自定义域名。



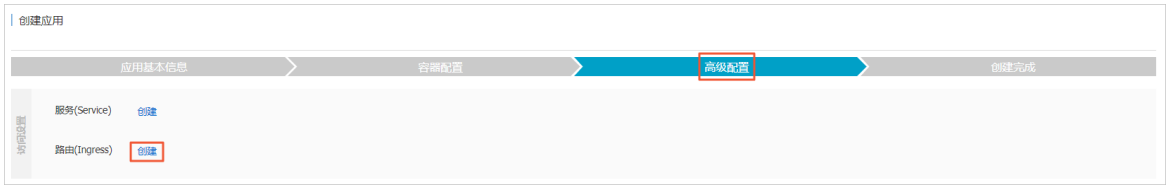
容器服务Kubernetes集群

在容器服务Kubernetes集群可以通过路由（Ingress）功能来实现，用户可以通过创建路由的方式进行相关功能的创建。同时容器服务Kubernetes的Ingress还提供了蓝绿发布、灰度发布等功能。更多信息，请参见通过Nginx Ingress实现灰度发布和蓝绿发布。

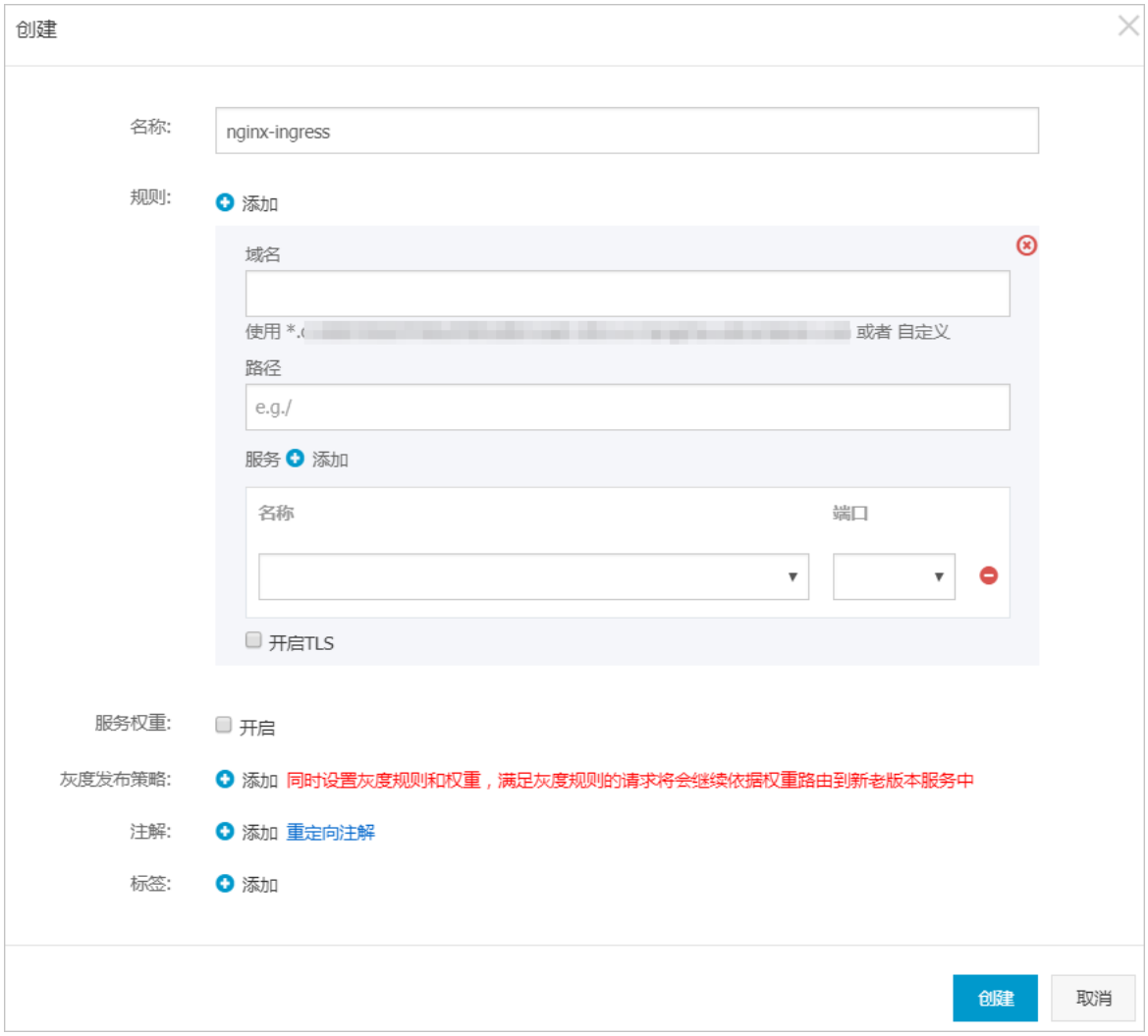
容器服务Kubernetes集群实现路由（Ingress）功能，有两种方法：

方法一：创建应用时配置

1. 部署完**容器配置**后，进行**高级配置**时，在**访问设置区域**，单击**路由（Ingress）**右侧的**创建**。

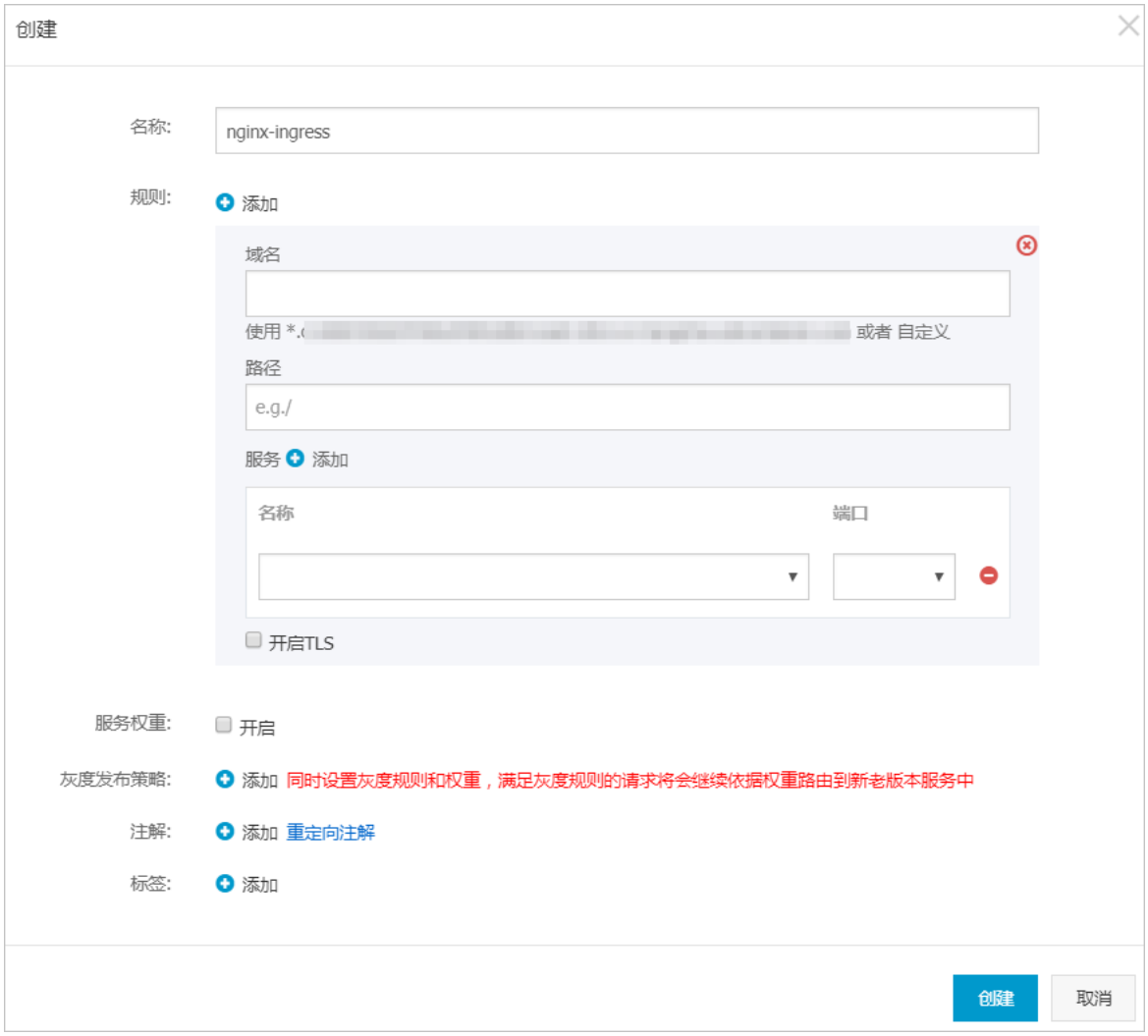


2. 使用镜像创建无状态应用。具体操作，请参见创建无状态工作负载Deployment。



方法二：通过创建路由配置

1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，单击**集群**。
3. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
4. 在集群管理页左侧导航栏中，选择**网络 > 路由**。
5. 选择命名空间，单击**创建**。具体操作，请参见控制台操作指导。



负载均衡路由配置

容器服务Swarm集群

容器服务Swarm集群的**负载均衡路由配置**为应用提供了通过阿里云负载均衡（Server Load Balancer）暴露应用访问方式的能力，用户首先自己创建SLB，然后将创建的SLB的IP及端口信息绑定到应用上，用户可以通过<SLB_IP>:<Port>的方式访问应用。



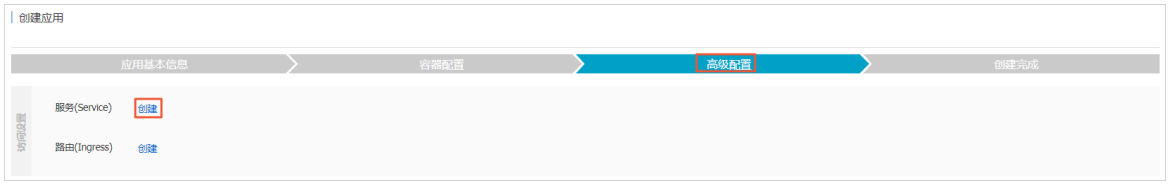
容器服务Kubernetes集群

容器服务Kubernetes集群同样支持通过绑定SLB的方式进行应用访问方式的暴露。容器服务Kubernetes集群上创建SLB是通过LoadBalancer类型的Service进行自动创建，不需要手工创建后配置。自动创建SLB时，可以选择公网访问或私网访问。同时如果使用YAML文件进行创建的话，还可以指定已有SLB及支持会话保持等配置。具体操作，请参见管理服务。

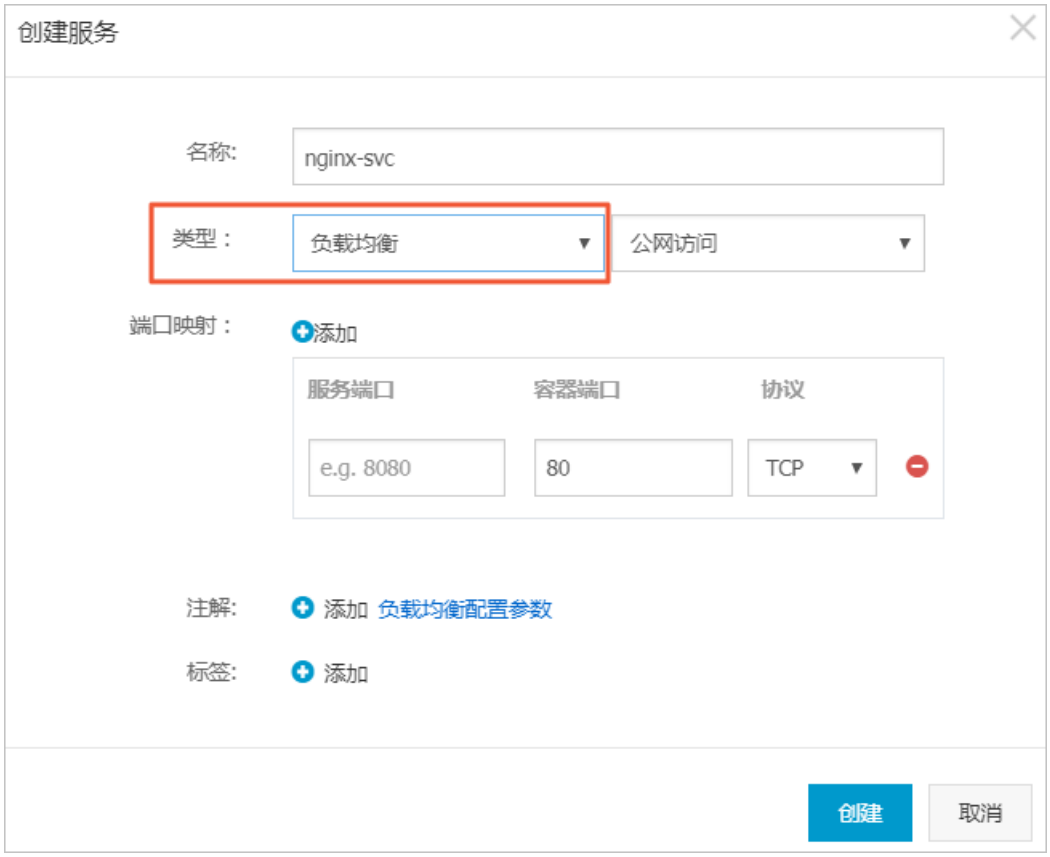
容器服务Kubernetes集群上创建Loadbalancer类型的Service有两种方式：

方法一：创建应用时配置

1. 部署完**容器配置**后，进行**高级配置**时，在**访问设置区域**，单击**服务（Service）**右侧的**创建**。

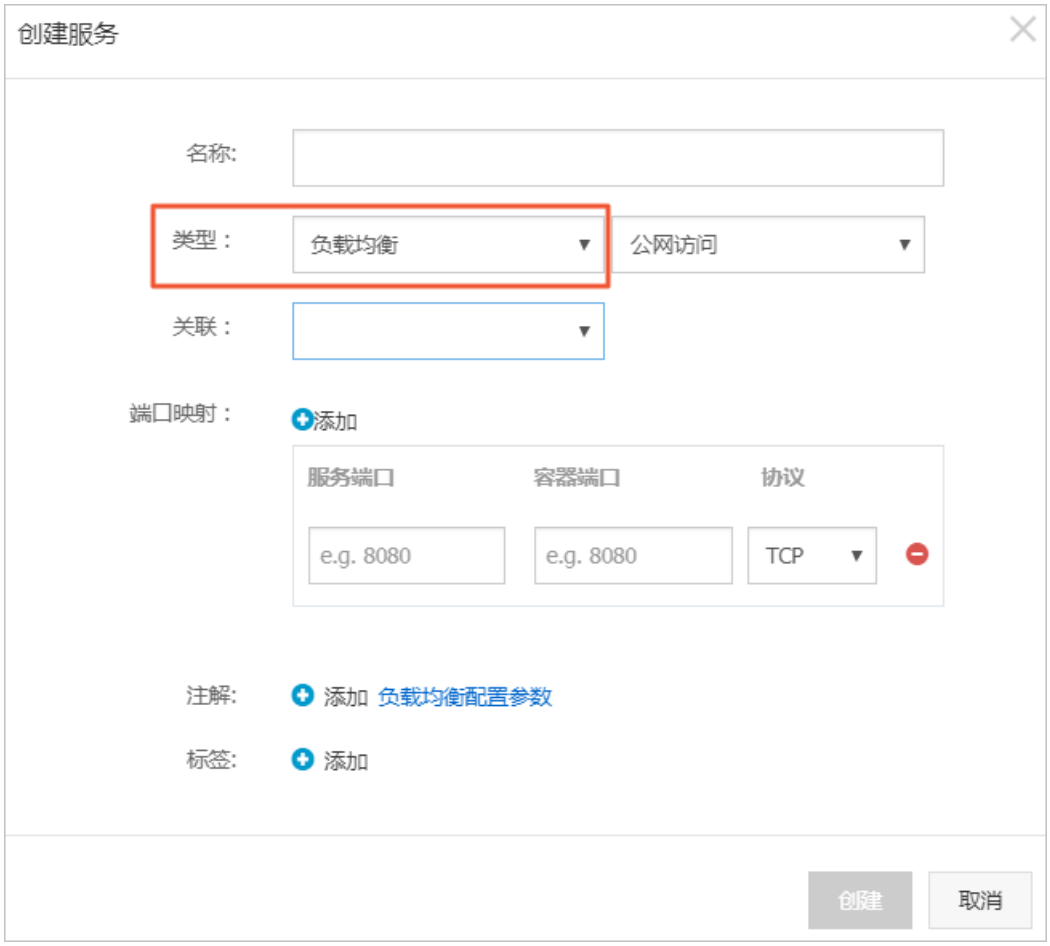


2. **类型**请选择**负载均衡**。具体操作，请参见创建无状态工作负载Deployment。



方法二：通过创建服务配置

1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，单击**集群**。
3. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
4. 在集群管理页左侧导航栏中，选择**网络 > 服务**。
5. 选择命名空间，单击**创建**，在**创建服务**页面，**类型**选**负载均衡**。更多信息，请参见管理服务。



13.1.5. 数据卷及环境变量配置比对（使用镜像创建应用）

本文介绍容器服务Swarm集群与Kubernetes集群使用镜像创建应用时，数据卷及环境变量的配置比对。

使用镜像创建应用

容器服务Swarm与Kubernetes集群在使用镜像创建应用时，部署界面差异较大。

- 容器服务Swarm集群，请参见创建应用。
- 容器服务Kubernetes集群，请参见创建无状态工作负载Deployment。

数据卷

容器服务Swarm集群

填写用户申请的云存储或者本地存储路径。



容器服务Kubernetes集群

容器服务Kubernetes集群使用存储的方式与容器服务Swarm集群一样，只是挂载的过程不一样，控制台界面基本一致。



用户可以选择本地存储，也可以选择云存储：

- 本地存储：包括主机目录、Kubernetes集群特有的配置项（configmap）、保密字典（secret）及临时目录。
- 云存储：包括云盘、NAS及OSS。

环境变量

容器服务Swarm集群与容器服务Kubernetes集群**环境变量**的配置一样。用户只需输入键值即可。



13.1.6. 容器配置及标签比对（使用镜像创建应用）

本文介绍容器服务Swarm集群与Kubernetes集群使用镜像创建应用时，容器配置及标签的比对。

使用镜像创建应用

容器服务Swarm与Kubernetes集群在使用镜像创建应用时，部署界面差异较大。

- 容器服务Swarm集群，请参见创建应用。
- 容器服务Kubernetes集群，请参见创建无状态工作负载Deployment。

容器配置

容器服务Swarm集群

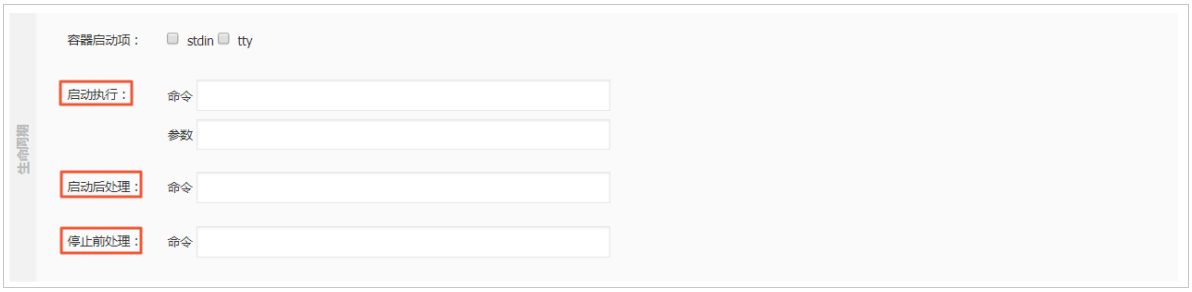
设置容器的启动命令（**Command**和**Entrypoint**）、资源限制（**CPU限制**和**内存限制**）及启动项等配置。



容器服务Kubernetes集群

容器服务swarm集群的**容器配置**，类似于容器服务Kubernetes集群的生命周期配置和基本配置。

- **生命周期**，详情请参见创建无状态工作负载Deployment。
  - **启动执行**
  - **启动后处理**
  - **停止前处理**



- **基本配置**，详情请参见创建无状态工作负载Deployment，推荐配置请参见高可靠推荐配置。
  - **资源限制**
  - **所需资源**

标签

容器服务Swarm集群：能够完成健康检查、设置访问域名、日志等功能。

容器服务Kubernetes集群：容器服务Kubernetes集群中的标签只能标识一个应用。容器服务Kubernetes部署的应用在创建时，会自动生成与应用名相同的标签，在使用镜像创建应用的配置界面没有展现，用户可以通过YAML文件的方式使用此标签。

13.1.7. 健康检查及自动伸缩比对（使用镜像创建应用）

本文介绍容器服务Swarm集群与Kubernetes集群使用镜像创建应用时，健康检查及自动伸缩的比对。

使用镜像创建应用

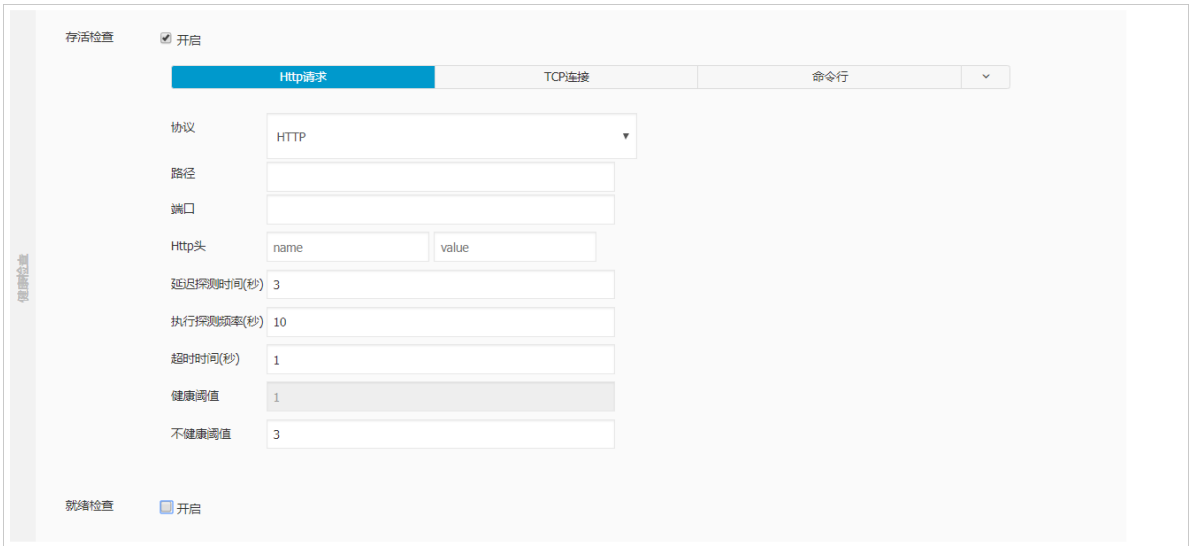
容器服务Swarm与Kubernetes集群在使用镜像创建应用时，部署界面差异较大。

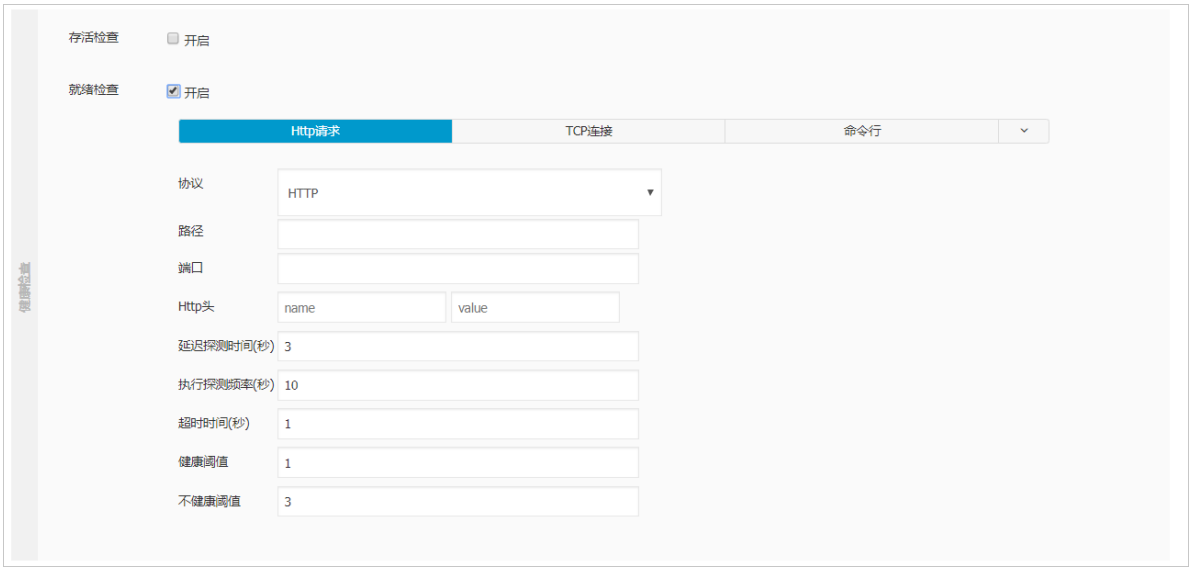
- 容器服务Swarm集群，请参见创建应用。
- 容器服务Kubernetes集群，请参见创建无状态工作负载Deployment。

健康检查

- 容器服务Swarm集群
  健康检查是通过标签的方式实现。

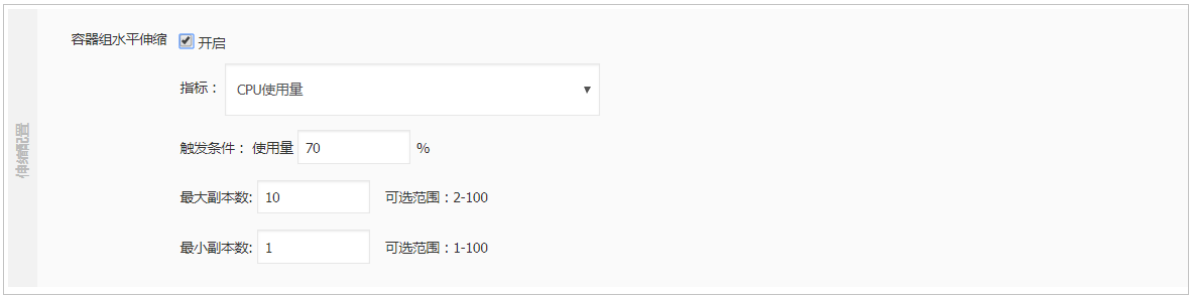
- 容器服务Kubernetes集群
  在使用镜像创建应用的**容器配置**页签，可进行健康检查的配置。目前支持**存活检查**和**就绪检查**。





自动伸缩

- 容器服务Swarm集群
  提供基于CPU和内存两个维度的自动伸缩。
- 容器服务Kubernetes集群
  在使用镜像创建应用的**高级配置**页签，可配置容器组的水平伸缩，同样提供CPU和内存两个维度的自动伸缩。



13.1.8. 使用YAML文件创建应用比对

本文介绍容器服务Swarm集群与Kubernetes集群使用YAML文件创建应用时，Swarm集群下的YAML文件与Kubernetes集群下的YAML文件的对应关系。

背景信息

在使用YAML文件创建应用时，Swarm集群与Kubernetes集群的YAML文件格式不一样：

- 您可以通过Kompose工具对Swarm集群的YAML文件进行自动转换。但转换后的内容仍需您核对检查。
  Kompose工具可以从Github上获取。
  Kompose工具下载地址：
  - Mac下载地址：Mac
  - Linux下载地址：Linux
  - Window下载地址：Window

  > **说明** 目前Kompose工具对阿里云的一些定制标签还不支持，如Kompose工具不支持的标签所示，阿里云容器服务团队会持续研发逐步覆盖各个标签的支持。

  Kompose工具不支持的标签

标签	参考
external	external
dns_options	dns_options
oom_kill_disable	oom_kill_disable
affinity:service	服务部署约束（affinity:service）

- 您也可以手动改写YAML文件。

本文将基于容器服务Swarm的YAML文件，介绍Kubernetes的YAML文件如何与之对应。文章中的YAML示例，仅作为示例，具体部署请依据具体情况添加及修改相关内容。

容器服务Swarm与Kubernetes集群的YAML文件比对

容器服务Swarm集群

容器服务Swarm集群的YAML文件*wordpress-swarm.yaml*如下，注释中的阿拉伯数字与容器服务Kubernetes集群的YAML文件的注释对应。

```
web:        #---1
  image: registry.aliyuncs.com/acs-sample/wordpress:4.5        #---2
  ports:        #---3
    - '80'
  environment:        #---4
    WORDPRESS_AUTH_KEY: changeme            #---5
    WORDPRESS_SECURE_AUTH_KEY: changeme        #---5
    WORDPRESS_LOGGED_IN_KEY: changeme        #---5
    WORDPRESS_NONCE_KEY: changeme            #---5
    WORDPRESS_AUTH_SALT: changeme            #---5
    WORDPRESS_SECURE_AUTH_SALT: changeme        #---5
    WORDPRESS_LOGGED_IN_SALT: changeme        #---5
    WORDPRESS_NONCE_SALT: changeme            #---5
    WORDPRESS_NONCE_AA: changeme            #---5
  restart: always        #---6
  links:            #---7
    - 'db:mysql'
  labels:            #---8
    aliyun.logs: /var/log        #---9
    aliyun.probe.url: http://container/license.txt        #---10
    aliyun.probe.initial_delay_seconds: '10'            #---10
    aliyun.routing.port_80: http://wordpress            #---11
    aliyun.scale: '3'                            #---12
db:             #---1
  image: registry.aliyuncs.com/acs-sample/mysql:5.7        #---2
  environment:        #---4
    MYSQL_ROOT_PASSWORD: password        #---5
  restart: always        #---6
  labels:        #---8
    aliyun.logs: /var/log/mysql        #---9
```

容器服务Kubernetes集群

通过容器服务Swarm集群的*wordpress-swarm.yaml*文件部署的WordPress应用，在容器服务Kubernetes集群中对应2个服务：web和db。

在容器服务Kubernetes集群上需要2个部署（Deployment）和2个服务（Service）。2个Deployment创建2个Service，2个服务分别暴露2个应用的访问方式。

容器服务Swarm集群中的Web应用对应Kubernetes集群的Deployment和Service如下：

> **说明** 以下YAML文件的内容仅作为示例说明与容器服务Swarm集群*wordpress-swarm.yaml*的对应关系，不可用作实际部署。

- *wordpress-kubernetes-web-deployment.yaml*内容如下：

```
apiVersion: apps/v1     # api版本
kind: Deployment        # 创建资源的类型
metadata:
  name: wordpress      #---1
  labels:            #---8 在这里的label只能做标识作用
    app: wordpress
spec:    #资源创建详细内容
  replicas: 2        #---12 设定实例（副本）个数
```

```
  selector:
    matchLabels:
      app: wordpress
      tier: frontend
strategy:
type: Recreate
  template:   #模板定义POD的详细信息
    metadata:
      labels:  #与前面保持一致
        app: wordpress
        tier: frontend
    spec:    #定义pod中container的详细信息
      containers:    #
      - image: wordpress:4   #---2  对应于镜像及版本
        name: wordpress
        env:    #---4 环境变量设置，kubernetes上configmap，secret都可以通过env的方式使用。
        - name: WORDPRESS_DB_HOST
          value: wordpress-mysql  #---7 通过名称指向需要访问的MySQL，该名称与MySQL Service的名
称相对应。
        - name: WORDPRESS_DB_PASSWORD    #---5 密码在这里使用，但Kubernetes提供了Secret进行密
码封装。
          valueFrom:
            secretKeyRef:
              name: mysql-pass
              key: password-wordpress
        ports:   #---3 容器内应用暴露的port
        - containerPort: 80
          name: wordpress
livenessProbe:        #add health check    ---10 健康检查
          httpGet:
            path: /
            port: 8080
          initialDelaySeconds: 30
          timeoutSeconds: 5
          periodSeconds: 5
        readinessProbe:       #add health check       ---10 健康检查
          httpGet:
            path: /
            port: 8080
          initialDelaySeconds: 5
          timeoutSeconds: 1
          periodSeconds: 5
        volumeMounts:  #使用存储卷，将存储卷真正挂到容器内部。
        - name: wordpress-pvc
          mountPath: /var/www/html
      volumes:   #获取存储卷，需先进行PV和PVC的创建。
      - name: wordpress-pvc
        persistentVolumeClaim:
          claimName: wordpress-pv-claim
```

- *wordpress-kubernetes-web-service.yam*l内容如下：

```
apiVersion: v1    #版本号
kind: Service     #创建资源类型，在这里为Service。
metadata:
  name: wordpress
  labels:
    app: wordpress
spec:
  ports:
    - port: 80    #服务端口号
  selector:  #通过label进行应用的关联
    app: wordpress
    tier: frontend
  type: LoadBalancer  #---11 定义访问方式，此处为LoadBalancer类型的Service，会自动创建SLB。
```

容器服务Swarm集群中的db应用对应Kubernetes集群的Deployment和Service如下：

> **说明** 以下YAML文件的内容仅作为示例说明与容器服务Swarm集群*wordpress-swarm.yaml*的对应关系，不可用作实际部署。

- *wordpress-kubernetes-db-deployment.yaml*内容如下：

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wordpress-mysql
  labels:
    app: wordpress
spec:
  selector:
    matchLabels:
      app: wordpress
      tier: mysql
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: wordpress
        tier: mysql
    spec:
      containers:
      - image: mysql:5.6
        name: mysql
        env:
        - name: MYSQL_ROOT_PASSWORD
          valueFrom:
            secretKeyRef:
              name: mysql-pass
              key: password-mysql
        ports:
        - containerPort: 3306
          name: mysql
        volumeMounts:
        - name: wordpress-mysql-pvc
          mountPath: /var/lib/mysql
      volumes:
      - name: wordpress-mysql-pvc
        persistentVolumeClaim:
          claimName: wordpress-mysql-pv-claim
```

- *wordpress-kubernetes-db-service.yaml*内容如下：

```
apiVersion: v1
kind: Service
metadata:
  name: wordpress-mysql
  labels:
    app: wordpress
spec:
  ports:
    - port: 3306
  selector:
    app: wordpress
    tier: mysql
      clusterIP: None
```

13.1.9. 网络比对

本文介绍容器服务Swarm集群与Kubernetes集群的网络比对。

容器服务Swarm集群

容器服务Swarm集群使用的网络有两种：

- VPC网络
- 经典网络

容器服务Kubernetes集群

容器服务Kubernetes集群的网络为VPC网络，可参见Kubernetes集群网络规划。

- 若容器服务Swarm集群使用的是VPC网络，在创建Kubernetes集群时，请选择与Swarm集群相同的VPC网络，以便Kubernetes集群与Swarm集群的网络是联通的。
- 若容器服务Swarm集群使用的是经典网络，在创建Kubernetes集群时，由于Kubernetes集群只能为VPC网络，请先打通Swarm集群与Kubernetes集群的网络，可参见迁移方案概述。
  Swarm集群与Kubernetes集群网络联通后，如果Swarm集群下存在OSS、NAS或RDS等存储产品或数据库，会获得VPC网络可访问的IP地址，即在Kubernetes集群的VPC网络中可通过此IP地址对Swarm集群的存储产品或数据库进行访问。

13.1.10. 日志及监控比对

本文介绍容器服务Swarm集群与Kubernetes集群的日志及监控功能的比对。

日志

容器服务Swarm集群：通过**标签**实现日志等功能。

容器服务Kubernetes集群：Kubernetes集群的日志功能，可在以下情况配置及使用。

- 创建Kubernetes集群：

在**创建Kubernetes集群**页面，选中**使用日志服务**后，将会在集群中自动配置日志服务插件。您可以使用已有的Project，也可以创建新的Project。



具体操作，请参见组件配置。
集群创建成功后，也可以手动安装日志服务组件，请参见步骤一：启用日志服务组件Logtail。

- 创建应用时配置日志服务，请参见步骤二：创建应用时配置日志服务。
- 创建应用后使用日志服务，请参见通过DaemonSet-控制台方式采集容器文本日志及通过DaemonSet-控制台方式采集容器标准输出。

监控

容器服务Swarm集群和Kubernetes集群，均在**创建集群**页面，选中**在ECS节点上安装云监控插件**，即可在云监控管理控制台查看所创建ECS实例的监控信息。具体操作，请参见组件配置。

容器服务Kubernetes集群与云监控的集成使用，请参见基础资源监控。

13.1.11. 应用访问比对

本文介绍容器服务Swarm集群与Kubernetes集群的应用访问比对，包括集群内部应用间访问及从集群外部访问应用。

集群内部应用间访问

容器服务Swarm集群

集群内部可以通过 `links` 标签，将需要被访问的服务名称设置到容器的环境变量中。

例如：使用YAML文件创建应用中，WordPress应用的Web服务与*MySQL*关联，在容器启动后，通过*MySQL*这个服务名称即可完成服务的访问。

```
links:            #---7
    - 'db:mysql'
```

容器服务Kubernetes集群

在容器服务Kubernetes集群内部，可通过Service的ClusterIP或服务名称进行应用间访问。推荐使用服务名称进行应用间的访问。

在创建应用时，可以将需要被访问服务的服务名称以环境变量的方式使用。

例如：使用yaml文件创建应用比对中，WordPress在调用*MySQL*服务时，就是通过环境变量的方式实现。

