天翼云·边缘容器集群

用户使用指南

天翼云科技有限公司

目 录

1 产品介绍

1.1 什么是边缘容器集群

边缘容器集群（ECK，Edge cloud Kubernetes）基于智能边缘云 ECX 节点资源，通过泛在异构资源纳管能力将算力延伸到任意需要的地方，以提供就近的高性能可伸缩的 Kubernetes 容器服务。通过 ECK，客户可快速创建云边一体化容器集群，轻松实现无处不在的应用部署和节点管理。

产品形态

边缘容器集群 ECK 提供标准版、云边协同版 2 种类型集群：

对比项	标准版	云边协同版
主要特点	标准的 Kubernetes 集群， Master 和 Worker 节点开在同一个区域	支持云边协同的 Kubernetes 集群，是基于 Kubernetes 增强的云边一体化集群。有如下特性： 支持管理跨区域的边缘节点； 支持单一控制平面管理； 支持边缘节点离线自治； 支持 Service 流量闭环； 支持边缘节点池负载均衡。

产品架构

核心功能

集群管理：一键创建云边一体化的 Kubernetes 集群，支持混合集群。

节点管理：支持购买新节点和纳管已有节点，支持多样异构资源纳管。

节点池管理：借助节点池批量管理节点，支持创建跨区域和不同规格的节点池。

应用管理：将容器应用快速部署到边缘节点，对应用进行全生命周期管理。

弹性伸缩：支持节点和工作负载维度的弹性伸缩，可手动和自动弹性伸缩。

调度配置：可配置跨区域的全局算力调度策略及节点和 Pod 层级的调度策略。

1.2 产品优势

多节点泛覆盖

依托天翼云 900+边缘节点，节点下沉至区县，资源覆盖广，离终端更近。

混合资源纳管

提供多样性算力统一纳管能力，边缘泛在异构资源快速接入，为云上云下多种资源共存的复

杂业务提供统一管理能力。

边缘节点自治

基于原生 Kubernetes 构建云边协同能力，在弱网络环境下提供边缘节点自治能力。

灵活调度引擎

与算力调度引擎无缝结合，通过配置即可快速实现一个属于自己的算力调度系统，轻松满足多样化的调度需求。

1.3 产品功能

集群管理

一键创建云边一体化的 Kubernetes 集群，支持混合集群。

节点管理

支持购买新节点和纳管已有节点，支持多样异构资源纳管。

节点池管理

借助节点池批量管理节点，支持创建跨区域和不同规格的节点池。

应用管理

将容器应用快速部署到边缘节点，对应用进行全生命周期管理。

弹性伸缩

支持节点和工作负载维度的弹性伸缩，可手动和自动弹性伸缩。

调度配置

可配置跨区域的全局算力调度策略及节点和 Pod 层级的调度策略。

1.4 应用场景

1.4.1 混合资源管理

适用场景

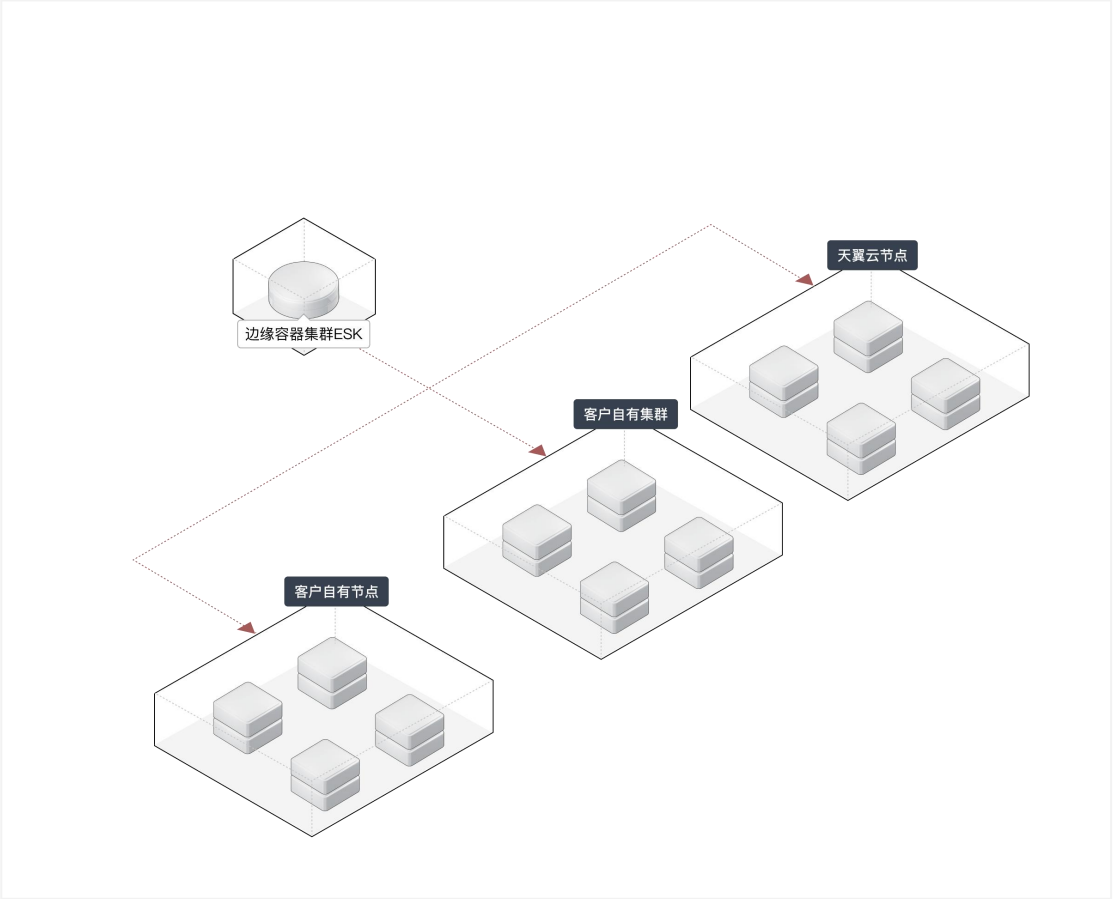
企业云上云下业务并存，希望云上云下混合资源统一管理，统一编排调度，统一管理界面管理。

解决方案

基于 ECK 快捷创建云上集群和节点，同时支持纳管用户自有节点或集群，统一管理云上云下资源及应用。

方案优势

实现算力资源的统一管理，提供一致的上云体验；降低上云复杂度，业务轻松上云，降低运营成本；支持纳管用户自有节点，最大合理化利用资源。



1.4.2 云边端协同

适用场景

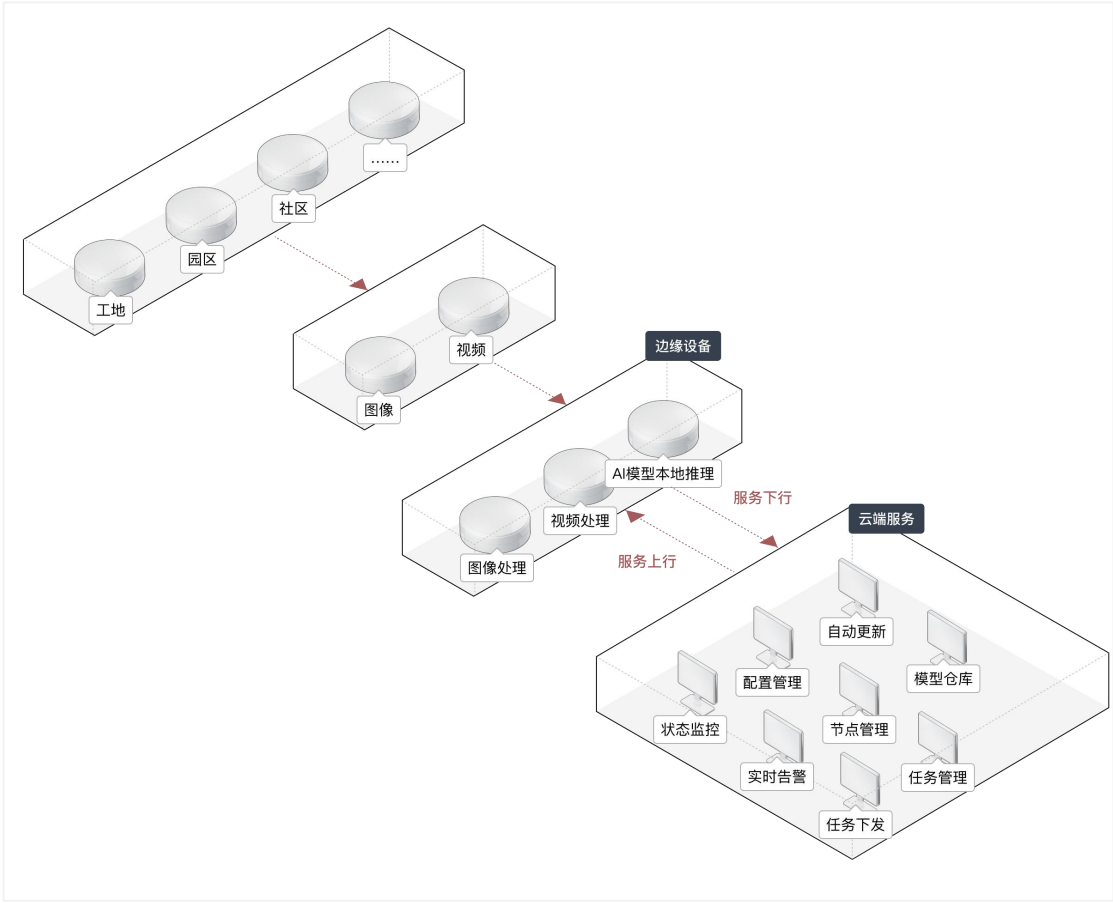
适用于需将算力下沉到边缘或现场、云边端协同、统一管理的场景，如监控视频 AI 分析、工业现场、智能楼宇等。

解决方案

客户现场部署天翼云的边缘盒子，与摄像头、传感器等现场设备连接，并对现场数据进行预分析，通过 ECK 纳管边缘盒子实现云端管理、建模、数据存储等与现场实时决策相结合。

方案优势

边缘盒子预置算法预置模型丰富；使用已有现场设备实现与云端协同。



1.4.3 企业数字化升级

适用场景

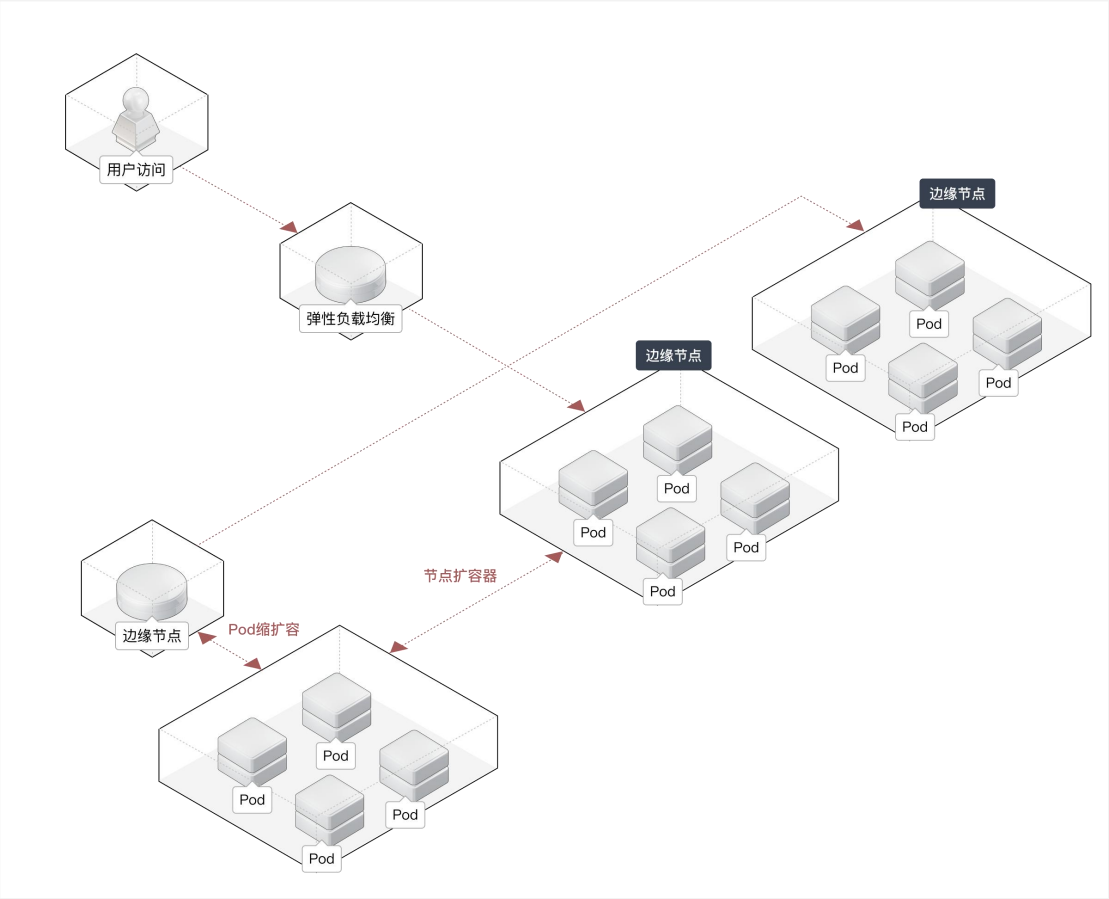
传统应用架构复杂，难以扩容更新，通过云化、容器化使系统更灵活，轻松应对需求变化。

解决方案

使用天翼云边缘容器集群 ECK，快速实现容器化服务的自动化部署、结合 ECK 的弹性伸缩能力可以随需扩缩容业务。

方案优势

通过将原应用拆分成多个松耦合的容器微服务，更便于维护、扩容和升级。



1.4.4 自建 CDN

适用场景

适用于快速构建和部署自建 CDN 节点，对边缘节点进行统一部署、升级、管理和运维的需求。

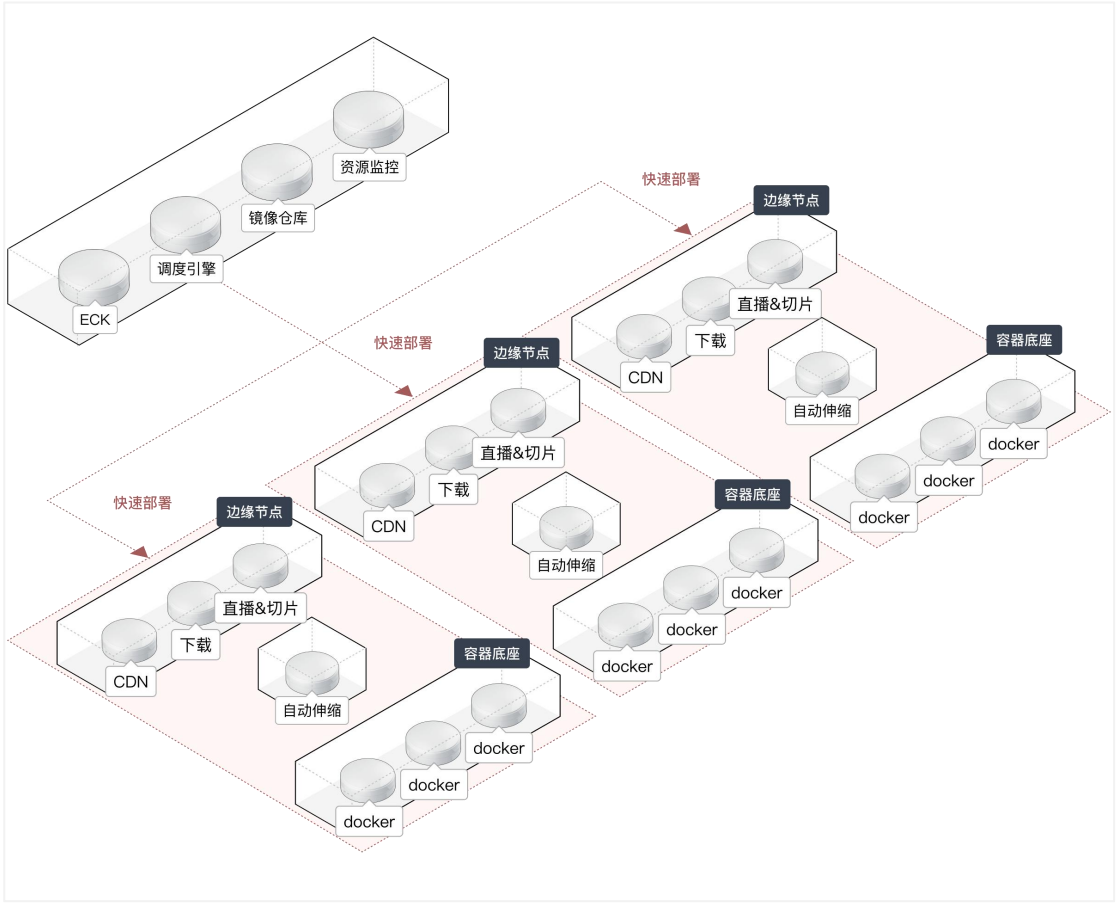
解决方案

利用天翼云的边缘容器集群 ECK 和边缘节点，可快速构建和部署自建 CDN 节点，通过 ECK 管理平台自动化部署应用至业务所需的区域。

方案优势：

边缘节点分布广泛，保障用户就近接入；按需弹性使用，更节约成本；结合算力调度引擎，

通过配置策略即可实现调度系统。



1.5 术语解释

统一身份认证 IAM

统一身份认证（Identity and Access Management，简称 IAM）服务，是提供用户权限管理的基础服务。

IAM 工作区

工作区是 IAM 进行权限管理的基本对象，你可以在工作区添加你的业务，然后邀请成员加入工作区，并赋予他们相应的业务角色，以实现成员的业务权限管理。

Kubernetes

Kubernetes 是一个开源平台，具有可移植性和可扩展性，用于管理容器化的工作负载和服务，简化了声明式配置和自动化。

集群（Cluster）

集群指容器运行所需要的云资源组合，关联了若干服务器节点、负载均衡、专有网络等云资源。ECK 目前支持的集群类型有 专有版集群；专有版集群 Master 节点由客户自行维护。

节点（Node）

每一个节点对应一台服务器，用于部署和管理容器。节点上运行着 Agent 代理程序（kubelet），用于管理节点上运行的容器实例。集群中的节点数量可以伸缩。

节点池（NodePool）

节点池是集群中全都具有相同配置的一组节点，节点池可以包含一个或多个节点。

管理节点（Master Node）

管理节点是 Kubernetes 集群的管理者，运行着的服务包括 kube-apiserver、kube-scheduler、kube-controller-manager、etcd 组件，和容器网络相关的组件。

工作节点（Worker Node）

工作节点是 Kubernetes 集群中承担工作负载的节点，可以是虚拟机也可以是物理机。工作节点承担实际的 Pod 调度以及与管理节点的通信等。一个工作节点上的服务包括 Docker 运行时环境、kubelet、Kube-Proxy 以及其它一些可选的组件。

专有网络 VPC

虚拟私有云是通过逻辑方式进行网络隔离，提供安全、隔离的网络环境。您可以在 VPC 中定义与传统网络无差别的虚拟网络，同时提供弹性 IP、安全组等高级网络服务。

安全组

安全组是一种虚拟防火墙，具备状态检测和数据包过滤能力，用于在云端划分安全域。安全组是一个逻辑上的分组，由同一地域内具有相同安全保护需求并相互信任的实例组成。

容器组（Pod）

Pod 是 Kubernetes 部署应用或服务的最小的基本单位。一个 Pod 封装多个应用容器（也可以只有一个容器）、存储资源、一个独立的网络 IP 以及管理控制容器运行方式的策略选项。

容器（Container）

一个通过 Docker 镜像创建的运行实例，一个节点可运行多个容器。容器的实质是进程，但与直接在宿主执行的进程不同，容器进程运行于属于自己的独立的命名空间。

工作负载（Workload）

工作负载是在 Kubernetes 上运行的应用程序。工作负载包括以下几种类型：

状态工作负载（Deployment）：即 kubernetes 中的“Deployment”，无状态工作负载支持弹性伸缩与滚动升级，适用于实例完全独立、功能相同的场景，如：nginx、wordpress 等。

有状态工作负载（StatefulSet）：即 kubernetes 中的“StatefulSet”，有状态工作负载支持实例有序部署和删除，支持持久化存储，适用于实例间存在互访的场景，如 ETCD、mysql-HA 等。

守护进程集（DaemonSet）：即 kubernetes 中的“DaemonSet”，守护进程集确保全

部（或者某些）节点都运行一个 Pod 实例，支持实例动态添加到新节点，适用于实例在每个节点上都需要运行的场景，如 ceph、fluentd、Prometheus Node Exporter 等。

任务（Job）：即 kubernetes 中的“Job”，普通任务是一次性运行的短任务，部署完成后即可执行。使用场景为在创建工作负载前，执行普通任务，将镜像上传至镜像仓库。

定时任务（CronJob）：即 kubernetes 中的“CronJob”，定时任务是按照指定时间周期运行的短任务。使用场景为在某个固定时间点，为所有运行中的节点做时间同步。

镜像（Image）

容器镜像是容器应用打包的标准格式，封装了应用程序及其所有软件依赖的二进制数据。在部署容器化应用时可以指定镜像，镜像可以来自于 Docker Hub，容器镜像服务，或者用户的私有镜像仓库。

镜像仓库（Image Registry）

容器镜像仓库是一种存储库，用于存储 Kubernetes 和基于容器应用开发的容器镜像。

命名空间（Namespace）

命名空间为 Kubernetes 集群提供虚拟的隔离作用。Kubernetes 集群初始有 3 个命名空间，分别是默认命名空间 default、系统命名空间 kube-system 和 kube-public，除此以外，管理员可以创建新的命名空间以满足需求。