```
spec:
      containers:
      - image: wordpress:4
        name: wordpress
        env:
        - name: WORDPRESS_DB_HOST
          value: wordpress-mysql    #---7 通过名称指向需要访问的MySQL，该名称与MySQL service的名称相对应。
        - name: WORDPRESS_DB_PASSWORD
```

从集群外部访问应用

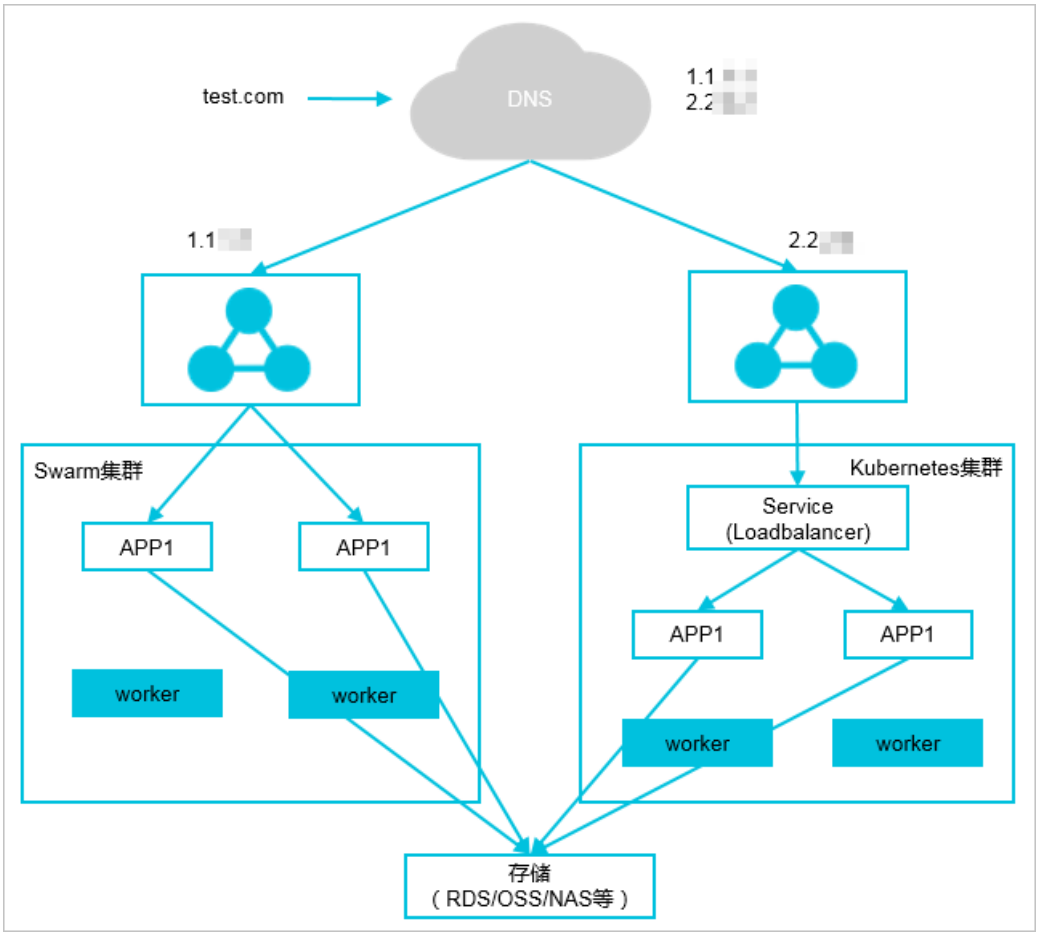
从集群外部访问有三种方式：通过域名访问、通过<HostIP>:<port>访问、通过负载均衡访问。

通过域名访问应用

> 说明
> - 无论经典网络还是VPC，请确保网络的连通。
> - DNS具有负载均衡的能力，可将流量分发到不同的后端IP。
> - 通过域名作为应用的访问入口，可以不停机地将业务从Swarm集群迁移到Kubernetes集群。

简单路由（域名绑定到容器服务Swarm集群默认的SLB上）

为实现应用从容器服务Swarm集群迁移到Kubernetes集群，需要先在容器服务Kubernetes集群上创建应用，测试可用后，再进行应用迁移。



迁移方法

- 在容器服务Kubernetes集群上通过以下步骤创建应用：
  i. 在容器服务Kubernetes集群上创建与Swarm集群相同类型的应用。
  ii. 在容器服务Kubernetes集群上为应用创建Loadbalancer类型的服务（Service）。
  iii. Loadbalancer类型的服务会创建SLB，在本例中为2.2.2.2。
  iv. 在DNS中*test.com*域名的后端IP增加2.2.2.2这个IP地址。
- 验证Kubernetes集群的应用可用

通过IP地址2.2.2.2可以正常访问应用，说明容器服务Kubernetes集群上的应用可用。

- 应用迁移
  将DNS中*test.com*域名的后端IP地址1.1.1.1删除。

后续流量经过DNS后全部流向Kubernetes集群上的应用。

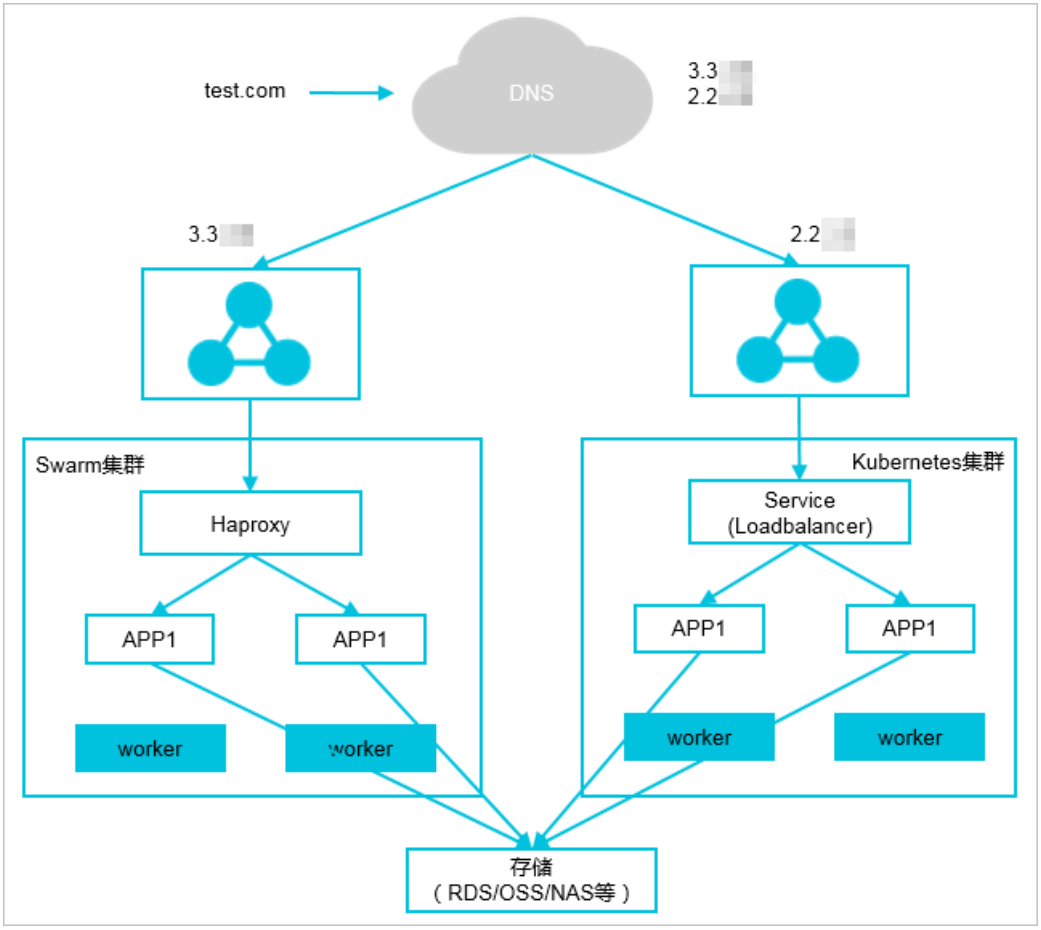
简单路由（域名绑定到应用在容器服务Swarm集群上自建的SLB）

域名绑定到容器服务Swarm集群上自建的SLB与绑定到默认的SLB的区别是：

- SLB不是集群默认的，而是自建的。
- DNS默认为云解析DNS，如果用户使用自己的域名，需要自行解析。

迁移方法

与域名绑定到容器服务Swarm集群默认的SLB的方法相同：在容器服务Kubernetes集群上创建应用，测试可用后，再进行应用迁移。



通过<HostIP>:<port>访问应用

如果您通过<HostIP>:<port>的方式访问应用，那么在原容器服务Swarm集群的基础上无法做到不停机迁移业务，请选择业务量小的时候进行业务的迁移。

迁移方法

1. 在容器服务Kubernetes集群上创建应用，并通过NodePort类型的服务完成应用对外访问方式的暴露，请参见网络配置比对。
2. 记录该<NodePort>，将Swarm集群的<port>替换为Kubernetes集群的<NodePort>。

> **说明** 此步骤需要逐个停止并修改应用实例。

3. 将Kubernetes集群内的worker节点，挂载到Swarm集群的SLB下。
4. 当有流量时，会有部分流量进入Kubernetes集群的应用，待Kubernetes集群的应用验证可用后，将SLB中Swarm集群的节点删除，完成集群的迁移。

通过负载均衡访问应用

如果您通过负载均衡的方式访问应用，那么在原容器服务Swarm集群的基础上无法做到不停机迁移业务，请选择业务量小的时候进行业务的迁移。

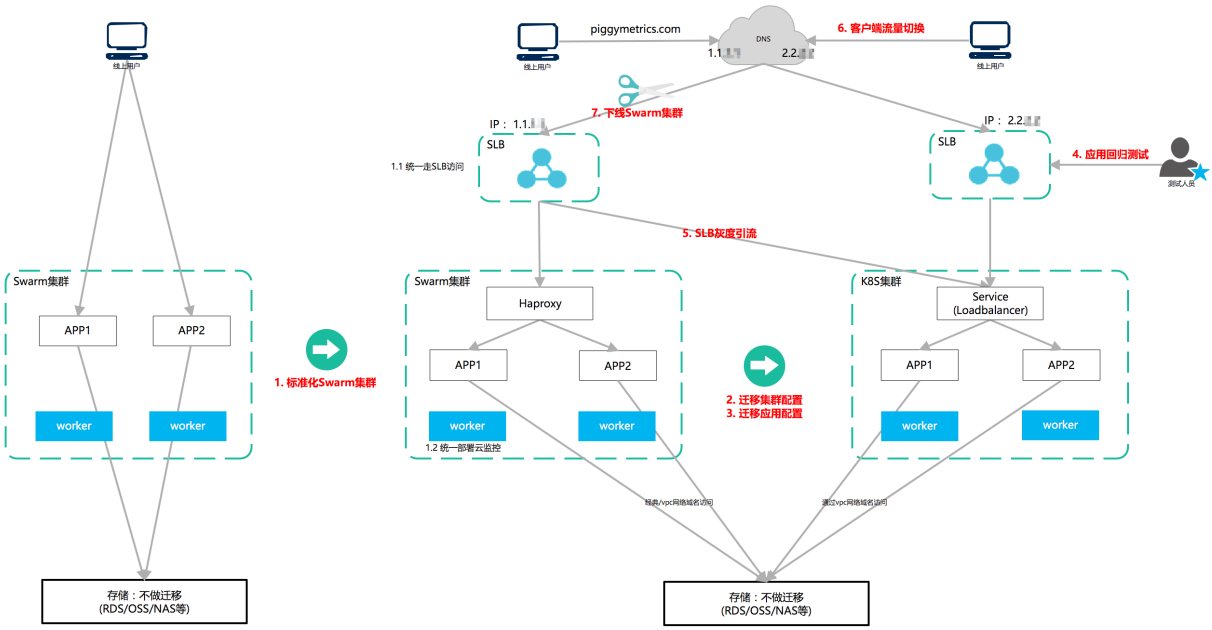
迁移方法

容器Kubernetes集群LoadBalancer的使用方法与容器服务Swarm的负载均衡一样，请参见网络配置比对。

13.2. 迁移方案概述

本文整体简单介绍下如何通过7个步骤，将容器服务 Swarm 集群平滑迁移到 Kubernetes 集群，并尽量确保迁移期间对业务无影响。

迁移方案



迁移步骤

1. 标准化Swarm集群。

   由运维人员对已有 Swarm 集群做少量运维操作，降低后续 Kubernetes 集群迁移成本和风险。

   i. 客户端统一通过SLB访问集群应用，保证后续**SLB灰度引流**实时生效。

   - 如果通过SLB 访问应用，能够快速引流验证或出问题时能实时回滚（SLB控制台变更实时生效）。
   - 如果不通过SLB而通过其他方式访问应用，则通过客户端发布或 DNS 回滚（时间区间为：0~48h）无法实时生效，导致对业务产生影响。

   ii. 统一部署云监控，方便后续监控各ECS运行情况，保证灰度引流生效。

2. 迁移 Swarm 集群配置。

   由运维人员完成 Kubernetes 集群创建及集群维度资源配置，降低开发人员应用迁移复杂度。

   i. 集群机器资源及网络迁移。
   ii. Node 标签迁移。
   iii. VPC 网络互联互通验证。
   iv. 数据卷迁移。
   v. 配置项迁移。

3. 迁移应用配置。

   由研发人员基于kompose转换工具完成应用配置迁移。

   i. 准备迁移环境。
   ii. 预处理 Swarm 编排文件。
   iii. 转换 Swarm 编排文件。
   iv. 部署 Kubernetes 资源文件。
   v. 手动迁移应用配置。
   vi. 应用启动调试。
   vii. 迁移应用日志配置。

4. 应用回归测试。

   应用在不影响线上流量情况下，由测试人员完成新集群业务功能回归测试。

   i. 配置应用测试域名。
   ii. 测试业务功能。
   iii. 确认应用日志采集。
   iv. 确认应用监。

5. SLB灰度引流。

   由运维人员逐步线上引流灰度验证，有问题能够快速回滚。

   i. 配置NodePort引流。
   ii. 灰度引流快速回。

6. 客户端流量切换。

   由运维人员做客户端或DNS切换，将流量切换至新集群。

   i. DNS流量切换：调整DNS解析配置实现流量切换。
   ii. 客户端流量切换：升级客户端代码或配置实现流量切换。

7. 下线Swarm集群。

   由运维人员确认新集群服务访问正常，下线老Swarm集群资源。

   i. 确认Swarm集群流。
   ii. 下线Swarm集群资源。

13.3. 标准化Swarm集群

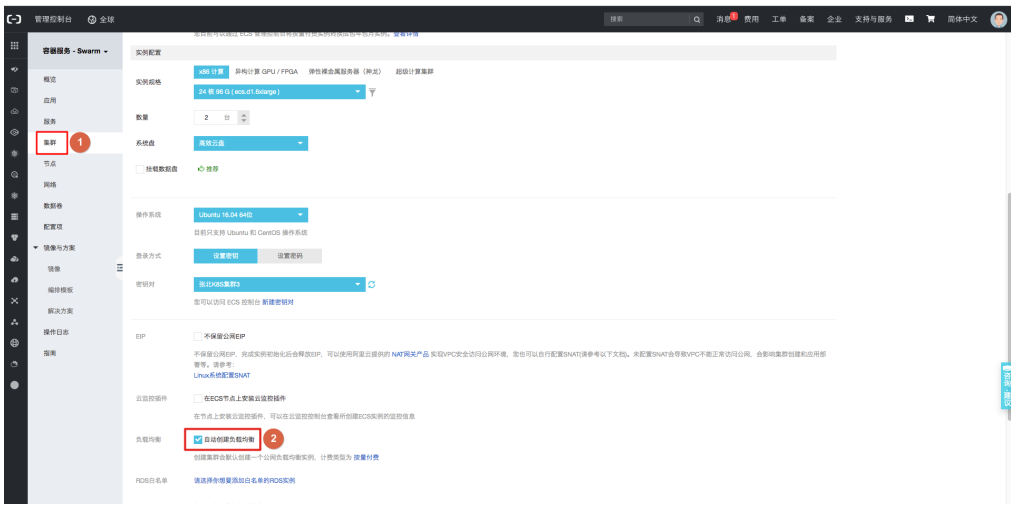
背景信息

从集群外部访问应用时，如果您通过*<HostIP>:<port>*访问应用或基于备案域名直接解析到NodeIP场景，则在迁移Swarm集群时，无法做到不停机迁移业务和实时回滚，所以先调整成统一通过SLB访问应用，再通过部署监控插件的方式，完成标准化Swarm集群，确保集群迁移时不停机迁移业务和实时回滚。

统一通过SLB访问Swarm集群

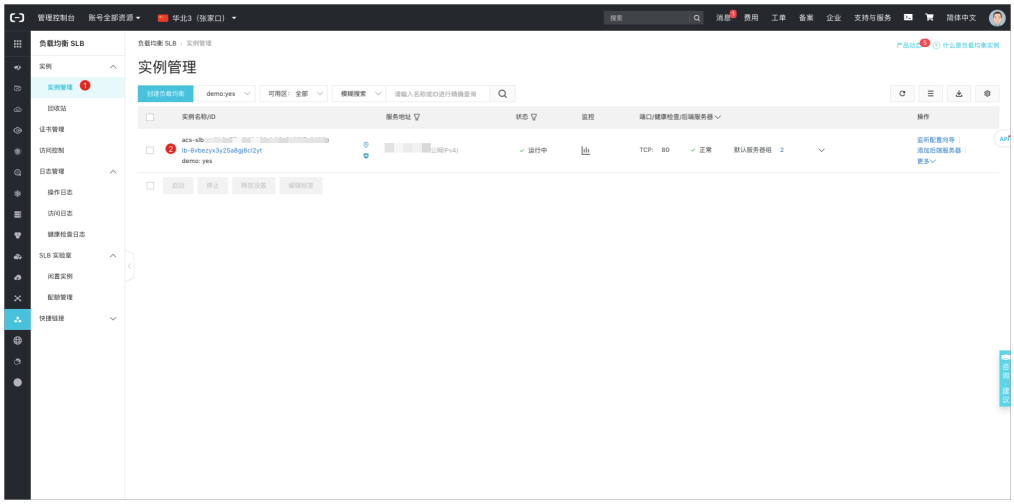
1. 创建SLB实例。
   - 自动创建SLB
     如果您在创建Swarm集群时，已勾选**自动创建负载均衡**，如设置SLB所示。表示您已自动创建一个按量付费的负载均衡实例。

     设置SLB

     您可以登录负载均衡管理控制台，查看已创建的SLB，确认该SLB已配置了监听后端服务器的 TCP 9080 端口，如查看服务端口所示。

     查看服务端口

   - 手动创建SLB
     如果您创建Swarm集群时，未勾选**自动创建负载均衡**，则只能在集群创建完成后，手动创建SLB并做绑定，具体步骤如下：
     a. 创建SLB。请参见创建和管理CLB实例。
     b. 为Swarm集群绑定SLB，保证后续该集群配置会自动更新SLB配置。请参见为集群绑定和解绑负载均衡。

2. 配置简单路由。

   原来通过*<HostIP>:<port>*访问应用，因将Host IP绑定在客户端，如后续出现Host替换或扩容，只能通过升级客户端方式来访问新Host服务。因此，建议您通过域名访问服务的方式。

   > **说明** 如果没有申请自定义域名，阿里云容器服务免费支持为每个应用或服务配置独立域名，格式为*XXX.$cluster_id.$region_id.alicontainer.com*，其中*XXX*域名前缀可以根据每个应用或服务自行定义。

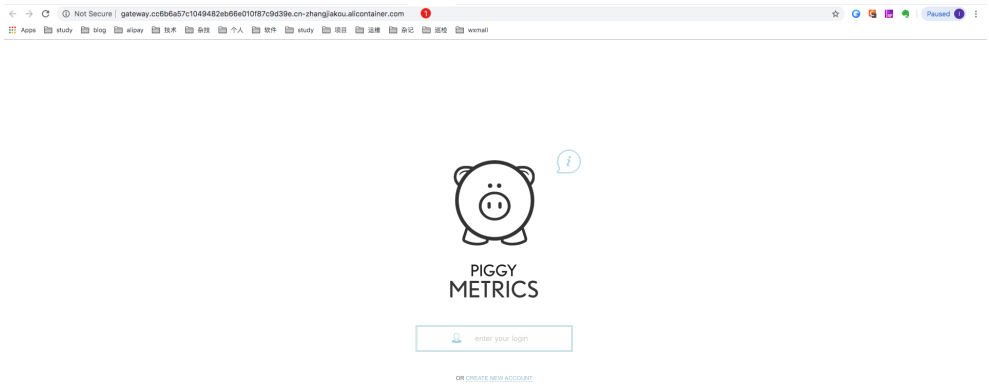
   通过这种方式，可以将服务访问切换成通过域名访问（用户自定义域名或容器服务提供的测试域名均可）的形式，而不再依赖于ECS或SLB的*<HostIP>:<port>*访问应用这种方式。详细操作请参见简单路由（支持 HTTP/HTTPS)。

3. SLB访问业务功能测试。

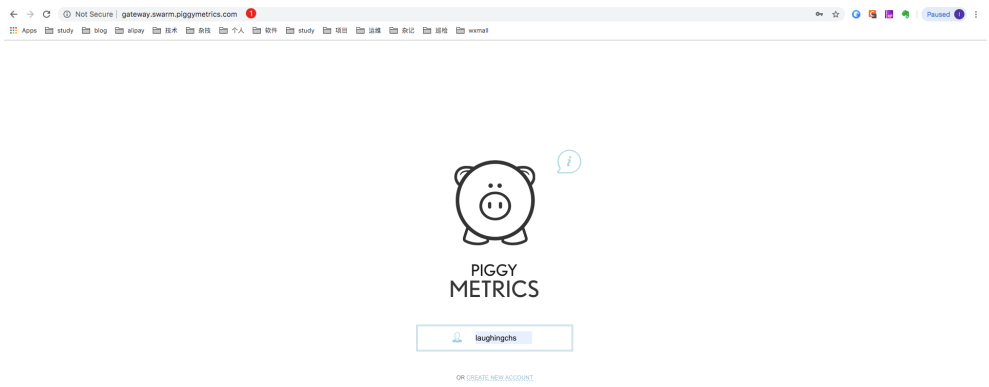
   您可以使用上个步骤配置的域名访问业务，测试业务功能是否正常，并观察SLB流量是否有流量进入。

   i. 使用域名访问业务。

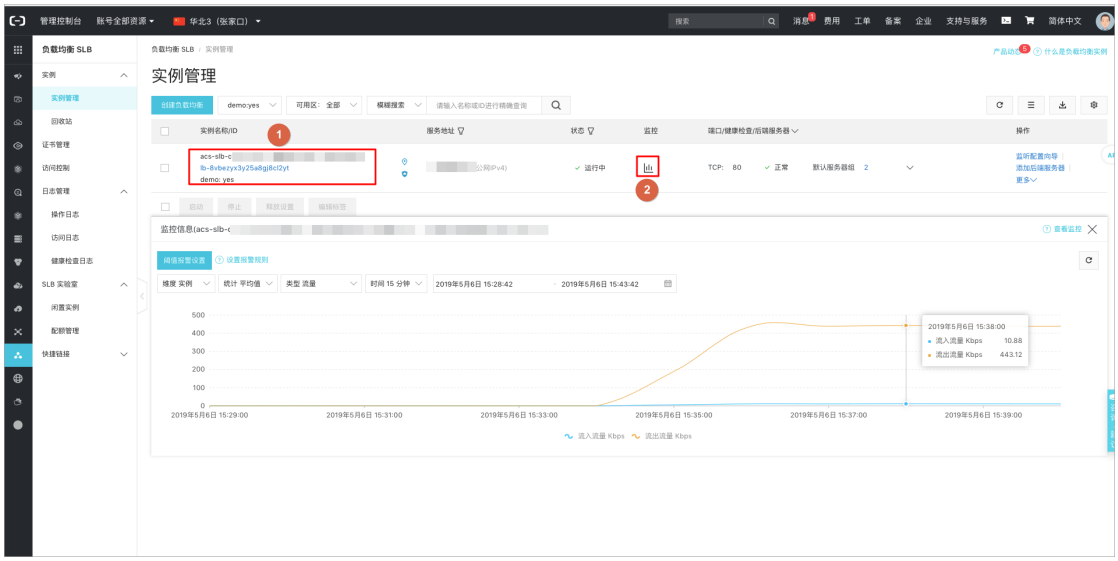
      容器服务提供的域名

      用户自定义的域名

ii. 查看SLB监控是否有流量进入。



> **说明** 请完成业务功能测试，确认业务流量通过SLB访问域名的方式对业务无影响。

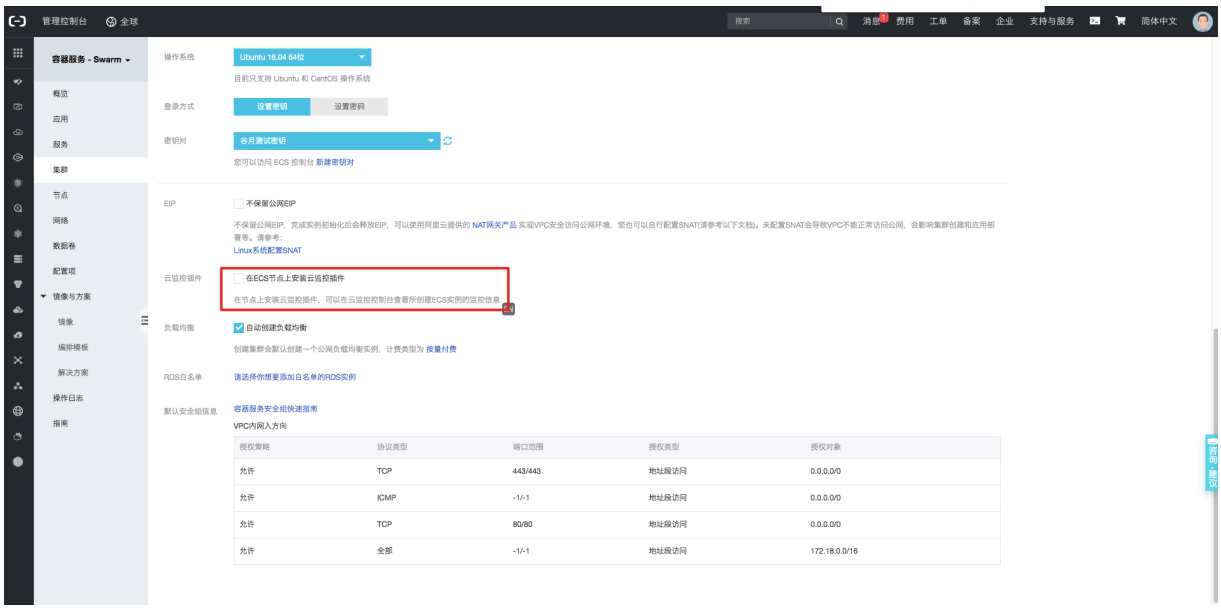
4. 修改流量解析入口。
   - 修改DNS解析地址：若原先客户端是通过域名方式访问服务的，请联系DNS服务商修改DNS解析，将原来Node Host IP修改成SLB IP。

     > **说明** 修改DNS解析地址后，各DNS服务商都有缓存，需要观察SLB监控看流量是否已切换完成。

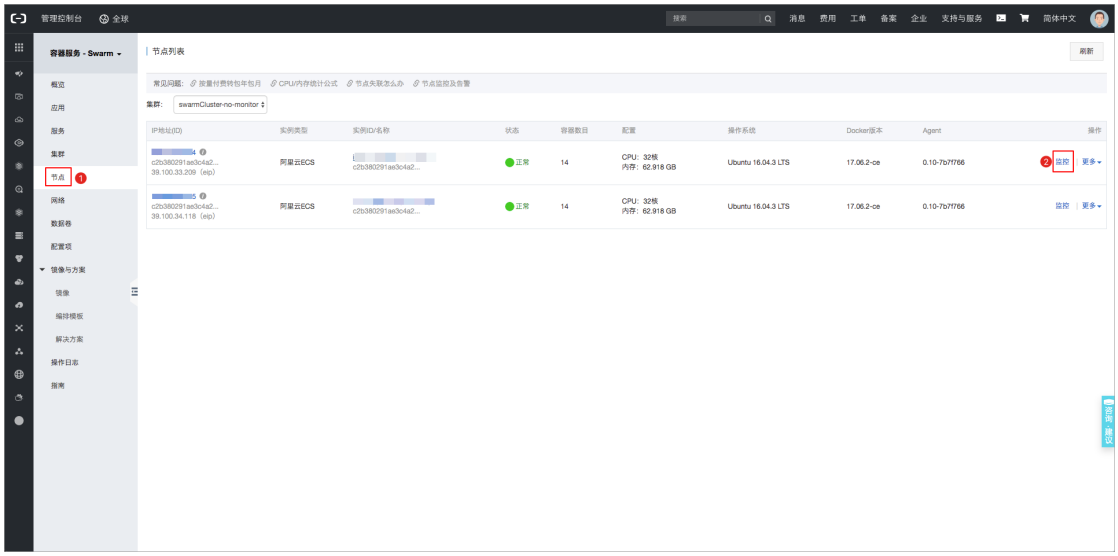
   - 修改客户端访问地址：若原先客户端是通过Node Host IP访问的，请升级客户端，修改成基于域名访问的方式。

统一部署云监控（可选）

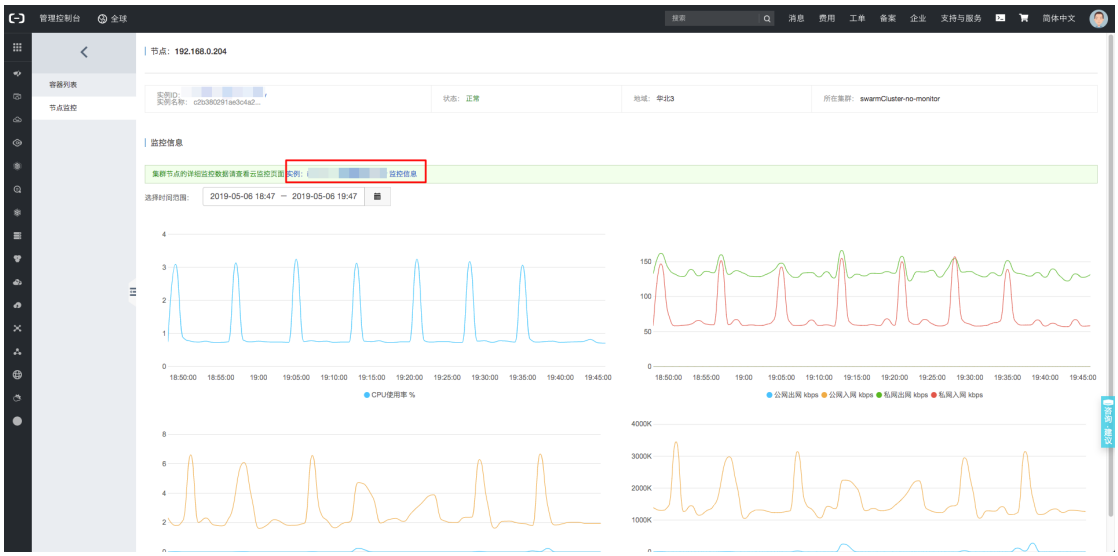
如果在创建集群时，没有勾选**在ECS节点上安装云监控插件**，则无法在云监控查看ECS实例的监控信息，不利于后续迁移时观察集群情况。例如下线Swarm集群时，无法通过云监控确认对应ECS机器是否还有流量。您可以通过如下操作，部署云监控。



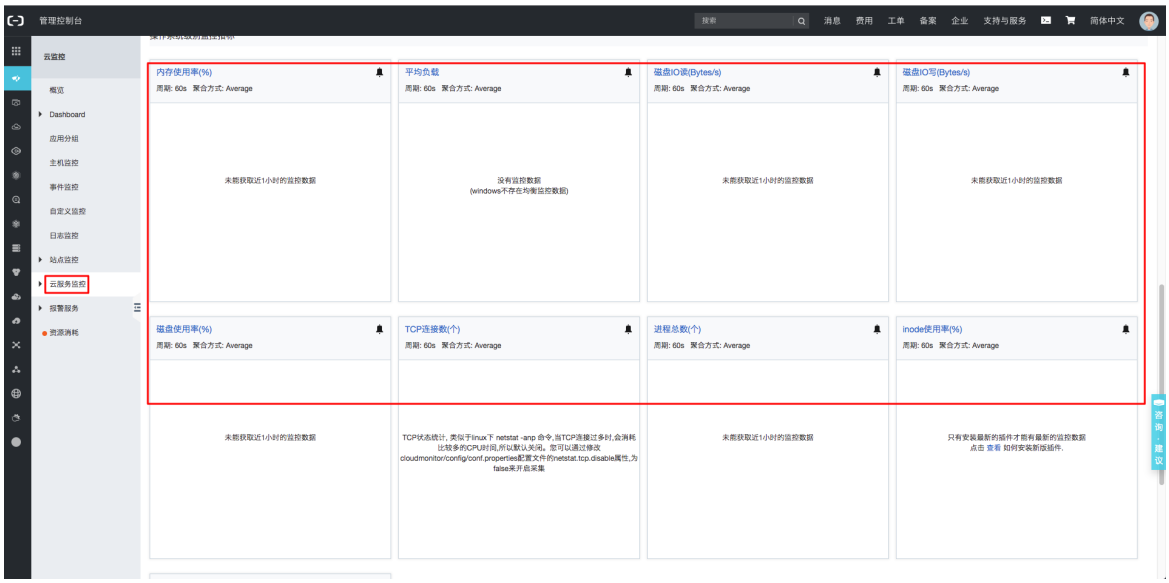
1. 查看Swarm集群节点ECS监控信息。
   i. 您可以登录容器服务管理控制台，在左侧菜单栏单击**节点**，在目标节点右侧单击**监控**，进入云监控入口链接页面。



ii. 单击**节点监控**，在**监控信息**区域可以查看到该节点在某个时间区间内的详细监控信息。

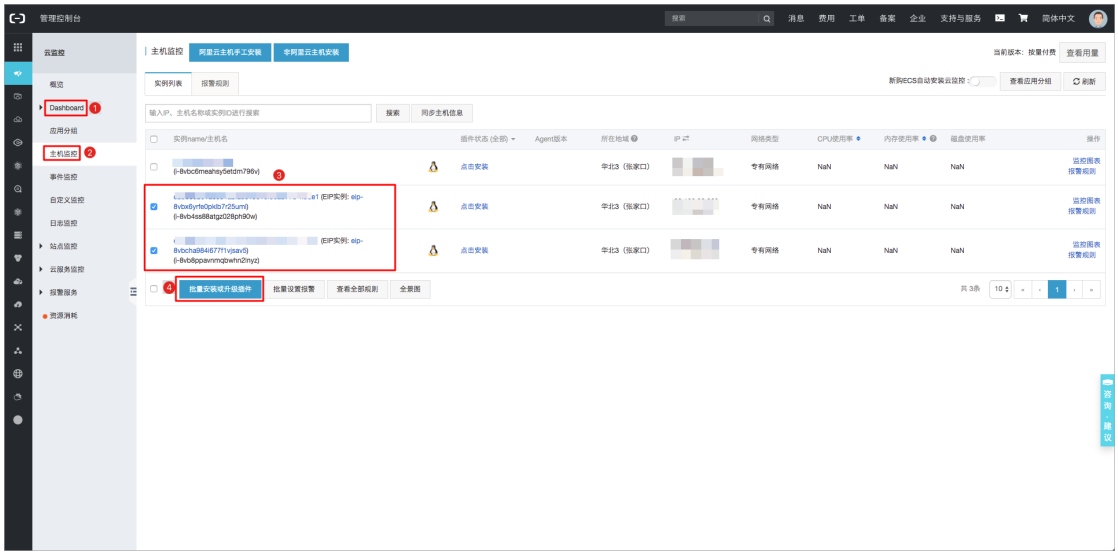


如果未安装云监控插件，则无法查看ECS详细监控信息如内存、磁盘、TCP连接数等。您可以按照如下操作，ECS云监控插件。

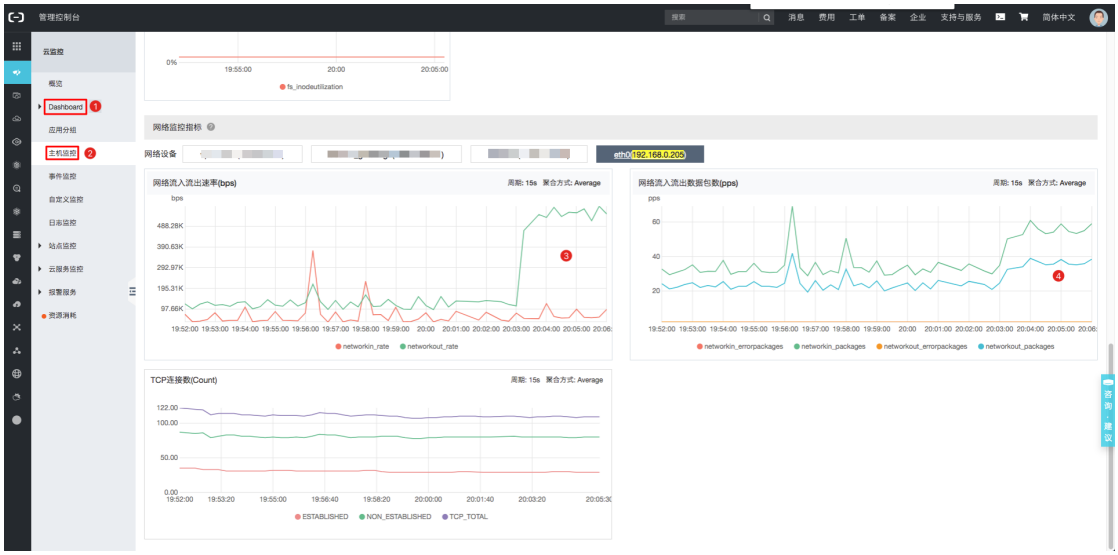


2. 在Swarm集群节点上手动安装ECS云监控插件。

i. 登录云监控控制台，在左侧导航栏选择**Dashborad > 主机监控**，在**实例列表**页签，勾选目标实例，并单击**批量安装或升级插件**，选择对应的ECS机器列表升级插件即可。



ii. 安装完成后，可以在**Dashborad > 主机监控**中，看到**网络监控指标**中，该网络设备的网络流入流出速率的曲线图。



13.4. 迁移集群配置

本文以swarm-piggymetrics-cluster集群为例，为您介绍如何迁移该集群的配置，配置迁移主要包含创建Kubernetes集群、VPC网络互联互通验证、数据卷迁移、配置项迁移等操作。

前提条件

待迁移的Swarm集群已完成标准化改造。详情请参见标准化Swarm集群。

背景信息

本文以swarm-piggymetrics-cluster集群为例，为您介绍如何迁移该集群的配置，配置迁移主要包含创建Kubernetes集群、VPC网络互联互通验证、数据卷迁移、配置项迁移等操作。

创建Kubernetes集群

基于已有Swarm集群（swarm-piggymetrics-cluster）创建一个Kubernetes集群，详情请参见创建Kubernetes托管版集群。创建集群时，您需要注意以下情况：

- 地域和可用区选择上，选择跟swarm-piggymetrics-cluster集群同地域同可用区，以保证像VPC网络这类无法跨地域的资源能正常迁移。
- Kubernetes 集群只支持VPC网络，不支持经典网络。
  - 如果swarm-piggymetrics-cluster集群为VPC网络，请创建Kubernetes托管版集群时共用swarm-piggymetrics-cluster集群的VPC网络。
  - 如果swarm-piggymetrics-cluster集群为经典网络，则需要做VPC网络迁移，详情请参见混挂混访方案。
- 请创建Kubernetes托管版集群时，请确保已安装云监控插件，方便后续灰度或切流时，可通过云监控平台查看ECS实例的监控信息。
- 如果swarm-piggymetrics-cluster集群有使用阿里云日志服务，请创建Kubernetes托管版集群时打开**日志服务**，并选择复用已有Project，避免需要重新迁移日志服务配置。
- 如果swarm-piggymetrics-cluster集群有使用RDS云产品，请在创建Kubernetes托管版集群时，选择对应的RDS白名单，避免出现网络权限问题。

迁移Node标签

如果swarm-piggymetrics-cluster集群有对Node节点打用户标签，并在应用配置里面通过用户标签指定Node节点部署，需要做Node用户标签的迁移。迁移过程如下：

1. 查看Swarm集群用户标签。
   i. 登录容器服务Swarm控制台，在左侧导航栏单击**集群**，在目标集群右侧单击**管理**。



   ii. 单击**用户标签**，在swarm-piggymetrics-cluster集群右侧可以看到对应的用户标签。



2. 在创建Kubernetes托管版集群中对应Node节点打上同样的标签。
   i. 登录容器服务管理控制台，选择**节点管理 > 节点**，在**节点列表**右侧单击**标签管理**。



ii. 在**标签管理**页面，勾选目标节点，单击**添加标签**，填写标签的名称及值，单击**确定**。



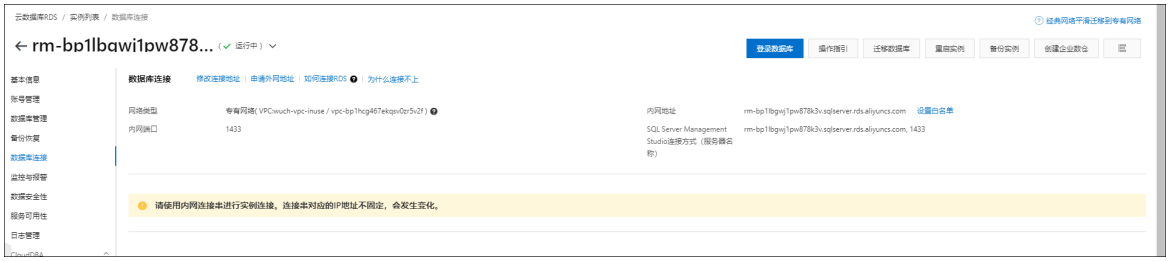
VPC互联互通验证

如果swarm-piggymetrics-cluster集群是经典网络，而创建Kubernetes托管版集群是VPC网络，则可能访问RDS、OSS 这类产品时，出现网络不通的情况。其中RDS会有白名单限制（在上述创建Kubernetes 托管版集群时，已选择添加RDS白名单），而OSS本身则针对经典网络和VPC网络提供两个不同的访问域名，故需要在创建Kubernetes 托管版集群上进行网络连通性验证。

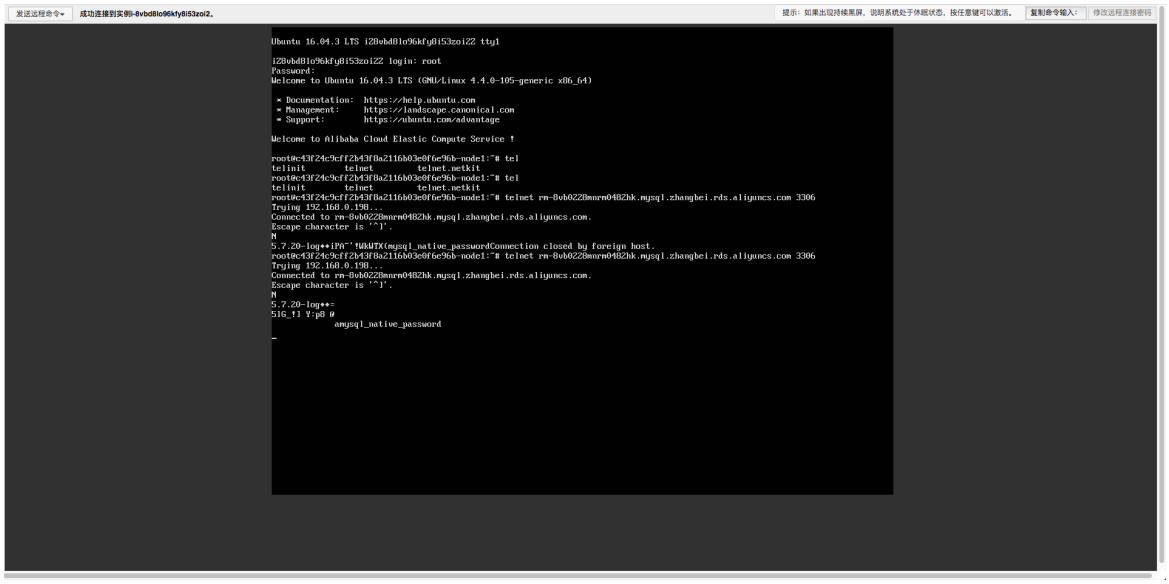
1. 登录RDS控制台，单击目标实例，进入实例基本信息页面，单击左侧导航栏**数据安全性**，在**白名单设置**中，确认RDS白名单已添加成功。



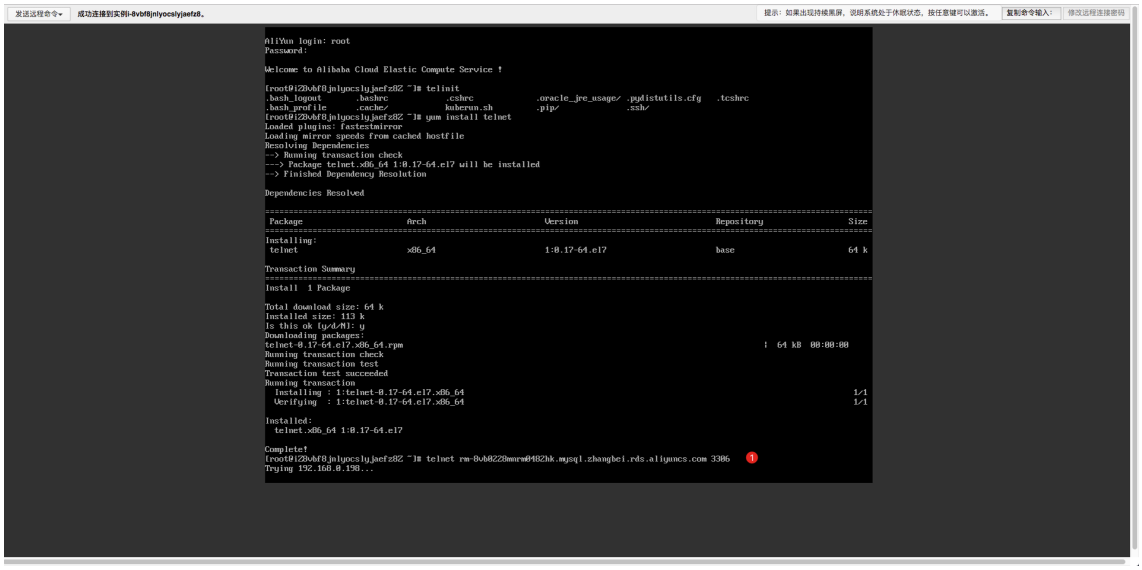
2. 单击**数据库连接**，在**实例连接**页签中，看到的内网地址即为RDS的VPC网络域名。



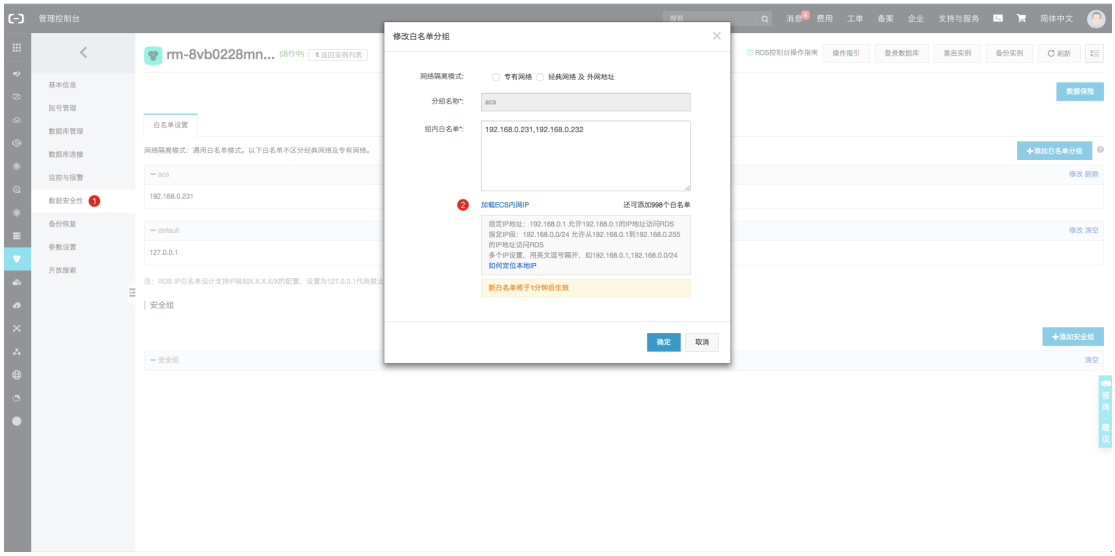
3. 登录 ECS 管理控制台，根据界面提示操作，确认网络连通性是否正常。



- 网络连通，进行下一步操作。
- 网络不通，可能存在以下原因。

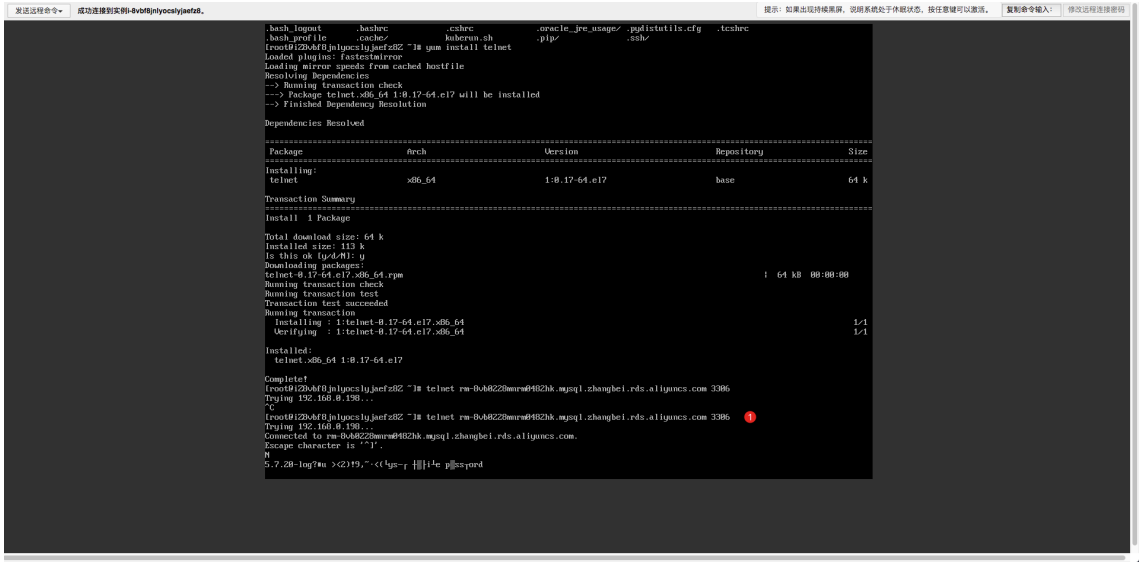


- RDS白名单添加失败。并没有将当前创建Kubernetes 托管版集群的ECS实例添加到RDS的白名单列表中。您可以登录RDS控制台，在**RDS实例 > 数据安全性**上，设置访问白名单，详情请参见设置白名单。

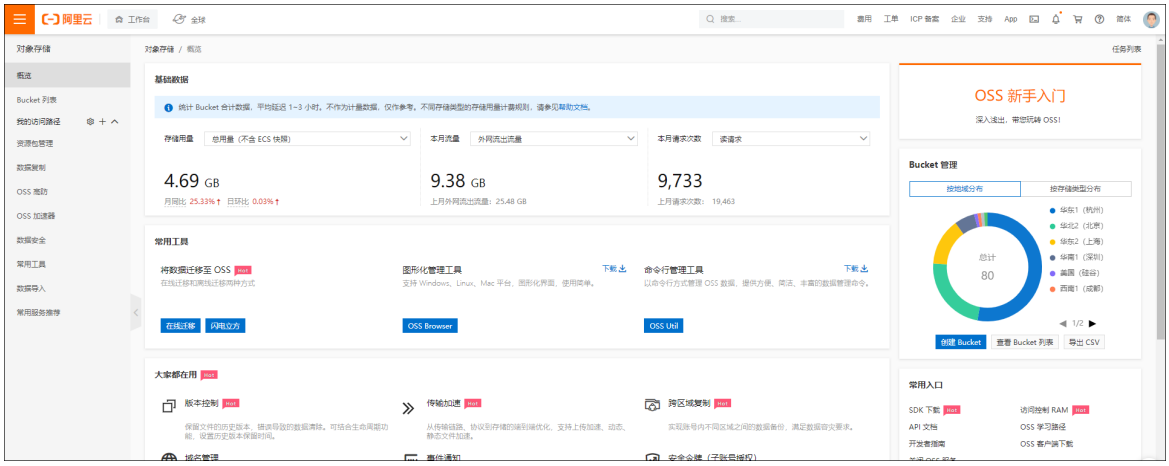


- 其他问题请参见解决无法连接实例问题。

请排查并修改完成后，再登录 ECS 管理控制台，确保网络已连通后，进行下一步操作。



4. 登录OSS 管理控制台，可以看到针对经典网络和VPC网络有各自对应的域名地址。



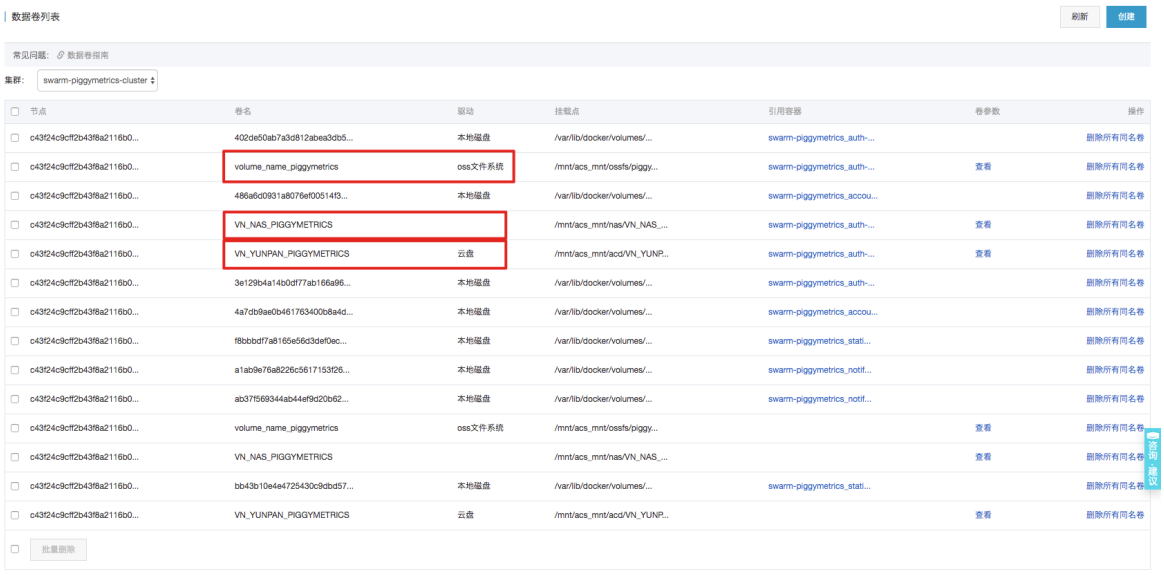
如果swarm-piggymetrics-cluster集群通过经典网络域名访问OSS的，后续需修改为通过VPC网络域名访问（后面集群数据卷迁移时也需要用到）。

数据卷迁移

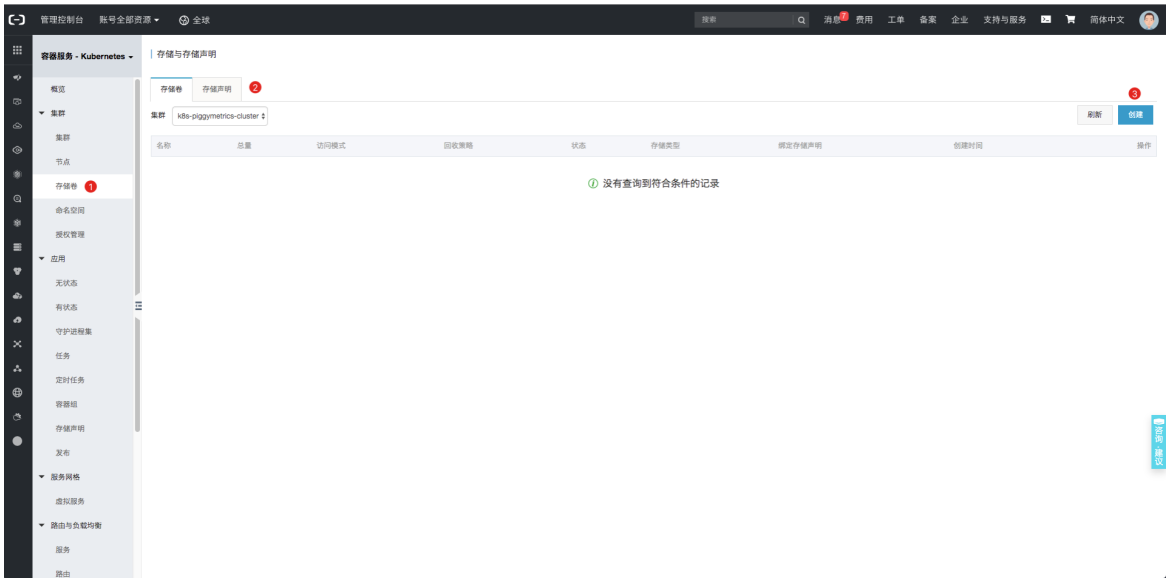
Swarm数据卷对应到Kubernetes的存储卷都是挂在集群维度下面的。因此可按集群维度先统一做数据卷配置本身的迁移，当后面应用迁移时，只需做数据卷引用迁移即可。一个Swarm数据卷对应到Kubernetes的存储卷PV和存储声明PVC两个概念。

Swarm数据卷支持NAS、云盘、OSS三种类型，而这三种类型在Kubernetes中使用方式上均支持**直接使用Volume方式**、**使用PV/PVC**两种方式来使用存储卷（详情请参见存储管理概述），本文档中以**使用PV/PVC**方式为例。

1. 登录容器服务Swarm控制台，在左侧导航栏单击**数据卷**，集群选择*swarm-piggymetrics-cluster*，在**数据卷列表**中分别找到NAS、云盘、OSS这三种类型的数据卷。

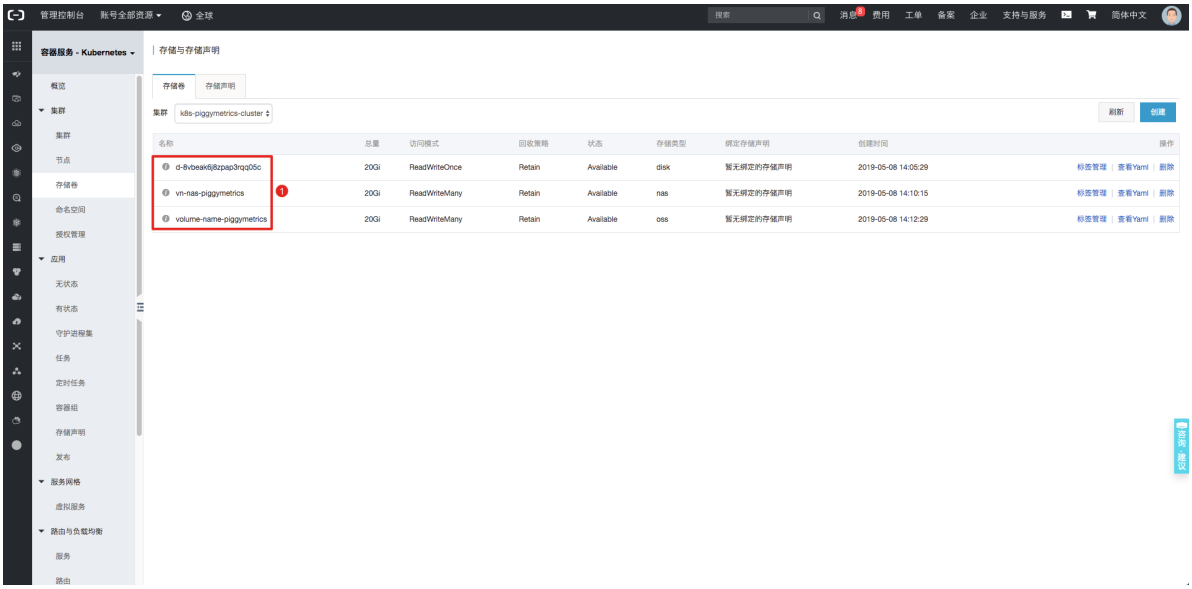


2. 在创建的Kubernetes托管版集群中，依次创建这三类数据卷对应的存储卷PV和存储声明PVC。详情请参见以下文档：
   - 云盘存储卷概述
   - NAS存储卷概述
   - OSS存储卷概述

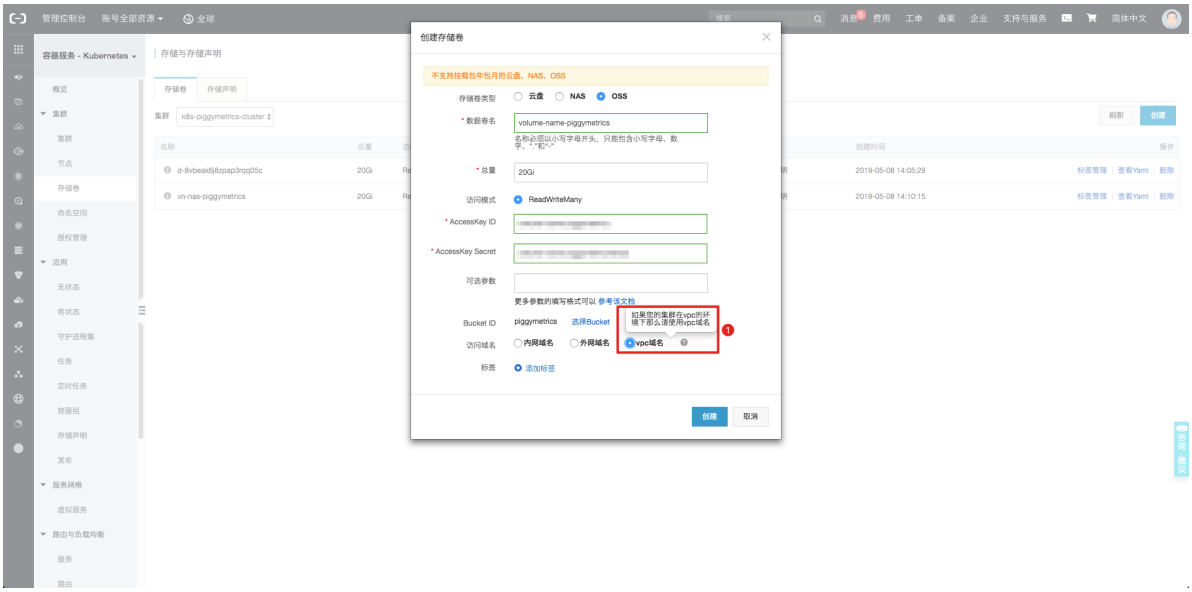


数据卷迁移说明

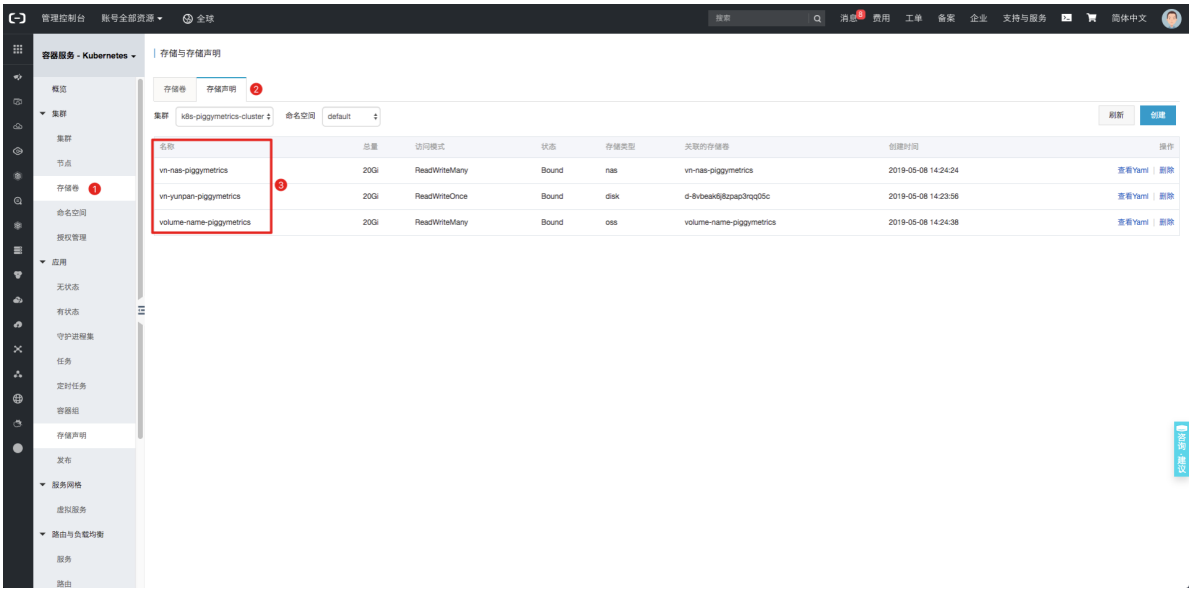
- 因为云盘是阿里云存储团队提供的非共享存储，只能同时被一个pod挂载，若Swarm和Kubernetes一起写入会报错。因此，若在Swarm集群里面存在使用云盘的场景则无法做到不停机迁移，请选择切换到共享存储NAS/OSS或在业务低峰期做停机迁移。只有将云盘先从Swarm集群卸载之后，才能在Kubernetes集群里面挂载。云盘操作请参见卸载数据盘和查看或删除数据卷。
- Kubernetes集群中，数据卷名不支持大写字母和下划线。如果数据卷名存在大写字母或下划线，请做如下统一转换：
  - 大写字母请转换为小写字母。
  - 下划线请转换为中划线。



- 在创建OSS存储卷时，访问域名需要选择为*VPC域名*，因为Kubernetes只支持VPC网络。



- 创建存储声明PVC，名称请默认与数据卷名一致（后面应用配置转换工具kompose工具会与此保持一致）。



配置项迁移

在Swarm里面，支持创建配置项，并通过配置参数传递配置，以方便对多个容器的环境变量进行统一管理，在应用发布时通过选择配置文件进而实现替换编排模板YAML文件里面的变量符$。

但是阿里云容器服务Swarm控制台提供的增强功能，在标准Kubernetes里面并没有完全对应的功能实现。其中，Kubernetes集群的配置项实体与此有差异，暂无法实现自动迁移，只能在迁移应用配置概述章节，手动处理Swarm编排模板YAML文件里面的变量符$。

13.5. 迁移应用配置

13.5.1. 迁移应用配置概述

本文将介绍在运维人员完成集群配置迁移后，开发人员如何将Swarm集群里面的应用配置迁移到Kubernetes集群。其主要包括准备应用迁移环境、预处理和转换应用的Swarm编排文件；然后部署转换后的Kubernetes资源文件，针对其中不支持自动转换的应用配置做手动迁移；最后再逐一调试解决应用启动过程中暴露的问题。

概念对比

Swarm和Kubernetes的概念较多，且二者在应用、服务、访问方式上均有较大不同；二者实体概念的差异请参见概念比对。

前提条件

您已经存在一个待迁移Swarm应用，且已完成集群维度配置迁移，这里以swarm-piggymetrics应用作为示例，通过迁移应用配置，将其部署到k8s-piggymetrics-cluster集群上。

其中，swarm-piggymetrics是个微服务架构应用，其来源于PiggyMetrics。PiggyMetrics是github上的一个SpringCloud应用项目。

迁移应用配置流程

1. 准备迁移环境。
2. 预处理Swarm编排文件。
3. 转换Swarm编排文件。
4. 部署 Kubernetes 资源文件。
5. 手动迁移应用配置。
6. 应用启动调试。
7. 迁移应用日志配置。

相关参考

迁移过程中常见问题，请参见应用配置迁移异常解决方案。

关于标签的映射关系，请参见配置类标签、发布类标签、网络配置类标签和日志配置类标签。

13.5.2. 准备迁移环境

将Swarm集群里面的应用配置迁移到Kubernetes集群时，您需要准备迁移环境。本文介绍如何准备迁移环境。

背景信息

在做应用配置迁移时，kompose是个开源工具，可以一键将Swarm compose文件转换为Kubernetes资源文件，阿里云在其基础上针对阿里云定制标签做了部分增强。

当我们通过kompose工具转换得到Kubernetes资源文件后，可以通过kubectl工具连接集群，并将资源文件部署到Kubernetes集群。

由于每个应用配置迁移中均需用到kompose和kubectl工具，建议找一台独立ECS部署，避免每次迁移时都需要重新准备迁移环境。

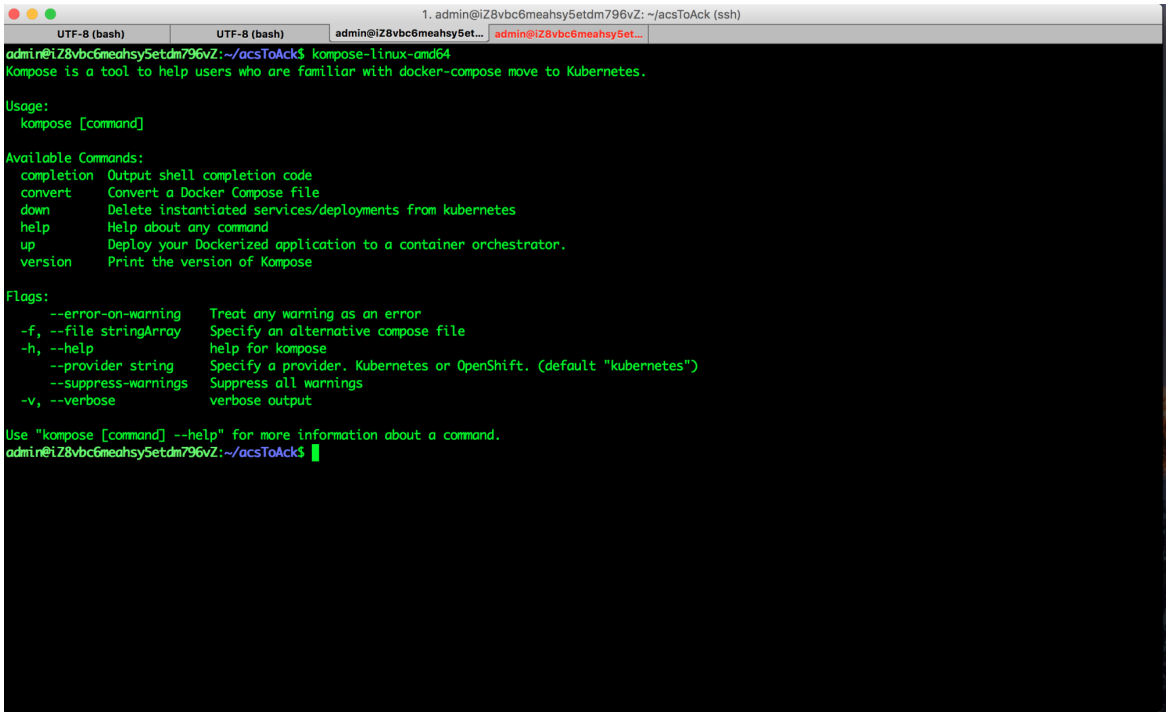
操作步骤

1. 安装kompose工具。

   kompose工具可以一键将Swarm compose文件转换为Kubernetes资源文件，阿里云在其基础上针对阿里云定制标签做了部分增强。阿里云kompose工具项目，请参见AliyunContainerService/kompose。

安装过程：直接到github上按OS类型下载最新版本的可执行文件AliyunContainerService/kompose/releases。