服务（Service）

Service 是将运行在一组 Pods 上的应用程序公开为网络服务的抽象方法，每一个服务后面都有很多对应的容器来提供支持，通过 Kube-Proxy 的 ports 和服务 selector 决定服务请求传递给后端的容器，对外表现为一个单一访问接口。

路由（Ingress）

Ingress 是为进入集群的请求提供路由规则的集合，可以给 service 提供集群外部访问的 URL、负载均衡、SSL 终止、HTTP 路由等。

配置项（ConfigMap）

ConfigMap 用于保存配置数据的键值对，可以用来保存单个属性，也可以用来保存配置文件。ConfigMap 跟 secret 很类似，但它可以更方便地处理不包含敏感信息的字符串。

保密字典（Secret）

保密字典用于存储在 Kubernetes 集群中使用一些敏感的配置，例如密码、证书等信息。

存储卷（Persistent Volume，PV）

PV 是集群内的存储资源，类似节点是集群资源一样。PV 独立于 Pod 的生命周期，可根据不同的 StorageClass 类型创建不同类型的 PV。

存储卷声明（Persistent Volume Claim，PVC）

PVC 是资源的使用者。类似 Pod 消耗节点资源一样，而 PVC 消耗 PV 资源。

弹性伸缩（HPA）

Horizontal Pod Autoscaling，简称 HPA，是 Kubernetes 中实现 POD 水平自动伸缩的功能。Kubernetes 集群可以通过 Replication Controller 的 scale 机制完成服务的扩容或缩容，实现具有伸缩性的服务。

标签（Label）

Labels 的实质是附着在资源对象上的一系列 Key/Value 键值对，用于指定对用户有意义的

对象的属性，标签对内核系统是没有直接意义的。标签可以在创建一个对象的时候直接赋予，也可以在后期随时修改，每一个对象可以拥有多个标签，但 key 值必须唯一。

污点（Taints）

污点和节点亲和性相反，它使节点能够排斥一类特定的 Pod。

容忍（Tolerations）

应用于 Pod 上，允许（但并不要求）Pod 调度到带有与之匹配的污点的节点上。

节点亲和性（nodeAffinity）

节点亲和性指通过 Worker 节点的 Label 标签控制 Pod 部署在特定的节点上。

应用亲和性（podAffinity）

指定工作负载部署在相同节点。通过应用亲和性调度，将其部署到同一节点中，容器间通信就近路由，减少网络消耗。

应用反亲和性（podAntiAffinity）

指定工作负载部署在不同节点。同个工作负载的多个实例反亲和部署，减少宕机影响；互相干扰的应用反亲和部署，避免干扰，以提高服务本身的稳定性。

2 购买指南

2.1 计费说明

边缘容器集群 ECK 专有版本身不收取费用，但在使用过程中会创建相关智能边缘云产品资源，您需要为您使用的这些资源付费，如虚拟机、云硬盘、带宽等，智能边缘云产品资源详细价格请参见相应云产品资源计费

2.2 计费方式

ECK 目前支持按需计费模式：

按需计费：一种先使用后付费的方式，从“开通”开启计费到“删除”结束计费，按实际购买时长计费。这种购买方式比较灵活，您可以按需取用资源，随时开启和释放，无需提前购买大量资源。

2.3 计费方式变更

目前 ECK 只支持按需计费模式，暂不支持计费方式变更。

2.4 到期欠费

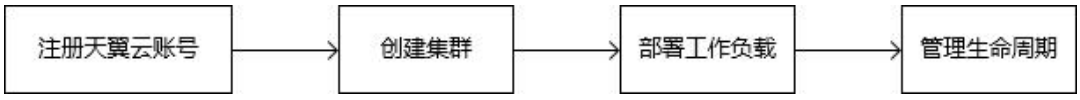
用户欠费后，ECK 集群创建的智能边缘云实例占用的计算资源将进入 1 小时的保留期，1 小时内若用户未充值则会进入冻结期。进入保留期时，资源可以正常使用并正常计费。进入冻结期时，虚拟机将不能使用，虚拟机的 CPU、内存、GPU 资源将会被释放，但系统盘和挂载的数据盘将会保留并按原价继续收费；边缘存储实例将不能使用，实例资源及数据将会保留并按原价继续收费；网络将无法访问。冻结期内，若用户仍未充值，冻结期结束时系统

会释放虚拟机、边缘存储实例、弹性 IP、NAT 网关、负载均衡等资源，VPC、专线、路由表、SSL 证书等资源或配置会被保留。

3 快速入门

3.1 入门指引

本节介绍边缘容器集群 ECK 的快速使用流程，帮助您快速上手边缘容器集群 ECK，ECK 的快速使用流程如下图所示:



1. **注册天翼云帐号**

注册一个天翼云帐号并完成实名认证。

2. **创建集群**

登录 ECK 控制台，创建 Kubernetes 集群及节点。

3. **创建工作负载（应用）**

使用镜像创建工作负载。

4. **管理生命周期**

查看部署后工作负载的状态和日志信息，对工作负载进行相应的升级、伸缩和监控等。

3.2 准备工作

在使用边缘容器集群 ECK 前，您需要完成本文中的准备工作。

注册天翼云账号并实名认证

如果您已有一个天翼云帐户，请跳到下一个任务。如果您还没有天翼云帐户，请参考以下步骤创建。

1. 打开 https://www.ctyun.cn/，单击“免费注册”。
2. 根据提示信息完成注册。详细请参考注册天翼云账号
3. 参考实名认证完成个人或企业帐号实名认证。

获取资源权限

由于 ECK 在运行中对智能边缘云服务资源（计算、存储和网络等）存在依赖关系，因此当您首次登录 ECK 控制台时 ,ECK 将自动请求获取智能边缘云资源权限 ,需您同意授权后 ECK 方可正常提供服务。您若没开通边缘智能云服务，授权过程需要您开通智能边缘云服务

创建虚拟私有云(可选)

VPC 为 ECK 集群提供一个隔离的、用户自主配置和管理的虚拟网络环境。创建首个集群前，您必须先确保要创建集群的区域已存在虚拟私有云，否则无法创建集群。

登录智能边缘云 ECX 专属网络控制台 创建 VPC，方法请参见创建虚拟私有云。若您已有虚拟私有云，可重复使用，无需重复创建。

注意：VPC 所在区域需与集群区域一致

创建公网 IP(可选)

ECK 专有版默认会为 API Server 创建一个外网 SLB 实例，并需绑定一个公网 IP，这样才能通过外网访问集群 API Server。创建集群前，您必须先确保要创建集群的区域已存在未被使用的公网 IP，否则无法创建集群。

登录智能边缘云 ECX 专属网络控制台 创建公网 IP，方法请参见 创建弹性公网 IP 。若您已有公网 IP，可重复使用，无需重复创建。

注意：公网 IP 所在区域需与集群区域一致

3.3 快速创建边缘容器集群

本节介绍使用控制台采用默认配置如何快速创建 Kubernetes 集群

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理。**
3. 在**集群列表**页面中，单击页面左上角的**创建专有集群。**
4. 完成专有版集群配置。

配置以下集群相关参数，未说明参数保留默认设置即可

配置项	描述
集群名称	填写集群的名称。 **说明** 集群名称应包含 1~63 个字符，可包含数字、汉字、英文字符、短划线（-）或下划线（_），且不能以下划线（_）开头。
虚拟私有云	选择集群所在的虚拟私有云 VPC，选择提前准备好的 VPC
子网	选择控制节点（即集群 Master 节点）所在子网，选择提前准备好的子网
Pod 网络 CIDR	设置容器使用的网段，使用推荐配置
Service CIDR	同一集群下容器互相访问时使用的 Service 资源的网段，使用推荐配置
API Server 访问	默认为 API Server 创建一个 SLB 实例，并为 SLB 绑定一个弹性公网 IP，选择提前准备好的公网 IP

5. 单击下一步：**Master 配置**，配置以下 Master 节点相关参数，未说明参数保留默认设置即可

配置项	描述
节点规格	选择 Master 节点的实例规格

操作系统	Master 节点的操作系统
系统盘	配置 Master 节点系统盘的类型和大小 ,如果选择的实例规格里已经包含系统盘，则不需要配置
登录方式	目前支持设置密码登录 **登录密码**：设置节点的登录密码。 **确认密码**：确认设置的节点登录密码。 **说明** 密码为 8~26 位，大、小写字母、数字、特殊字符 4 种字符组合

6. 单击**下一步：节点池配置**，配置以下 Worker 节点相关参数，未说明参数保留默认设置即可

配置项	**描述**
节点池名称	填写节点池名称
节点规格	选择 Worker 节点的实例规格
操作系统	Worker 节点的操作系统，至少需要 1 个节点
系统盘	配置 Worker 节点系统盘的类型和大小 ,如果选择的实例规格里已经包含系统盘，则不需要配置
登录方式	目前支持设置密码登录 **登录密码**：设置节点的登录密码。 **确认密码**：确认设置的节点登录密码。 **说明** 密码为 8~26 位，大、小写字母、数字、特殊字符 4 种字符组合

7. 单击下一步：**组件配置**，使用默认配置。

配置项	**描述**
监控插件	在节点上安装监控插件和使用 Prometheus 监控服务

8. 单击下一步：**确认配置。**

9. 配置确认界面会做依赖检查，检查通过，单击**创建集群** 开始创建集群

检查不通过，请按说明进行调整，调整完可重新检查，通过后可创建集群。

集群的创建时间一般约为 10-20 分钟。

3.4 使用镜像快速创建无状态 Deployment

1. 登录边缘容器集群控制台。

2. 在控制台左侧导航栏中，单击**集群管理。**

3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情。**

4. 在控制台左侧导航栏中，单击**工作负载>无状态。**

5. 在**无状态**负载列表，单击左上角的**创建无状态负载。**

6. 在应用配置页面，设置应用的配置信息

完成应用基本信息配置，未说明参数保留默认设置即可

配置项	**描述**
应用名称	设置应用的名称，名称为 1~63 个字符，支持小写字母、数字、 "-"以及"."等字符

完成容器配置，未说明参数保留默认设置即可

配置项	**描述**
容器镜像	镜像类型：可以选择公有镜像或私有镜像 公有镜像：填入需要使用的公有镜像地址 私有镜像：选择上传到镜像仓库的私有镜像 支持以下三种镜像拉取策略（imagePullPolicy）： **优先使用本地镜像（IfNotPresent）**：如果本地有该镜像（之前拉取

	过该镜像至宿主机中），则使用本地镜像，本地不存在时拉取镜像。 **总是拉取镜像（Always）**：表示每次部署或扩容都会从容器镜像服务重新拉取镜像，而不会从本地拉取镜像。 **仅使用本地镜像（Never）**：仅使用本地镜像。 单击**设置镜像密钥**，您可以实现免密拉取镜像

7. 点击**创建**，即可开始创建无状态 Deployment。

4 操作指南

4.1 集群管理

4.1.1 创建专有版集群

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击页面左上角的**创建专有集群**。
4. 完成专有版集群配置。

完成集群配置

配置项	描述
ECK 版本	**ECK 标准版**：标准 kubernetes 版本，Master 和 Worker 节点在同一个区域，节点池区域不可选，默认与 Master 节点同区域 **ECK Octopus 版**：ECK 专有集群集群边缘版，针对边缘计算场景推出的云边一体化方案，采用非侵入方式增强，提供云边协同、缘边缘自治等能力，可在不同的区域创建创建节点池和节点，实现跨区域资源的管理和调度
计费模式	边缘容器集群(ECK 专有版)目前支持 按量计费
区域集群	选择集群 master 节点所在的地域
描述	根据需要填写描述
标签	为集群绑定标签，最多绑定 20 个标签
容器运行时	支持 Containerd 和 Docker。

kube-proxy 代理模式	支持 iptables 和 IPVS 两种模式。 iptables：成熟稳定的 kube-proxy 代理模式，Kubernetes Service 的服务发现和负载均衡使用 iptables 规则配置，但性能一般，受规模影响较大，适用于集群存在少量的 Service。 IPVS：高性能的 kube-proxy 代理模式，Kubernetes Service 的服务发现和负载均衡使用 Linux ipvs 模块进行配置，适用于集群存在大量的 service，对负载均衡有高性能要求的场景
集群名称	填写集群的名称。 **说明** 集群名称应包含 1~63 个字符，可包含数字、汉字、英文字符、短划线（-）或下划线（_），且不能以下划线（_）开头。
版本信息	当前 ECK 支持的 Kubernetes 版本
虚拟私有云	选择集群所在的虚拟私有云 VPC，如没有可选项可以单击下面“VPC 管理”到专属网络控制台创建。集群创建后 VPC 不可修改
子网	选择控制节点（即集群 Master 节点）所在子网，如没有可选项可以单击下面“VPC 管理”到专属网络控制台创建 VPC 和子网。创建后控制节点子网不可修改。
节点 IP 数量	指可分配给一个节点的 IP 数量
Pod 网络 CIDR	设置容器使用的网段，网段不能和 VPC 及 VPC 已有 Kubernetes 集群使用的网段重复，Service 地址段也不能和 Pod 地址段重复，创建成功后不能修改
Service CIDR	同一集群下容器互相访问时使用的 Service 资源的网段，决定了 Service 资源的上限。网段不能与 VPC 及 VPC 内已有 Kubernetes 集群使用的网段重复，Service 地址段也不能和 Pod 地址段重复，创建后不可修改

自定义证书 SAN	在集群 API Server 服务端证书的 SAN（Subject Alternative Name）字段中添加自定义的 IP 或域名，以实现对客户端的访问控制。
配置 SNAT	创建集群时，默认为专有网络配置 SNAT，并自动配置 SNAT 规则，使用集群具备公网访问能力
APIServer 访问	默认为 API Server 创建一个 SLB 实例，并为 SLB 绑定一个弹性公网 IP，这样 Master 节点的 6443 端口（对应 API Server）暴露出来，可以在外网通过 kubeconfig 连接并操作集群。可通过 可以免费申请 IP 和使用已有 IP 方式提供弹性 IP 可以免费申请 IP：创建新的 IP，设置绑定的弹性 IP 的带宽上限，如果 IP 资源不足则会导致创建集群失败 使用已有 IP：可选用已经创建的 IP
集群删除保护	设置是否启用集群删除保护。为防止通过控制台或 API 误释放集群。

5. 单击下一步：Master 配置，完成 Master 节点配置。

配置项	**描述**
Master 实例数量	设置您所需的 Master 节点数量。目前支持创建 1 个或者 3 个 Master 节点。
节点规格	选择 Master 节点的实例规格
操作系统	Master 节点的操作系统
系统盘	配置 Master 节点系统盘的类型和大小，如果选择的实例规格里已经包含系统盘，则不需要配置
数据盘	可选择给 Master 挂载数据盘系统盘，最多挂载 100 块

登录方式	目前支持设置密码登录 **登录密码**：设置节点的登录密码。 **确认密码**：确认设置的节点登录密码。 **说明** 密码为 8~26 位，大、小写字母、数字、特殊字符 4 种字符组合

6. 单击**下一步：节点池配置**，完成 Worker 节点配置。

配置项	**描述**
节点池名称	填写节点池名称
节点规格	选择 Worker 节点的实例规格
操作系统	Worker 节点的操作系统，至少需要 1 个节点
实例数量	设置您所需的 Worker 节点数量
系统盘	配置 Worker 节点系统盘的类型和大小，如果选择的实例规格里已经包含系统盘，则不需要配置
数据盘	可选择给 Worker 挂载数据盘系统盘，最多挂载 100 块
登录方式	目前支持设置密码登录 **登录密码**：设置节点的登录密码。 **确认密码**：确认设置的节点登录密码。 **说明** 密码为 8~26 位，大、小写字母、数字、特殊字符 4 种字符组合

7. 单击下一步：组件配置，完成组件配置。

配置项	**描述**
监控插件	在节点上安装监控插件和使用 Prometheus 监控服务

8. 单击下一步：确认配置。

9. 配置确认界面会做依赖检查，检查通过，单击**创建集群** 开始创建集群。

检查不通过，请按说明进行调整，调整完可重新检查，通过后可创建集群。

4.1.2 查看集群信息

您可以查看集群的基本信息、连接信息、集群日志。

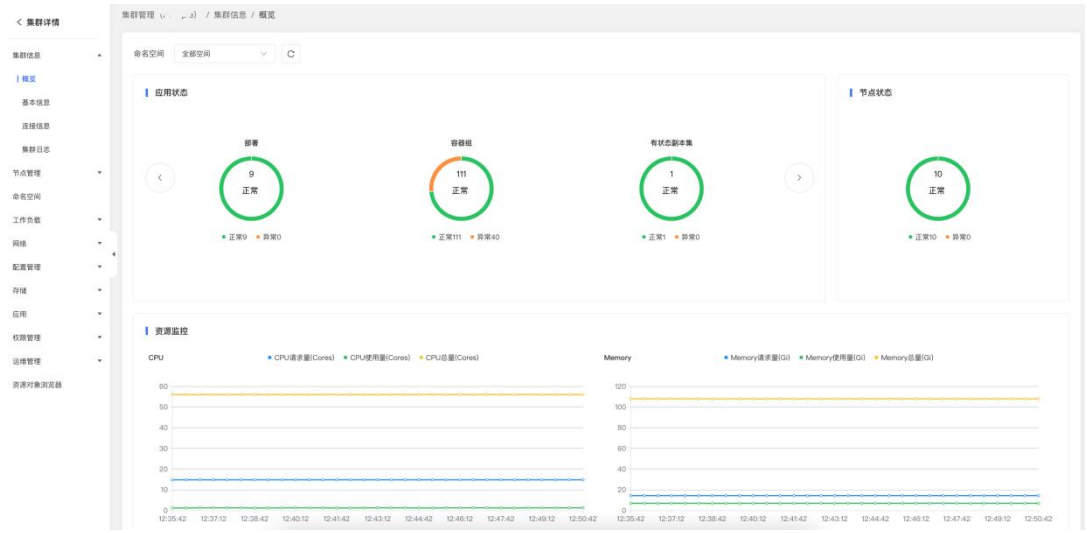
1、登录边缘容器集群控制台。

2、在控制台左侧导航栏中，单击**集群管理**。

3、在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

查看基本信息

单击**概览**，查看集群的应用状态、节点状态及资源监控(CPU、内存及 GPU 的使用情况)。



查看基本信息

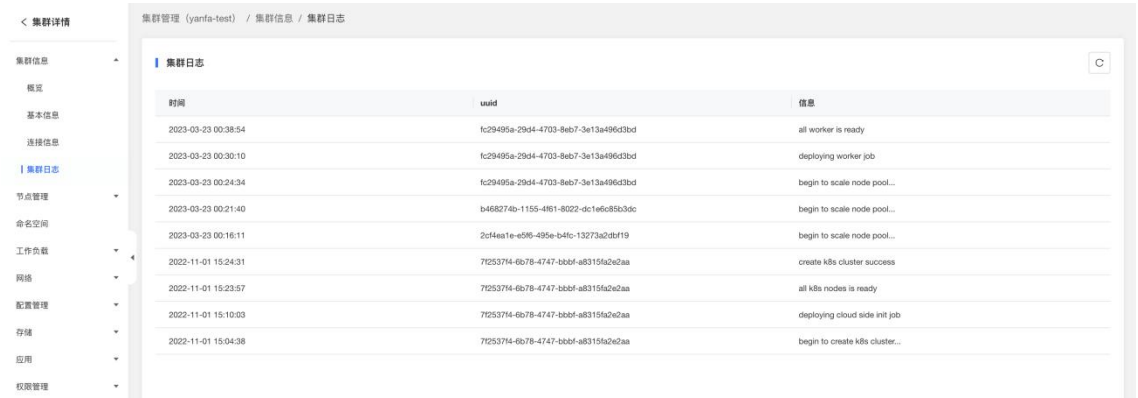
单击**基本信息**，查看集群 ID、地域、API Server 连接端点以及其他网络信息。

查看连接信息

单击**连接信息**，您可以获取公网和内网环境下 KubeConfig 文件的配置内容，用于配置通过 Kubectl 客户端访问集群。

查看集群日志

单击**集群日志**页签，您可以查看集群日志。



4.1.3 连接集群

4.1.3.1 通过 kubectl 工具连接集群

除了通过控制台来管理集群之外，您还可以通过 Kubernetes 命令行工具 kubectl 来管理集群以及应用。本文介绍如何通过 kubectl 客户端连接 ECK 集群。

安装 kubectl 工具

1）根据需要，确定安装 kubectl 的客户端机器

通过公网连接，可以选择公网中的任一机器作为客户端安装 kubectl

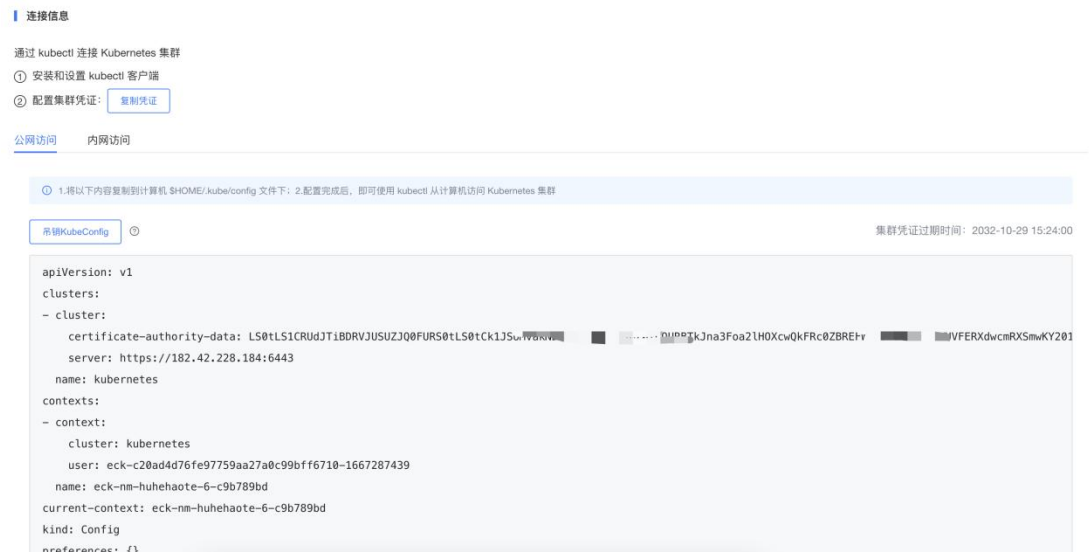
通过私网连接，安装 kubectl 客户端机器必须与集群位于同一 VPC

2）下载 kubectl 工具，并安装至对应的客户端机器。

获取集群凭证

ECK 集群提供了两种集群凭证（即 KubeConfig），分别用于公网访问和私网访问。

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在**集群信息**目录，单击**连接信息**子目录，根据需要选择公网或私网访问凭证。单击**复制凭证**可复制凭证