```
kompose-linux-amd64
```

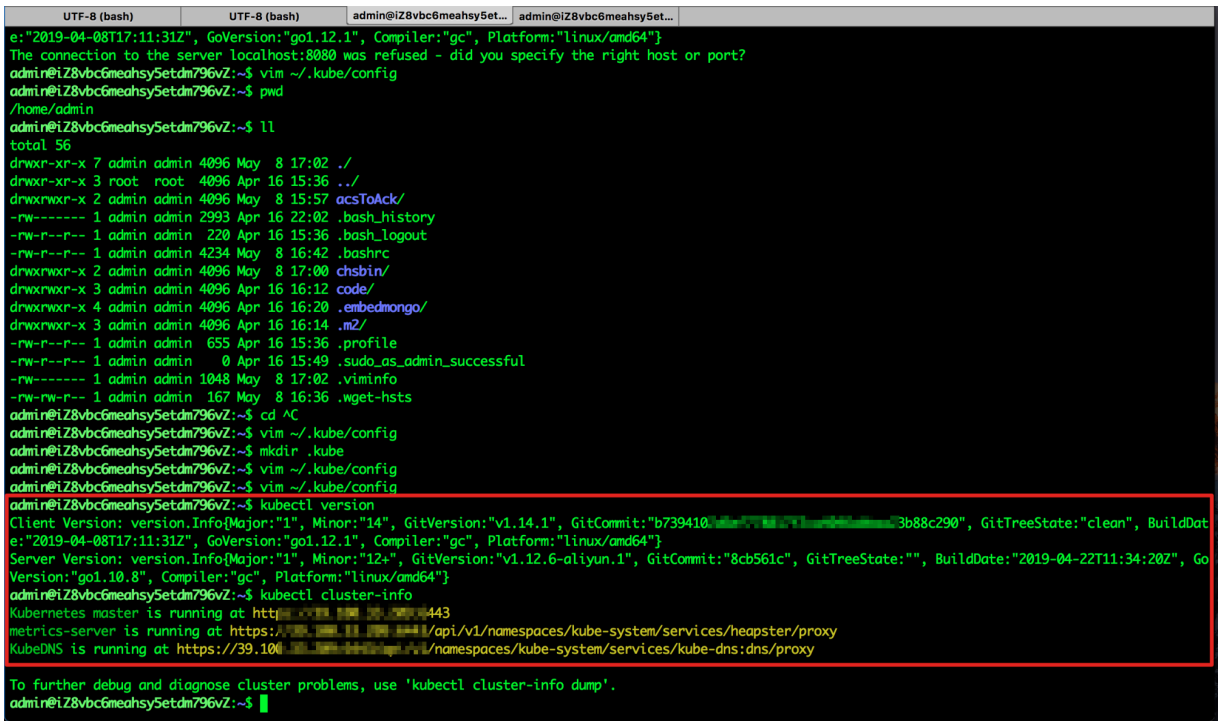


2. 配置kubectl环境。
   i. 从Kubernetes 版本页面下载最新版kubectl客户端。
   ii. 安装和设置kubectl客户端。详情请参见安装和设置 kubectl。
   iii. 配置Kubernetes集群票据。
      a. 登录容器服务管理控制台。
      b. 在控制台左侧导航栏中，单击**集群**。
      c. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
      d. 在集群管理页面的**连接信息**页签，将KubeConfig的内容粘贴到计算机*$HOME/.kube/config*。
   iv. 安装配置成功后，可以通过以下命令测试是否安装成功，集群票据配置是否成功。

```
kubectl version
kubectl cluster-info
```

执行结果

当出现以下红框内容时，表示kubectl环境准备完成。



13.5.3. 预处理Swarm编排文件

本文介绍如何预处理Swarm编排文件。

操作步骤

1. 下载Swarm应用编排文件。
   i. 登录容器服务Swarm控制台，在左侧导航栏单击**应用**，在目标应用右侧单击**变更配置**。



   ii. 拷贝编排文件并保存到本地，文件名后缀为*.yaml*，本文命名为*swarm-piggymetrics.yaml*。
2. 预处理Swarm编排文件里面环境变量。

   Swarm编排文件支持配置项替换，是由阿里云Swarm控制台实现的功能，在Kubernetes里面没有与其对应的实体支持，需要在转换前，手动将编排文件里面的变量符*$*替换成该配置项真实值。具体的操作如下：
   i. 登录容器服务Swarm控制台，单击**配置项**，选择对应的地域，找到目标配置文件。



   ii. 参考该配置文件，手动将编排文件*swarm-piggymetrics.yaml*里面的变量符*$*替换成该配置项真实值。
3. 预处理部分标签。

您需要预处理部分Swarm标签，否则kompose因格式或不支持等原因，会报错并中断整个转换过程。标签的预处理主要包含以下两个方面，具体针对每个标签的详细操作请参见配置类标签、发布类标签、网络配置类标签、日志配置类标签。

- 以下标签的值需要将布尔型修改为字符串类型，如：true/false 修改为 `"true"` / `"false"` 。
  - aliyun.global
  - aliyun.latest_image
- 包含以下标签的Service无法做迁移，需要先手动删除，后续再人工迁移。
  - external

4. 修改编排文件版本号。

   因kompose工具目前主要支持V2版本的compose文件，所以需要将文件头的版本声明从version: '2.X'修改为version: '2'，否则转换时会报错。

13.5.4. 转换Swarm编排文件

本文介绍如何转换Swarm编排文件。

操作步骤

1. 使用kompose工具转换文件。

   在对Swarm编排文件预处理之后，接下来可以用kompose工具做转换，操作命令如下。

```
kompose-linux-amd64 convert -f source/swarm-piggymetrics.yaml --volumes PersistentVolumeClaimOrHostPath
```