配置集群凭证

kubectl 工具默认会从客户端机器的$HOME/.kube 目录下查找名为 config 的文件，该文件用于存储所要管理集群的访问凭证，kubectl 会根据该配置文件连接至集群。

将复制的集群凭证内容粘贴至$HOME/.kube 目录下的 config 文件中，保存并退出。

说明 如果$HOME/目录下没有.kube 目录和 config 文件，请自行创建。

验证集群连通性

集群凭证配置之后，您可以执行 **kubectl** 命令以验证集群的联通性。以查询命名空间为例，执行以下命令。

kubectl **get namespace**

预期输出：

NAME STATUS AGE

default Active 4h39m

kube-node-lease Active 4h39m

kube-public Active 4h39m

kube-system Active 4h39m

4.1.3.2 通过 SSH 连接集群的 Master 节点

ECK 专有版集群的 master 节点是在 ECX 上构建而成，用户可以通过智能边缘云的网络能力使用 SSH 进行登录。

1.登录边缘容器集群控制台。

2.在控制台左侧导航栏中，单击**集群管理** 。

3.在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情** ，可以看到集群的所在区域和所在 VPC。

4.登录智能边缘云控制台，点击左侧**边缘网络-NAT 网关**菜单，找到指定 VPC 下的 NAT，点击右侧**设置规则**按钮。

5. 点击 **DNAT** 页签，点击**添加 DNAT 规则**按钮，给希望连接的 Master 节点对应的示例添加 DNAT 规则，详情参考创建和管理 DNAT 条目。

6.通过 ssh 工具访问刚刚设置的 NAT 规则中的弹性公网 IP 和公网端口，使用 root 用户名即可登录到 Master 节点。

4.1.3.3 通过 webShell 管理集群

webShell 是命令行管理工具，您可以在浏览器上打开 webShell 命令管理 ECK 集群。本文介绍如何在边缘容器集群 ECK 控制台上利用 webShell 通过 kubectl 管理集群。

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

4. 点击左侧通过 **webShell 管理集群**，打开终端。

5. 在终端中可以使用 kubectl 命令来管理集群。

4.1.3.4 吊销集群的 KubeConfig 凭证

原 KubeConfig 凭证遭到疑似泄露等情况时，通过吊销该集群的 KubeConfig 可有效保障集群的安全。吊销 KubeConfig 后，用户就无法使用原 KuberConfig 连接集群，请谨慎操作。

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在**集群信息**目录，单击**连接信息**子目录，单击**吊销凭证**。

5. 在弹出的对话框中单击**确定**。

吊销原来的 KubeConfig 凭证，并自动为您分配新的 KubeConfig 凭证。

4.1.3.5 控制集群 API Server 的公网访问能力

ECK 提供通过外网 EIP 访问集群 APIServer 的能力。这样您就可以使用对应 CLI 工具使用 kubeconfig 访问集群。为了保护您的集群安全，您可以根据自己的需要，自行配置安全组。ECK 仅仅只能在创建的时候，给集群 APIServer 绑定对应的 EIP。

创建集群时绑定 EIP

你可以通过下列方式进行指定 EIP：

1.登录边缘容器集群控制台。

2.在控制台左侧导航栏中，单击 **集群管理** 。

3.在**集群列表**页面中，单击页面左上角的 **创建专有集群** 。

4.在创建页面，**APIServer 访问**勾选**免费申请 IP** ，创建集群时即为您自动分配 EIP，并且绑定到您的公网 SLB 中。如果您已经存在了对应的 EIP，则可以直接选择**使用已有 IP。**

注意：创建集群的时候，您需要自行检查对应配额是否充足，以免开通失败。

4.1.4 扩容集群

通过控制台，您可以根据实际业务需要对集群的节点池进行扩容。

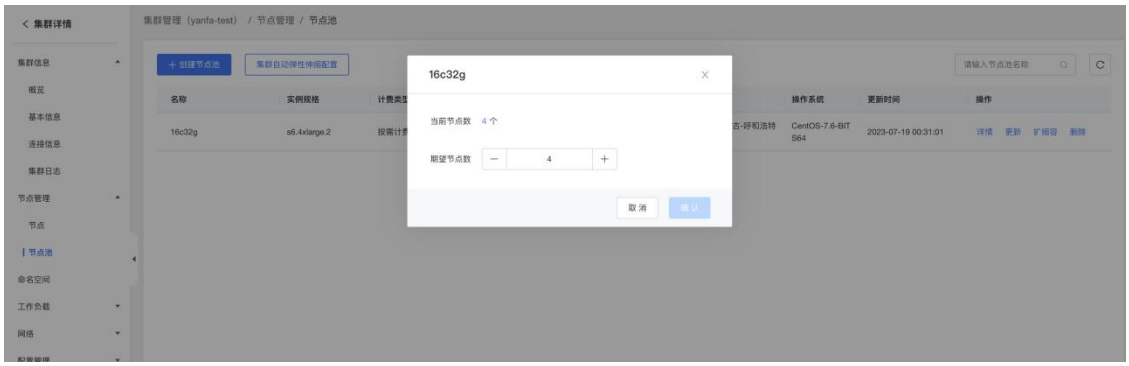
1、登录边缘容器集群控制台。

2、在控制台左侧导航栏中，单击**集群管理**。

3、在集群列表页面，选择目标集群，并在目标集群右侧操作列下，选择**节点池**。

4、在节点池页面，单击目标节点池操作列的**扩缩容**。

5、在期望节点数输入数值，单击**确认**，然后在修改确认的对话框中，单击**确定**。



在节点池页面，如果节点池状态显示扩容中，则说明节点池正在扩容中。扩容完成后，状态显示为已激活。

4.1.5 删除集群

您可以通过控制台删除不再使用的集群。

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群**。
3. 在**集群列表**页面，选择所需的集群并单击右侧的**... > 删除**。

4. 在**删除集群**对话框中，您可选择需要保留的集群资源，确认内容无误后，单击**确定**。

注意：

1）如果集群的删除保护已打开，则不能删除集群，需要先关闭删除保护，可通过以下方式关闭删除保护：在**集群列表**页面，选择所需的集群并单击右侧的**... > 修改集群删除保护状态**

2）集群创建时自动创建的资源(如 NAT 网关、SNAT、SLB、EVM 等)，集群删除时将自动删除这些资源；手动创建的云产品在删除集群时不会被自动释放，集群删除后会继续计费，请您根据需要手动释放相关资源

4.1.6 升级集群

您可以通过控制台升级集群的 kubernetes 版本。

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群**。
3. 在**集群列表**页面，选择所需的集群并单击右侧的**... > 集群升级**。

4. ECK 进行升级检查，有可升级版本和前置检查通过即可进行升级，点击**确认**开始升级。

ECK 将对节点逐个进行升级

4.1.7 最小化集群访问规则

通过添加安全组规则，可以控制集群内节点在出/入方向上的访问。详细的介绍可以参考安全组管理。

入方向

协议	端口	来源	策略	说明
ICMP	全部	0.0.0.0/0	允许	
TCP	10250	集群 Pod 网络地址段	允许	用于访问各个节点上的 kubelet。
TCP	6443	集群公网与内网 slb 地址	允许	用于访问控制平面的 api server。
指定协议	希望被访问的指定端口	希望被访问的来源地址	允许	用于访问集群内部署的组件。

出方向

通常情况下出方向不建议限制，建议配置允许 0.0.0.0/0 的所有端口所有协议。

4.2 节点管理

4.2.1 管理节点标签

您可以通过控制台对节点进行标签管理，包括批量添加节点标签和快速删除节点标签。

批量添加节点标签

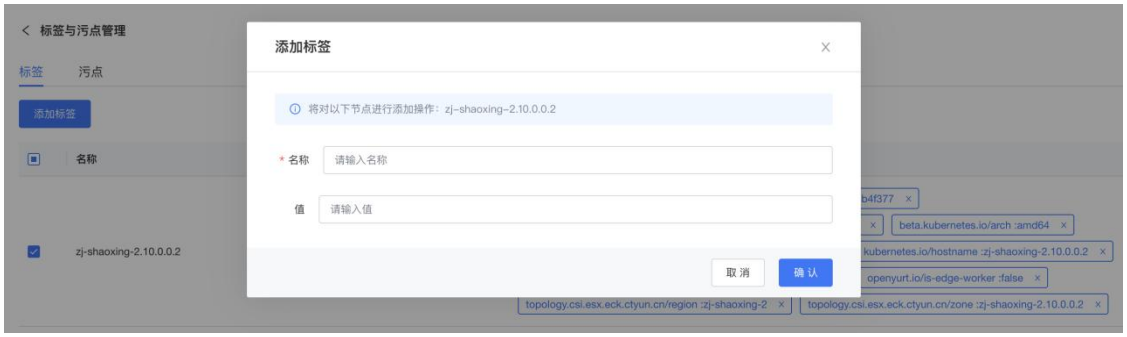
1. 登录边缘容器集群控制台。
2. 进入**标签与污点管理**页面。
   1） 在控制台左侧导航栏中，单击**集群管理**。
   2） 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

3） 在左侧导航栏中选择**节点管理 > 节点**。

4） 在**节点**页面左上角单击**标签与污点管理**。

3. 单击**标签**页签，批量选择节点，然后单击**添加标签**。

4. 在**添加**对话框中，输入标签的**名称**和**值**，然后单击**确定**。



在**标签**页面，可以看到为批量选择的节点添加了相同的标签。

删除节点标签

1. 登录边缘容器集群控制台。

2. 进入**标签与污点管理**页面。

   1） 在控制台左侧导航栏中，单击**集群管理**。

   2） 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

   3） 在左侧导航栏中选择**节点管理 > 节点**。

   4） 在**节点**页面左上角单击**标签与污点管理**。

3. 单击**标签**页签，单击标签的删除图标。

4.2.2 管理节点污点

污点可以使 Pod 排斥一类特定的节点，每个节点上都可以应用一个或多个污点，您可以通过控制台对节点进行污点管理，包括批量添加节点污点和快速删除节点污点

批量添加节点污点

1. 登录边缘容器集群控制台。
2. 进入**标签与污点管理**页面。
   1）在控制台左侧导航栏中，单击**集群管理**。
   2）在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
   3）在左侧导航栏中选择**节点管理 > 节点**。
   4）在**节点**页面左上角单击**标签与污点管理**。
3. 单击**污点**页签，批量选择节点，然后单击**添加污点**。

4. 在**添加**对话框中，输入标签的**名称**、**值**和 **Effect**，

参数名称	参数说明
NoSchedule	如果污点中存在至少一个 Effect 值为 NoSchedule 的污点， 则系统不会将 Pod 分配到该节点。
NoExecute	任何不能忍受这个污点的 Pod 都会被驱逐，任何可以忍受这个污点的 Pod 都不会被驱逐。
PreferNoSchedule	系统会尽量避免将 Pod 调度到存在其不能容忍污点的节点上， 但这不是强制的。

5. 然后单击**确定**。在**污点**页面，可以看到为批量选择的节点添加了相同的污点。

删除节点污点

1. 登录边缘容器集群控制台。
2. 进入**标签与污点管理**页面。
   1）在控制台左侧导航栏中，单击**集群管理**。
   2）在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
   3）在左侧导航栏中选择**节点管理 > 节点**。
   4）在**节点**页面左上角单击**标签与污点管理**。
3. 单击**污点**页签，单击污点的删除图标。

4.2.3 设置节点调度

通过控制台设置节点调度，从而合理分配各节点的负载。

1、 登录边缘容器集群控制台。

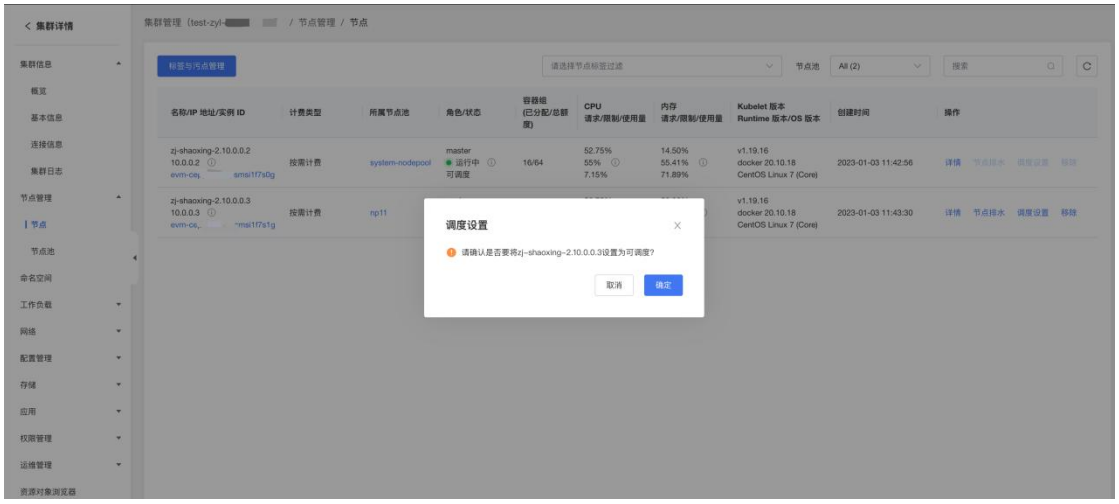
2、在控制台左侧导航栏中，单击**集群管理**。

3、在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

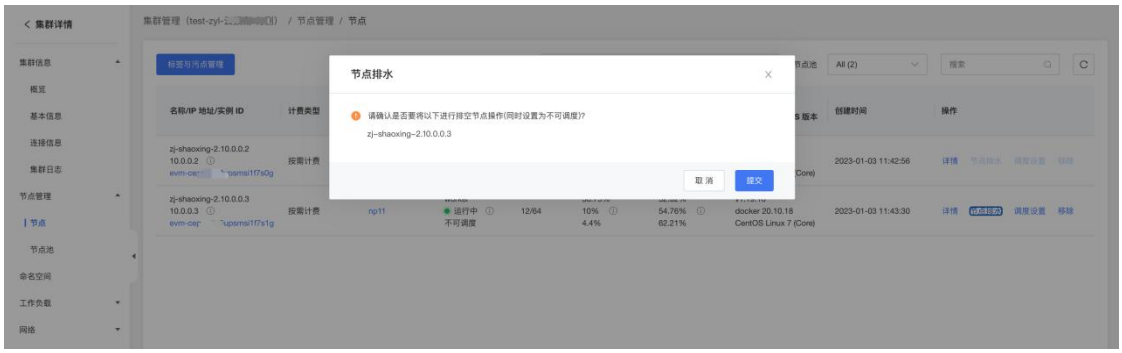
4、 在控制台左侧导航栏中，单击**节点管理>节点**。

5、 在节点列表页面中，在目标节点右侧操作列下进行以下操作。

选择**调度设置**，切换节点的调度状态，点击确定完成调度设置。设置为不可调，则在后续进行应用部署时，Pod 不会再调度到该节点。



选择**节点排水**，设置节点为不可调度，并排空节点，单击**提交**，完成节点排水设置。



4.2.4 查看节点详情

1、 登录边缘容器集群控制台。

2、在控制台左侧导航栏中，单击**集群管理**。

3、在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

4、 在控制台左侧导航栏中，单击**节点管理>节点**。

5、 在节点页面目标节点的操作列选择 **详情**。

说明：您也通过边缘智能云 ECX 控制台查看节点对应虚拟机 EVM 的详情

4.2.5 移除节点

除了了使用 kubectl，您还可以使用使用控制台移除节点

说明

· 移除节点会涉及 Pod 迁移，可能会影响业务，请在业务低峰期操作。

· 操作过程中可能存在非预期风险，请提前做好相关的数据备份。

· 移除节点仅支持移除 Worker 节点，不会移除 Master 节点。

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**节点管理>节点**。
5. 在节点列表页面中，单击目标节点右侧操作列下的**更多 > 移除**
6. **可选：**在**移除节点**对话框中，可选中**同时释放 EVM** 和 **自动排空节点（drain）**。

同时释放 EVM：仅释放按量付费类型的 EVM 实例，不被释放的 EVM 会继续收费，若不选择同时释放 EVM，该节点对应的 EVM 实例会继续计费。

自动排空节点（drain）：把待移除节点上的 Pod 转移到其他节点。请确保集群其他节点的资源充足。

7. 单击**确定**。

4.2.6 为容器节点添加数据盘

容器节点下载镜像的操作，例如运行容器运行时、存储容器的临时存储、记录容器的stdout 日志等，会占用大量系统盘资源。影响节点运行的稳定性。您可以使用数据盘作为容器运行时的根目录，从而节省系统盘的资源，提升节点运行的稳定性。本文介绍如何为容器的新建节点和已有节点添加数据盘。

为新建节点添加数据盘

1.登录边缘容器集群控制台。

2.在控制台左侧导航栏中，单击**集群管理** 。

3.在**集群列表**页面中，单击目标集群右侧**操作**列下的**节点池。**

4.在节点池列表页面，单击左上角的**创建节点池。**

5.在创建节点池页面，存储配置里面有个数据盘，点击勾上**是否将容器运行时目录挂载到最后一块数据盘。**

增加挂载数据盘后，容器服务 ECK 在节点初始化过程中，会自动将 Containerd、Docker 的使用目录放到数据盘。

为容器节点添加数据盘

如果您的集群中存在一些老旧的节点，这些节点的 Kubelet 和容器运行时的使用目录在系统盘中，会影响节点运行的稳定性。建议您通过切换节点池的方式，为节点添加数据盘，使用数据盘作为容器运行时的根目录。

1.登录边缘容器集群控制台。

2.在控制台左侧导航栏中，单击**集群管理** 。

3.在**集群列表**页面中，单击目标集群右侧**操作**列下的**节点池。**

4.在节点池列表页面，单击左上角的**创建节点池**，创建一个新的节点池，新节点池需挂载数据盘，期望节点数可以与旧节点池保持一致。

5.在节点页面，过滤条件选择旧节点池，筛选得到旧节点池的节点，点击节点排水和调度设置 ,将节点置为不可调度和排水中 ,集群会自动将旧节点池的容器逐步迁移到新的节点池中。

6.下线旧节点池：在节点池页面，选中旧节点池，点击**扩缩容**，将节点池中的节点逐步变成0，然后点击**删除节点池。**

4.2.7 监控节点

ECK 集成了 Prometheus 监控服务，用户可以根据自己的需要，将 Prometheus 部署到 ECK 容器中，并可在节点详情中查看监控数据。

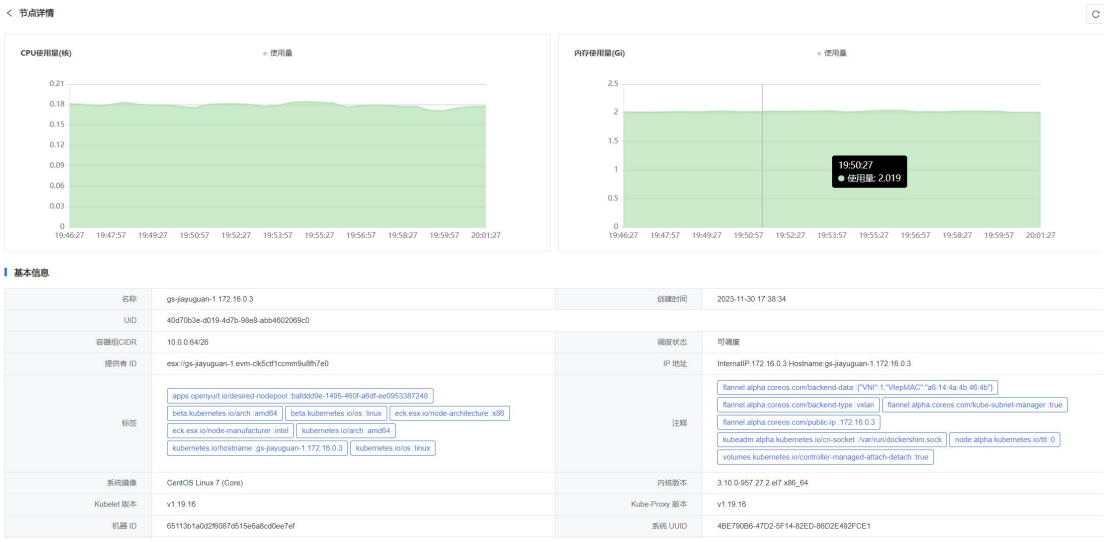
1.登录边缘容器集群控制台。

2.在控制台左侧导航栏中，单击**集群管理** 。

3.在**集群列表**页面，选择目标集群并单击左侧的**更多** > **组件管理** 。

4.进入**组件管理**页面，找到 Prometheus，单击**安装**安装组件。

5.安装成功之后，进入节点列表，选择目标节点点击**详情**，进入节点详情页面，可以查看节点的详情信息，详情信息数据都来源于 Prometheus。



4.3 节点池管理

4.3.1 节点池概述

节点池是一种用于管理计算节点的抽象概念，它可以将一组具有相同配置和用途的计算节点进行分组管理，提供更方便的节点管理和调度，帮助用户更好地利用计算资源。你可以通过一些组合动作，整体对一组逻辑划分的资源进行管理，例如可以节点配置、开启节点自动弹性伸缩、指定调度等。本文将通过介绍节点池的概念、节点池与自建节点池对比、节点