```
admin@iZ8vbc6meahsy5etdm796vZ:~/acsToAck$ kompose convert -f source/swarm-piggymetrics.yaml --volumes PersistentVolumeClaimOrHostPath
WARN Unsupported memswap_limit key - ignoring
WARN Unsupported kernel_memory key - ignoring
WARN Unsupported memswap_reservation key - ignoring
WARN Unsupported name key - ignoring
WARN Unsupported hostname key - ignoring
WARN Unsupported shm_size key - ignoring
WARN Unsupported kernel_memory key - ignoring
WARN Unsupported memswap_limit key - ignoring
WARN Unsupported shm_size key - ignoring
WARN Unsupported memswap_reservation key - ignoring
WARN Unsupported hostname key - ignoring
WARN Unsupported depends_on key - ignoring
WARN Unsupported name key - ignoring
WARN Unsupported hostname key - ignoring
WARN Unsupported depends_on key - ignoring
WARN Unsupported memswap_limit key - ignoring
WARN Unsupported name key - ignoring
WARN Unsupported shm_size key - ignoring
WARN Unsupported kernel_memory key - ignoring
WARN Unsupported memswap_reservation key - ignoring
WARN Unsupported external_links key - ignoring
WARN Unsupported memswap_limit key - ignoring
WARN Unsupported shm_size key - ignoring
WARN Unsupported memswap_reservation key - ignoring
WARN Unsupported kernel_memory key - ignoring
WARN Unsupported hostname key - ignoring
WARN Unsupported depends_on key - ignoring
WARN Unsupported aliyun.log.timestamp label key - ignoring
WARN Unsupported aliyun.routing.session_sticky label key - ignoring
WARN Unsupported name key - ignoring
WARN Unsupported hostname key - ignoring
WARN Unsupported aliyun.log.timestamp label key - ignoring
WARN Unsupported aliyun.depends label key - ignoring
WARN Unsupported links key - ignoring
WARN Unsupported memswap_limit key - ignoring
WARN Unsupported shm_size key - ignoring
WARN Unsupported memswap_reservation key - ignoring
WARN Unsupported name key - ignoring
WARN Unsupported kernel_memory key - ignoring
WARN Unsupported hostname key - ignoring
WARN Unsupported depends_on key - ignoring
WARN Unsupported name key - ignoring
WARN Unsupported memswap_limit key - ignoring
WARN Unsupported shm_size key - ignoring
WARN Unsupported memswap_reservation key - ignoring
WARN Unsupported kernel_memory key - ignoring
WARN Unsupported hostname key - ignoring
WARN Unsupported hostname key - ignoring
WARN Unsupported links key - ignoring
WARN Unsupported aliyun.log.timestamp label key - ignoring
WARN Unsupported hostname key - ignoring
WARN Unsupported external_links key - ignoring
WARN Unsupported depends_on key - ignoring
WARN Unsupported hostname key - ignoring
WARN Unsupported external_links key - ignoring
WARN Unsupported depends_on key - ignoring
WARN Unsupported aliyun.log.timestamp label key - ignoring
WARN Unsupported depends_on key - ignoring
WARN Unsupported aliyun.log.timestamp label key - ignoring
WARN Unsupported hostname key - ignoring
WARN Unsupported external_links key - ignoring
WARN Unsupported hostname key - ignoring
WARN Unsupported logging key - ignoring
WARN Unsupported devices key - ignoring
INFO Kubernetes file "gateway-service.yaml" created
INFO Kubernetes file "monitoring-service.yaml" created
INFO Kubernetes file "rabbitmq-service.yaml" created
INFO Kubernetes file "registry-service.yaml" created
INFO Kubernetes file "turbine-stream-service-service.yaml" created
INFO Kubernetes file "account-mongodb-deployment.yaml" created
INFO Kubernetes file "account-service-deployment.yaml" created
INFO Kubernetes file "auth-mongodb-deployment.yaml" created
INFO Kubernetes file "auth-service-deployment.yaml" created
INFO Kubernetes file "config-deployment.yaml" created
INFO Kubernetes file "gateway-deployment.yaml" created
INFO Kubernetes file "monitoring-deployment.yaml" created
INFO Kubernetes file "notification-mongodb-deployment.yaml" created
INFO Kubernetes file "notification-service-deployment.yaml" created
INFO Kubernetes file "rabbitmq-deployment.yaml" created
INFO Kubernetes file "registry-deployment.yaml" created
INFO Kubernetes file "statistics-mongodb-deployment.yaml" created
INFO Kubernetes file "statistics-service-deployment.yaml" created
INFO Kubernetes file "turbine-stream-service-deployment.yaml" created
admin@iZ8vbc6meahsy5etdm796vZ:~/acsToAck$
```

> **说明** 正常情况下，会生成对应Kubernetes资源文件，针对不支持自动转换的标签，会有WARN提示。这部分标签请按后面的转换异常进行处理，其一般会有以下3种处理方式：
>
> - 手动调整原Swarm编排文件，然后重新通过kompose转换。
> - 手动修改kompose工具转换Kubernetes资源文件。
> - 暂时忽略，先部署转换好的Kubernetes资源文件，再通过容器服务控制台手动迁移配置。

2. 手动修复转换问题。

   kompose工具支持转换大部分标签，但少量Swarm标签无法支持自动转换（后续我们会进一步增强），需要我们根据kompose工具输出的警告信息，手动调整Swarm编排文件并重新用kompose工具转换；具体各类异常信息及对应解决方案参见应用配置迁移异常解决方案。

3. 手动完善Kubernetes资源文件。

   在通过kompose工具转换之后，部分kompose无法支持的标签，需要我们手动编写或完善Kubernetes资源文件。主要包括以下几类标签：

   - aliyun.routing.port_
   - aliyun.global
   - external
   - environment: constraint
   - extra_hosts
   - net
   - dns

   针对每个标签的详细操作请参见配置类标签、发布类标签、网络配置类标签、日志配置类标签。

13.5.5. 部署 Kubernetes 资源文件

本文介绍如何部署 Kubernetes 资源文件。

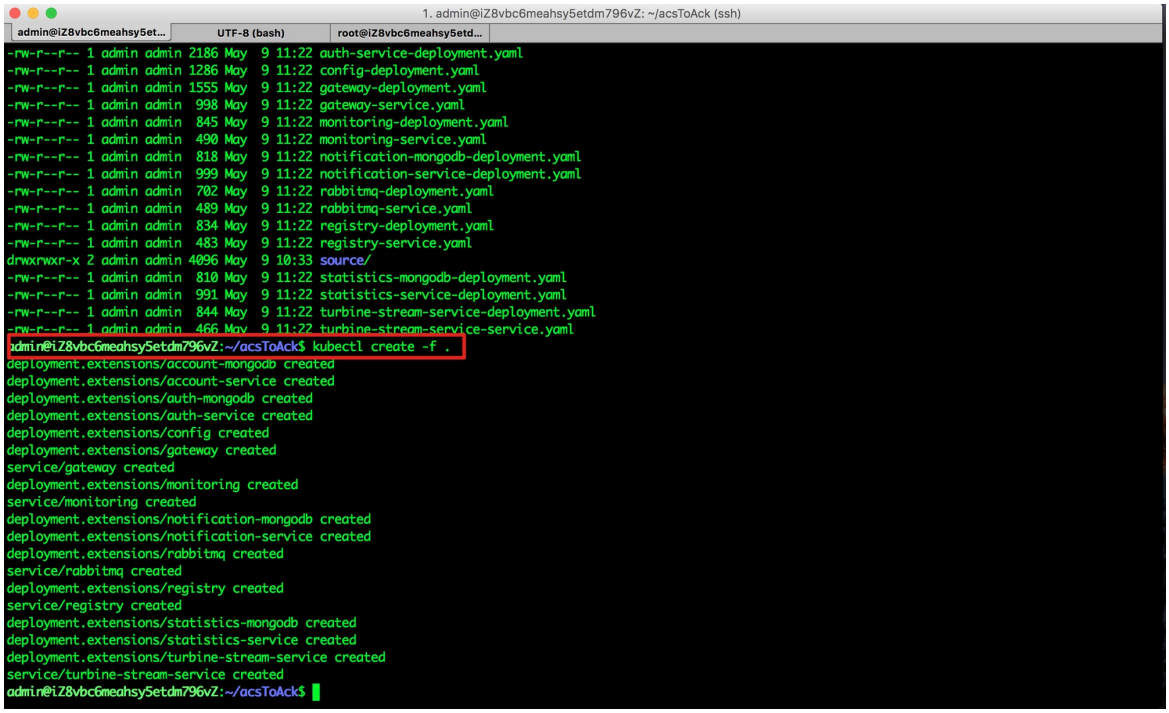
背景信息

针对前面 kompose 转换成功并预处理后的 Kubernetes 资源文件，我们可以通过 kubectl 工具部署到 Kubernetes 集群中。其中，集群票据在配置 kubectl 环境已配置完成。

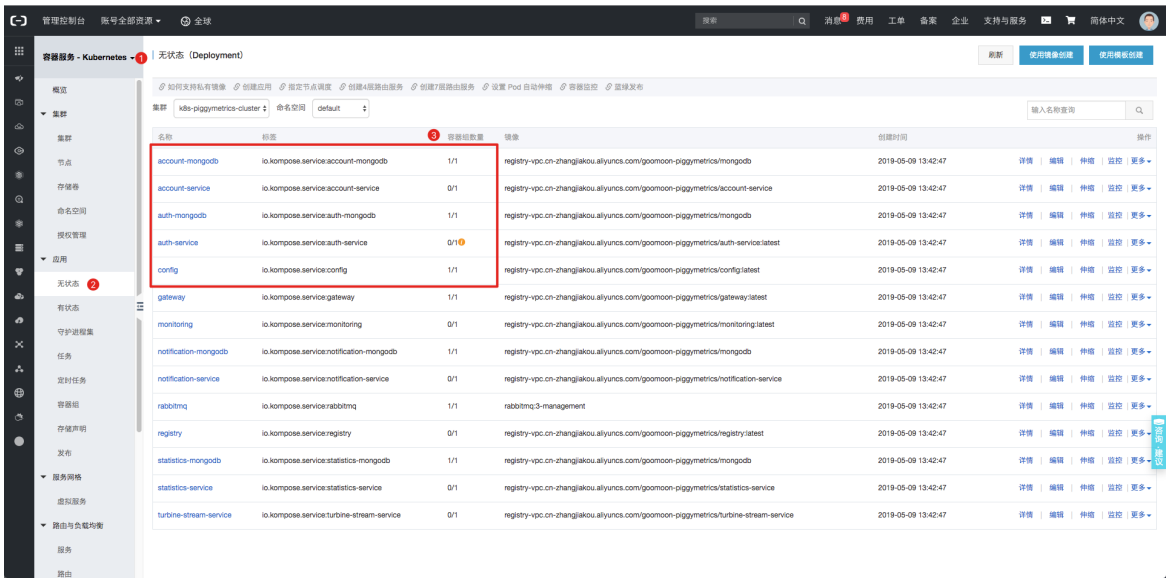
操作步骤

1. 可通过如下命令，批量部署当前目录下的所有资源文件。

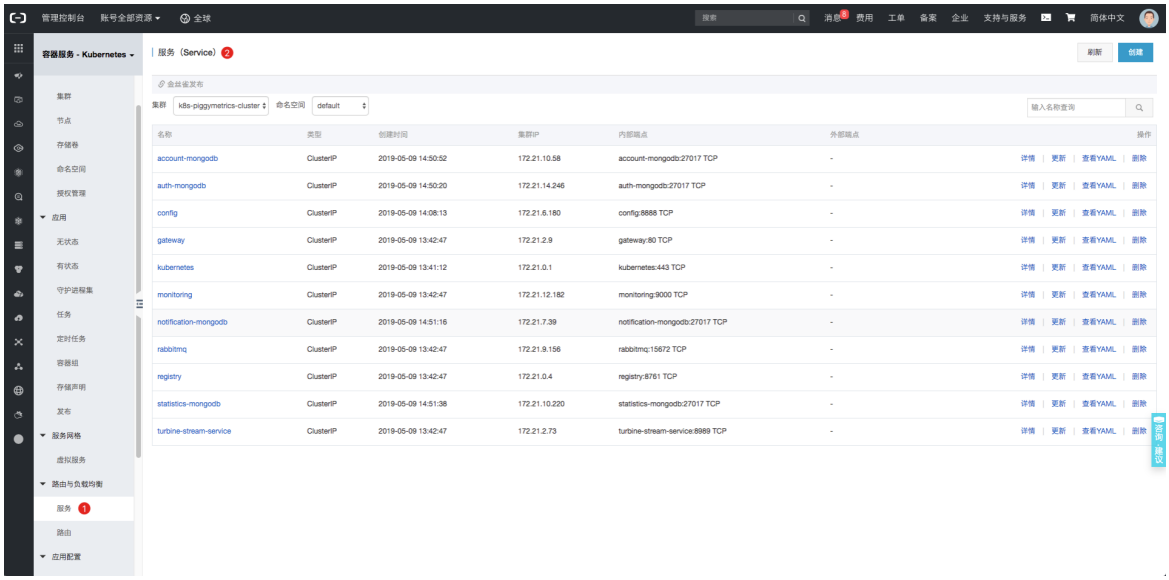
```
kubectl create -f .            #通过文件名或标准输入创建资源，.表示基于当前目录下的资源文件创建
```



2. 登录容器服务管理控制台，在左侧导航栏选择**应用 > 无状态**，查看应用部署情况。



3. 在**路由与负载均衡**下，单击**路由**或者**服务**，查看服务及路由配置情况。



13.5.6. 手动迁移应用配置

针对少量kompose工具无法支持的Swarm标签，需要您登录容器服务管理控制台进行手动迁移。主要包括以下几类标签，针对每个标签的详细操作请参见配置类标签、发布类标签、网络配置类标签和日志配置类标签。

- aliyun.auto_scaling
- hostname
- links
- external_links
- aliyun.lb.port_
- aliyun.log_ttl_

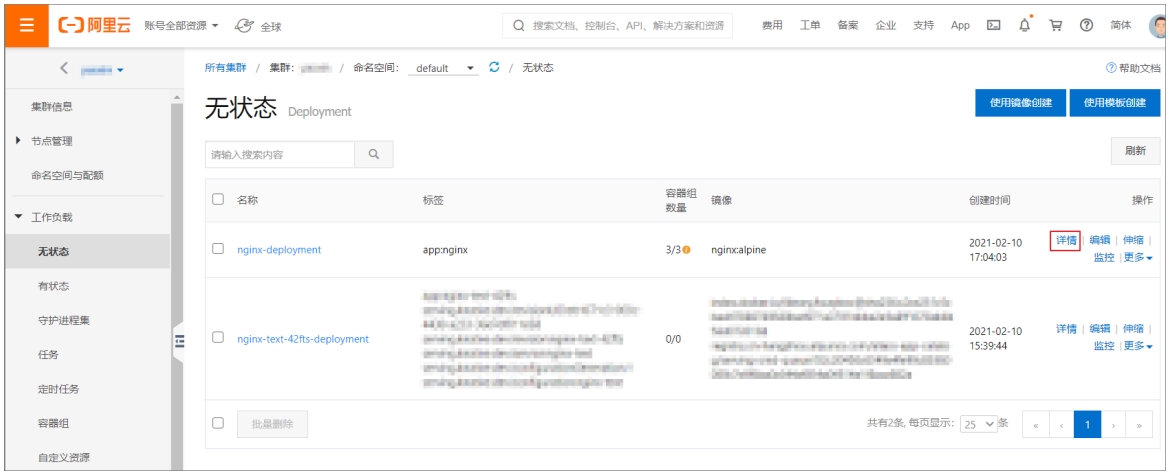
13.5.7. 应用启动调试

本文介绍应用配置迁移之后，如何通过控制台查看应用运行状态及应用启动出现问题时的分析与修复。

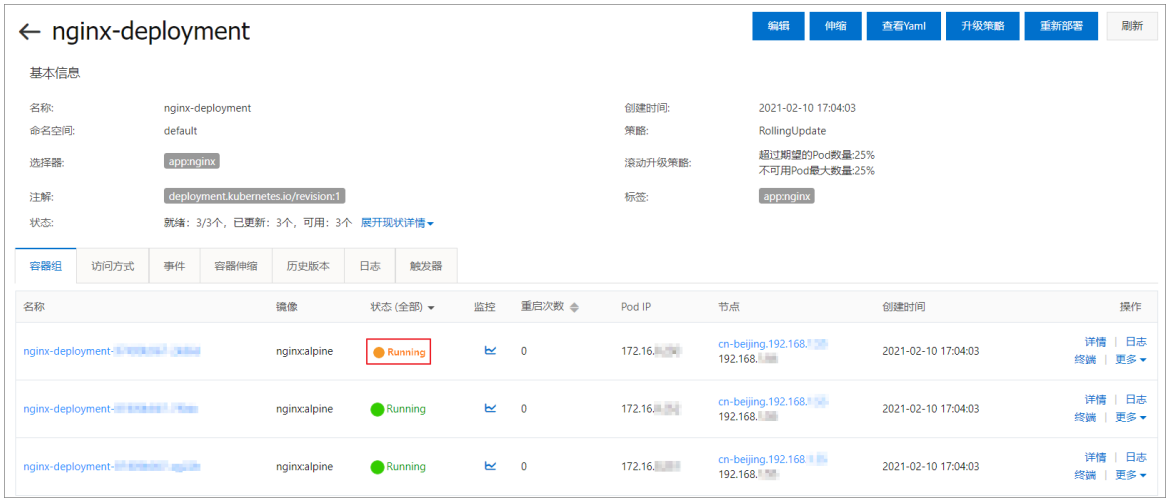
查看应用启动情况

通过kompose工具或手动完成应用配置迁移之后，接下来我们可以通过控制台查看应用是否正常启动，或在启动过程中是否有问题。我们可以通过K8s控制台查看应用启动日志定位问题。

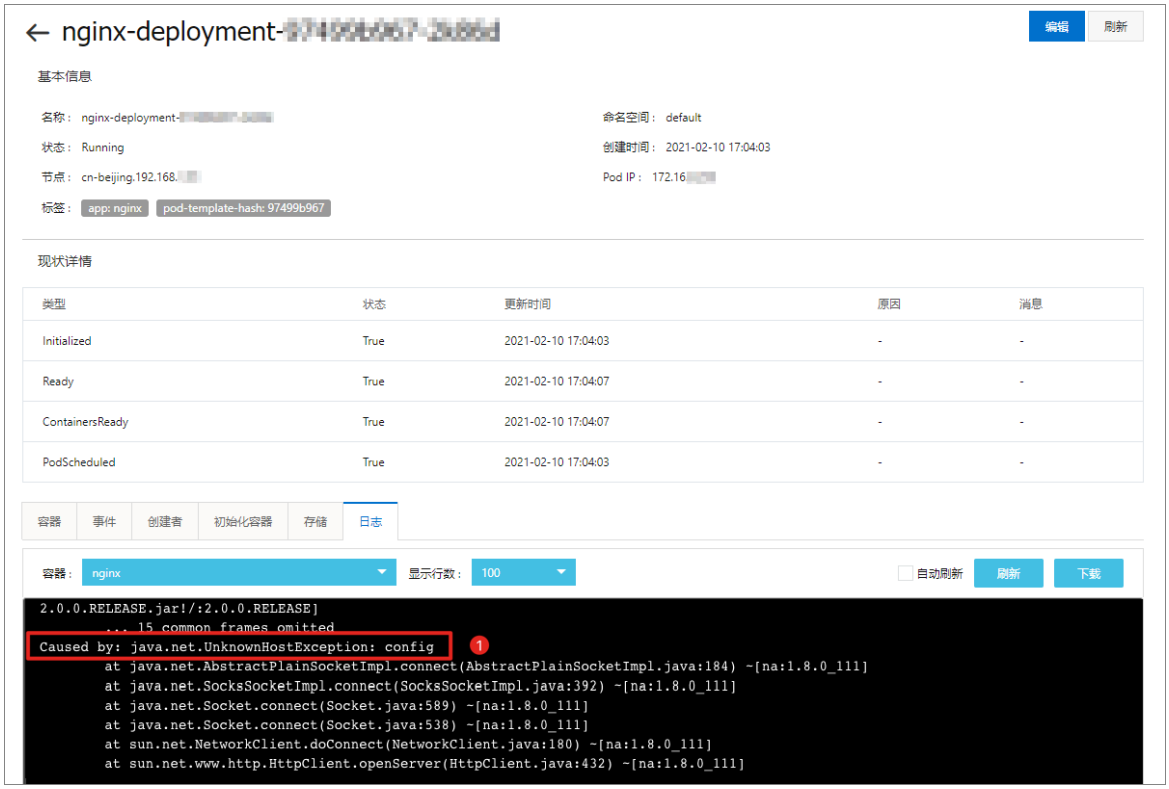
1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，单击**集群**。
3. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
4. 在集群管理页左侧导航栏中，选择**工作负载 > 无状态**。
5. 选择对应的命名空间，在目标应用（即，运行状态异常的应用）右侧单击**详情**。



6. 在**容器组**页签，在目标容器右侧选择**日志**。



7. 在**日志**页签，查看日志内容。

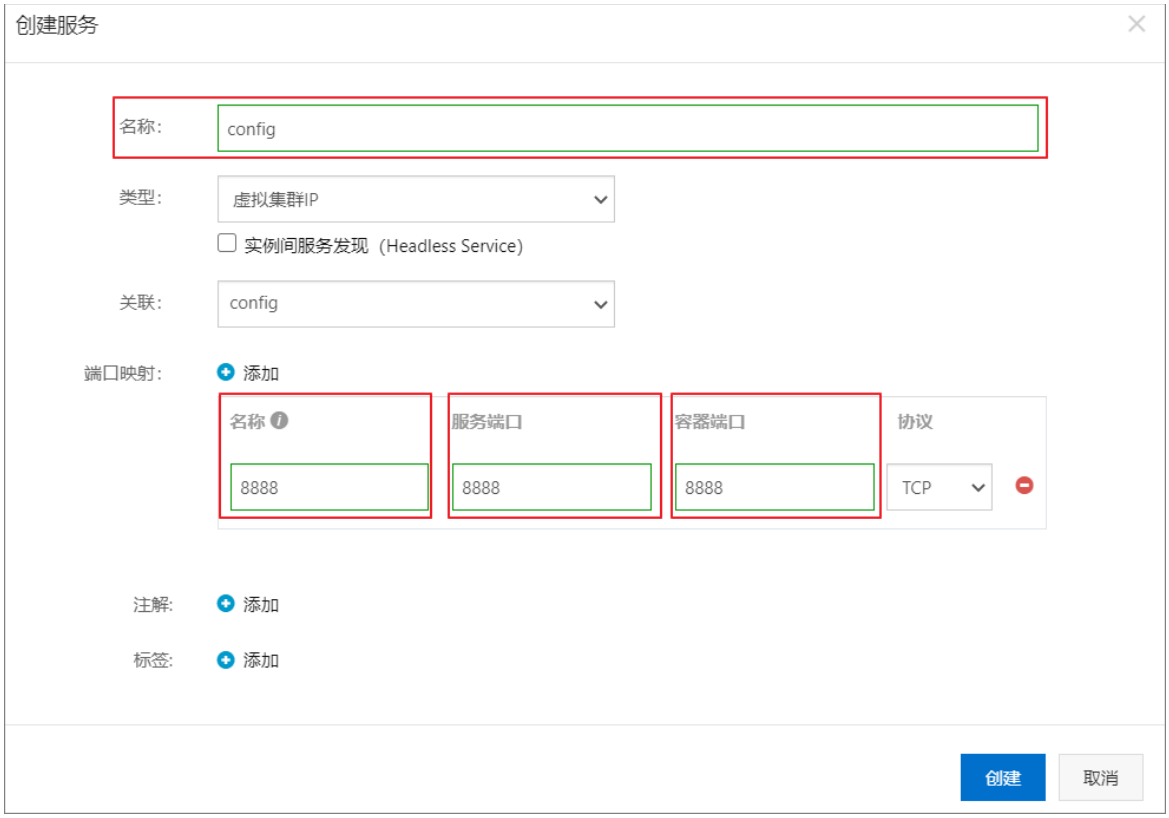


修复应用启动问题

针对应用启动出现的问题，我们需要逐个分析并修复。例如，上述日志显示故障原因为应用nginx-deployment无法访问域名config。解决方案如下：

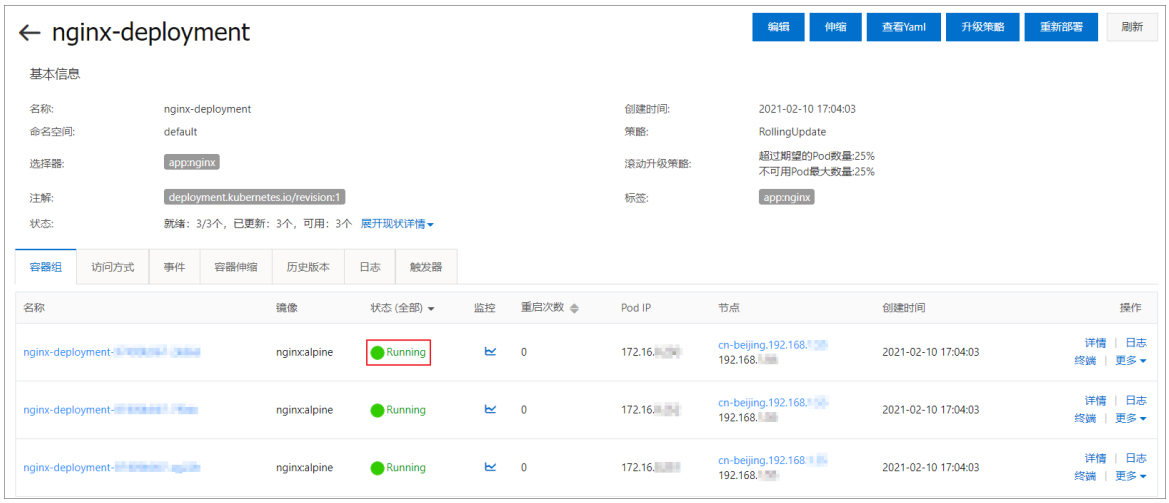
1. 通过手动创建一个Kubernetes集群的服务，来支持其它应用访问config应用即可。

   其中下图中的**名称**表示服务名，必须要跟hostname保持一致。**端口映射**没有特殊要求，可以将端口**名称**、**服务端口**与**容器端口**配置一致。详细操作，请参见控制台操作指导。



2. 修复完成后，容器重新部署之后再确认问题是否已修复。

   如下图所示表示容器状态为**运行中**，表示容器已正常启动。



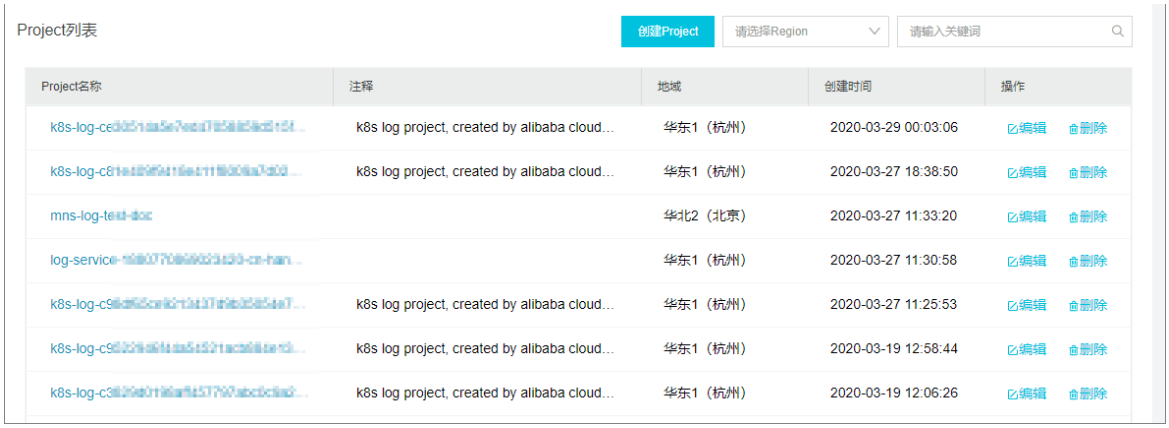
各个业务在应用启动过程中可能会出现各种异常，具体各类异常信息及对应解决方案参见应用配置迁移异常解决方案。

13.5.8. 迁移应用日志配置

本文介绍如何迁移应用日志配置。

如果您的应用里面使用到阿里云日志服务，则在迁移时，需要确认Kubernetes集群的日志输出是否正常，以及基于日志服务上的云监控等产品是否正常。

1. 登录日志服务控制台，在**Project列表**中，单击**Project名称**。



2. 在**日志库**页签，在Logstore列表中，可以看到对应的Logstore。

> **说明** 我们在创建Kubernetes集群时，复用了已有Swarm集群的日志Project，并通过kompose工具自动完成应用日志配置的迁移。但由于Swarm容器服务和K8s容器服务对于Logstore的生成规则不一样，Kubernetes的应用在对应日志Project下面，还是会生成不同的Logstore。
>
> - Swarm容器服务自动生成的Logstore，其生成格式为*acslog-${app}-${name}*。
> - Kubernetes容器自动生成的Logstore，其生成格式为*${name}*，并与Swarm容器服务Logstore一一对应。
> - 由于Kubernetes的Logstore不支持大写，所以kompose转换工具在转换时，会自动将大写字母转换化小写字母。

因此，所有底层依赖于日志服务，通过swarm容器服务创建的Project采集应用日志的其它阿里云产品的配置均需要做配置迁移或切换。目前主要包含以下几类，具体使用方式参见日志服务-实时消费完成日志切换和配置变更：

- 实时消费： SDK、Storm Spout、Spark Client、Web Console、云监控、业务实时监控ARMS等。
- 实时查询：用于日志的实时查询分析。
- 投递存储：通过将日志投递到种类存储，用于后续加工。
- 安全日志服务：日志服务与安全云产品对接，可通过ISV消费云产品日志。

13.5.9. 应用配置迁移异常解决方案

您可通过本文了解应用配置迁移过程中的常见问题及解决方法。

文件版本不正确

异常输出

```
FATA Version 2.1 of Docker Compose is not supported. Please use version 1, 2 or 3
```

异常原因

kompose工具只支持V1、V2、V3版本的compose格式，不支持V2.X版本，导致整个转换中断。

解决方案

将 `version: '2.X'` 修改为 `version: '2'` ，重新转换。

标签解析异常

- 异常输出

```
ERRO Could not parse config for project source : Unsupported config option for account-db service: 'external'
```

异常原因
kompose工具无法解析external标签，导致转换中断。
解决方案
出现**ERRO**或**FATA**级别的错误，修改原编排文件，移除配置，后续手动迁移配置，迁移方式请参见配置类标签、发布类标签、网络配置类标签、日志配置类标签。

- 异常输出

```
ERRO Could not parse config for project source : Unsupported config option for gateway service: 'net'
```

异常原因
kompose工具无法解析net标签，导致转换中断。
解决方案
修改原编排文件，移除配置，后续手动迁移配置，迁移方式请参见配置类标签、发布类标签、网络配置类标签、日志配置类标签。

标签值格式不正确

- 异常输出

```
ERRO Could not parse config for project source : Service 'auth-service' configuration key 'labels' contains an invalid type, it should be an array or object Unsupported config option for auth-service service: 'latest_image'
```

异常原因
根据提示是latest_image配置值格式无效，kompose工具无法转换。
解决方案
修改原编排文件，修复值配置， 将bool值修改成字符串形式，true修改成 `'true'` 。

> **说明** 以下标签列表均有此问题：
> - aliyun.latest_image: true
> - aliyun.global: true

- 异常输出

```
ERRO Could not parse config for project source : Cannot unmarshal '30' of type int into a string value
```

异常原因
值类型不对，查看compose文件*aliyun.log_ *标签值是否为30，但未带单引号 `''` ，因为其对应结构体应该是string不是int，所以得加引号。
解决方案
修改原编排文件，修复值配置，然后重新转换，将30修改成 `'30'` 。

标签不支持提醒

- 异常输出

```
WARN Unsupported hostname key - ignoring
```

异常原因
Swarm里面，支持hostName作为域名来访问服务，而在compose文件里面并没有声明对应的容器端口，使得kompose工具无法自动创建对应的Kubernetes服务，导致在Kubernetes里面无法通过hostName来访问应用。
解决方案
先部署资源文件，然后在Kubernetes控制台手动创建Kubernetes服务，其中服务名称为hostName，服务端口与容器端口号一致。

- 异常输出

```
WARN Unsupported links key - ignoring
```

异常原因
links底层跟hostName一样，支持通过links名称或别名访问服务。而在compose文件里面并没有声明对应的容器端口，使得kompose工具无法自动创建对应的Kubernetes服务，导致在Kubernetes里面无法通过links名称或别名来访问应用。
解决方案
先部署资源文件，然后在Kubernetes控制台手动创建Kubernetes Service，其中Service名称即为links名称或别名，服务端口与容器端口号一致。

标签转换有缺陷

- 异常输出

```
WARN Handling aliyun routings label: [{rabbitmq.your-cluster-id.alicontainer.com  http 15
672}]
```

异常原因
swarm简单路由里面配置了测试域名（swarm容器服务提供的域名），其跟集群标识cluster-id有关。因为不知道Kubernetes新集群id，kompose无法自动完成新测试域名的转换，需要手动修改对应的ingress文件，更新域名。
解决方案
找到各ingress文件*-ingress.yaml，将其中的*host: your-cluster-id.alicontainer.com*修改成对应Kubernetes集群的测试域名，您可以通过以下操作查看测试域名：

i. 登录容器服务管理控制台，在左侧导航栏选择**集群 > 集群**，在目标集群Kubernetes-piggymetrics-cluster右侧单击**管理**。
ii. 在**基本信息**页面，可以看到测试域名的值为*.c7f537c92438f415b943e7c2f8ca30b3b.cn-zhangjiakou.alicontainer.com，将其替换各个ingress文件的.your-cluster-id.alicontainer.com*字符串。

- 异常输出

```
WARN Handling aliyun routings label: [{gateway.your-cluster-id.alicontainer.com  http 400
0} {gateway.swarm.piggymetrics.com  http 4000}]
```

异常原因
Swarm里面简单路由配置多域名，但kompose工具转换时只支持单域名转名，后续域名需要自己添加ingress规则。
解决方案
修改转换后的Kubernetes资源文件，修复方式请参见配置类标签、发布类标签、网络配置类标签、日志配置类标签。

部署应用时报错

- 异常输出

```
error: error validating "logtail2-daemonset.yaml": error validating data: ValidationError
(DaemonSet.status): missing required field "numberReady" in io.Kubernetes.api.extensions.
v1beta1.DaemonSetStatus; if you choose to ignore these errors, turn validation off with -
-validate=false
```

异常原因
针对Swarm里面的**aliyun.global**：true这类服务，kompose工具会转成Kubernetes里面的DaemonSet，但自动转换后的资源文件里面会有status信息，用来记录中间状态，进而导致部署失败。
status：

- currentNumberScheduled: 0
- desiredNumberScheduled: 0
- numberMisscheduled: 0

解决方案
在转换后的Kubernetes资源文件*-daemonset.yaml*里面，删除status信息，再部署应用。

- 异常输出

```
error: error parsing auth-service-deployment.yaml: error converting YAML to JSON: yaml: l
ine 26: did not find expected key
```

异常原因
对应Kubernetes资源文件对应行的标签定义不符合要求，解析失败。最常见的是缩进问题，将当前标签误以为上一个标签的子标签，其实应该是兄弟标签等。
解决方案
按照标签列表和Kubernetes官方文档，确认当前行的正确配置，并修改Kubernetes资源文件。

- 异常输出

```
error: error parsing auth-service-deployment.yaml: error converting YAML to JSON: yaml: l
ine 34: found character that cannot start any token
```

异常原因
Kubernetes资源文件对应行上存在非token字符，最常见是tab按键。
解决方案
在转换后的Kubernetes资源文件*-daemonset.yaml*里面，将对应行tab修改成空格键。

启动失败

- 异常输出
  Pod容器一直在重启，并显示健康检查失败。

```
Initialized: True
Ready: False
ContainersReady: False
PodScheduled: True
```

异常原因
在Kubernetes里面会通过liveness probe和readiness probe来验证业务容器是否存活或就绪。类似于Swarm里面的aliyun.probe.url、aliyun.probe.cmd。但aliyun.probe.url、aliyun.probe.cmd如果有问题只会标注容器的健康状态，kompose工具会将其转换为liveness probe。而在Kubernetes中，如果liveness probe设置有问题，检查失败，则会自动触发重启容器，强制中断原有容器业务的初始化。
解决方案

i. 先验证是否为探针设置问题。

先去掉liveness probe或readiness probe探针，如果应用能启动成功，则说明是探针时间设置有问题。

ii. 修复问题。
根据业务需要，确认容器真正启动完成耗时，到Kubernetes对应应用配置里合理设置liveness probe的配置，尤其是其中的延迟探测时间、执行探测频率、超时时间等字段，详情请参见创建无状态工作负载Deployment。

- 异常输出
Pod容器一直在重启，并显示健康检查失败。

```
Initialized: True
Ready: False
ContainersReady: False
PodScheduled: True
```

进入容器查看日志如下：

```
2019-05-22 06:42:09.245  INFO [gateway,,,] 1 --- [           main] c.c.c.ConfigServicePro
pertySourceLocator : Connect Timeout Exception on Url - http://config:8888. Will be tryin
g the next url if available
```

异常原因
在这里请求 `config:8888` 超时，之前是因为该Pod的网络模型设置出错，设置了 `hostNetwork: true` ，通过ECS网络栈导致解析不了Kubernetes service名称。
若已经修复gateway-deployment.yaml里面的配置，并通过kubectl apply重新部署，但仍一直报错。
这种情况是因其他原因导致新配置 *-deployment.yaml没有真正生效。
解决方案
可以在保存配置后，登录容器服务管理控制台，删除对应的应用，再通过kubectl重新部署。

13.6. 附录：标签映射

13.6.1. 配置类标签

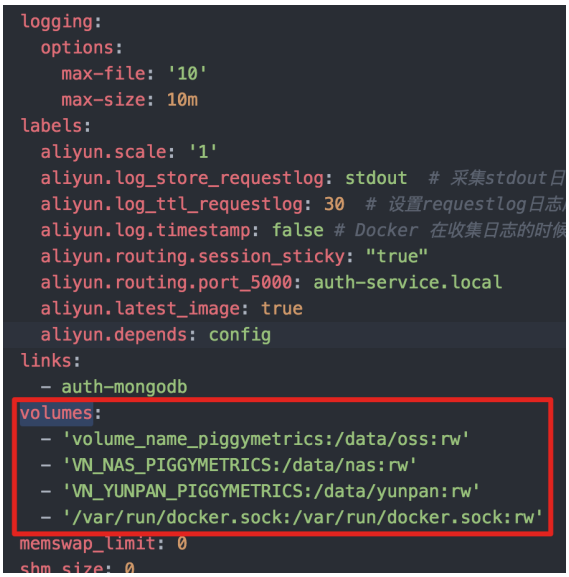
本文介绍容器服务Swarm集群相关的配置类标签。

Swarm标签名	Swarm标签含义	对应 Kubernetes配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
name	service名称，无特殊作用，可忽略。	Kubernetes不支持，迁移忽略。	略	略	无法迁移。
image	容器镜像。	image标签。	略	略	kompose自动迁移。
ports	容器端口映射。	containerPort标签若有对外访问诉求，则需要自己配置NodePort。	略	略	kompose自动迁移。
environment	容器环境变量。	env标签。	略	略	kompose自动迁移。

Swarm标签名	Swarm标签含义	对应 Kubernetes配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
volumes	挂载宿主机目录或存储卷。	volumeMounts标签。volumes标签。	请参见volumes的Swarm配置示例。	请参见volumes的Kubernetes配置示例。	kompose自动迁移。
cap_add/cap_drop	增加或移除容器对内核的修改权限。	securityContext:capabilities标签。	请参见cap_add/cap_drop的Swarm配置示例。	请参见cap_add/cap_drop的Kubernetes配置示例。	kompose自动迁移。
privileged: true	使用该参数，container内的root拥有真正的root权限。 否则，container内的root只是外部的一个普通用户权限。privileged启动的容器，可以看到很多host上的设备，并且可以执行mount。甚至允许您在docker容器中启动docker容器。	securityContext:privileged标签。	请参见privileged: true的Swarm配置示例。	请参见privileged: true的Kubernetes配置示例。	kompose自动迁移。
mem_limit	指定容器对内存资源限制。	resource:request/limits标签。	请参见mem_limit的Swarm配置示例。	请参见mem_limit的Kubernetes配置示例。	kompose自动迁移。
cpu_shares	指定容器对cpu资源限制。	resource:request/limits标签。	请参见cpu_shares的Swarm配置示例。	请参见cpu_shares的Kubernetes配置示例。	kompose自动迁移。
kernel_memory	同 docker run 参数。	Kubernetes不支持，迁移忽略。	略	略	无法迁移。
memswap_reservation	同 docker run 参数。	Kubernetes不支持，迁移忽略。	略	略	无法迁移。
memswap_limit	同 docker run 参数。	Kubernetes不支持，迁移忽略。	略	略	无法迁移。

Swarm标签名	Swarm标签含义	对应 Kubernetes配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
shm_size	同 docker run 参数。	Kubernetes不支持，迁移忽略。	略	略	无法迁移。
oom-kill-disable	设置是否禁止OOM Killer，和 `docker run` 命令中的--oom-kill-disable 参数语义一致。	Kubernetes不支持，迁移忽略。	略	略	无法迁移。

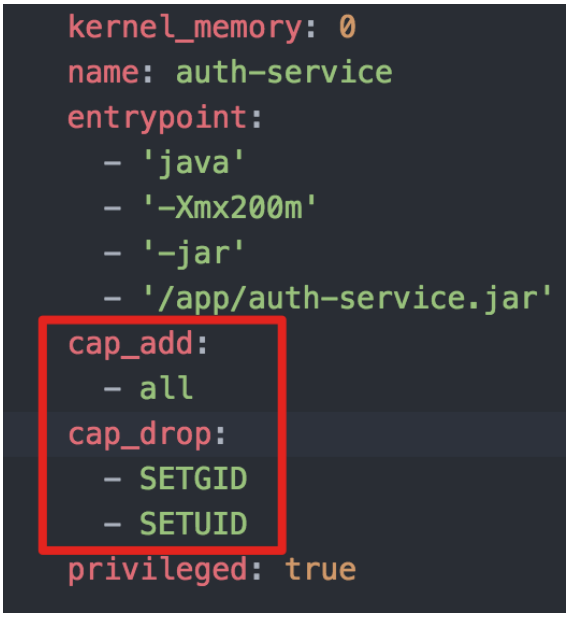
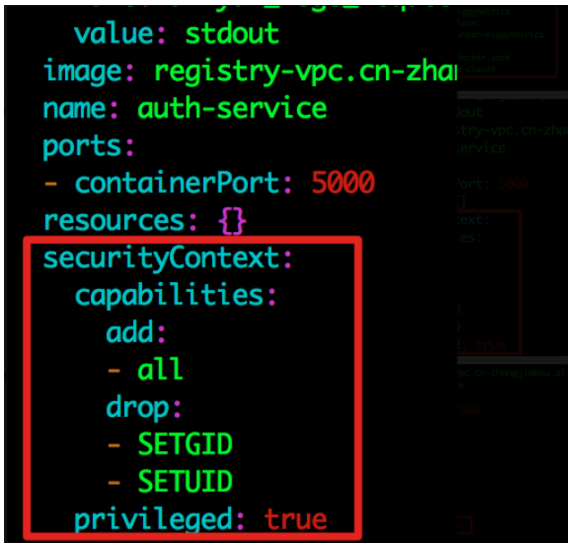
volumes

Swarm配置示例	Kubernetes配置示例
	

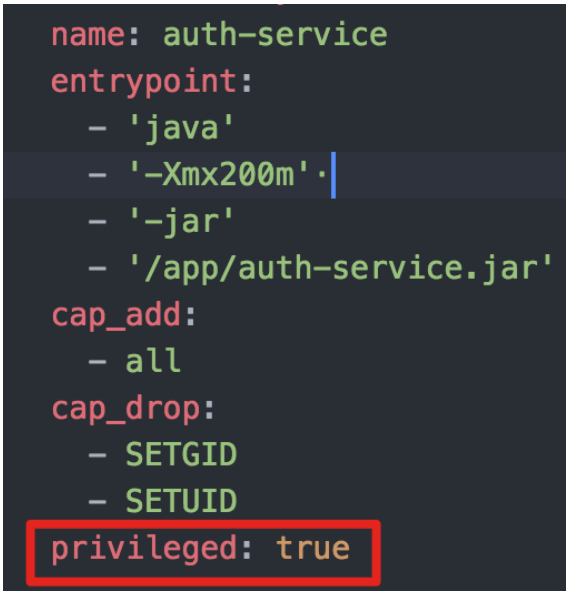
配置详情，请参见auth-service-deployment.yaml 。

Swarm配置示例	Kubernetes配置示例

cap_add/cap_drop

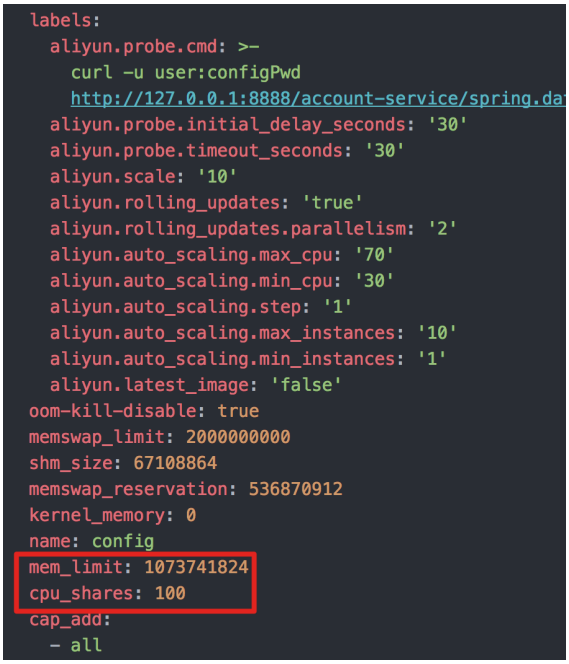
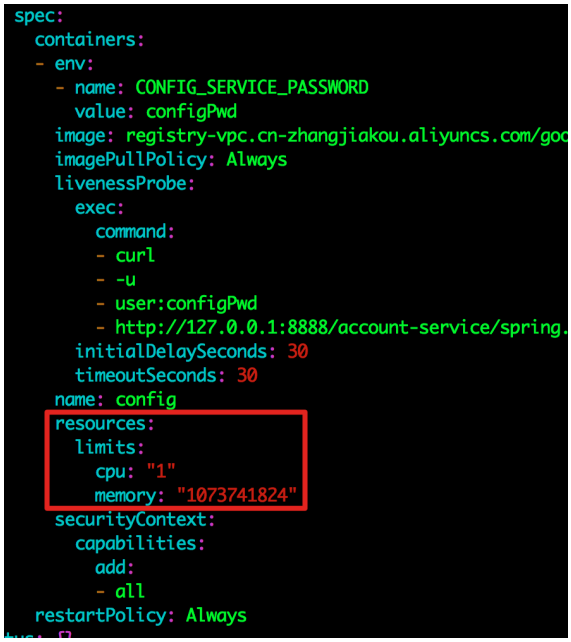
Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见 auth-service-deployment.yaml

privileged: true

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见 auth-service-deployment.yaml。

mem_limit

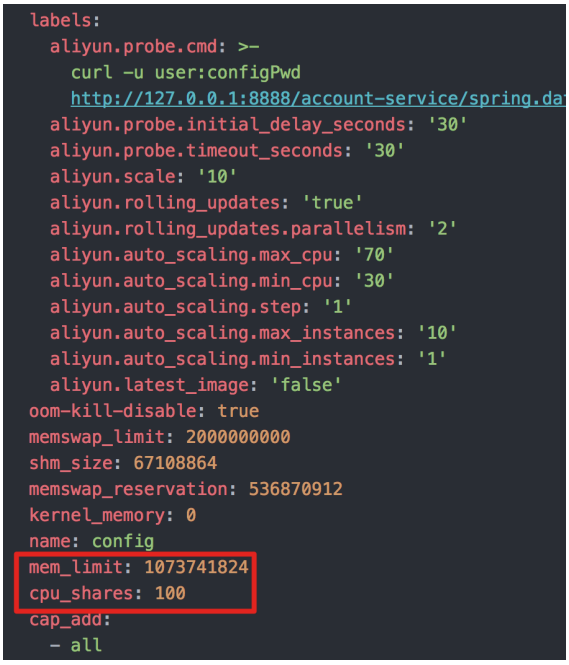
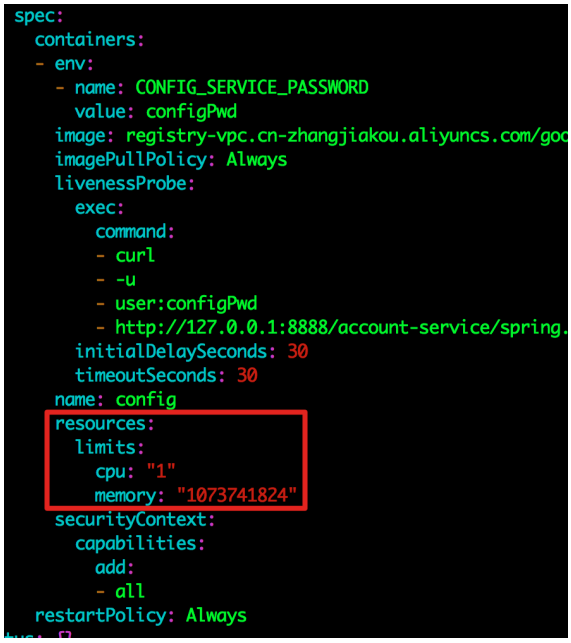
Swarm配置示例	Kubernetes配置示例

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见config-deployment.yaml 。

```
labels:
  aliyun.probe.cmd: >-
    curl -u user:configPwd
    http://127.0.0.1:8888/account-service/spring.da
  aliyun.probe.initial_delay_seconds: '30'
  aliyun.probe.timeout_seconds: '30'
  aliyun.scale: '10'
  aliyun.rolling_updates: 'true'
  aliyun.rolling_updates.parallelism: '2'
  aliyun.auto_scaling.max_cpu: '70'
  aliyun.auto_scaling.min_cpu: '30'
  aliyun.auto_scaling.step: '1'
  aliyun.auto_scaling.max_instances: '10'
  aliyun.auto_scaling.min_instances: '1'
  aliyun.latest_image: 'false'
oom-kill-disable: true
memswap_limit: 2000000000
shm_size: 67108864
memswap_reservation: 536870912
kernel_memory: 0
name: config
mem_limit: 1073741824
cpu_shares: 100
cap_add:
  - all
```

```
spec:
  containers:
  - env:
    - name: CONFIG_SERVICE_PASSWORD
      value: configPwd
    image: registry-vpc.cn-zhangjiakou.aliyuncs.com/goo
    imagePullPolicy: Always
    livenessProbe:
      exec:
        command:
        - curl
        - -u
        - user:configPwd
        - http://127.0.0.1:8888/account-service/spring.
      initialDelaySeconds: 30
      timeoutSeconds: 30
    name: config
    resources:
      limits:
        cpu: "1"
        memory: "1073741824"
    securityContext:
      capabilities:
        add:
        - all
  restartPolicy: Always
```

cpu_shares

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见config-deployment.yaml 。

13.6.2. 发布类标签

本文介绍容器服务Swarm集群相关的发布类标签。

Swarm标签名	Swarm标签含义	对应 Kubernetes配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
restart	容器重启策略。	restartPolicy标签。	略	略	kompose自动迁移。
aliyun.latest_image	是否拉取最新镜像。	imagePullPolicy标签。	aliyun.latest_image: true	请参见aliyun.latest_image的Kubernetes配置示例。	手动迁移：kompose转换后，修改该应用对应资源文件 *-deployment.yaml，添加imagePullPolicy配置。如果不填写，默认策略是always。
depends_on	设置服务的依赖关系。	利用init container，liveness/readiness探针等技术实现服务健康检查、依赖检查等功能。详细参见解决服务依赖。	略	略	依赖业务实现。

Swarm标签名	Swarm标签含义	对应 Kubernetes配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
aliyun.depends	设置服务的依赖关系。	利用init container，liveness/readiness探针等技术实现服务健康检查、依赖检查等功能。详细参见解决服务依赖。	略	略	依赖业务实现。
environment:affinity:service!=db	设置服务的部署约束条件，参见服务部署约束（affinity:service）。	在 Kubernetes中可以通过affinity.nodeAffinity affinity.podAffinity affinity.podAntiAffinity标签做Node或Pod亲和性配置，参见Affinity and anti-affinity。	略	略	依赖业务实现。
environment:constraint:group==1	表示在所有带有*group:1*标签的节点上部署。	在Kubernetes可以通过nodeSelector来做Node过滤，参见nodeSelector。	请参见environment:constraint:group==1的Swarm配置示例。	请参见environment:constraint:group==1的Kubernetes配置示例。	手动迁移：kompose转换后，修改该应用对应资源文件 *-deployment.yaml*，添加nodeSelector配置。
aliyun.global	设置该服务为全局服务。	通过DaemonSet类型应用支持，创建方式，参见DaemonSet。	请参见aliyun.global的Swarm配置示例。	aliyun.global的Kubernetes配置示例。	kompose自动迁移+手动修复 手动修复：kompose转换后，需要删除资源文件中的status章节（用来记录中间状态），才能部署成功。
aliyun.scale	设置该服务的容器数量，横向扩展服务。	replicas标签。	aliyun.scale: '10'	replicas: 10	kompose自动迁移。
aliyun.rolling_updates	是否开启服务滚动更新。	strategy.type.RollingUpdate标签，参见Strategy。	请参见aliyun.rolling_updates和aliyun.rolling_updates.parallelism的Swarm配置示例。	请参见aliyun.rolling_updates和aliyun.rolling_updates.parallelism的Kubernetes配置示例。	手动迁移：kompose转换后，需要手动配置滚动更新策略。
aliyun.rolling_updates.parallelism	每次并行更新的容器数量。	maxUnavailable标签，参见Strategy。			手动迁移：kompose转换后，需要手动配置滚动更新策略。

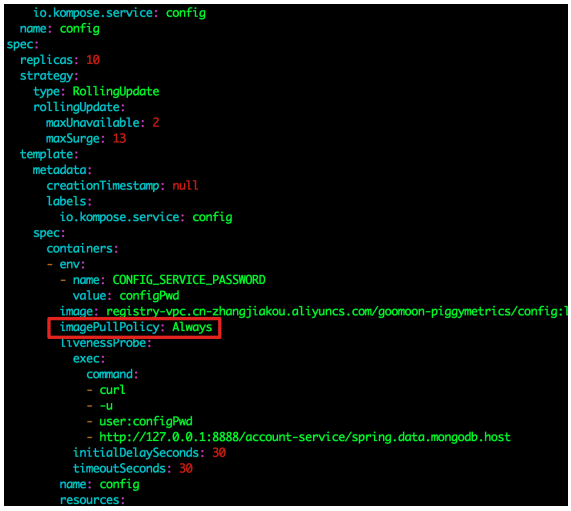
Swarm标签名	Swarm标签含义	对应 Kubernetes配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
aliyun.auto_scaling.*	容器自动伸缩，根据服务的容器资源占用情况自动调整容器数量。参见容器自动伸缩。	无法简单做标签映射，阿里云容器服务支持在控制台界面上快速创建支持HPA的应用，实现容器资源的弹性伸缩。参见容器水平伸缩（HPA）。	请参见aliyun.auto_scaling.*的Swarm配置示例。	请参见aliyun.auto_scaling.*的Kubernetes配置示例。	手动迁移：部署Kubernetes资源文件后，在控制台配置。
aliyun.probe.cmd	健康检查执行的检查 Shell 命令，参见probe。	livenessProbe标签。	请参见aliyun.probe.cmd、aliyun.probe.	请参见aliyun.probe.cmd、aliyun.probe.	kompose自动迁移。

Swarm标签名	Swarm标签含义	对应 Kubernetes配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
aliyun.probe.initial_delay_seconds	在容器启动后延迟几秒开始健康检查，参见probe。	livenessProbe 标签 。 **说明** • 在Swarm里面健康检查只是用来表示容器状态，并不会影响其对外提供服务。 • 但在Kubernetes中，如果livenessProbe检查失败，则Kubernetes会强制重启容器，所以延迟该标签的值一定要设置合理，否则会导致Kubernetes不断重启容器，导致服务不可用。	aliyun.probe.initial_delay_seconds和aliyun.probe.timeout_seconds的Swarm配置示例。	aliyun.probe.initial_delay_seconds和aliyun.probe.timeout_seconds的Kubernetes配置示例。	kompose自动迁移。
aliyun.probe.timeout_seconds	健康检查的超时时间，参见probe。	livenessProbe标签			kompose自动迁移。
aliyun.probe.url	HTTP、TCP 请求的 URL，参见probe。	livenessProbe标签	请参见aliyun.probe.url的Swarm配置示例。	请参见aliyun.probe.url的Kubernetes配置示例。	kompose自动迁移。

Swarm标签名	Swarm标签含义	对应 Kubernetes配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
extra_hosts	添加容器Host 文件绑定配置。	对应 Kubernetes的hostAliases配置，参见Adding Additional Entries with HostAliases。	请参见extra_hosts的Swarm配置示例。	请参见extra_hosts的Kubernetes配置示例。	手动迁移：kompose转换后，修改该应用对应资源文件 *-*deployment.yaml*，添加hostAliases配置。
entrypoint	覆盖默认入口点。	command标签	请参见entrypoint的Swarm配置示例。	请参见entrypoint的Kubernetes配置示例。	kompose自动迁移。
command	重写默认命令。	args标签	请参见command的Swarm配置示例。	请参见command的Kubernetes配置示例。	kompose自动迁移。

aliyun.latest_image

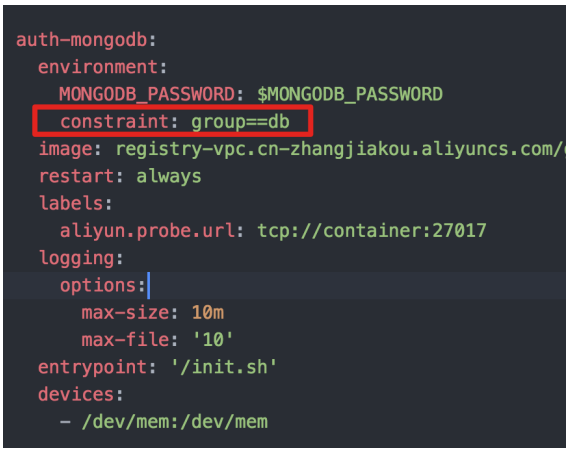
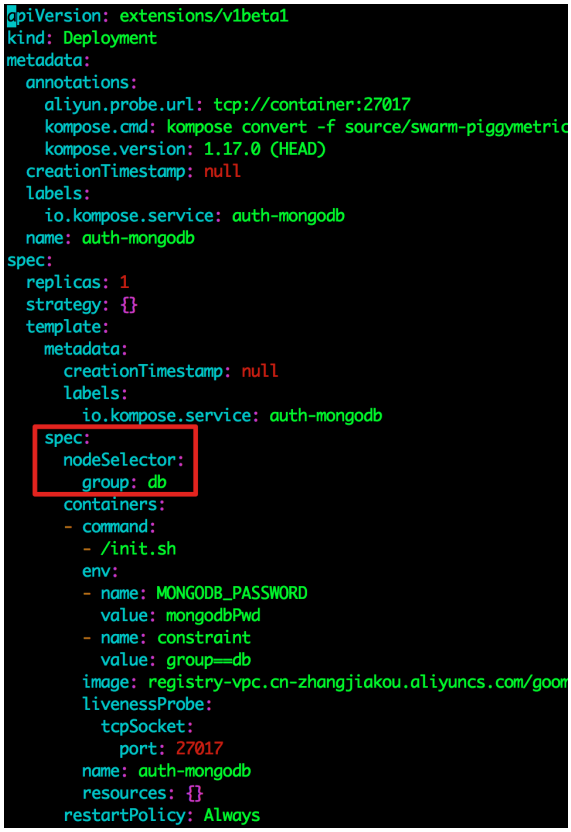
Swarm配置示例	Kubernetes配置示例

Swarm配置示例	Kubernetes配置示例
aliyun.latest_image: true	 详细配置，请参见config-deployment.yaml 。

```
    io.kompose.service: config
  name: config
spec:
  replicas: 10
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxUnavailable: 2
      maxSurge: 13
  template:
    metadata:
      creationTimestamp: null
      labels:
        io.kompose.service: config
    spec:
      containers:
      - env:
        - name: CONFIG_SERVICE_PASSWORD
          value: configPwd
        image: registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/config:1
        imagePullPolicy: Always
        livenessProbe:
          exec:
            command:
            - curl
            - -u
            - user:configPwd
            - http://127.0.0.1:8888/account-service/spring.data.mongodb.host
          initialDelaySeconds: 30
          timeoutSeconds: 30
        name: config
        resources:
```

environment: constraint:group==1

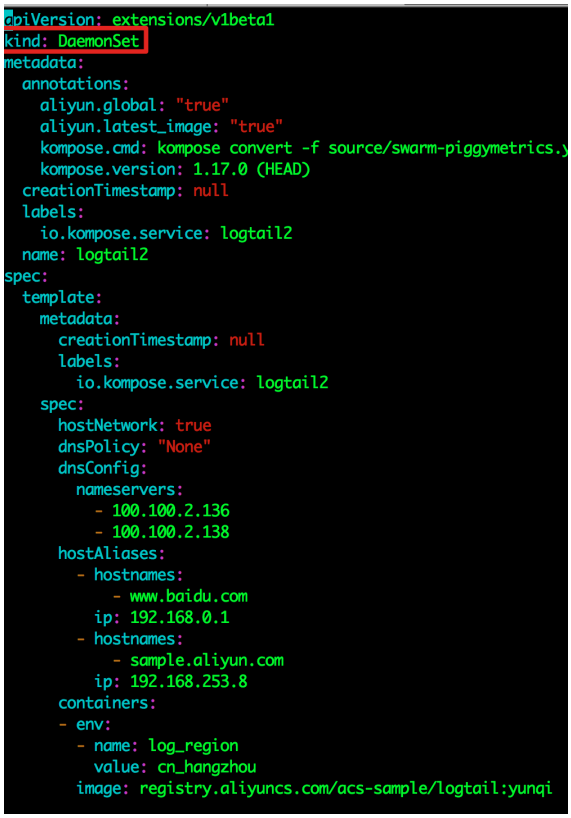
Swarm配置示例	Kubernetes配置示例

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见auth-mongodb-deployment.yaml 。

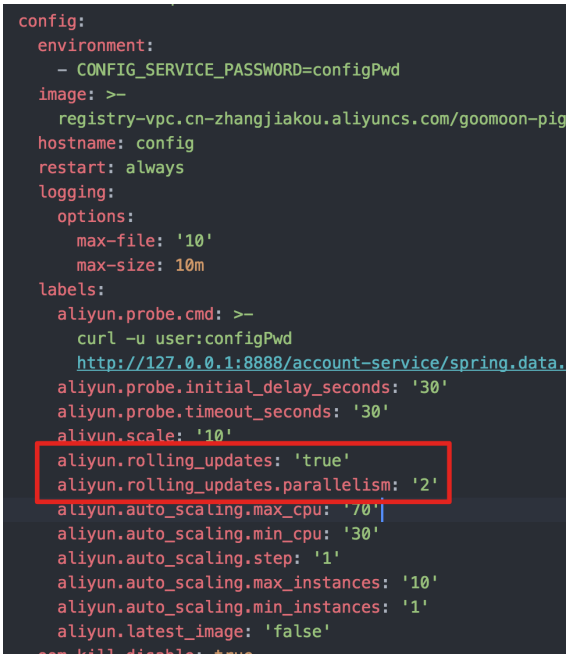
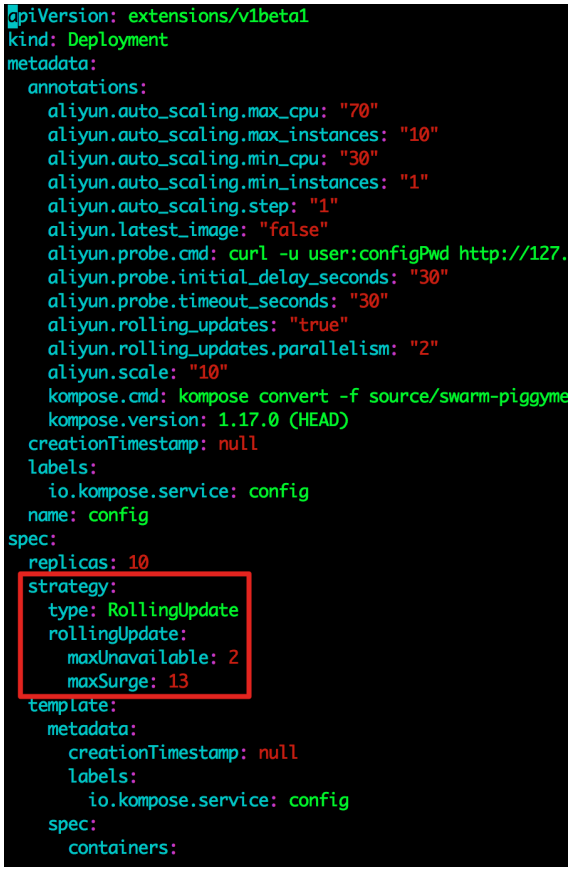
```
auth-mongodb:
  environment:
    MONGODB_PASSWORD: $MONGODB_PASSWORD
    constraint: group==db
  image: registry-vpc.cn-zhangjiakou.aliyuncs.com/
  restart: always
  labels:
    aliyun.probe.url: tcp://container:27017
  logging:
    options:
      max-size: 10m
      max-file: '10'
  entrypoint: '/init.sh'
  devices:
    - /dev/mem:/dev/mem
```

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
    aliyun.probe.url: tcp://container:27017
    kompose.cmd: kompose convert -f source/swarm-piggymetric
    kompose.version: 1.17.0 (HEAD)
  creationTimestamp: null
  labels:
    io.kompose.service: auth-mongodb
  name: auth-mongodb
spec:
  replicas: 1
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        io.kompose.service: auth-mongodb
    spec:
      nodeSelector:
        group: db
      containers:
      - command:
        - /init.sh
        env:
        - name: MONGODB_PASSWORD
          value: mongodbPwd
        - name: constraint
          value: group==db
        image: registry-vpc.cn-zhangjiakou.aliyuncs.com/goom
        livenessProbe:
          tcpSocket:
            port: 27017
        name: auth-mongodb
        resources: {}
      restartPolicy: Always
```

aliyun.global

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见logtail2-daemonset.yaml 。

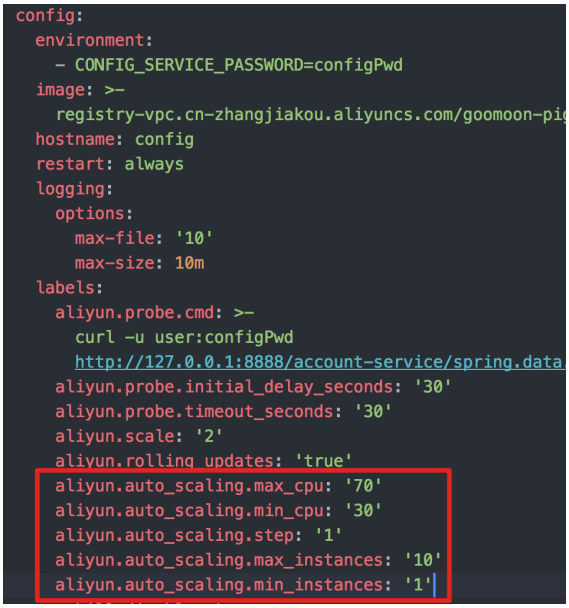
aliyun.rolling_updates和aliyun.rolling_updates.parallelism

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见config-deployment.yaml 。

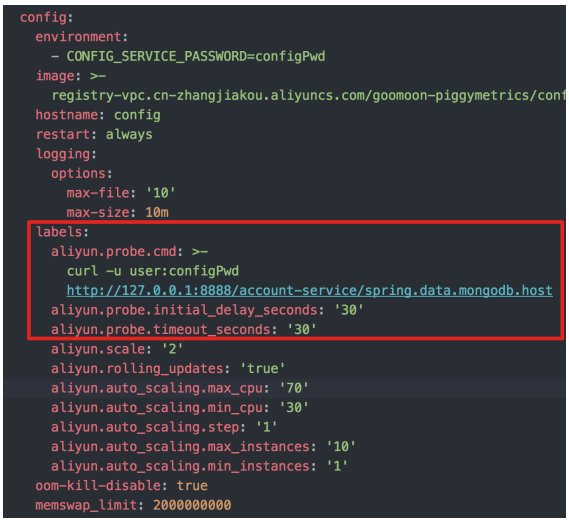
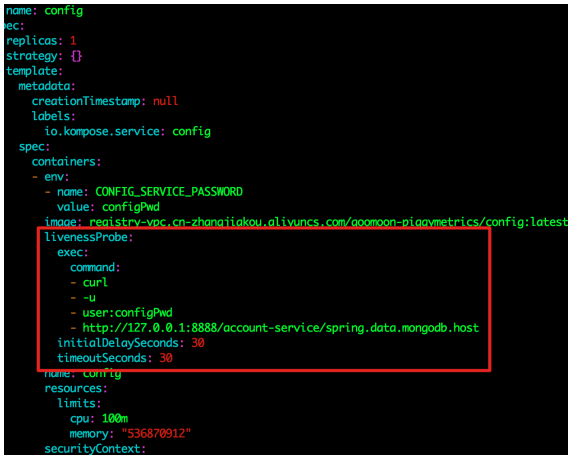
```
config:
  environment:
    - CONFIG_SERVICE_PASSWORD=configPwd
  image: >-
    registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-pig
  hostname: config
  restart: always
  logging:
    options:
      max-file: '10'
      max-size: 10m
  labels:
    aliyun.probe.cmd: >-
      curl -u user:configPwd
      http://127.0.0.1:8888/account-service/spring.data.
    aliyun.probe.initial_delay_seconds: '30'
    aliyun.probe.timeout_seconds: '30'
    aliyun.scale: '10'
    aliyun.rolling_updates: 'true'
    aliyun.rolling_updates.parallelism: '2'
    aliyun.auto_scaling.max_cpu: '70'
    aliyun.auto_scaling.min_cpu: '30'
    aliyun.auto_scaling.step: '1'
    aliyun.auto_scaling.max_instances: '10'
    aliyun.auto_scaling.min_instances: '1'
    aliyun.latest_image: 'false'
```

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
    aliyun.auto_scaling.max_cpu: "70"
    aliyun.auto_scaling.max_instances: "10"
    aliyun.auto_scaling.min_cpu: "30"
    aliyun.auto_scaling.min_instances: "1"
    aliyun.auto_scaling.step: "1"
    aliyun.latest_image: "false"
    aliyun.probe.cmd: curl -u user:configPwd http://127.
    aliyun.probe.initial_delay_seconds: "30"
    aliyun.probe.timeout_seconds: "30"
    aliyun.rolling_updates: "true"
    aliyun.rolling_updates.parallelism: "2"
    aliyun.scale: "10"
    kompose.cmd: kompose convert -f source/swarm-piggyme
    kompose.version: 1.17.0 (HEAD)
  creationTimestamp: null
  labels:
    io.kompose.service: config
  name: config
spec:
  replicas: 10
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxUnavailable: 2
      maxSurge: 13
  template:
    metadata:
      creationTimestamp: null
      labels:
        io.kompose.service: config
    spec:
      containers:
```

aliyun.auto_scaling.*

Swarm配置示例	Kubernetes配置示例
	

aliyun.probe.cmd、aliyun.probe.initial_delay_seconds和aliyun.probe.timeout_seconds

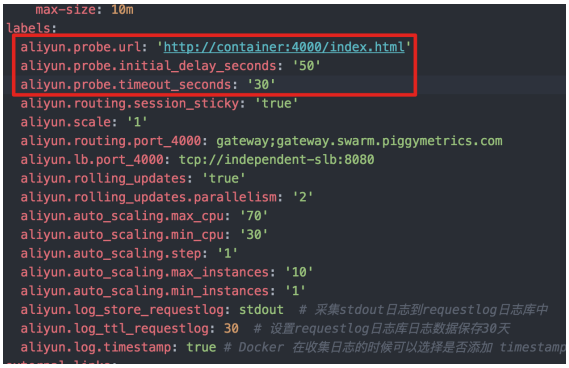
Swarm配置示例	Kubernetes配置示例
 aliyun.latest_image: true	 详细配置，请参见config-deployment.yaml 。

```
config:
  environment:
    - CONFIG_SERVICE_PASSWORD=configPwd
  image: >-
    registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/conf
  hostname: config
  restart: always
  logging:
    options:
      max-file: '10'
      max-size: 10m
  labels:
    aliyun.probe.cmd: >-
      curl -u user:configPwd
      http://127.0.0.1:8888/account-service/spring.data.mongodb.host
    aliyun.probe.initial_delay_seconds: '30'
    aliyun.probe.timeout_seconds: '30'
    aliyun.scale: '2'
    aliyun.rolling_updates: 'true'
    aliyun.auto_scaling.max_cpu: '70'
    aliyun.auto_scaling.min_cpu: '30'
    aliyun.auto_scaling.step: '1'
    aliyun.auto_scaling.max_instances: '10'
    aliyun.auto_scaling.min_instances: '1'
  oom-kill-disable: true
  memswap_limit: 2000000000
```

```
name: config
ec:
replicas: 1
strategy: {}
template:
  metadata:
    creationTimestamp: null
    labels:
      io.kompose.service: config
  spec:
    containers:
    - env:
      - name: CONFIG_SERVICE_PASSWORD
        value: configPwd
      image: registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/config:latest
      livenessProbe:
        exec:
          command:
          - curl
          - -u
          - user:configPwd
          - http://127.0.0.1:8888/account-service/spring.data.mongodb.host
        initialDelaySeconds: 30
        timeoutSeconds: 30
      name: config
      resources:
        limits:
          cpu: 100m
          memory: "536870912"
      securityContext:
```

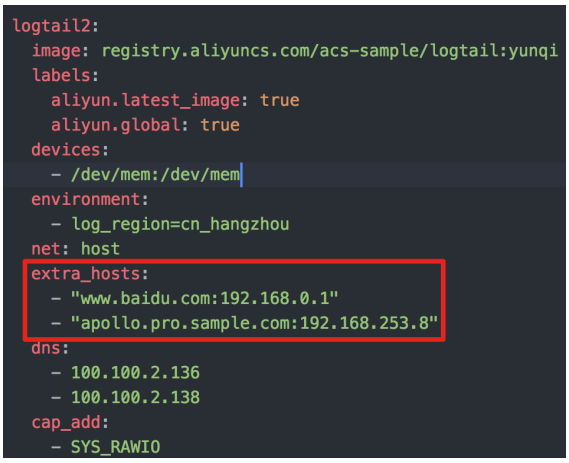
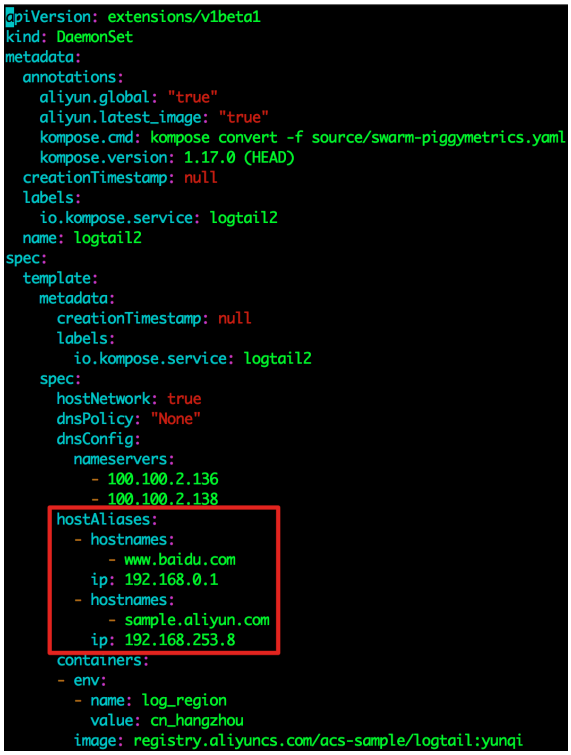
aliyun.probe.url

Swarm配置示例	Kubernetes配置示例

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见gateway-deployment.yaml 。

extra_hosts

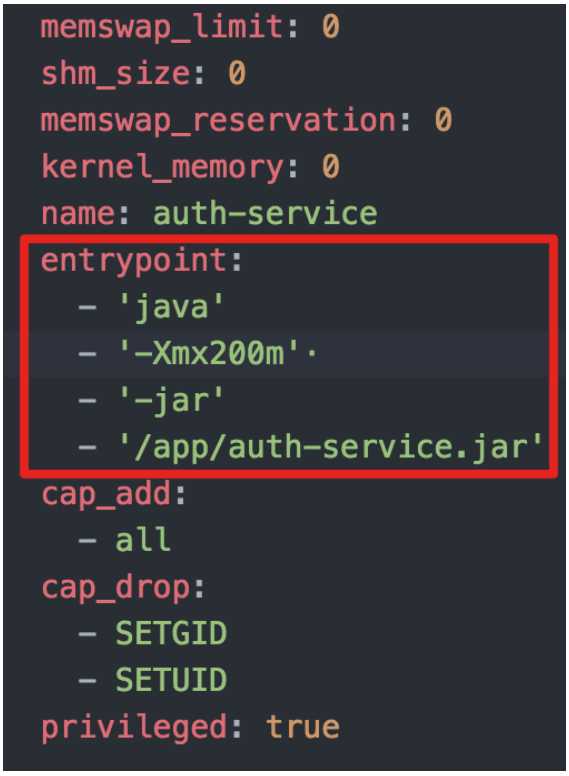
Swarm配置示例	Kubernetes配置示例

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见logtail2-daemonset.yaml 。

```
logtail2:
  image: registry.aliyuncs.com/acs-sample/logtail:yunqi
  labels:
    aliyun.latest_image: true
    aliyun.global: true
  devices:
    - /dev/mem:/dev/mem
  environment:
    - log_region=cn_hangzhou
  net: host
  extra_hosts:
    - "www.baidu.com:192.168.0.1"
    - "apollo.pro.sample.com:192.168.253.8"
  dns:
    - 100.100.2.136
    - 100.100.2.138
  cap_add:
    - SYS_RAWIO
```

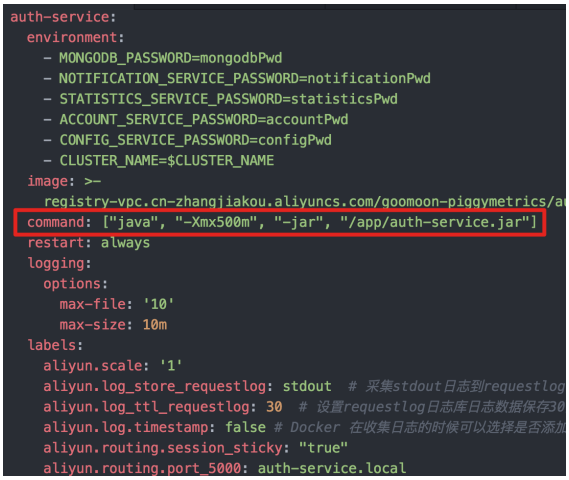
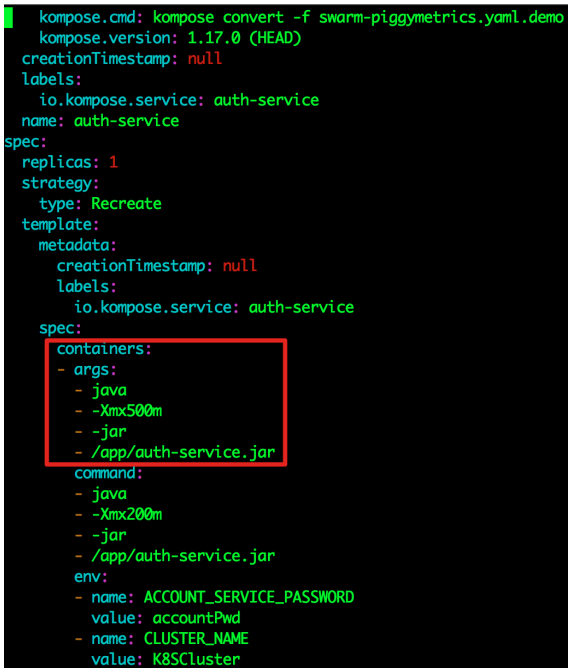
```
apiVersion: extensions/v1beta1
kind: DaemonSet
metadata:
  annotations:
    aliyun.global: "true"
    aliyun.latest_image: "true"
    kompose.cmd: kompose convert -f source/swarm-piggymetrics.yaml
    kompose.version: 1.17.0 (HEAD)
  creationTimestamp: null
  labels:
    io.kompose.service: logtail2
  name: logtail2
spec:
  template:
    metadata:
      creationTimestamp: null
      labels:
        io.kompose.service: logtail2
    spec:
      hostNetwork: true
      dnsPolicy: "None"
      dnsConfig:
        nameservers:
          - 100.100.2.136
          - 100.100.2.138
      hostAliases:
        - hostnames:
            - www.baidu.com
          ip: 192.168.0.1
        - hostnames:
            - sample.aliyun.com
          ip: 192.168.253.8
      containers:
      - env:
        - name: log_region
          value: cn_hangzhou
        image: registry.aliyuncs.com/acs-sample/logtail:yunqi
```

entrypoint

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见 auth-service-deployment.yaml。

command

Swarm配置示例	Kubernetes配置示例

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见 auth-service-deployment.yaml。

13.6.3. 网络配置类标签

本文介绍容器服务Swarm集群相关的网络配置类标签。

Swarm标签名	Swarm标签含义	对应 Kubernetes 配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
net	V1 版compose文件格式，设置网络模式，同docker命令行参数--net；参见net。	部分值，可以直接转换，部分值在Kubernetes不支持定制，其中：net: host可直接转换为hostNetwork: true，其余值在Kubernetes里面不支持。参见Host namespaces。	请参见net的Swarm配置示例。	请参见net的Kubernetes配置示例。	手动迁移：kompose转换后，需要手动修改不支持的资源配置。

Swarm标签名	Swarm标签含义	对应 Kubernetes 配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移

aliyun.routing.port_	• Swarm简单路由，用于设置该服务的访问域名并支持多域名设置，参见routing。
	• Swarm简单路由，当设置了.local后缀域名时，仅用于集群内服务间路由和负载均衡，参见集群内服务间路由和负载均衡。
		对应Kubernetes ingress，但kompose工具支持部分转换，但转换过程只支持转换第一个域名，后续域名规则需人工添加，且针对测试域名需要人工替换。参见Nginx Ingress高级用法。	请参见aliyun.routing.port_的Swarm配置示例。	请参见aliyun.routing.port_的Kubernetes配置示例。	kompose自动迁移或手动修复。
					手动修复：
					• 普通域名：kompose转换后，需要手动修改Kubernetes资源文件，找到Kubernetes对应ingress资源文件*-ingress.yaml。
					  ◦ 将其中的host: your-cluster-id.alicontainer.com修改成对应Kubernetes 集群的测试域名，其中，k8s-piggymetrics-cluster示例的测试域名为*.c7f537c92438f415b943e7c2f8ca30b3b.cn-zhangjiakou.alicontainer.com；将其替换到各个ingress文件的.your-cluster-id.alicontainer.com字符串。
					  ◦ 针对分号后面的第二个域名，需要自己在ingress文件里面添加rule规则。
					• .local域名：
					  ◦ 部署Kubernetes文件后完成。
					  ◦ 手动到控制台创建Service。如果是.local域名，在Swarm里面只有集群内可以访问且不用带端口，而Kubernetes Service名只支持小写和中划线-，不支持.local这类带 . 字符的服务名。在迁移Kubernetes时，只能修改应用代码适配。

Swarm标签名	Swarm标签含义	对应 Kubernetes 配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
aliyun.routing.session_sticky	设置routing在做请求路由的时候，是否保持session sticky，即会话保持。 **说明** 此标签必须和 `routing` 配合使用，单独使用无效。	Kubernetes不支持，迁移忽略。	略	略	无法迁移。
hostname	可用作跨Node容器间访问（基于DNS+LB），常用于跨服务访问。	转换成service的方式，实现跨容器访问；参见管理服务。	请参见hostname的Swarm配置示例。	请参见hostname的Kubernetes配置示例。	手动迁移： 1. 先部署Kubernetes资源文件。 2. 然后从控制台界面创建Service（cluster IP）。由于在Swarm编排文件里面不知道容器对应端口，因此无法通过 kompose 工具自动转换。
links	1. 通过服务名或别名访问跨主机容器组。 2. 跟depends on类似，决定启动顺序。	1. 基于服务名和别名，分别创建service 的方式，实现跨容器访问；参见管理服务。 2. 启动顺序依赖，需要根据业务特点处理。	请参见links的Swarm配置示例。	请参见links的Kubernetes配置示例。	

Swarm标签名	Swarm标签含义	对应 Kubernetes 配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
external_links	将两个服务链接起来，使之可以通信；其相对links标签来说，支持链接到其它编排文件或其它方式创建的容器。	Kubernetes只支持通过service方式来实现跨容器访问；且 service对应的后端容器必须由Kubernetes管理。 迁移方案： 1. 被external_links链接的容器通过Kubernetes集群创建和管理。 2. 基于服务名和别名，分别创建service 的方式，实现跨容器访问。参见管理服务。	请参见external_links的Swarm配置示例。	请参见external_links的Kubernetes配置示例。	**说明** 由于Kubernetes Service名只支持小写和中划线-。如果hostname、links、external_links等带有其它字符如.local这类带 . 字符的，迁移Kubernetes时，只能修改应用代码适配。

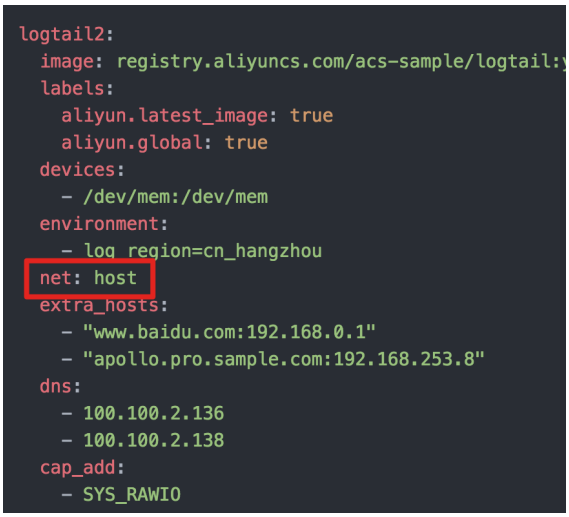
`external:`
`host: $RDS_URL`
`ports:`
`- 3306`
	设置该服务直接链接到外部地址。
	• host：设置链接的域名。
	• ports：设置链接的端口。
		对应Kubernetes的headless service，参见Headless services。	请参见external: ***的Swarm配置示例。	请参见external: ***的Kubernetes配置示例。	手动迁移：
					1. kompose转换前删除配置，避免中断转换流程。
					2. kompose 转换后，手动编写headless service（ExternalName类型）资源文件。
					3. 跟其它资源文件，一起通过kubectl部署。

Swarm标签名	Swarm标签含义	对应 Kubernetes 配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
aliyun.lb.port_*	通过自定义阿里云负载均衡 na映射的方式来暴露服务端口到公网或者到内网。参见负载均衡路由。	通过LoadBalancer（类似于Swarm集群的负载均衡）访问服务。参见通过Annotation配置负载均衡。	请参见aliyun.lb.port_*的Swarm配置示例。	请参见aliyun.lb.port_*的Kubernetes配置示例。	手动迁移： 部署后，通过Kubernetes LoadBalancer（类似于Swarm集群的负载均衡）访问服务。参见通过Annotation配置负载均衡。 **说明** • 这里仅做示例，需要客户结合业务需要按Kubernetes文档操作。 • 一经挂载到SLB就有可能触发生产流量进入，建议先挂载到测试端口完成测试，再挂载到正式端口。

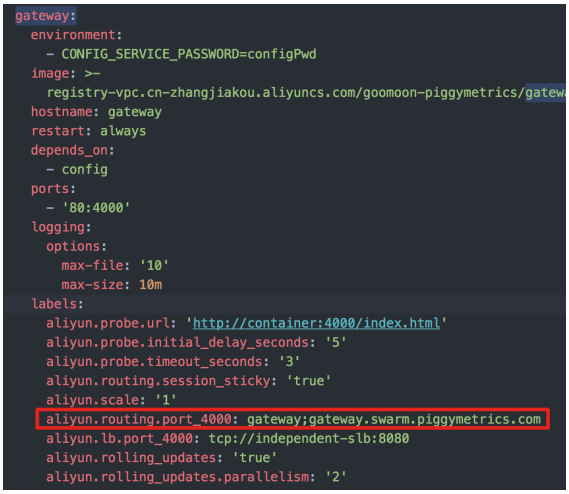
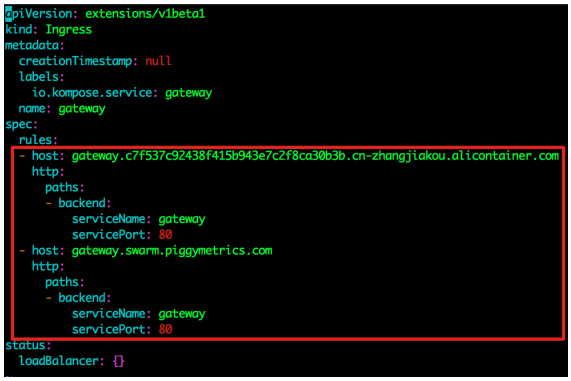
Swarm标签名	Swarm标签含义	对应 Kubernetes 配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
dns	指定容器上游DNS服务器	对应Kubernetes的Pod级别的DNS配置，详细参见Pod's DNS Config。 **说明** 如果指定了DNS，则对应的Service名称的解析可能会失效。	请参见DNS的Swarm配置示例。	请参见DNS的Kubernetes配置示例。	手动迁移：kompose转换后，修改该应用对应资源文件 *-deployment.yaml，添加dnsConfig配置。

net

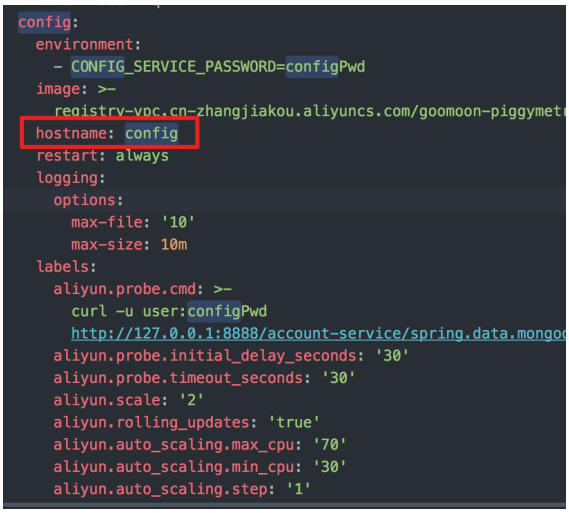
Swarm配置示例	Kubernetes配置示例

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见logtail2-daemonset.yaml 。

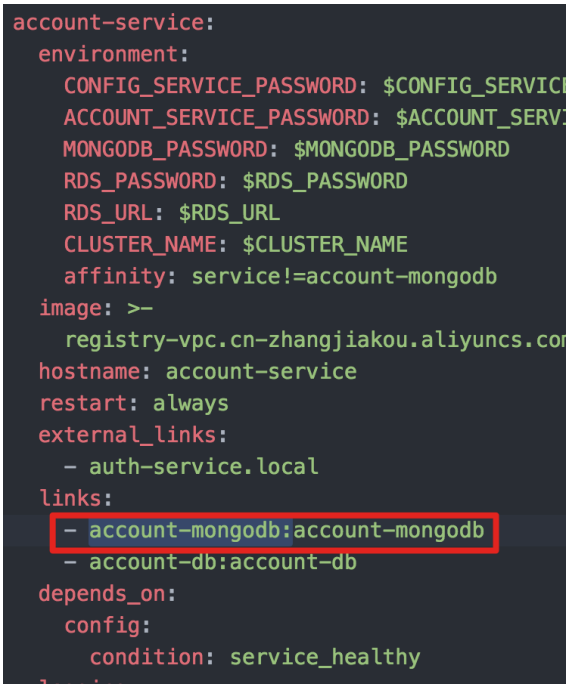
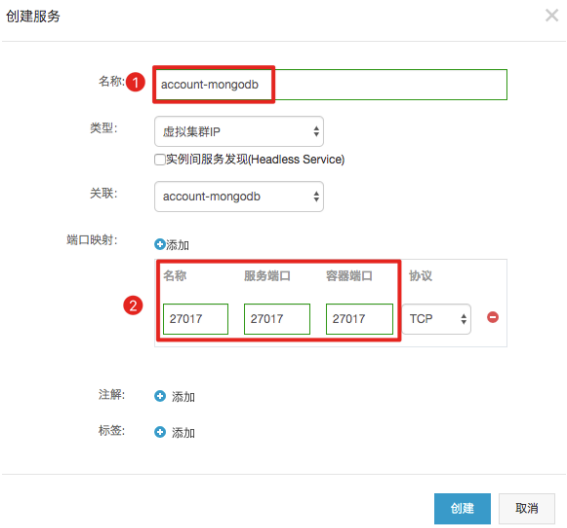
aliyun.routing.port_

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见 gateway-ingress.yaml。

hostname

Swarm配置示例	Kubernetes配置示例
	 **说明** 通过控制台创建一个同名（标注1）的service用于服务访问，服务端口为容器端口。

links

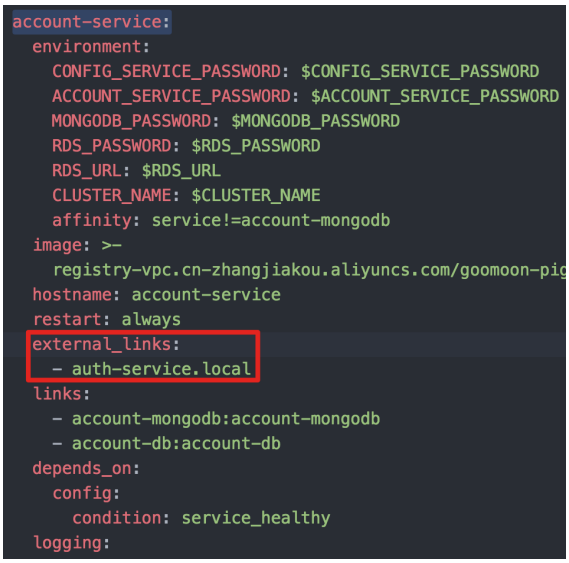
Swarm配置示例	Kubernetes配置示例
	