池功能、相关术语、生命周期等，来帮助您了解节点池的概念。

节点池概念

节点池是集群中一个或一组节点的逻辑集合 ,集群中可以创建多个不同配置和类型的节点池。在 ECK 中，期望的是同一个节点池，节点资源基本配置都是类似的，比如镜像，节点规格，标签、污点等。在逻辑上，您期望这一组资源用于特定的功能，或者用于某些特定场景。这些属性可以在创建节点池时指定 ,部分属性可以在创建完成后进行编辑修改(比如操作系统、污点、标签）。

以下是节点池的功能：

节点扩缩容。可以弹出一个或者多个相同配置的 ECX 的节点。一次扩容多个节点。同时，为了适配某些突发应用场景，可以设置自动伸缩功能，能够根据需求，弹出节点。

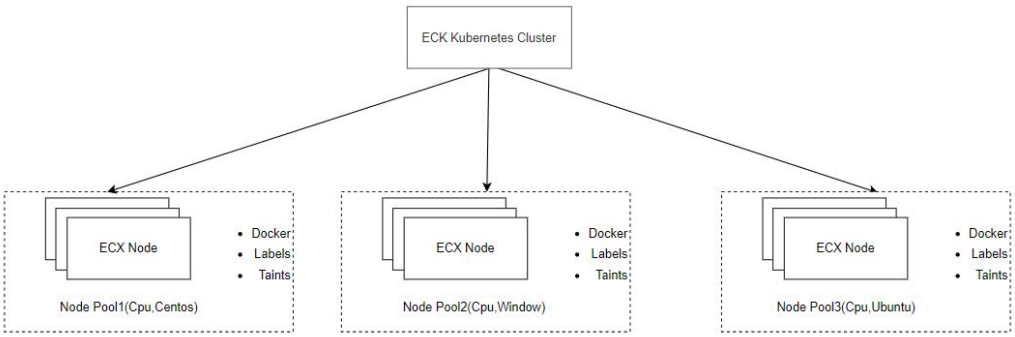
分组管理和运维。可以根据您的需要，可以设置组资源节点的配置、调度应用至指定节点池等。例如可以通过节点亲和性 ,将应用只部署到某个节点池中 ,原理是匹配节点池的 label。

节点池可以创建的类型：

允许创建不同操作系统（CentOS、Windows、Ubuntu）的节点池。

允许创建不同容器运行时（Containerd、Docker）的节点池。

允许创建多个开启自动弹性伸缩的节点池。



节点池与自建节点池介绍

节点池类型

ECK 节点池类型分为节点池和自建节点池。

节点池类型	描述
节点池	节点池是集群中具有相同配置的一组节点，节点池可以包含一个或多个节点。节点池与弹性伸缩组实例一比一对应。当对节点池进行扩容和缩容时，ACK 通过弹性伸缩服务下发扩容和移除节点的操作。您可以根据自己的需要创建和管理多个节点池。
自建节点池	自建节点池只有创建云边协同版的 ECK 集群的时候才能使用的功能。自建节点池可以在任意节点上部署。不限于 ECX 节点。您可以根据自身业务的功能，对资源池划分，将私有资源接入到 ECK 集群中来。需要私有资源能够网络上访问 ECK 集群的 Apiserver。

节点池功能

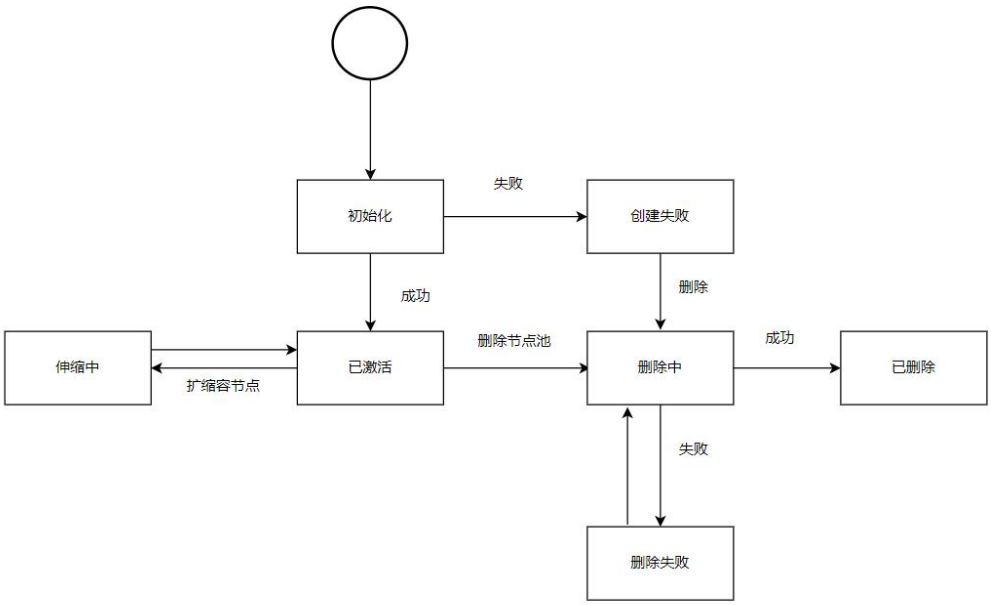
节点池目前支持以下功能。

功能	说明
创建节点池	创建节点池，需要指定节点池的配置。
编辑节点池	修改节点池的配置(仅仅能够修改节点池非节点生命周期的修改项，比如名称、标签等）。
扩缩容节点池	调整节点池内的节点数量。 1.根据实际的节点池数量，使用原始创建的配置进行变更节点池的数量。 2.节点池释放节点，会释放创建时间最新的节点。
添加已有节点（仅仅自建节点池才提供）	可添加不属于集群的节点到节点池中。
移除节点	移除节点池内指定的一个或多个节点，移除后节点将不再属于集群和节点池。您可以在移除节点前选择是否排水以及是否释放实例。
节点池自动伸缩（仅仅普通节点池提供）	弹性伸缩可以根据业务负载和策略，按需弹出实例。可以提供缩容阈值、GPU 缩容阈值、缩容触发时延、静默时间、弹性灵敏度、节点池扩容顺序策略等。

节点池相关术语

术语	描述
缩容阀值	节点上 Request 的资源与总资源量的比值。
缩容触发时延	节点缩容时需要连续满足触发时延所设定的时间，方可进行缩容。
静默时间	扩容出的节点，在静默时间过后，方可进入缩容判断。
弹性灵敏度	判断伸缩的间隔时间。
节点池扩容顺序策略	依据此策略决定执行伸缩的节点池的顺序。提供随机、最小资源策略。

节点池生命周期

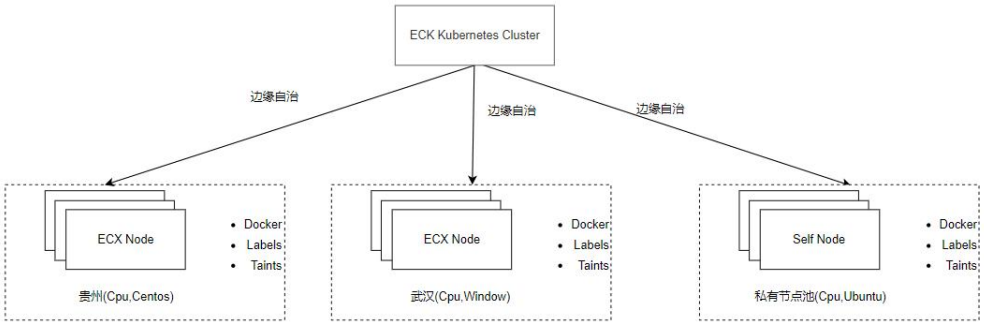


状态	说明
初始化	正在初始化节点池。
创建失败	创建节点池失败。
已激活	成功创建节点池、成功扩缩容节点。
删除中	正在删除节点池。
伸缩中	扩缩容节点。
已删除(用户不可见)	删除成功。
删除失败	删除失败。可以是从创建失败中转化，也可以是从已激活中转化。

4.3.2 自建节点池概述

通过自建节点池，用户可以自行将自己不在集群管理的资源纳入到 eck 集群中管理，可以有效的提升资源使用率。本文介绍自建节点池的基本信息、适用场景、主要特征以及与普通节点池之间的差异。

自建节点池的概览图



适用场景

拥有自己私有资源，希望能够使用 eck 集群纳管，部署应用。

期望能够边缘自治，在和 master 失联的情况下，依然希望边缘应用能够提供服务。

主要特征

边缘自治，能够保证边缘服务不会因为集群异常而中断服务。

可以支持纳管自己的私有资源，并且通过 eck 集群进行管理部署。

支持 centos 和 ubuntu 的资源池。

自建节点池和普通节点池对比

普通节点池 ：为您提供管理一组同质节点的能力，同一个节点池内具有相同的节点配置，例如规格、标签（Label）、污点（Taint）。您可以自行运维普通节点池内节点。

自建节点池： 您可以自行将自己的私有资源，或者 ecx 资源纳入到自建节点池中管理，要求自建节点池内的节点都能相互通信，以免应用异常。

自建节点池与普通节点池具体的对比项及说明如下表所示。

功能对比

对比项	普通节点池	自建节点池
使用场景	不需要边缘自治功能的应用部署，即 master 节点与 node 通信断了，服务会受影响。	拥有边缘自治能力，即 master 与 node 通信断了，边缘服务并不受影响。
支持系统	Centos,Window,Ubuntu。	Centos,Ubuntu。
是否支持纳管自有资源池	不支持。	支持。
是否支持部署边缘应用	不支持。	支持。
区域支持	仅支持 Master 相同的区域。	支持可以和 Apiserver 通信的所有区域。
是否支持弹性伸缩	支持。	不支持。

4.3.3 创建节点池

1. 登录边缘容器集群控制台。

2. 在控制台左侧导航栏中，单击**集群管理**。

3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

4. 在控制台左侧导航栏中，单击**节点管理>节点池**。

5. 在节点池列表页面，单击左上解的**创建节点池**

6. 在**创建节点池**页面，设置创建节点池的配置项。

配置项	描述
节点池名称	填写节点池名称
节点规格	选择 Worker 节点的实例规格
操作系统	Worker 节点的操作系统，至少需要 1 个节点
实例数量	设置您所需的 Worker 节点数量
系统盘	配置 Worker 节点系统盘的类型和大小，如果选择的实例规格里已经包含系统盘，则不需要配置
数据盘	可选择给 Worker 挂载数据盘系统盘，最多挂载 100 块
登录方式	目前支持设置密码登录 **登录密码**：设置节点的登录密码。 **确认密码**：确认设置的节点登录密码。 **说明** 密码为 8~26 位，大、小写字母、数字、特殊字符 4 种字符组合

7. 单击**确认配置**。进入**配置确认**页面，确认配置无误并且依赖项检查通过，然后单击**创建节点池**开始创建。在**节点池**页面，如果节点池**状态**显示**初始化中**，则说明节点池正在创建中。创建完成后，**状态**显示为**已激活**。