```
account-service:
  environment:
    CONFIG_SERVICE_PASSWORD: $CONFIG_SERVICE
    ACCOUNT_SERVICE_PASSWORD: $ACCOUNT_SERVI
    MONGODB_PASSWORD: $MONGODB_PASSWORD
    RDS_PASSWORD: $RDS_PASSWORD
    RDS_URL: $RDS_URL
    CLUSTER_NAME: $CLUSTER_NAME
    affinity: service!=account-mongodb
  image: >-
    registry-vpc.cn-zhangjiakou.aliyuncs.com
  hostname: account-service
  restart: always
  external_links:
    - auth-service.local
  links:
    - account-mongodb:account-mongodb
    - account-db:account-db
  depends_on:
    config:
      condition: service_healthy
```


说明 通过控制台分别创建一个同名的service用于服务访问。

external_links

Swarm配置示例	Kubernetes配置示例

Swarm配置示例	Kubernetes配置示例
	

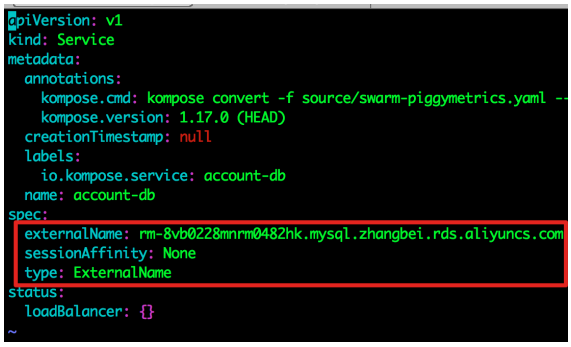
```
account-service:
  environment:
    CONFIG_SERVICE_PASSWORD: $CONFIG_SERVICE_PASSWORD
    ACCOUNT_SERVICE_PASSWORD: $ACCOUNT_SERVICE_PASSWORD
    MONGODB_PASSWORD: $MONGODB_PASSWORD
    RDS_PASSWORD: $RDS_PASSWORD
    RDS_URL: $RDS_URL
    CLUSTER_NAME: $CLUSTER_NAME
    affinity: service!=account-mongodb
  image: >-
    registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-pig
  hostname: account-service
  restart: always
  external_links:
    - auth-service.local
  links:
    - account-mongodb:account-mongodb
    - account-db:account-db
  depends_on:
    config:
      condition: service_healthy
  logging:
```


> **说明** 通过控制台创建一个同名的service用于服务访问。

external: ***

external: ***标签的全部显示如下。

```
external:
     host: $RDS_URL
     ports:
       - 3306
```

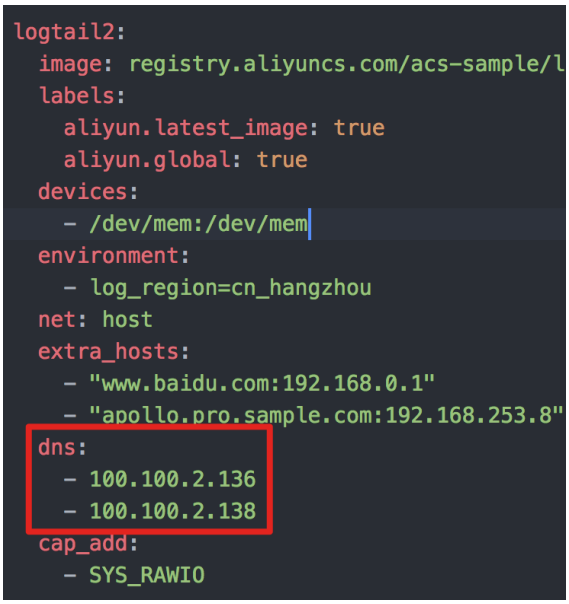
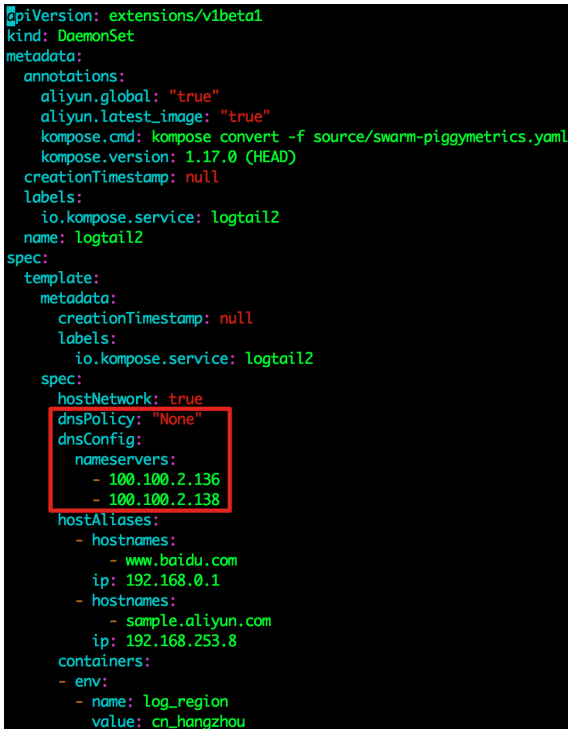
Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见account-db-service.yaml。

aliyun.lb.port_*

Swarm配置示例	Kubernetes配置示例

Swarm配置示例	Kubernetes配置示例
	

dns

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见logtail2-daemonset.yaml 。

13.6.4. 日志配置类标签

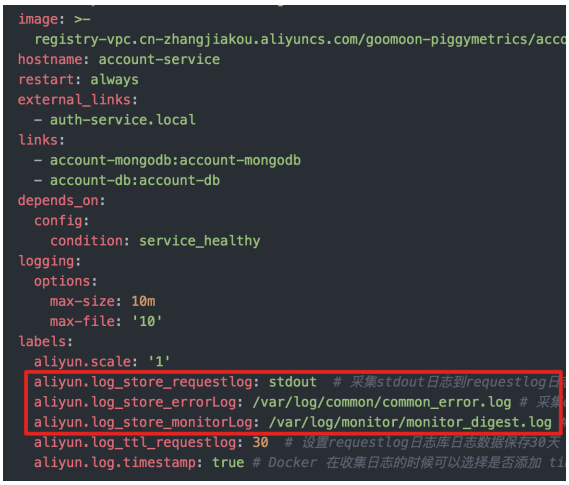
本文介绍容器服务Swarm集群相关的日志配置类标签。

Swarm标签名	Swarm标签含义	对应Kubernetes配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
aliyun.log_store_<logstore_name>	将容器日志保存到阿里云日志服务。	aliyun_logs环境变量，参见通过日志服务采集Kubernetes容器日志。	请参见aliyun.log_store_<logstore_name>的Swarm配置示例。	请参见aliyun.log_store_<logstore_name>的Kubernetes示例。	kompose自动迁移。

Swarm标签名	Swarm标签含义	对应Kubernetes配置方案	Swarm配置示例	Kubernetes配置示例	如何迁移
aliyun.log_ttl_<logstore_name>	阿里云日志服务定制标签，用来设置日志库初始日志保存时间，单位为天，有效值为 1~365 天，不配置则默认初始化为 2 天。参见集成日志服务。 这里设置的是初始配置值，如果后期您需要修改日志保存时间，需要到日志服务控制台进行设置。	没有对应标签支持，需要自己到日志服务控制台进行设置。	aliyun.log_ttl_requestlog: '30'	请参见aliyun.log_ttl_<logstore_name>的Kubernetes配置示例。	手动迁移过程： 1. 先部署Kubernetes资源文件，并调试应用通过（这时会在日志服务生成对应的Logstore）。 2. 登录日志服务控制台修改Logstore配置，将数据保存时间调长。
aliyun.log.timestamp	Docker在收集日志的时候选择是否添加timestamp。参见集成日志服务。	Kubernetes不支持，迁移忽略。	略	略	无法迁移，一般输出都带这个默认的time。

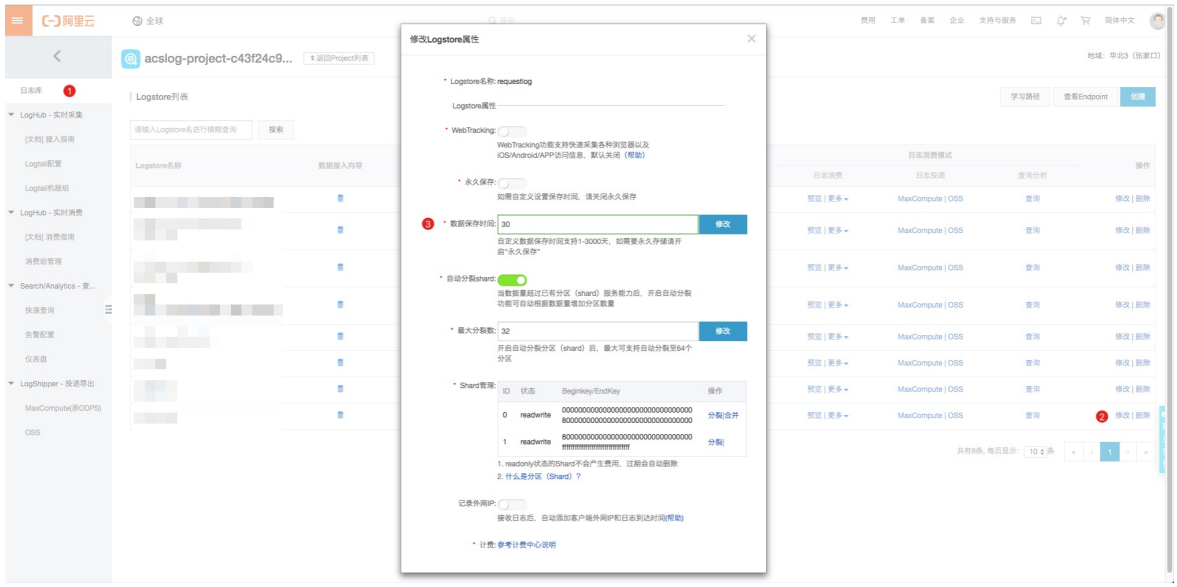
aliyun.log_store_<logstore_name>

Swarm配置示例	Kubernetes配置示例

Swarm配置示例	Kubernetes配置示例
	 详细配置，请参见account-service-deployment.yaml。

aliyun.log_ttl_<logstore_name>

Kubernetes配置示例



13.7. 附录：标签配置样例

account-db-service.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: account-db
  namespace: default
spec:
  type: ExternalName
  externalName: rm-8vb0228mnrm0482hk.mysql.zhangbei.rds.aliyuncs.com
```

account-service-deployment.yaml

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
    aliyun.log_store_errorLog: /var/log/common/common_error.log
    aliyun.log_store_monitorLog: /var/log/monitor/monitor_digest.log
    aliyun.log_store_requestlog: stdout
    aliyun.scale: "1"
    kompose.cmd: kompose convert -f source/swarm-piggymetrics.yaml --volumes PersistentVolu
meClaimOrHostPath
    kompose.version: 1.17.0 (HEAD)
  creationTimestamp: null
  labels:
    io.kompose.service: account-service
  name: account-service
spec:
```

```
  replicas: 1
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        io.kompose.service: account-service
    spec:
      containers:
      - env:
        - name: ACCOUNT_SERVICE_PASSWORD
          value: accountPwd
        - name: CLUSTER_NAME
          value: K8SCluster
        - name: CONFIG_SERVICE_PASSWORD
          value: configPwd
        - name: MONGODB_PASSWORD
          value: mongodbPwd
        - name: RDS_PASSWORD
          value: hello@1235
        - name: RDS_URL
          value: rm-8vb0228mnrm0482hk.mysql.zhangbei.rds.aliyuncs.com
        - name: affinity
          value: service!=account-mongodb
        - name: aliyun_logs_errorlog
          value: /var/log/common/common_error.log
        - name: aliyun_logs_monitorlog
          value: /var/log/monitor/monitor_digest.log
        - name: aliyun_logs_requestlog
          value: stdout
        image: registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/account-servic
e
        name: account-service
        resources: {}
        volumeMounts:
        - mountPath: /var/log/common
          name: volumn-sls-errorlog
        - mountPath: /var/log/monitor
          name: volumn-sls-monitorlog
      restartPolicy: Always
      volumes:
      - emptyDir: {}
        name: volumn-sls-errorlog
      - emptyDir: {}
        name: volumn-sls-monitorlog
status: {}
```

auth-mongodb-deployment.yaml

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
    aliyun.probe.url: tcp://container:27017
    kompose.cmd: kompose convert -f source/swarm-piggymetrics.yaml --volumes PersistentVolu
meClaimOrHostPath
    kompose.version: 1.17.0 (HEAD)
  creationTimestamp: null
  labels:
    io.kompose.service: auth-mongodb
  name: auth-mongodb
spec:
  replicas: 1
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        io.kompose.service: auth-mongodb
    spec:
      nodeSelector:
        group: db
      containers:
      - command:
        - /init.sh
        env:
        - name: MONGODB_PASSWORD
          value: mongodbPwd
        - name: constraint
          value: group==db
        image: registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/mongodb
        livenessProbe:
          tcpSocket:
            port: 27017
        name: auth-mongodb
        resources: {}
      restartPolicy: Always
status: {}
```

auth-service-deployment.yaml

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
    aliyun.latest_image: "true"
    aliyun.log_store_requestlog: stdout
    aliyun.routing.port_5000: auth-service.local
    aliyun.scale: "1"
    kompose.cmd: kompose convert -f source/swarm-piggymetrics.yaml --volumes PersistentVolu
meClaimOrHostPath
    kompose.version: 1.17.0 (HEAD)
```

```
    kompose.version: 1.17.0 (HEAD)
  creationTimestamp: null
  labels:
    io.kompose.service: auth-service
  name: auth-service
spec:
  replicas: 1
  strategy:
    type: Recreate
  template:
    metadata:
      creationTimestamp: null
      labels:
        io.kompose.service: auth-service
    spec:
      containers:
      - args:
        - java
        - -Xmx500m
        - -jar
        - /app/auth-service.jar
        command:
        - java
        - -Xmx200m
        - -jar
        - /app/auth-service.jar
        env:
        - name: ACCOUNT_SERVICE_PASSWORD
          value: accountPwd
        - name: CLUSTER_NAME
          value: K8SCluster
        - name: CONFIG_SERVICE_PASSWORD
          value: configPwd
        - name: MONGODB_PASSWORD
          value: mongodbPwd
        - name: NOTIFICATION_SERVICE_PASSWORD
          value: notificationPwd
        - name: STATISTICS_SERVICE_PASSWORD
          value: statisticsPwd
        - name: aliyun_logs_requestlog
          value: stdout
        image: registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/auth-service:l
atest
        name: auth-service
        ports:
        - containerPort: 5000
        resources: {}
        securityContext:
          capabilities:
            add:
            - all
            drop:
            - SETGID
            - SETUID
          privileged: true
```

```
        volumeMounts:
        - mountPath: /data/oss
          name: volume-name-piggymetrics
        - mountPath: /data/nas
          name: vn-nas-piggymetrics
        - mountPath: /data/yunpan
          name: vn-yunpan-piggymetrics
        - mountPath: /var/run/docker.sock
          name: auth-service-claim3
        - mountPath: /data/tmp
          name: auth-service-claim4
      restartPolicy: Always
      volumes:
      - name: volume-name-piggymetrics
        persistentVolumeClaim:
          claimName: volume-name-piggymetrics
      - name: vn-nas-piggymetrics
        persistentVolumeClaim:
          claimName: vn-nas-piggymetrics
      - name: vn-yunpan-piggymetrics
        persistentVolumeClaim:
          claimName: vn-yunpan-piggymetrics
      - hostPath:
          path: /var/run/docker.sock
        name: auth-service-claim3
      - hostPath:
          path: /tmp
        name: auth-service-claim4
status: {}
```

config-deployment.yaml

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
    aliyun.auto_scaling.max_cpu: "70"
    aliyun.auto_scaling.max_instances: "10"
    aliyun.auto_scaling.min_cpu: "30"
    aliyun.auto_scaling.min_instances: "1"
    aliyun.auto_scaling.step: "1"
    aliyun.latest_image: "false"
    aliyun.probe.cmd: curl -u user:configPwd http://127.0.0.1:8888/account-service/spring.d
ata.mongodb.host
    aliyun.probe.initial_delay_seconds: "30"
    aliyun.probe.timeout_seconds: "30"
    aliyun.rolling_updates: "true"
    aliyun.rolling_updates.parallelism: "2"
    aliyun.scale: "3"
    kompose.cmd: kompose convert -f source/swarm-piggymetrics.yaml --volumes PersistentVolu
meClaimOrHostPath
    kompose.version: 1.17.0 (HEAD)
  creationTimestamp: null
  labels:
```

```
    io.kompose.service: config
  name: config
spec:
  replicas: 10
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxUnavailable: 2
      maxSurge: 13
  template:
    metadata:
      creationTimestamp: null
      labels:
        io.kompose.service: config
    spec:
      containers:
      - env:
        - name: CONFIG_SERVICE_PASSWORD
          value: configPwd
        image: registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/config:latest
        imagePullPolicy: Always
        livenessProbe:
          exec:
            command:
            - curl
            - -u
            - user:configPwd
            - http://127.0.0.1:8888/account-service/spring.data.mongodb.host
          initialDelaySeconds: 30
          timeoutSeconds: 30
        name: config
        resources:
          limits:
            cpu: "1"
            memory: "1073741824"
        securityContext:
          capabilities:
            add:
            - all
      restartPolicy: Always
status: {}
```

gateway-deployment.yaml

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
    aliyun.auto_scaling.max_cpu: "70"
    aliyun.auto_scaling.max_instances: "10"
    aliyun.auto_scaling.min_cpu: "30"
    aliyun.auto_scaling.min_instances: "1"
    aliyun.auto_scaling.step: "1"
    aliyun.log_store_requestlog: stdout
    aliyun.probe.initial_delay_seconds: "50"
    aliyun.probe.timeout_seconds: "30"
    aliyun.probe.url: http://container:4000/index.html
    aliyun.rolling_updates: "true"
    aliyun.rolling_updates.parallelism: "2"
    aliyun.routing.port_4000: gateway;gateway.swarm.piggymetrics.com
    aliyun.scale: "1"
    kompose.cmd: kompose convert -f source/swarm-piggymetrics.yaml --volumes PersistentVolu
meClaimOrHostPath
    kompose.version: 1.17.0 (HEAD)
  creationTimestamp: null
  labels:
    io.kompose.service: gateway
  name: gateway
spec:
  replicas: 1
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        io.kompose.service: gateway
    spec:
      containers:
      - env:
        - name: CONFIG_SERVICE_PASSWORD
          value: configPwd
        - name: aliyun_logs_requestlog
          value: stdout
        image: registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/gateway:latest
        livenessProbe:
          httpGet:
            path: /index.html
            port: 4000
          initialDelaySeconds: 50
          timeoutSeconds: 30
        name: gateway
        ports:
        - containerPort: 4000
      restartPolicy: Always
status: {}
```

gateway-ingress.yaml

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  creationTimestamp: null
  labels:
    io.kompose.service: gateway
  name: gateway
spec:
  rules:
  - host: gateway.c7f537c92438f415b943e7c2f8ca30b3b.cn-zhangjiakou.alicontainer.com
    http:
      paths:
      - backend:
          serviceName: gateway
          servicePort: 80
  - host: gateway.swarm.piggymetrics.com
    http:
      paths:
      - backend:
          serviceName: gateway
          servicePort: 80
status:
  loadBalancer: {}
```

logtail2-daemonset.yaml

```
apiVersion: extensions/v1beta1
kind: DaemonSet
metadata:
  annotations:
    aliyun.global: "true"
    aliyun.latest_image: "false"
    kompose.cmd: kompose convert -f source/swarm-piggymetrics.yaml --volumes PersistentVolu
meClaimOrHostPath
    kompose.version: 1.17.0 (HEAD)
  creationTimestamp: null
  labels:
    io.kompose.service: logtail2
  name: logtail2
spec:
  template:
    metadata:
      creationTimestamp: null
      labels:
        io.kompose.service: logtail2
    spec:
      hostNetwork: true
      dnsPolicy: "None"
      dnsConfig:
        nameservers:
        - 100.100.2.136
        - 100.100.2.138
      hostAliases:
      - ip: "192.168.0.1"
        hostnames:
        - "www.baidu.com"
      - ip: "192.168.253.8"
        hostnames:
        - "sample.aliyun.com"
      containers:
      - env:
        - name: log_region
          value: cn_hangzhou
        image: registry.aliyuncs.com/acs-sample/logtail:yunqi
        name: logtail2
        resources: {}
        securityContext:
          capabilities:
            add:
            - SYS_RAWIO
      restartPolicy: Always
```

swarm-piggymetrics.yaml

```
version: '2'
services:
  rabbitmq:
    image: >-
      registry-vpc.cn-hangzhou.aliyuncs.com/hd_docker_images/rabbitmq:latest
    hostname: rabbitmq
```

```
    hostname: rabbitmq
    restart: always
    ports:
      - '15672:15672'
    logging:
      options:
        max-file: '10'
        max-size: 10m
    labels:
      aliyun.scale: '1'
      aliyun.routing.port_15672: rabbitmq
    memswap_limit: 0
    shm_size: 0
    memswap_reservation: 0
    kernel_memory: 0
    name: rabbitmq
  config:
    environment:
      - CONFIG_SERVICE_PASSWORD=configPwd
    image: >-
      registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/config:latest
    hostname: config
    restart: always
    logging:
      options:
        max-file: '10'
        max-size: 10m
    labels:
      aliyun.probe.cmd: >-
        curl -u user:configPwd
        http://127.0.0.1:8888/account-service/spring.data.mongodb.host
      aliyun.probe.initial_delay_seconds: '30'
      aliyun.probe.timeout_seconds: '30'
      aliyun.scale: '10'
      aliyun.rolling_updates: 'true'
      aliyun.rolling_updates.parallelism: '2'
      aliyun.auto_scaling.max_cpu: '70'
      aliyun.auto_scaling.min_cpu: '30'
      aliyun.auto_scaling.step: '1'
      aliyun.auto_scaling.max_instances: '10'
      aliyun.auto_scaling.min_instances: '1'
      aliyun.latest_image: 'false'
    oom-kill-disable: true
    memswap_limit: 2000000000
    shm_size: 67108864
    memswap_reservation: 536870912
    kernel_memory: 0
    name: config
    mem_limit: 1073741824
    cpu_shares: 100
    cap_add:
      - all
  registry:
    environment:
      - CONFIG_SERVICE_PASSWORD=configPwd
```

```
    image: >-
      registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/registry:latest
    hostname: registry
    restart: always
    depends_on:
      - config
    ports:
      - '8761:8761'
    logging:
      options:
        max-file: '10'
        max-size: 10m
    labels:
      aliyun.scale: '1'
      aliyun.routing.port_8761: eureka
    memswap_limit: 0
    shm_size: 0
    memswap_reservation: 0
    kernel_memory: 0
    name: registry
  gateway:
    environment:
      - CONFIG_SERVICE_PASSWORD=configPwd
    image: >-
      registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/gateway:latest
    hostname: gateway
    restart: always
    depends_on:
      - config
    ports:
      - '80:4000'
    logging:
      options:
        max-file: '10'
        max-size: 10m
    labels:
      aliyun.probe.url: 'http://container:4000/index.html'
      aliyun.probe.initial_delay_seconds: '50'
      aliyun.probe.timeout_seconds: '30'
      aliyun.routing.session_sticky: 'true'
      aliyun.scale: '1'
      aliyun.routing.port_4000: gateway;gateway.swarm.piggymetrics.com
      aliyun.lb.port_4000: tcp://independent-slb:8080
      aliyun.rolling_updates: 'true'
      aliyun.rolling_updates.parallelism: '2'
      aliyun.auto_scaling.max_cpu: '70'
      aliyun.auto_scaling.min_cpu: '30'
      aliyun.auto_scaling.step: '1'
      aliyun.auto_scaling.max_instances: '10'
      aliyun.auto_scaling.min_instances: '1'
      aliyun.log_store_requestlog: stdout  # 采集stdout日志到requestlog日志库中
      aliyun.log_ttl_requestlog: 30  # 设置requestlog日志库日志数据保存30天
      aliyun.log.timestamp: true # Docker 在收集日志的时候可以选择是否添加 timestamp
    external_links:
```

```
      - auth-service.local
    oom-kill-disable: true
    name: gateway
  auth-service:
    environment:
      - MONGODB_PASSWORD=mongodbPwd
      - NOTIFICATION_SERVICE_PASSWORD=notificationPwd
      - STATISTICS_SERVICE_PASSWORD=statisticsPwd
      - ACCOUNT_SERVICE_PASSWORD=accountPwd
      - CONFIG_SERVICE_PASSWORD=configPwd
      - CLUSTER_NAME=$CLUSTER_NAME
    image: >-
      registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/auth-service:latest
    command: ["java", "-Xmx500m", "-jar", "/app/auth-service.jar"]
    restart: always
    logging:
      options:
        max-file: '10'
        max-size: 10m
    labels:
      aliyun.scale: '1'
      aliyun.log_store_requestlog: stdout  # 采集stdout日志到requestlog日志库中
      aliyun.log_ttl_requestlog: 30  # 设置requestlog日志库日志数据保存30天
      aliyun.log.timestamp: false # Docker 在收集日志的时候可以选择是否添加 timestamp
      aliyun.routing.session_sticky: "true"
      aliyun.routing.port_5000: auth-service.local
      aliyun.latest_image: 'true'
      aliyun.depends: config
    links:
      - auth-mongodb
    volumes:
      - 'volume_name_piggymetrics:/data/oss:rw'
      - 'VN_NAS_PIGGYMETRICS:/data/nas:rw'
      - 'VN_YUNPAN_PIGGYMETRICS:/data/yunpan:rw'
      - '/var/run/docker.sock:/var/run/docker.sock:rw'
      - '/tmp:/data/tmp'
    memswap_limit: 0
    shm_size: 0
    memswap_reservation: 0
    kernel_memory: 0
    name: auth-service
    hostname: auth-service
    entrypoint:
      - 'java'
      - '-Xmx200m'
      - '-jar'
      - '/app/auth-service.jar'
    cap_add:
      - all
    cap_drop:
      - SETGID
      - SETUID
    privileged: true
  auth-mongodb:
```

```
    environment:
      MONGODB_PASSWORD: $MONGODB_PASSWORD
      constraint: group==db
    image: registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/mongodb
    restart: always
    labels:
      aliyun.probe.url: tcp://container:27017
    logging:
      options:
        max-size: 10m
        max-file: '10'
    entrypoint: '/init.sh'
    devices:
      - /dev/mem:/dev/mem
  account-service:
    environment:
      CONFIG_SERVICE_PASSWORD: $CONFIG_SERVICE_PASSWORD
      ACCOUNT_SERVICE_PASSWORD: $ACCOUNT_SERVICE_PASSWORD
      MONGODB_PASSWORD: $MONGODB_PASSWORD
      RDS_PASSWORD: $RDS_PASSWORD
      RDS_URL: $RDS_URL
      CLUSTER_NAME: $CLUSTER_NAME
      affinity: service!=account-mongodb
    image: >-
      registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/account-service
    hostname: account-service
    restart: always
    external_links:
      - auth-service.local
    links:
      - account-mongodb:account-mongodb
      - account-db:account-db
    depends_on:
      config:
        condition: service_healthy
    logging:
      options:
        max-size: 10m
        max-file: '10'
    labels:
      aliyun.scale: '1'
      aliyun.log_store_requestlog: stdout  # 采集stdout日志到requestlog日志库中
      aliyun.log_store_errorLog: /var/log/common/common_error.log # 采集error日志到requestlo
g日志库中
      aliyun.log_store_monitorLog: /var/log/monitor/monitor_digest.log # 采集monitor_digest
日志到requestlog日志库中
      aliyun.log_ttl_requestlog: 30  # 设置requestlog日志库日志数据保存30天
      aliyun.log.timestamp: true # Docker 在收集日志的时候可以选择是否添加 timestamp
  account-mongodb:
    environment:
      INIT_DUMP: account-service-dump.js
      MONGODB_PASSWORD: $MONGODB_PASSWORD
      constraint: group==db
    image: registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/mongodb
```

```
    restart: always
    logging:
      options:
        max-size: 10m
        max-file: '10'
  statistics-service:
    environment:
      CONFIG_SERVICE_PASSWORD: $CONFIG_SERVICE_PASSWORD
      MONGODB_PASSWORD: $MONGODB_PASSWORD
      STATISTICS_SERVICE_PASSWORD: $STATISTICS_SERVICE_PASSWORD
    image: >-
      registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/statistics-service
    hostname: statistics-service
    external_links:
      - auth-service.local
    restart: always
    depends_on:
      config:
        condition: service_healthy
    logging:
      options:
        max-size: 10m
        max-file: '10'
    labels:
      aliyun.log_store_requestlog: stdout  # 采集stdout日志到requestlog日志库中
      aliyun.log_ttl_requestlog: 30  # 设置requestlog日志库日志数据保存30天
      aliyun.log.timestamp: true # Docker 在收集日志的时候可以选择是否添加 timestamp
  statistics-mongodb:
    environment:
      MONGODB_PASSWORD: $MONGODB_PASSWORD
      constraint: group==db
    image: registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/mongodb
    hostname: statistics-mongodb
    restart: always
    logging:
      options:
        max-size: 10m
        max-file: '10'
  notification-service:
    environment:
      CONFIG_SERVICE_PASSWORD: $CONFIG_SERVICE_PASSWORD
      MONGODB_PASSWORD: $MONGODB_PASSWORD
      NOTIFICATION_SERVICE_PASSWORD: $NOTIFICATION_SERVICE_PASSWORD
    image: >-
      registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/notification-service
    hostname: notification-service
    external_links:
      - auth-service.local
    restart: always
    depends_on:
      config:
        condition: service_healthy
    logging:
      options:
        max-size: 10m
```

```
        max-size: 10m
        max-file: '10'
    labels:
      aliyun.log_store_requestlog: stdout  # 采集stdout日志到requestlog日志库中
      aliyun.log_ttl_requestlog: 30  # 设置requestlog日志库日志数据保存30天
      aliyun.log.timestamp: true # Docker 在收集日志的时候可以选择是否添加 timestamp
  notification-mongodb:
    image: registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/mongodb
    hostname: notification-mongodb
    restart: always
    environment:
      MONGODB_PASSWORD: $MONGODB_PASSWORD
      constraint: group==db
    logging:
      options:
        max-size: 10m
        max-file: '10'
  monitoring:
    environment:
      - CONFIG_SERVICE_PASSWORD=configPwd
    image: >-
      registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/monitoring:latest
    hostname: monitoring
    restart: always
    depends_on:
      - config
    ports:
      - '9000:9000'
    logging:
      options:
        max-size: 10m
        max-file: '10'
    labels:
      aliyun.scale: '1'
      aliyun.routing.port_9000: hystrix
    memswap_limit: 0
    shm_size: 0
    memswap_reservation: 0
    kernel_memory: 0
    name: monitoring
  turbine-stream-service:
    environment:
      CONFIG_SERVICE_PASSWORD: $CONFIG_SERVICE_PASSWORD
    image: >-
      registry-vpc.cn-zhangjiakou.aliyuncs.com/goomoon-piggymetrics/turbine-stream-service
    hostname: turbine-stream-service
    restart: always
    depends_on:
      config:
        condition: service_healthy
    ports:
      - '8989:8989'
    logging:
      options:
        max-size: 10m
```

```
        max-file: '10'
  logtail2:
    image: registry.aliyuncs.com/acs-sample/logtail:yunqi
    labels:
      aliyun.latest_image: 'false'
      aliyun.global: 'true'
    devices:
      - /dev/mem:/dev/mem
    environment:
      - log_region=cn_hangzhou
    net: host
    extra_hosts:
      - "www.baidu.com:192.168.0.1"
      - "sample.aliyun.com:192.168.253.8"
    dns:
      - 100.100.2.136
      - 100.100.2.138
    cap_add:
      - SYS_RAWIO
```

13.8. 应用回归测试

当应用启动成功后，您需要通过业务功能回归测试，确认应用没有问题。

配置测试域名

在Kubernetes集群中，可以通过配置服务端口实现集群内部不同服务之间的相互调用，通过配置路由实现从集群外部访问集群服务。

而在配置路由时，可以配置多域名，包括生产域名（客户自己备案的域名，完整域名）和测试域名（由容器平台提供域名，提供域名前缀即可）。您可以按照以下方法进行操作。

1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，单击**集群**。
3. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
4.
5. 单击右上角的**创建**。

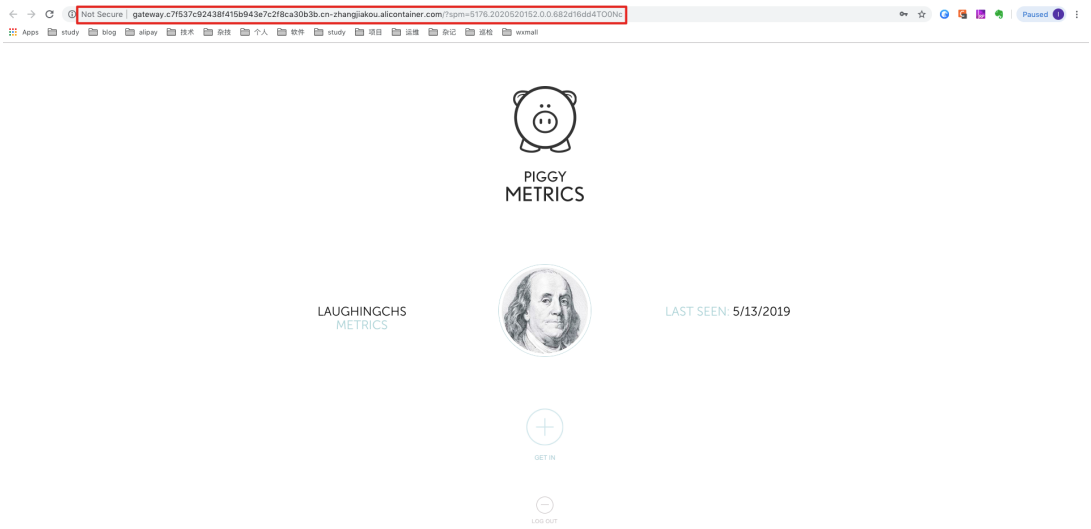
   创建路由的具体操作及参数说明，请参见创建路由。
6. 创建完成后，您可以在路由列表中看到对应访问域名及端点。

> **说明** 此处创建的路由的访问端点即为与集群关联的SLB实例的公网IP地址。

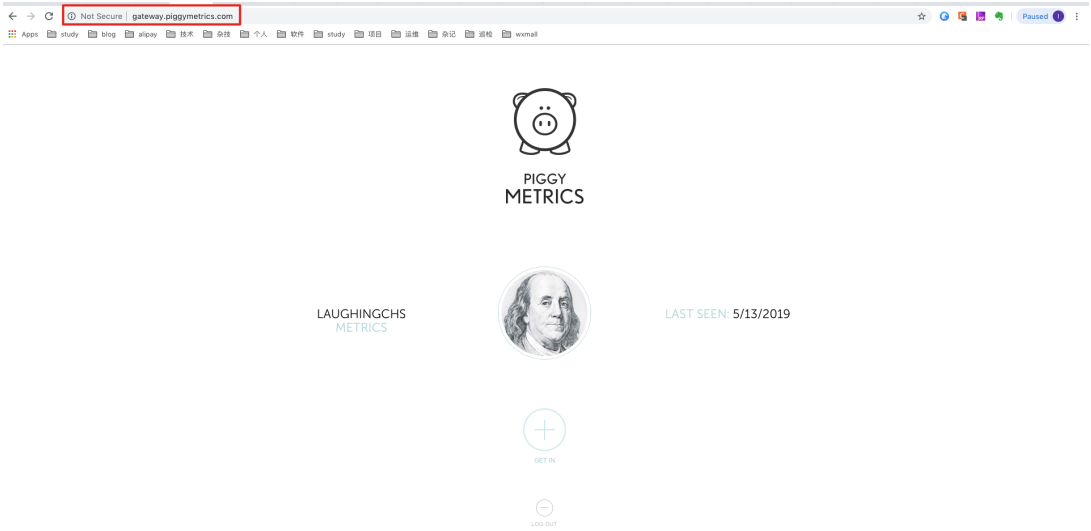
测试业务功能

客户可以通过上述配置的测试域名或客户自己的备案域名（备案域名因尚未更新DNS解析到该SLB公网IP，需在本地做域名绑定）就可以访问Kubernetes集群提供的业务功能。示例域名如下：

测试域名



用户备案域名



> **说明** 请根据测试域名或备案域名完成业务功能回归测试，以保证新集群提供的业务服务没有问题。

确认应用日志

在迁移应用日志配置中，Swarm和Kubernetes的Logstore生成规则不同。所以，Swarm集群生成的应用日志和Kubernetes生成的应用日志在日志服务同个Project，不同Logstore下面。在测试过程中，可以查看Kubernetes创建的Logstore是否有业务日志输出，并确认输出是否正确。

1. 登录日志服务控制台，在**Project列表**区域，单击目标Project名称。
2. 在**日志库**页签，在Logstore列表中，可以看到对应的Logstore。
3. 单击**查询**，即可查看Kubernetes集群的日志信息，确认输出是否正常。

确认应用监控

在确认Kubernetes集群在新Logstore日志输出正常之后，需要进一步确认基于该日志Logstore基础之上的云监控、业务实时监控ARMS、消息投递存储等云产品是否正常配置，详情请参见日志服务-实时消费并按自身业务实际使用情况，确认各消费场景配置是否正常。

本例中，主要通过示例应用swarm-piggymetrics云监控中的主机监控和日志监控确认Kubernetes集群的服务运行情况，具体操作如下：

1. 查看主机监控。详情请参见如何通过云监控查看指定日期的监控数据？。

   在创建Kubernetes集群时，选择安装**云监控插件**，可以监控Kubernetes集群中各个Node节点机器的系统负载及网络监控信息，并从中观察测试流量是否有进入到新Kubernetes集群。

2. 查看日志监控。详情请参见管理日志监控项。

   更详细的业务监控可以通过云日志监控或业务实时监控服务等云产品实现。这里以云监控中的日志监控为例，在原来Swarm集群的日志监控规则基础上，新创建一个日志监控findAccountCount_K8s用于监听K8s集群的机器日志，并配置对应的解析规则，然后可以通过监控图表查看到有新的测试流量进入。

   > **说明** 通过上图，可以明显看到Kubernetes集群账户查询请求从无到有开始上涨，也验证了测试流量确实是由新集群K8s-piggymetrics-cluster在对外提供服务。

13.9. SLB灰度引流

13.9.1. SLB灰度引流概述

本文主要为您介绍灰度部分生产流量到Kubernetes集群，验证Kubernetes集群的业务功能，避免回归测试未能覆盖所有业务功能。

前提条件

您已经完成业务功能回归测试，请参见应用回归测试。

NodePort引流

灰度引流主要有两种方式，基于Kubernetes集群SLB引流或基于Swarm集群SLB引流，具体两种引流方式的情况如下：

- 基于Kubernetes SLB引流：更新客户端或DNS解析，将Kubernetes集群SLB地址追加到客户端或DNS中，实现流量引入。
  - 引流成本高
    更新客户端或DNS解析配置
  - 回滚风险高
    - 万一Kubernetes集群有问题，只能再次更新客户端或DNS配置。
    - 由于客户端发布时间较长或DNS运营商缓存问题，一般无法实时生效，导致线上业务有损。
- 基于Swarm SLB引流：将Kubernetes集群服务挂到Swarm SLB下面并设置权重，实现流量引入。
  - 引流成本较低
    - 通过SLB控制台，更新Swarm SLB配置即可。
    - 可以在SLB控制台权重配置，实现更精细的流量引入。
  - 回滚风险低
    - 通过SLB控制台，更新Swarm SLB配置即可。
    - 实时生效。

操作步骤

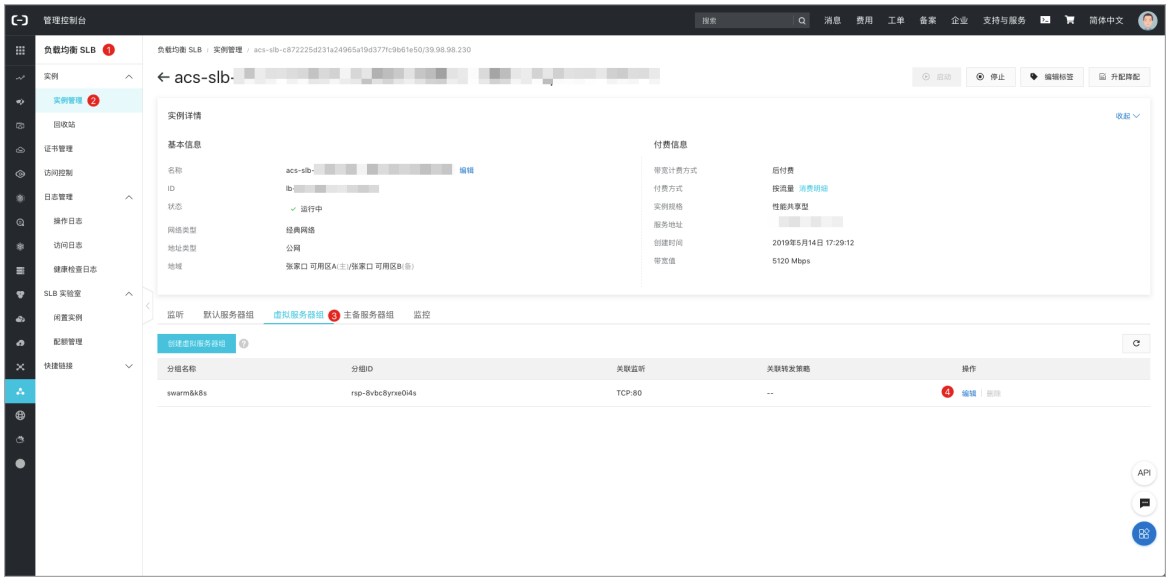
1. 创建Kubernetes NodePort服务。
2. 验证Kubernetes NodePort服务。
3. 修改Swarm SLB配置。

引流快速回滚

如果引入生产流量之后，发现Kubernetes集群业务服务有问题，则调整虚拟服务器组，将Kubernetes集群机器流量权重设置成*0*或将其剔除即可。

操作步骤

1. 登录负载均衡管理控制台，在**实例管理**页面单击目标实例。
2. 在实例详情页面，单击**虚拟服务器组**，在目标分组右侧单击**编辑**。



3. 从右侧滑出**编辑虚拟服务器组**页面，在目标虚拟服务右侧单击**删除**，并单击**确定**。



灰度引流总结

主要介绍如何通过Swarm集群SLB**虚拟服务器组**机制，将生产流量部分引流到Kubernetes的NodePort Service，进一步验证Kubernetes 集群业务服务是否正常。

通过Swarm SLB+Kubernetes NodePort引流方式，相较于传统客户端或DNS切流方式，成本更低（只需在SLB控制台操作），风险更可控（切流/回滚实时生效，流量控制更精细）。

13.9.2. 创建Kubernetes NodePort服务

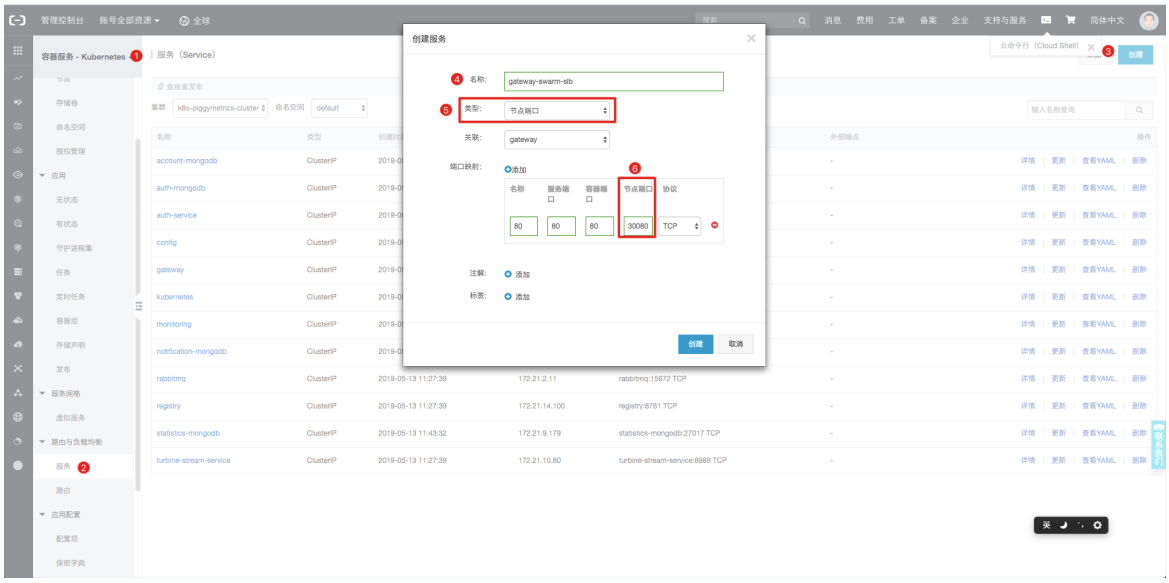
本文主要介绍在Kubernetes-piggymetrics-cluster集群中，创建一个NodePort类型的Service，用于后续从Swarm集群的SLB上引入生产流量。

前提条件

您已经创建好一个Kubernetes托管版集群。请参见创建Kubernetes托管版集群。

操作步骤

1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，单击**集群**。
3. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
4. 在集群管理页左侧导航栏中，选择**网络 > 服务**。
5. 在目标集群Kubernetes-piggymetrics-cluster右侧单击**创建**。
6. 在**创建服务**页面，填写对应的内容，并单击**创建**。请参见管理服务。



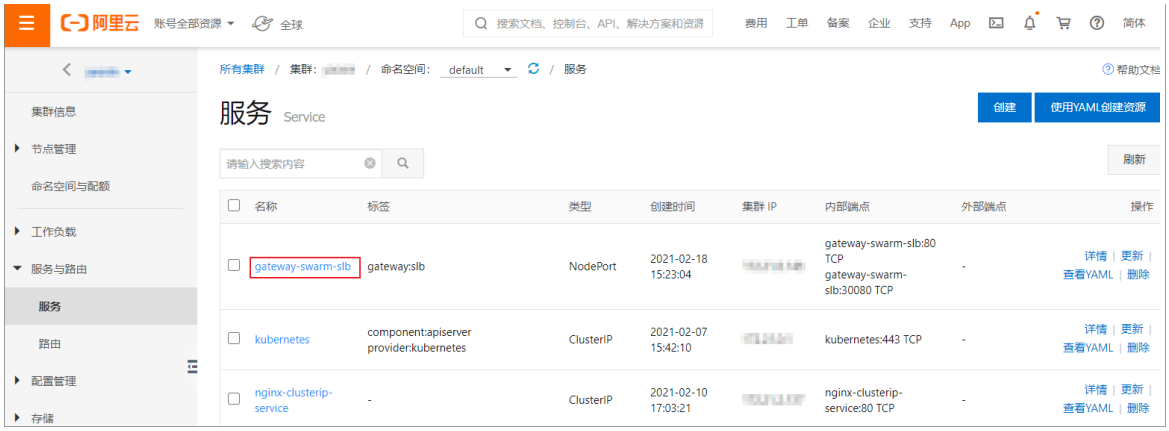
> 说明
>
> - 名称：输入服务的名称，本例中为gateway-swarm-slb。
> - 类型：选择服务类型，即服务访问方式，请选择*节点端口*类型。
>   节点端口：即NodePort，通过每个Node上的IP和静态端口（NodePort）暴露服务。NodePort服务会路由到ClusterIP服务，这个ClusterIP服务会自动创建。通过请求 `<NodeIP>:<NodePort>` ，可以从集群的外部访问一个 NodePort服务。
> - 关联部署：选择服务要绑定的后端对象，本例中是前面创建的gateway服务。
> - 端口映射：添加服务端口和容器端口，容器端口需要与后端的pod中暴露的容器端口一致。节点端口范围为：30000~32767。

13.9.3. 验证Kubernetes NodePort服务

在Kubernetes-piggymetrics-cluster集群中，NodePort类型的Service用于从Swarm集群的SLB上引入生产流量。本文主要介绍如何验证Kubernetes NodePort服务。

1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，单击**集群**。
3. 在**集群列表**页面中，单击目标集群名称或者目标集群右侧**操作**列下的**详情**。
4. 在集群管理页左侧导航栏中，选择**网络 > 服务**。
5. 查看服务是否创建成功。

   在服务列表中，查看gateway-swarm-slb为*NodePort*。

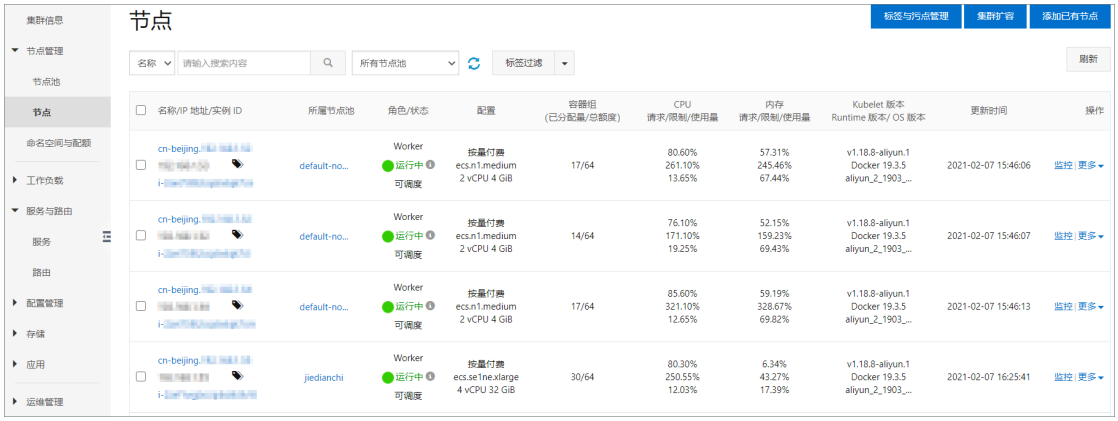


6. 验证服务是否正常。

   本例中，我们通过随机登录一台Kubernetes集群的ECS服务器，请求对应的NodePort服务，看能否正常访问服务；示例是30080端口，ECS机器IP为 192.168.XX.X0。

   i. 在集群管理页左侧导航栏中，选择**节点管理 > 节点**。

   ii. 在**节点**页面，选择目标集群Kubernetes-piggymetrics-cluster，单击目标实例。



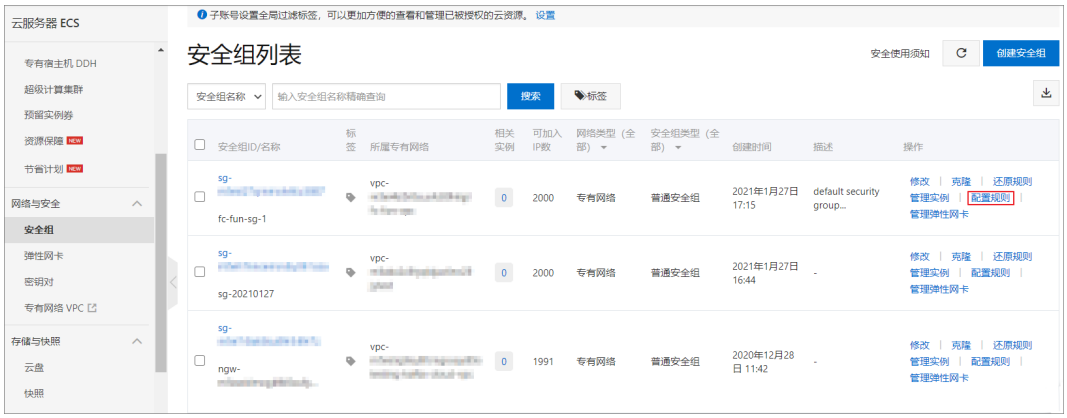
   iii. 在实例基本信息页面，单击**远程连接**，根据界面提示登录该实例。

   iv. 在192.168.XX.X0机器上，通过**ping**、**telnet**、**wget**等命令访问 192.168.XX.X1:30080，确认该NodePort Service是否正常。

```
Transaction Summary
================================================================================
Install  1 Package

Total download size: 64 k
Installed size: 113 k
Is this ok [y/d/N]: y
Downloading packages:
telnet-0.17-64.el7.x86_64.rpm                               |  64 kB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : 1:telnet-0.17-64.el7.x86_64                                  1/1
  Verifying  : 1:telnet-0.17-64.el7.x86_64                                  1/1

Installed:
  telnet.x86_64 1:0.17-64.el7

Complete!
[root@iZ8vbf8jnlyocsy4k5epzyZ ~]# ping 192.168.■.■1
PING 192.168.■.■1 (192.168.■.■1) 56(84) bytes of data.
64 bytes from 192.168.■.■1: icmp_seq=1 ttl=64 time=1.05 ms
64 bytes from 192.168.■.■1: icmp_seq=2 ttl=64 time=0.283 ms
64 bytes from 192.168.■.■1: icmp_seq=3 ttl=64 time=0.324 ms
^C
--- 192.168.■.■1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 0.283/0.552/1.050/0.352 ms
[root@iZ8vbf8jnlyocsy4k5epzyZ ~]# telnet 192.168.■.251 30080
Trying 192.168.■.■1...
Connected to 192.168.■.■1.
Escape character is '^]'.
^CConnection closed by foreign host.
[root@iZ8vbf8jnlyocsy4k5epzyZ ~]# cd /tmp/
[root@iZ8vbf8jnlyocsy4k5epzyZ tmp]# wget http://192.168.■.■1:30080
--2019-05-15 14:53:30--  http://192.168.■.■1:30080/
Connecting to 192.168.■.■1:30080... connected.
HTTP request sent, awaiting response... 200
Length: 32164 (31K) [text/html]
Saving to: 'index.html'

100%[==========================================================================>] 32,164      --.-K/s   in 0s

2019-05-15 14:53:30 (356 MB/s) - 'index.html' saved [32164/32164]

[root@iZ8vbf8jnlyocsy4k5epzyZ tmp]# _
```

由于容器服务针对每个集群ECS实例都会自动创建一个网络安全组，其默认只允许集群内容器所属网段IP访问各个ECS实例端口，集群外部无法访问（SLB不受影响）。因此，我们在上述NodePort Service测试时选择利用Kubernetes 集群内的ECS实例做测试，您可以通过如下操作，访问网络安全组。

a. 登录ECS管理控制台，在右侧导航栏选择**网络和安全 > 安全组**，在目标安全组右侧单击**配置规则**。



b. 在**安全组规则**页面，在**入方向**页签，可以看到该网络安全组的IP及端口。



验证NodePort服务正常后，您需要修改Swarm集群所属SLB的监听规则，将我们的Kubernetes 集群的NodePort服务挂载到Swarm集群的SLB中。更多信息，请参见修改Swarm SLB配置。

13.9.4. 修改Swarm SLB配置

本文介绍如何修改Swarm SLB配置帮助您给Kubernetes集群机器引流。

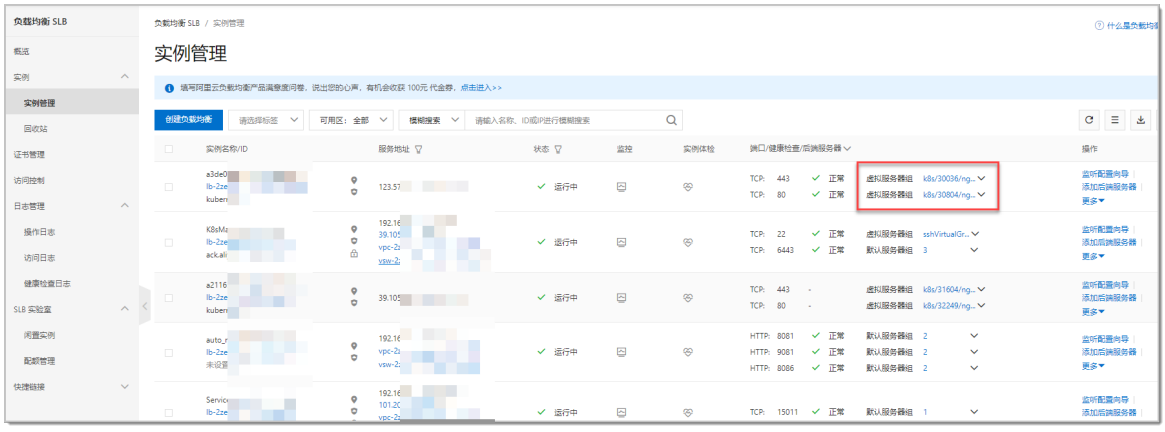
在SLB中，监听到请求之后，需要将请求转发到后端服务器；而后端服务器可以通过3种方式挂载到监听规则中，分别为**默认服务器组**、**主备服务器组**、**虚拟服务器组**。

其中，默认服务器组和主备服务器组要求组内机器的后端机器端口是一样的，无法满足我们Kubernetes引流的诉求（Swarm机器默认是通过9080端口从SLB引流，而Kubernetes集群在我们前一步配置NodePort服务时，只能通过30000~32767之间端口做引流）。所以，我们只能通过虚拟服务器组的方式配置后端服务器，进而实现给Kubernetes集群机器引流。您可以按照以下步骤操作。

确认后端服务器组配置

确认当前Swarm SLB是否是已通过**虚拟服务器组**方式挂载后端机器。

1. 登录负载均衡管理控制台，在左侧导航栏选择**实例 > 实例管理**。
2. 进入**实例管理**页面，在目标实例右侧查看后端服务器的类型。



- 如果后端服务器类型为*默认服务器*，请继续下一步骤。
- 如果后端服务器类型为*虚拟服务器组*，则直接进入修改监听规则，切换到新服务器组步骤。

创建虚拟服务器组

1. 在**实例管理**页面，单击目标实例进入**监听**页签，然后单击**虚拟服务器组**页签。

2. 单击**创建虚拟服务器组**进入**创建虚拟服务器组**页面。



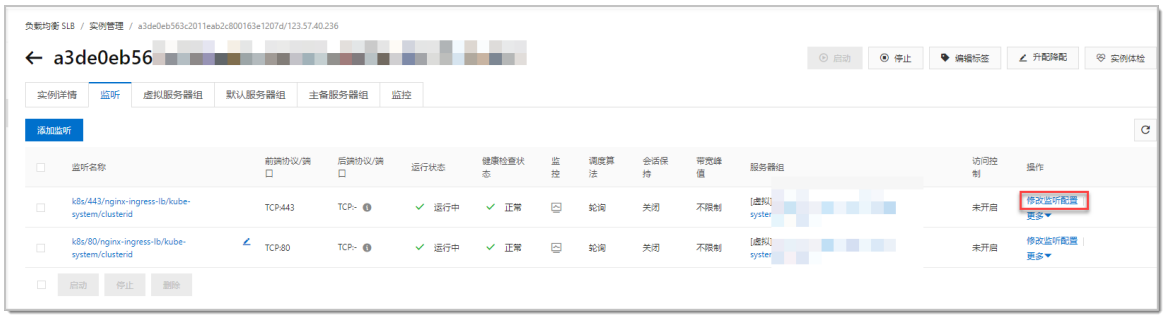
3. 单击**添加**进入**我的服务器**对话框。
4. 选中对应的信息并单击**下一步**创建虚拟服务器组。

> **说明**
>
> - 添加云服务时，Swarm集群机器需全部选择，避免万一切换到该虚拟服务器组时，后端服务器撑不住已有流量，而Kubernetes集群机器可以任选一台或多台。
> - Swarm监听端口和权重设置一定要跟原先**默认服务器组**或**主备服务器组**配置保持一致。

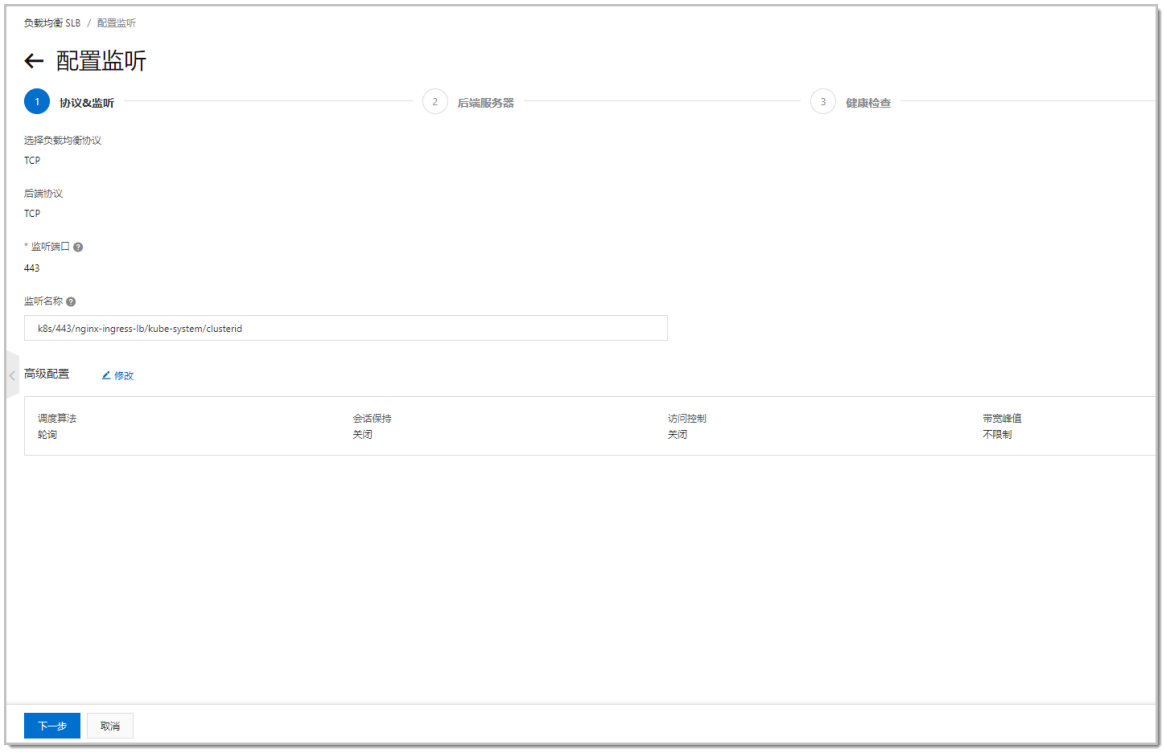
修改监听规则，切换到新服务器组

在创建好虚拟服务器组后，您需要修改SLB监听规则，将服务器组从**默认服务器组**或**主备服务器组**，切换到上述创建的虚拟服务器组上。

1. 在**实例管理**页面，单击目标实例进入**监听**页面。在**监听**页签目标端口右侧单击**修改监听配置**。



2. 在**配置监听**页面，修改**协议&监听**配置并单击**下一步**。



3. 在**后端服务器**配置中，单击**虚拟服务器组**，选择服务器群组为*swarm&K8s*，确认公网/内网IP地址及端口（需要重点确认虚拟服务器组中的swarm机器列表及端口是否正确）正确后，单击**下一步**保存更新。
4. 设置**健康检查**配置，然后单击**下一步**。
5. 设置**配置审核**配置，然后单击**提交**。
6. 返回**监听**页签，可以看到目标端口的服务器组已切换为*虚拟swarm&k8s*。

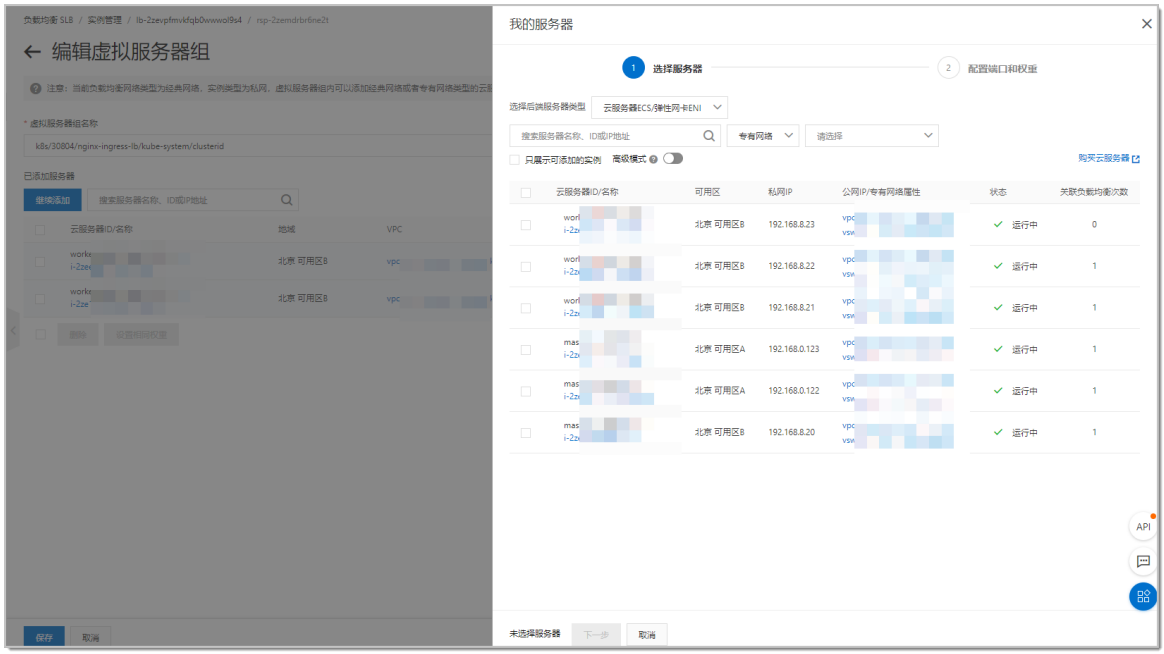
确认业务请求是否正常

请重点监控一段时间，确认线上流量业务请求处理是否正常，可通过**云监控 > 主机监控**，**云监控 > 日志监控**或者客户自身搭建的监控体系等手动观察业务服务响应情况。

调整虚拟服务器组

在确认将SLB的后端服务器切换成*虚拟服务器组*类型没问题之后，下一步我们要做的事情就是调整虚拟服务器组，加入Kubernetes机器，从而实现将生产流量部分引流到Kubernetes集群。

1. 在**实例管理**页面，单击目标实例进入**监听**页签。在**监听**页签监听名称右侧单击服务器组。
2. 进入**编辑虚拟服务器编辑**页面中，可通过单击**继续添加**选择添加Kubernetes服务器并保存，即完成Kubernetes机器的挂载引流。



> **说明**
>
> - Kubernetes集群机器可以按业务灰度流量需要选择加入一台或多台，方便做流量控制。
> - Kubernetes机器端口设置，需要跟前面NodePort设置的端口号一样。
>   区间应该为30000~32767，这里示例值为30080端口，映射到Kubernetes中的gateway-swarm-slb服务。
> - 如有需要，可以通过权重更精细地控制Kubernetes集群的流量占比。

确认流量进去Kubernetes集群

请重点监控一段时间，确认线上流量业务请求处理是否正常；并确认Kubernetes集群是否按虚拟服务器组设置的机器比例及权重配置引流成功。

> **说明** 这里只是通过Swarm SLB引入部分生产流量来验Kubernetes集群业务功能是否正常；对应的Swarm SLB此时并不会下线，需要等到后续客户端切流完成之后，在下线Swarm集群章节下线Swarm集群及Swarm SLB。

13.10. 客户端流量切换

本文档主要通过生产环境灰度引流验证，确认Kubernetes集群业务服务正常之后，如何逐步将线上流量切换到Kubernetes集群。主要包括基于DNS域名解析切流、通过客户端切流两种方式。

前提条件

为了降低业务风险，切流之前，请务必确保Kubernetes集群应用服务正常，您可通过测试域名或备案域名对Kubernetes集群再做一次业务功能回归验证，请参见应用回归测试。

DNS切流

若客户端是通过备案域名访问集群服务，可通过调整DNS配置逐步实现生产流量不停机迁移。

操作步骤

1. 获取新Kubernetes集群默认SLB IP地址。本示例为*39.XX.XX.112*。

登录容器服务管理控制台，在左侧导航栏选择**网络 > 路由**，选择命名空间*default*，获取目标路由的端点为*39.XX.XX.112*。

2. 前往备案域名所在DNS服务商，将Kubernetes集群的SLB的IP地址*39.XX.XX.112*添加到备案域名*piggymetrics.com*的后端IP列表中。
3. 请重点观察线上生产流量访问Kubernetes集群是否正常，请参见应用回归测试。

> **说明** 您也可基于自身的监控体系，确认Kubernetes集群的流量和业务服务是否正常。

4. 观察一段时间，在确认生产流量访问Kubernetes集群正常后，请前往备案域名所在DNS服务商，将Swarm集群的SLB IP地址从备案域名的后端IP地址列表中删除。

   本示例，将*39.XX.XX.230*这个Swarm集群SLB IP地址从 `piggymetrics.com` 域名的后端IP地址中删除。

客户端切流

若客户端通过Host IP访问Swarm SLB IP或者通过测试域名访问Swarm集群，则由于测试域名跟集群标识有关，导致Swarm集群测试域名与Kubernetes集群测试域名不一致，那么在原容器服务Swarm集群基础上无法做到不停机迁移业务，请选择业务量小的时候进行业务迁移。

操作步骤

1. 获取新Kubernetes集群默认SLB IP地址。本示例为*39.XX.XX.112*。

   登录容器服务管理控制台，在左侧导航栏选择**网络 > 路由**，选择命名空间*default*，获取目标路由的端点为*39.XX.XX.112*。

2. 通过升级客户配置或代码方式，将*39.XX.XX.112*地址添加到客户端访问列表中。
3. 请重点观察线上生产流量访问Kubernetes集群是否正常。请参见应用回归测试。

> **说明** 您也可基于自身的监控体系，确认Kubernetes集群的流量和业务服务是否正常。

4. 观察一段时间，在确认生产流量访问Kubernetes集群正常后，请通过升级客户配置或代码方式，将Swarm集群的SLB IP地址从客户端访问列表中删除。

   本示例，将Swarm集群SLB IP地址*39.XX.XX.230*从客户端访问列表中删除。

13.11. 下线Swarm集群

针对不需要的Swarm集群，本文介绍如何下线Swarm集群。

确认Swarm集群流量

1. 确认SLB流量。

   在下线Swarm集群前，需要重点确认老的Swarm SLB已无流量引入，您可以通过SLB提供的监控视图进行查看。监控视图支持按照SLB实例或监听规则视角查看，您可以通过以下操作查看监控视图，确认Swarm集群流量。

   i. 登录负载均衡管理控制台。
   ii. 在左侧导航栏中，选择**传统型负载均衡 CLB(原SLB) > 实例管理**。
   iii. 在**实例管理**页面，单击目标实例，进入实例详情页面。
   iv. 在**监控**页签，设置监听规则等，查看相应的监控视图。

2. 确认应用流量。

建议您通过自己的业务监控体系重点观察一段时间，确认老的Swarm集群已无业务流量引入。避免出现客户端升级未完成或DNS有缓存，导致有残余流量存在，进而影响线上业务。

下线Swarm集群

在确认K8s集群正常对外提供服务，老的Swarm集群没有残余流量后，您可以开始着手准备下线老Swarm集群。主要包括以下3个步骤：

1. 备份Swarm集群配置。

   为了确保K8s集群运行中出现未知问题时，能快速回溯原Swarm集群配置，建议备份一份老Swarm集群的配置，避免配置丢失不可恢复，包括但不仅限于以下维度的配置：

   - 集群Node标签、网络、数据卷配置、配置项、SLB配置等。
   - 各应用编排文件、日志等。

2. 下线Swarm集群资源。

   > **注意** 在Swarm集群中，Node节点ECS、应用、服务、SLB等云产品都挂在Swarm集群上，集群被删除，其关联的云产品都会被删除，请慎重确认。

   i. 登录容器服务管理控制台。

   ii. 在左侧导航栏选择**集群**。

   iii. 在目标集群右侧单击**更多>删除**，详情请参见删除集群。

3. 下线云盘资源。

   云盘只能挂载在单一ECS机器上，如果云盘是作为数据盘需要迁移时，只能选择停机迁移到K8s集群。更多信息，请参见迁移集群配置。
   如果云盘只是作为数据盘用于数据收集且无须带数据迁移的，则在K8s集群里面可以新建一个云盘作为数据存储，而在老的Swarm集群下线之后，其对应挂载的云盘并不会自动释放，只会变成**待挂载**状态。

   > **说明** 请按业务需要先收集或备份好云盘上的数据再释放云盘，避免数据丢失不可恢复。

   若需释放Swarm集群云盘，您可以按照以下步骤在云盘控制台进行手动释放。

   i. 登录ECS管理控制台。

   ii. 在左侧导航栏选择**存储与快照 > 云盘**。

   iii. 在目标云盘右侧选择**更多 > 释放**，完成Swarm集群云盘的释放。

14.基因计算

14.1. 基因计算工作流

本文介绍基因计算工作流的场景描述、解决问题、架构图及操作参考链接。

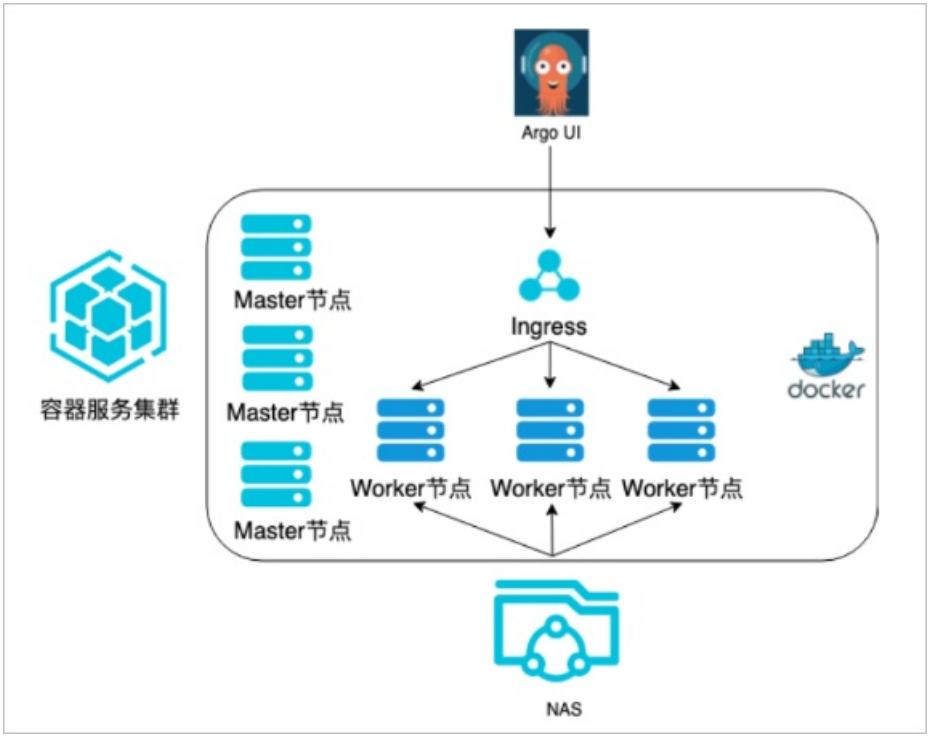
场景描述

本文适合利用容器服务Kubernetes版上的工作流引擎进行大规模基因测序的场景。阿里云工作流引擎基于开源项目Argo实现，支持并发、循环、重试等多种执行策略典型的基因计算过程，把数据分批进行计算，按照规定好的步骤依次完成计算，这符合工作流的特点：多层次，有向无环图。

解决问题

- 构建基因计算工作流。
- 容器服务Kubernetes版结合共享文件存储NAS提供数据服务。
- 构建单Pod工作流。

架构图



参考链接

有关基因计算工作流的详情，请参见基因计算工作流。

15.微服务

15.1. MSE微服务治理最佳实践

本文介绍通过微服务引擎MSE（Microservices Engine）进行微服务治理的最佳实践。

- 如何在MSE上实现多语言微服务治理
- 使用Cloud Toolkit实现微服务的端云互联
- 为K8s集群命名空间中的应用开启微服务治理
- 基于Ingress-nginx网关实现全链路灰度
- 基于MSE云原生网关实现全链路灰度
- 基于自建Spring Cloud Gateway或Zuul网关实现全链路灰度
- 基于消息队列RocketMQ版实现全链路灰度
- 基于MSE实现微服务应用无损上下线
- 通过Jenkins构建CI/CD实现金丝雀发布
- 微服务敏捷开发最佳实践

15.2. 微服务应用问题定位及故障演练

本文介绍微服务应用问题定位及故障演练的场景描述、解决问题、架构图及操作参考链接。

场景描述

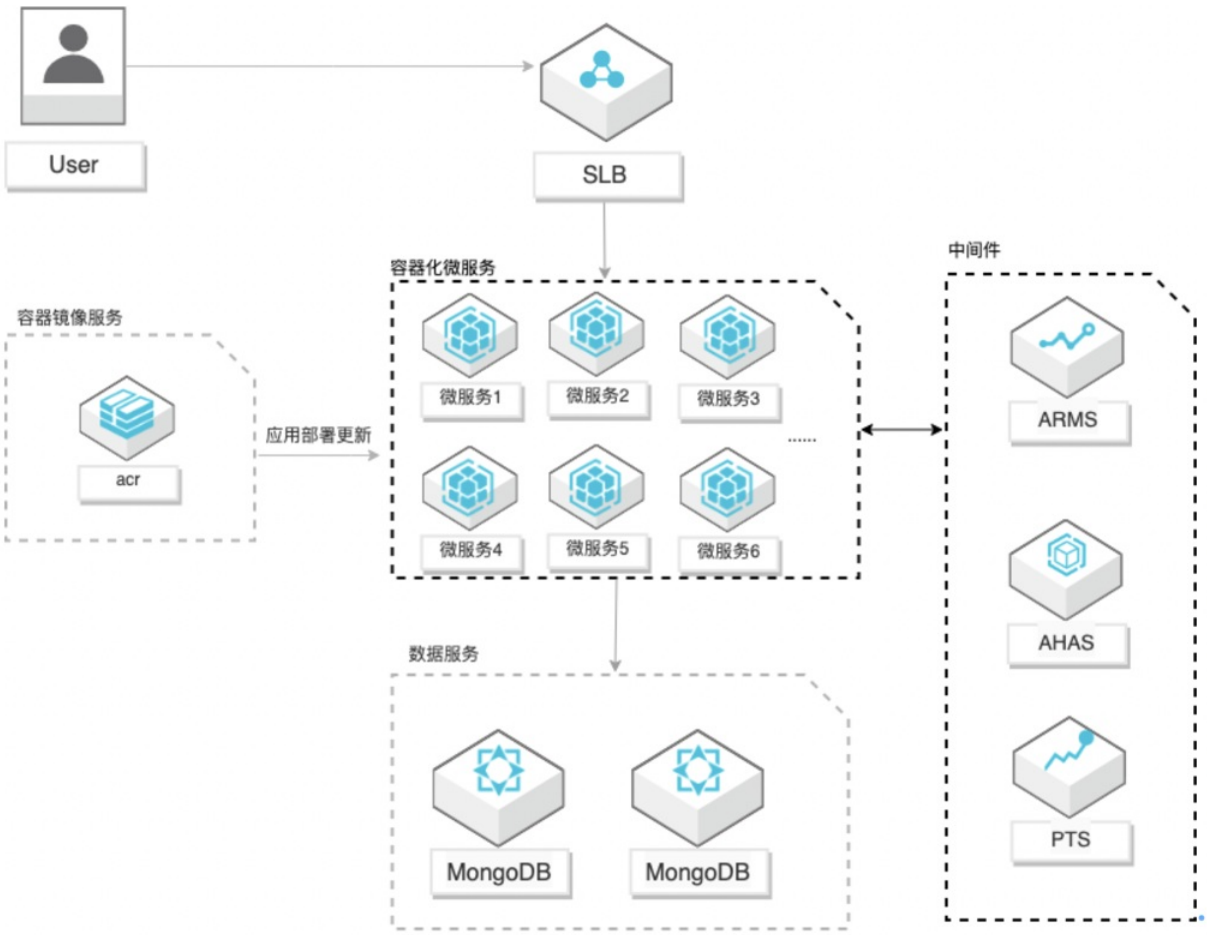
您微服务架构应用基于阿里云容器服务Kubernetes（ACK）部署，通过阿里云业务实时监控ARMS发现微服务应用的容错能力是否健壮、容器编排配置是否合理、节点故障引发的问题等。方案通过AHAS故障注入模拟生产环境产生的故障，通过ARMS及时发现问题、定位问题，并结合ARMS的告警功能快速发现并解决问题。

- 发现调用下游一个服务实例出现异常。
- 发现业务Pod故障，验证业务的稳定性。
- 发现调用数据库延迟故障，验证Pod的水平扩容能力。

解决问题

- 定位并解决微服务应用的容错能力。
- 定位并解决微服务应用编排合理性。
- 发现故障并验证系统故障的告警时效性。

架构图



参考链接

有关微服务应用问题定位及故障演练的详情，请参见微服务应用问题定位及故障演练最佳实践。

15.3. 微服务架构日志采集运维管理

本文介绍微服务架构日志采集运维管理的场景描述、方案优势、解决问题、架构图及操作参考链接。

场景描述

阿里云日志服务LOG（原SLS）结合阿里云容器服务Kubernetes版（ACK）集群日志特点以及应用场景，提供全方位的容器微服务应用环境下的日志采集、处理以及分析的实践解决方案。

方案优势

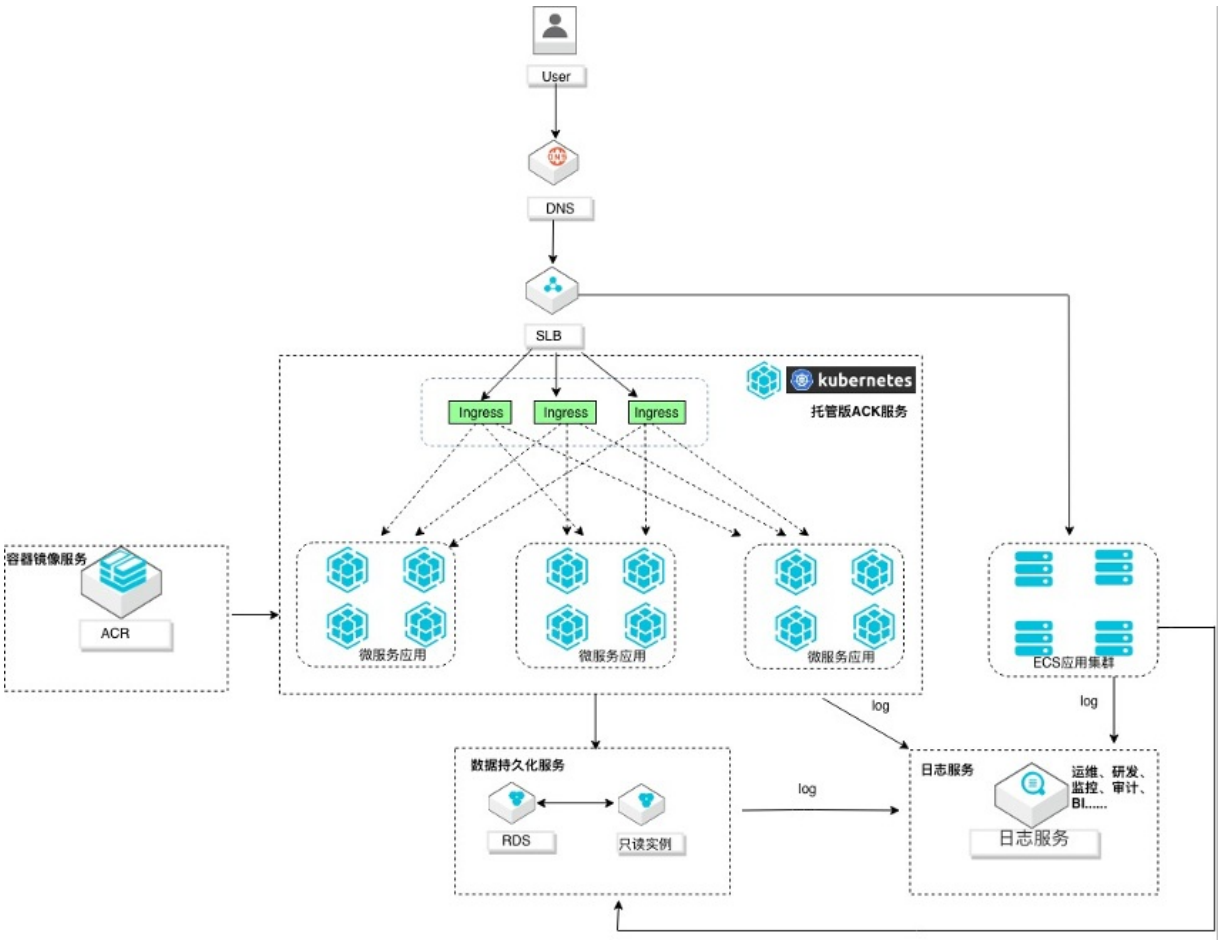
- 日志采集配置灵活，支持多种日志形态、日志采集与ACK完美集成。
- 日志采集Agent占用资源少。
- 安装部署方便、运维成本低。
- 便捷实用，支持多种日志智能分析能力。

解决问题

- 容器微服务架构下日志采集困难。
- 日志采集Agent资源消耗多。
- 运维日志平台系统成本代价高。

- 日志分析方式少、日志分析不方便。

架构图



参考链接

有关微服务架构日志采集运维管理的详情，请参见微服务架构日志采集运维管理最佳实践。

15.4. 微服务引擎的线上流量治理

本文介绍微服务引擎的线上流量治理的场景描述、解决问题、架构图及操作参考链接。

场景描述

随着业务不断创新，微服务架构及数字化转型不断落地，在这个过程中大量企业采用了开源的组件构建微服务，例如Spring Cloud体系或Apache Dubbo等。微服务在快速迭代过程中保障线上流量不受损是一个难题。同时开源产品无运维工具，常常需要投入较大的运维人力和成本。

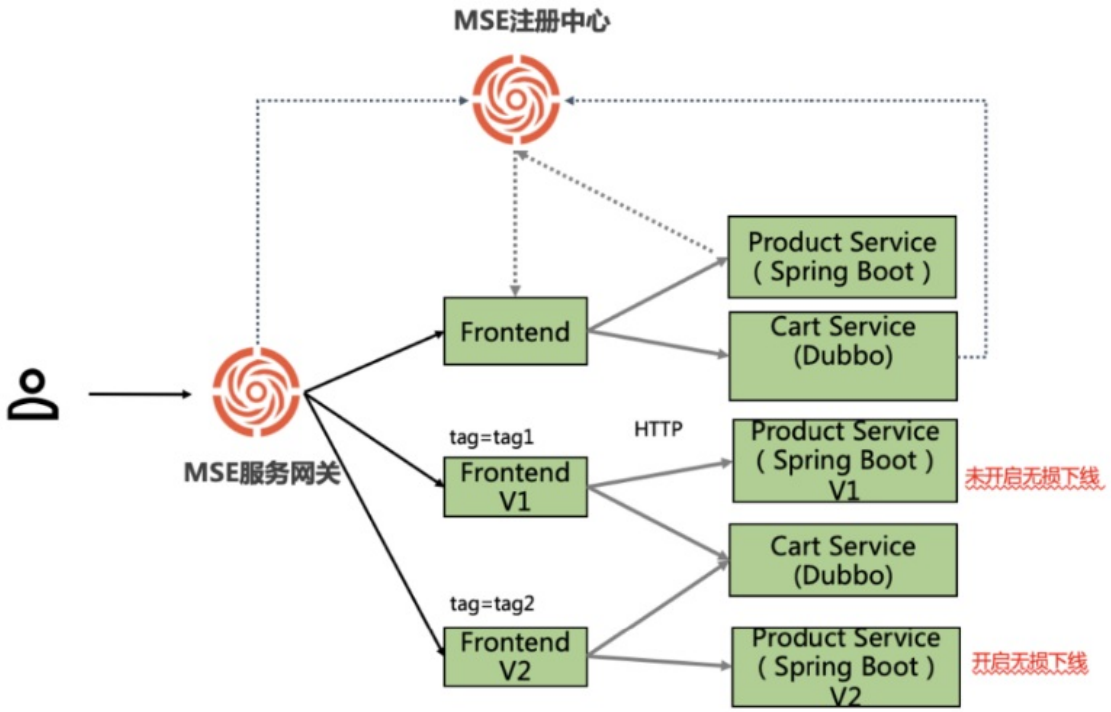
使用微服务引擎的线上流量治理，可以快速集成主流开源微服务框架，实现业务零改造，解决开源框架在生产落地过程中的痛点，例如无损上下线、标签路由等。通过托管微服务开源组件（API网关、注册中心、配置中心等）的服务，提供白屏化监控告警、容灾、宕机重启、扩缩容等能力，帮助企业释放业务无关的运维成本，聚焦业务本身的运维和发展。

解决问题

- 快速集成：通过JavaAgent技术实现Sping Cloud和Dubbo框架，实现业务零改造接入。
- 免运维：托管微服务依赖开源中间件的服务，提供白屏化监控告警、容灾、宕机重启、扩缩容等能力，帮助释放业务无关的运维成本，聚焦自身业务本身的运维和发展。

- 开源增强：提供开源框架在生产落地过程中的痛点，例如应用无损上下线、金丝雀发布、南北东西流量打通等，帮助提高业务自身SLA和降低自研成本。

架构图



参考链接

有关微服务引擎的线上流量治理的详情，请参见微服务引擎的线上流量治理。

15.5. 异地双活场景下的数据双向同步

本文介绍异地双活场景下数据双向同步的场景描述、解决问题、架构图及操作参考链接。

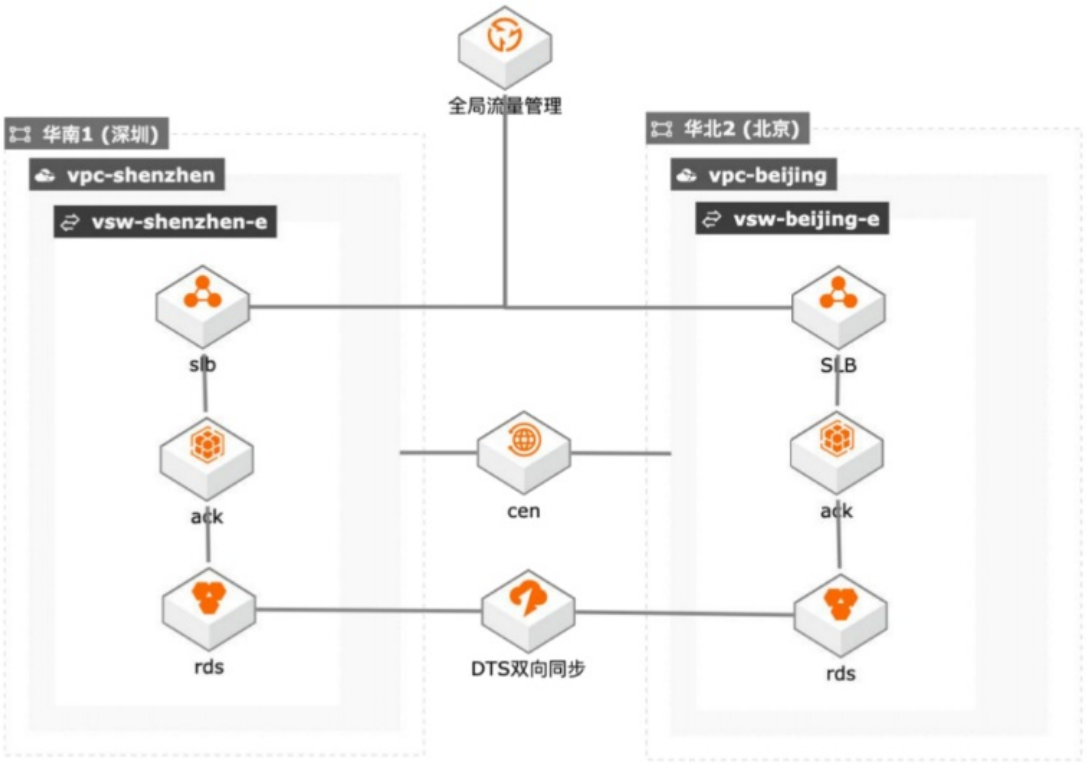
场景描述

随着业务规模的扩大，对系统高可用性要求越来越高，越来越多客户采用异地双活或多活架构。多活架构往往涉及业务侧做单元化改造，本方案仅模拟客户已做单元化改造后的数据双向同步，数据库采用双主架构，本地写本地读，同时保证双库的数据一致性，为业务增加可用性和灵活性。

解决问题

- 数据库双向同步。
- 数据库全局ID不冲突。
- 双活架构的数据库建设问题。

架构图



参考链接

有关异地双活场景下的数据双向同步详情，请参见异地双活场景下的数据双向同步。

15.6. 微服务分布式事务

本文介绍微服务分布式事务的场景描述、解决问题、架构图及操作参考链接。

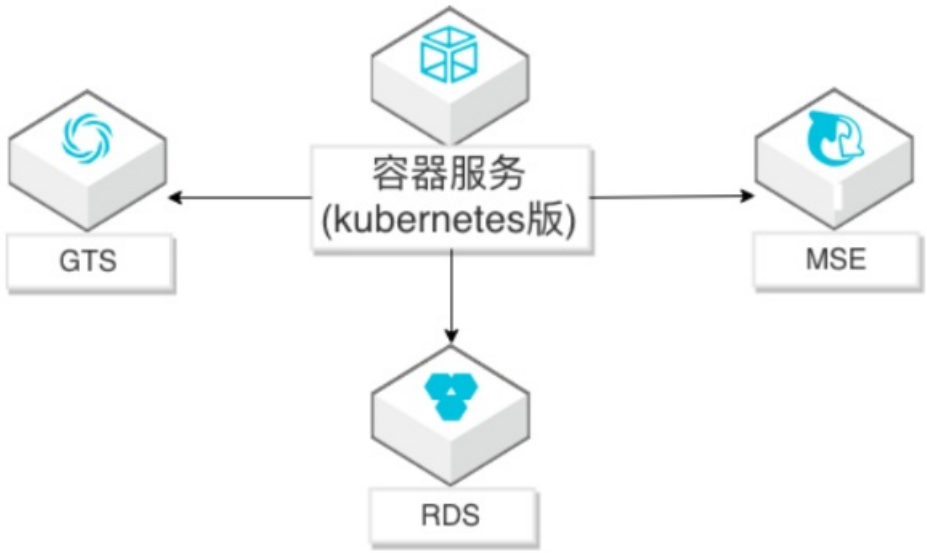
场景描述

本文档适用于在分布式应用中，多个服务间存在分布式事务的场景。通过阿里云全局事务服务（GTS）保障数据的最终一致性，提供简单、易用、高效的分布式事务解决方案。

解决问题

- 微服务应用的分布式事务解决方案，保障数据的最终一致性。
- 和业务解耦，降低架构复杂性，提供对业务无侵入的方案。

架构图



参考链接

有关微服务分布式事务的详情，请参见微服务分布式事务最佳实践。

15.7. 基于容器快速搭建小程序后端服务

本文介绍基于容器快速搭建小程序后端服务的场景描述、解决问题、架构图及操作参考链接。

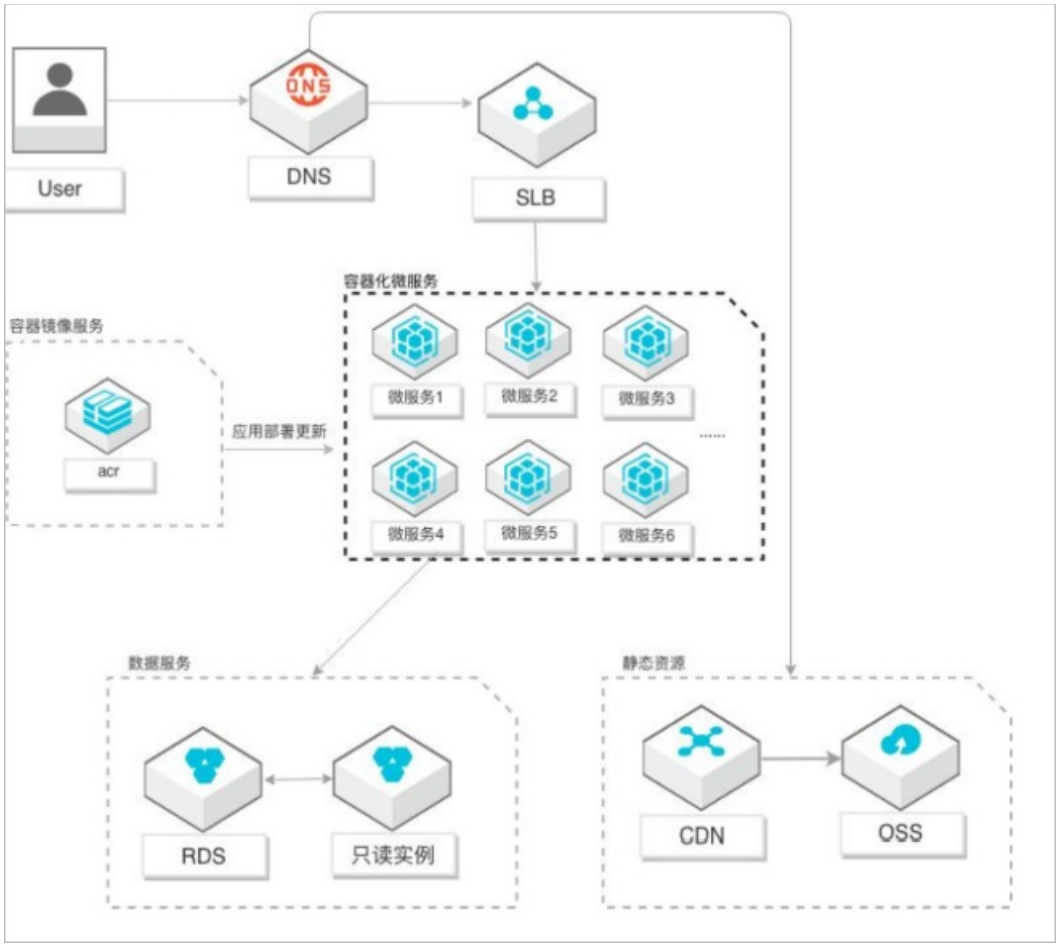
场景描述

小程序作为一种新的开放能力被广泛应用于各业务场景，开发者可以快速地开发一个小程序，在微信可便捷地获取和传播，同时依赖微信平台原生能力获取出色的使用体验。阿里云容器服务结合阿里云云原生能力帮助您快速构建小程序的后端服务。

解决问题

- 微服务架构开发实践。
- ACK部署微服务实践指导。

架构图



参考链接

有关基于容器快速搭建小程序后端服务的详情，请参见基于容器快速搭建小程序后端服务。

16.Serverless

16.1. 基于Knative的消息流事件处理

本文介绍基于Knative的消息流事件处理的场景描述、方案优势、解决问题、架构图及操作参考链接。

场景描述

本文描述了事件通过阿里云消息队列Kafka实时推送到服务端进行数据和业务处理的过程，并基于Knative的弹性来实现满足资源高峰需求，做到真正的按需使用，实时弹性，整个过程完全自动化，极大的减少了业务开发人员在基础设施上的心智负担。

方案优势

- 极致弹性：基于业务负载情况，自动进行容器实例的秒级弹性伸缩。
- 事件处理：基于阿里云容器对消息队列进行集成，弹性处理海量业务消息。
- 异构计算：支持GPU、NPU等实例类型。
- K8s兼容：完全兼容K8s，您可以开通阿里云容器服务ACK或ASK后一键安装并使用，支持容器控制台和kubctl两种方式进行管理操作。

解决问题

- 集群资源利用率低。
- 集群资源可扩展性差。

架构图



参考链接

有关基于Knative的消息流事件处理的详情，请参见基于Knative的消息流事件处理。

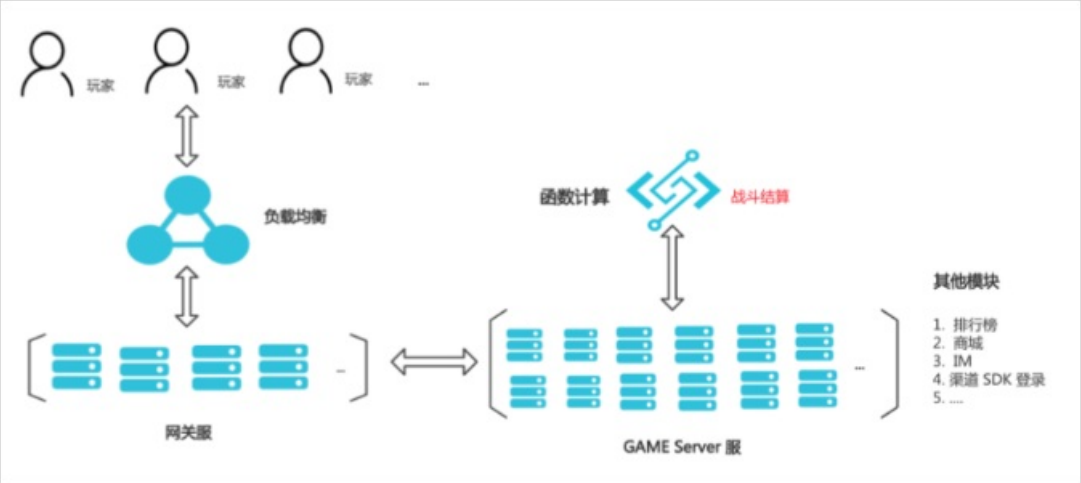
16.2. 基于ECI和FaaS构建游戏战斗结算服务

本文介绍基于ECI和FaaS构建游戏战斗结算服务的背景信息、架构图及操作参考链接。

背景信息

在游戏行业的很多游戏类型中，尤其是SLG，为了防止客户端作弊，在每局战斗之后，在客户端预判玩家胜利的情况下，需要服务端来进行战斗数据的结算，从而确定玩家是不是真正的胜利。战斗结算是强CPU密集型，结算系统每日需要大量的计算力，尤其是开服或者活动期间忽然涌入的大量玩家，导致需要的计算量瞬间几倍增长，同时需要结算系统保持稳定的延时来保证玩家的用户体验。

架构图



参考链接

有关基于ECI和FaaS构建游戏战斗结算服务的详情，请参见基于ECI+FaaS构建游戏战斗结算服务。

16.3. 基于ECI、RI和EIP降低成本方案

本文介绍基于ECI、RI和EIP降低成本方案的场景描述、方案优势、架构图及操作参考链接。

场景描述

随着企业逐步将自己的业务搬移到云上，越来越多的企业也开始重视云上的使用成本。

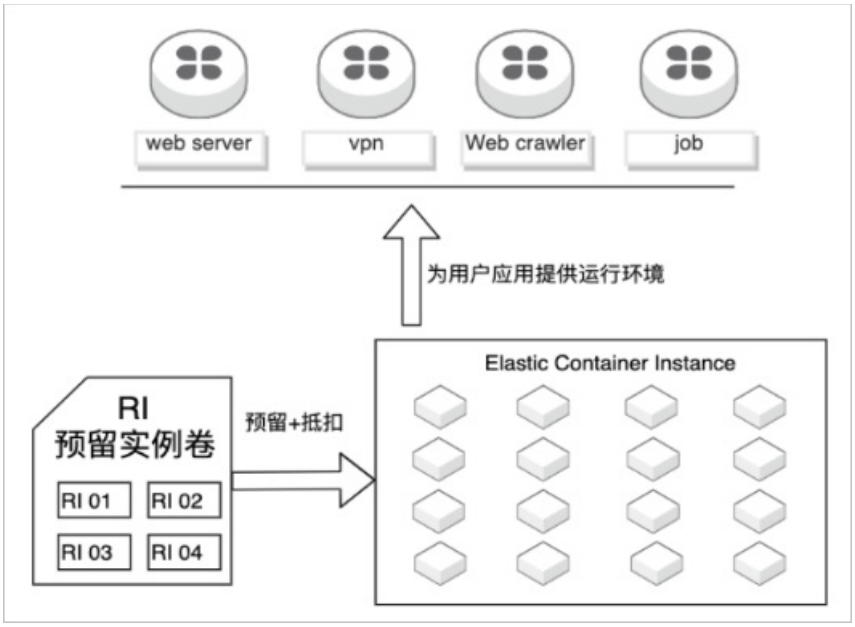
针对具体的使用场景，特别是无状态服务、易于容器化的场景，基于ECI、RI和EIP的组合方式能有效降您得成本。预留实例券RI，可以灵活的抵扣相关资源费用；弹性容器实例ECI，在资源使用率、启动速度、灵活性和成本上具有一定的优势。

本文以部署一个无状态的网站为例，使用ECI、RI和EIP的组合方式，通过RI抵扣计算资源费用，按量支付流量费用，帮助您快速验证云上服务，有效控制云上资源成本。

方案优势

- 使用RI，可以灵活的抵扣相关资源费用。
- 使用ECI作为应用运行环境，在灵活性、成本和资源使用率上更有优势。
- 使用ECI、RI和EIP的组合方式，资源使用率高，通过RI抵扣计算资源费用，按量支付流量费用，能有效降低整体成本费用。

架构图



参考链接

关于基于ECI、RI和EIP降低成本方案的详情，请参见基于ECI、RI和EIP降低成本方案。

16.4. 混合云IDC自有K8s弹性使用ECI

本文介绍混合云IDC自有K8s弹性使用ECI的场景描述、解决问题、架构图及操作参考链接。

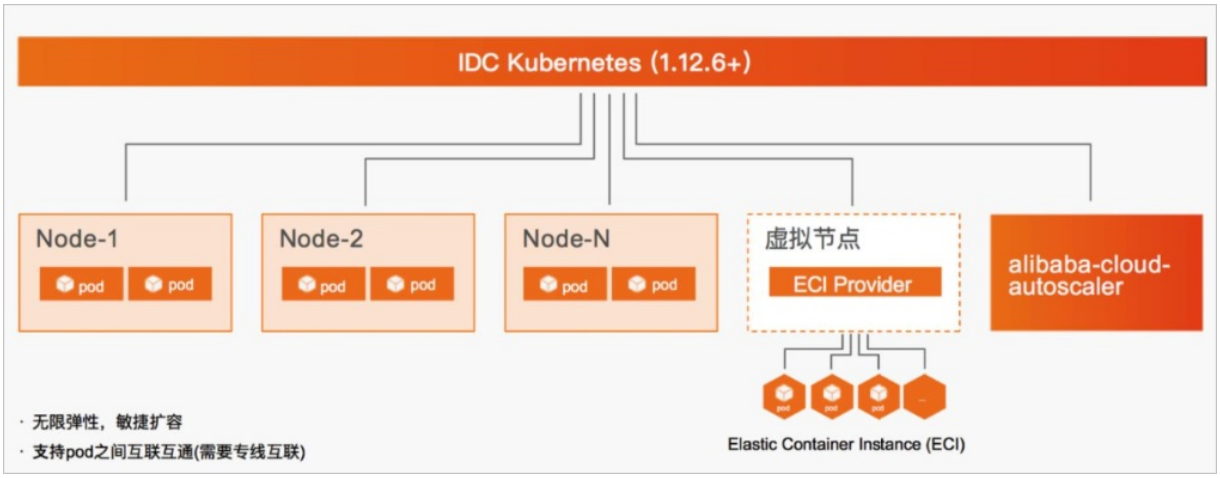
场景描述

本文介绍线下IDC与云端通过专线构建混合云架构，自有K8s利用虚拟节点弹性调用ECI承载业务高峰期资源需求的最佳实践。

解决问题

- 在混合云环境下，将自有K8s集群注册至ACK，实现云端纳管。
- 纳管K8s集群部署Virtual Node，使集群具备ECI资源调度能力。
- 在以上环境中部署Web及离线作业应用，并使用ECI资源作为弹性资源池满足业务波峰需求。

架构图



参考链接

有关混合云IDC自有K8s弹性使用ECI的详情，请参见混合云IDC自有K8s弹性使用ECI。

17.AI

17.1. ACK集群实现GPU成本优化

本文介绍ACK实现GPU成本优化的场景描述、解决问题、架构图及操作参考链接。

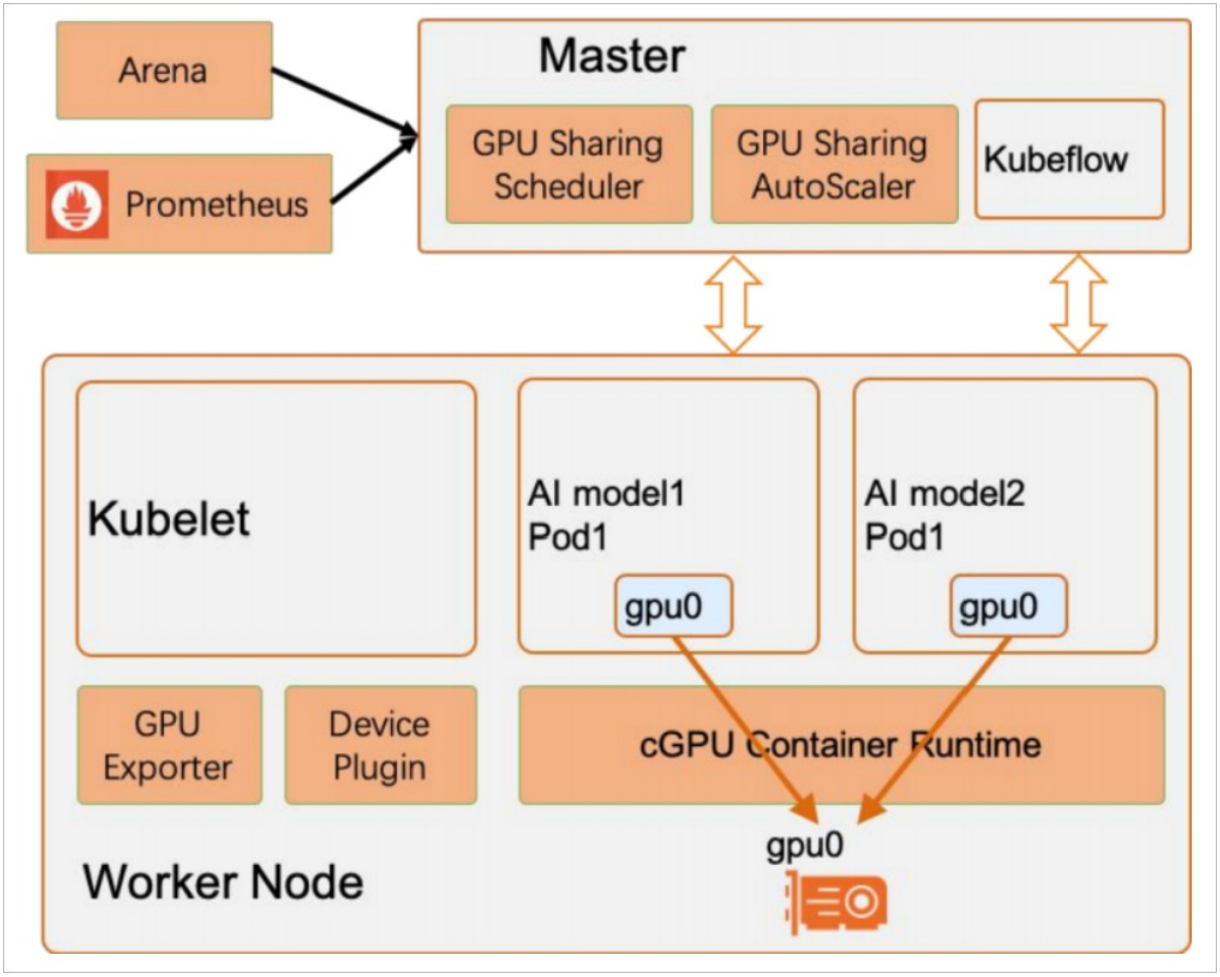
场景描述

利用阿里云容器服务ACK部署GPU集群之后，出于成本优化的考虑，对于集群中GPU利用率不高的应用， 例如推理的应用，建议利用阿里云cGPU技术将一定数量的应用运行至一 块GPU卡上，以提高利用率。对于GPU利用率比较高的应用，不做改动。这样可以实现灵活管理的同时， 降低整体成本。

解决问题

- 在ACK服务中实现共享GPU，灵活拆分。
- 提高GPU利用率，降低TCO。
- 应用无需修改，保障多个容器共享同一个GPU时，彼此互相隔离。

架构图



参考链接

有关ACK实现GPU成本优化的详情，请参见ACK实现GPU成本优化。

17.2. ACK集群实现基于弹性计算的AI推理

本文介绍基于弹性计算的AI推理的场景描述、解决问题、架构图及操作参考链接。

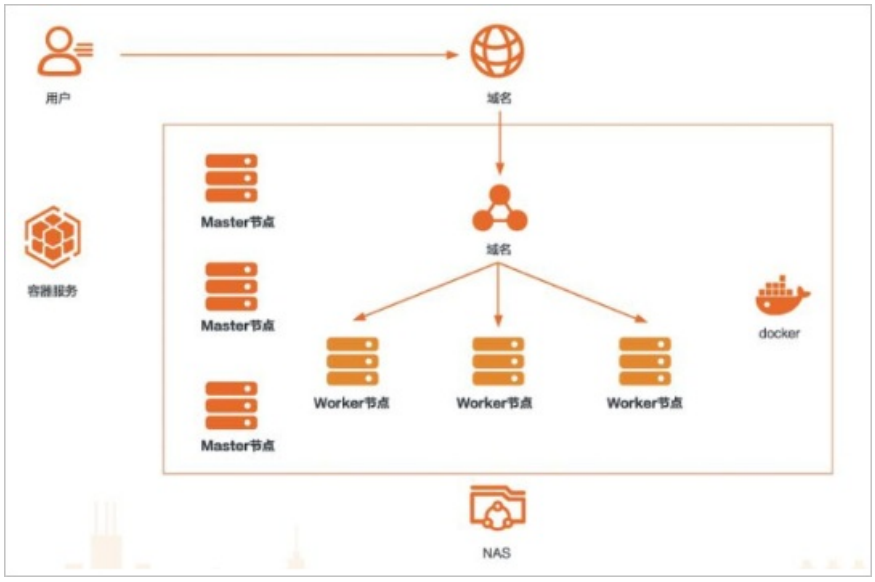
场景描述

本方案适用于使用GPU进行AI在线推理的场景。在推理之前，模型已经训练完成。比如，刷脸支付的时候，就是推理的一个过程。另外，图像分类、目标检测、语音识别、语义分析等返回结果的过程都是使用GPU进行AI在线推理的场景。

解决问题

- 使用GPU云服务器搭建推理环境。
- 使用容器服务Kubernetes版构建推理环境。
- 使用NAS存储模型数据。
- 使用飞天AI加速推理工具加速推理。

架构图



参考链接

有关基于弹性计算的AI推理的详情，请参见基于弹性计算的AI推理。

17.3. ACK集群实现弹性裸金属AI训练

本文介绍弹性裸金属AI训练的场景描述、解决问题、架构图及操作参考链接。

场景描述

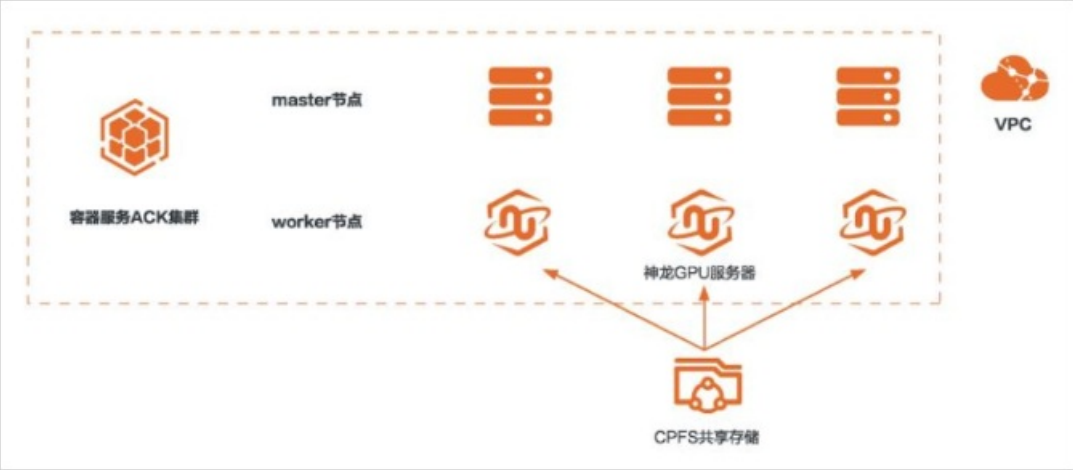
本方案适用于AI图片训练场景，尤其是对性能要求苛刻，业务交付紧迫的场景。例如自动驾驶的模型训练（图片）等AI模型训练的场景。 本方案使用了SCC超级计算集群，采用弹性裸金属GPU服务器、并行文件系统CPFS、RDMA网络、阿里云容器服务Kubernetes版和飞天AI加速训练工具，提供性能稳定的训练环境，保障业务能力。

解决问题

- 搭建AI图片训练基础环境。
- 使用CPFS存储训练数据。

- 使用飞天AI加速训练工具加速训练。
- 使用Arena一键提交作业。

架构图



参考链接

有关弹性裸金属AI训练的详情，请参见弹性裸金属AI训练。

17.4. ACK集群实现GPU AI模型训练

本文介绍GPU AI模型训练的场景描述、解决问题、架构图及操作参考链接。

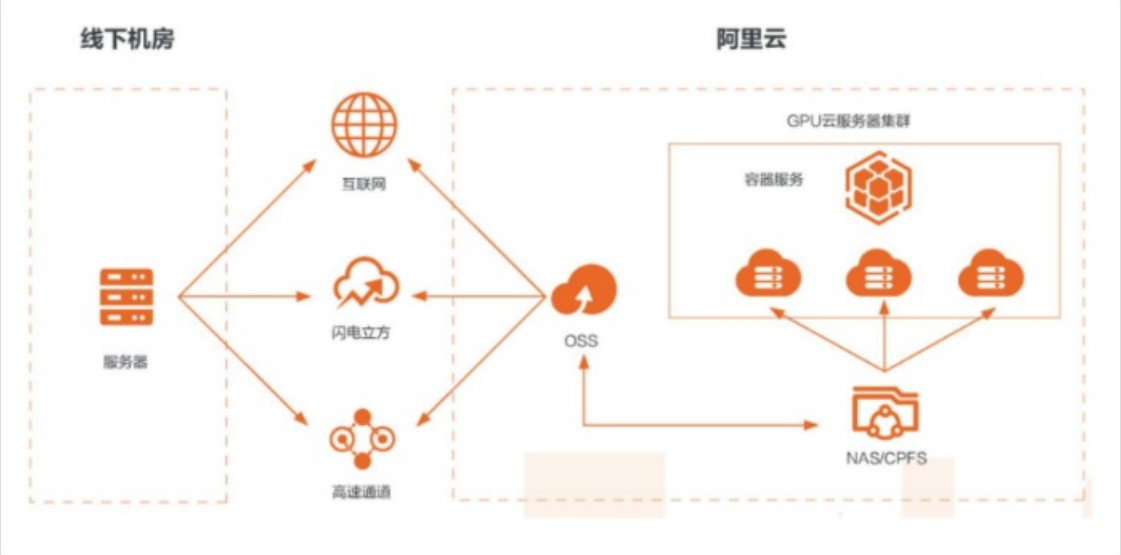
场景描述

本方案适用于AI图片训练场景，使用CPFS和NAS作为共享存储，利用容器服务Kubernetes版管理GPU云服务器集群进行图片AI训练。

解决问题

- 搭建AI图片训练基础环境。
- 使用CPFS存储训练数据。
- 使用飞天AI加速训练工具加速训练。
- 使用Arena一键提交作业。

架构图



参考链接

有关GPU AI模型训练的详情，请参见GPU AI模型训练。

18.云原生架构下日志服务数据预处理

本文介绍云原生架构下日志服务数据预处理的场景描述、方案优势、解决问题、架构图及操作参考链接。

场景描述

某家国际教育机构，提供在线教育服务，其用户主要分布在中美两地，该企业拥抱云计算，在架构设计上也全面采用了云原生服务，本实践将重点聚焦以下几个典型场景：

- 跨地域/跨账号数据汇集。
- 数据内容富化（join维表）。
- 数据投递/归档、入湖分析。
- 统一采集，按业务分发分析。
- 数据监控与智能告警。

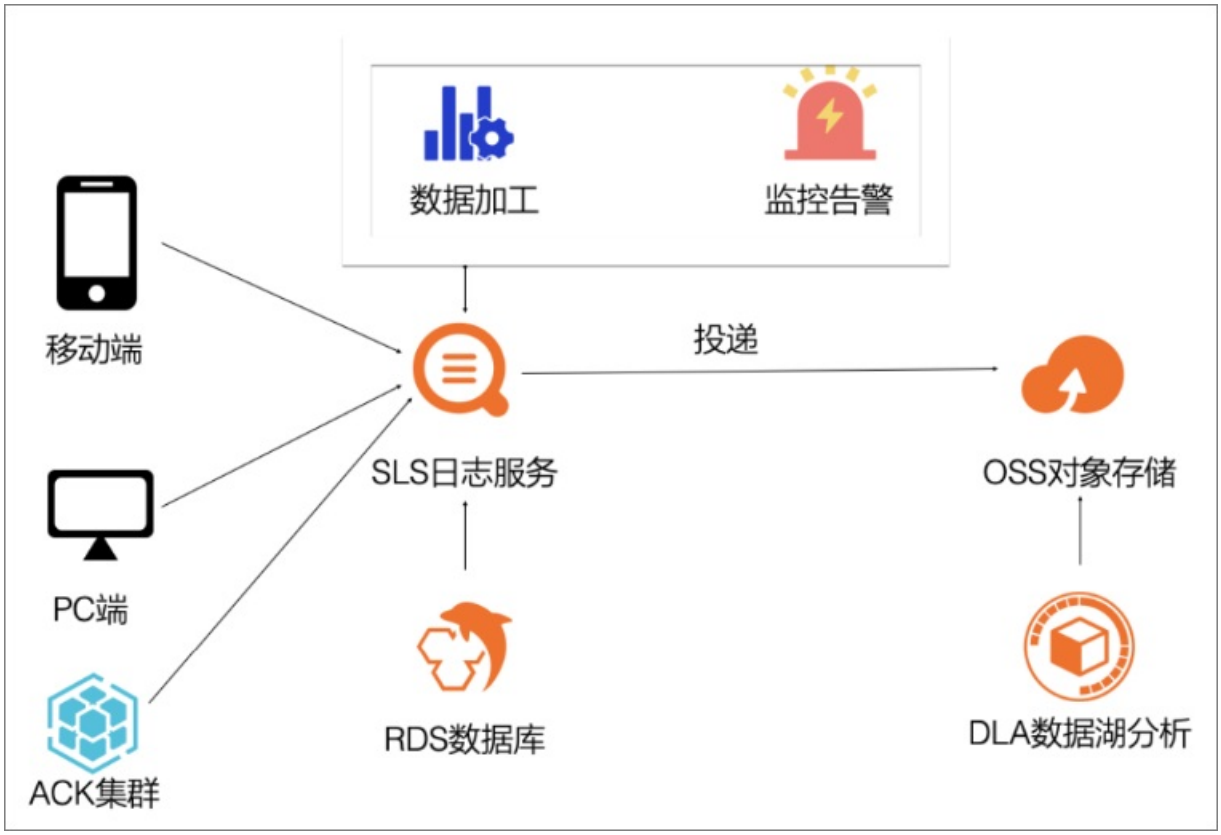
方案优势

- 数据规整：对混乱格式的日志进行字段提取、格式转换，获取结构化数据以支持后续的流处理、数仓计算。
- 数据富化：日志（例如业务订单）与维表（例如用户信息MySQL表）进行字段join，为日志添加更多维度信息供分析。
- 数据分发/汇聚：将全量日志按转发规则分别提取到多个下游存储供不同业务使用。
- 监控告警：监控大规模数据，触发告警，支持机器学习算法（例如，智能巡检），通过告警策略配置，对触发的告警进行降噪处理（去重、静默、合并、抑制、路由分派等），然后分派给特定通知渠道（短信、邮件等）。

解决问题

如何对云上架构方案的成本、部署、运维、 回收进行全生命周期的管理。

架构图



参考链接

关于云原生架构下日志服务数据预处理的详情，请参见云原生架构下日志服务数据预处理。