4.3.4 查看节点池

1.登录边缘容器集群控制台。

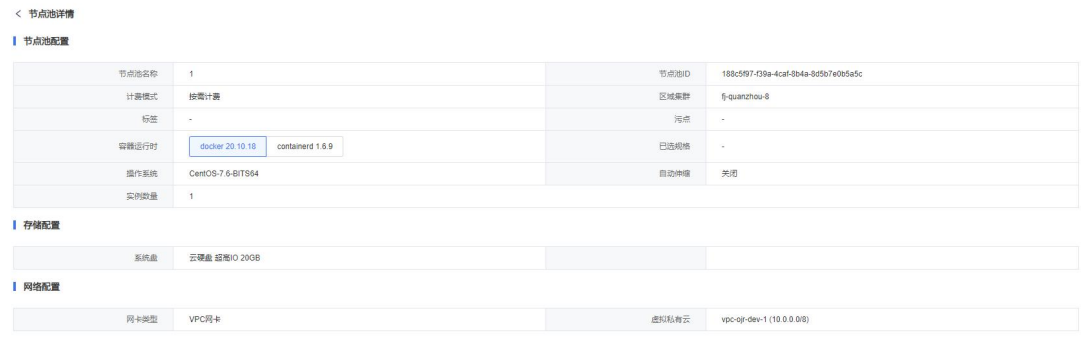
2.点击目标集群右侧的**节点池**按钮，即可查看节点池列表。

名称	实例规格	计费类型	类型	状态	节点数(正常/全部)	区域	操作系统	更新时间	操作
1	s6.large.2	按需计费	节点池	已激活	1 / 1	福建-泉州8	CentOS-7.6-BITS 64	2023-12-07 11:36:27	详情 更新 扩缩容 删除

如果节点池处于创建中、伸缩中等中间状态时，可以点击状态值，即可查看进度。

名称	实例规格	计费类型	类型	状态	节点数(正常/全部)	区域	操作系统	更新时间	操作
1	s6.large.2	按需计费	节点池	伸缩中	1 / 1	福建-泉州8	CentOS-7.6-BITS 64	2024-01-17 17:19:58	详情 更新 扩缩容 删除

3.点击目标节点池右侧的**详情**按钮，即可观察到节点池详细信息。

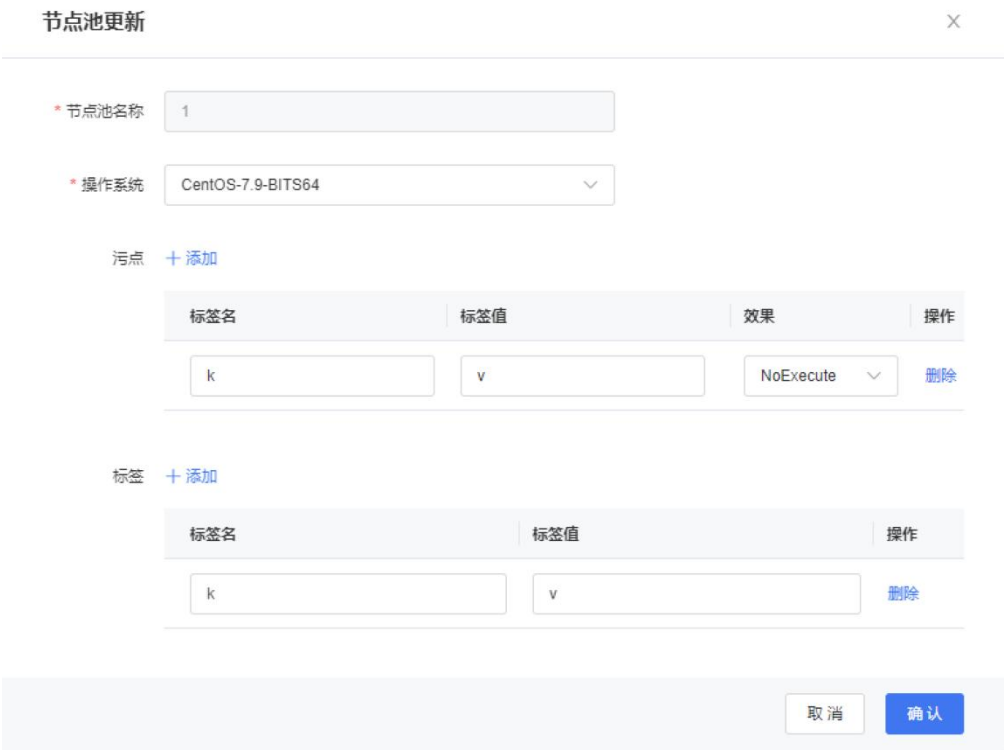


4.3.5 编辑节点池

1.登录边缘容器集群控制台。

2.点击目标集群右侧的**节点池**按钮，即可查看节点池列表。

3.点击目标节点池右侧的**更新**按钮，可以对节点池的操作系统、污点以及标签进行修改。



注意：

更新节点池之后进行扩容 ,新创建的节点会使用更新后的节点池操作系统和打上更新后的节点池污点和标签，但节点池更新不会影响节点池下属原有的节点。

4.3.6 删除节点池

删除节点池，会先删除节点池中的节点，节点删除后，原有节点上的工作负载实例会自动迁移至其他节点池的可用节点。如果工作负载实例具有特定的节点选择器，且如果集群中的其

他节点均不符合标准，则工作负载实例可能仍处于无法安排的状态；手动添加到节点池的节点不会被释放

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**节点管理>节点池**。
5. 在节点池列表，在目标节点池右侧的**操作**列中，单击**删除**。

6. 在弹出的对话框中核对信息后，单击**确认**即可删除资源池。

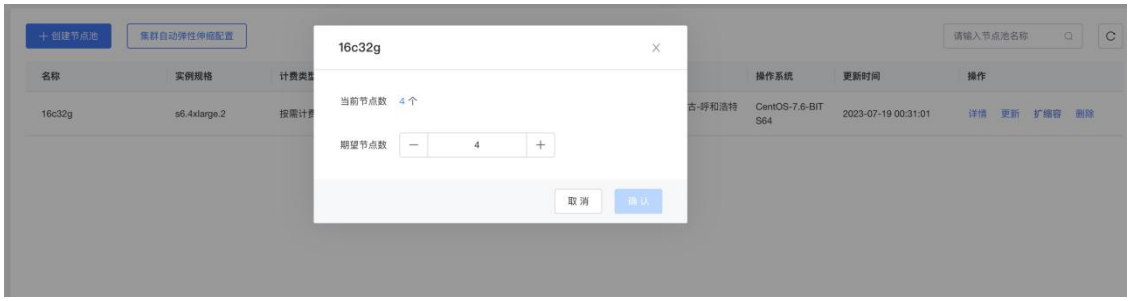
4.3.7 扩缩容节点池

4.3.7.1 调整期望节点数

您可以通过调整期望节点数，达到扩容或缩容节点池的目的，具体操作如下：

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**节点管理>节点池**。
5. 在节点池列表，在目标节点池右侧的**操作**列中，单击**扩缩容**。
6. 在弹出对话框中，设置期望节点数

扩容节点：设定期望节点数大于当前节点池的节点数，系统将触发节点池扩容。



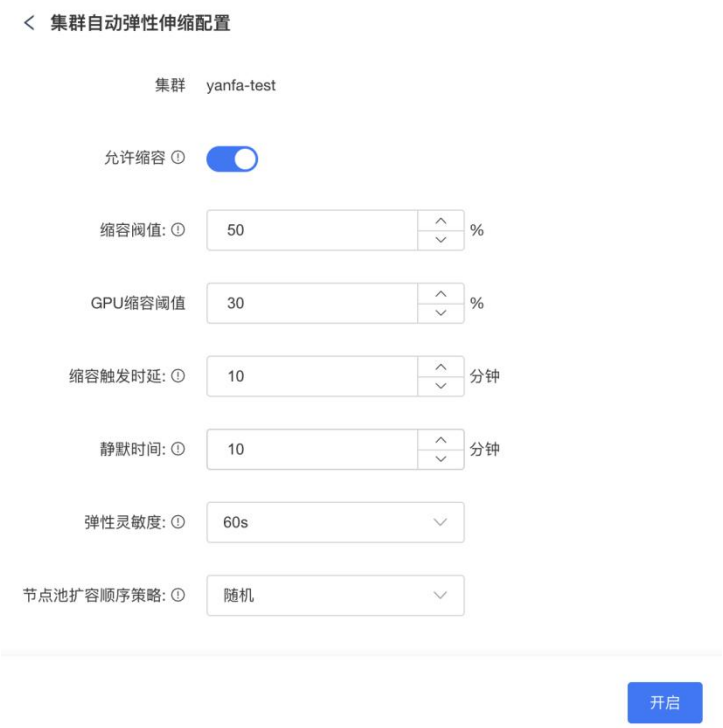
缩容节点：设定期望节点数小于当前节点池的节点数，系统将触发节点池缩容，缩容可以选择是否移除节点。

单击**确认**，进入**配置确认**页面，确认配置无误并且依赖项检查通过，然后单击**确认**开始扩缩容

在**节点池**页面，如果节点池**状态**显示**伸缩中**，则说明节点池正在扩容中。扩容完成后，**状态**显示为**已激活**。

4.3.7.2 自动弹性伸缩

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**节点管理>节点池**。
5. 在节点池列表页面，单击左上角的**集群自动弹性伸缩配置**



5. 在配置页面配置相应参数

配置项	描述
允许缩容	是否允许进行节点缩容。若设置为不允许缩容，缩容相关配置将不生效。
缩容阈值	每一个节点的资源申请值（Request）/ 每一个节点的资源容量。当低于配置的阈值时，节点会进行缩容。
GPU 缩容阈值	GPU 类型实例的缩容阈值，当低于配置的阈值时，GPU 类型节点会进行缩容。
缩容触发时延	集群满足配置的缩容阈值时，在配置的缩容触发时延到达后，集群开始缩容。单位：分钟。默认情况下是 10 分钟。
静默时间	扩容出的节点，在静默时间过后，方可进入缩容判断。单位：分钟。默认情况下是 10 分钟。
弹性灵敏度	集群自动弹性伸缩配置支持设置弹性灵敏度，以调整系统判断伸缩

	的间隔时间。目前支持 15s、30s、60s 的选项，默认值是 60s
节点池扩容顺序策略	**最小资源**：如果可扩容节点池有多个，从中选择一个资源浪费最少的节点池进行扩容。 **随机**：随机策略，如果可扩容节点池有多个，从中任意选择一个节点池进行扩容，默认为随机策略。

6. 单击**开启**启动自动弹性伸缩（注意：如允许缩容选择“否”的话，将不会自动缩容），在自动弹性伸缩界面点击**创建节点池**，在此处创建的节点池将启动自动弹性伸缩功能。您也可以通过**修改**、**禁用** 来修改或关闭自动弹性伸缩。

4.3.8 添加已有节点

通过创建自建节点池，然后将已有节点添加到自建节点池，以实现将已有的节点添加到 Kubernetes 集群中，目前仅支持添加 Worker 节点。ECK 集群纳管非 ECX 节点的第 3 方节点需要收取节点管理费(当前为公测阶段暂不收费)

使用限制

1）确保节点与集群之间的网络可互通

2）只支持添加以下操作系统的节点：Ubuntu、CentOS

具体操作如下：

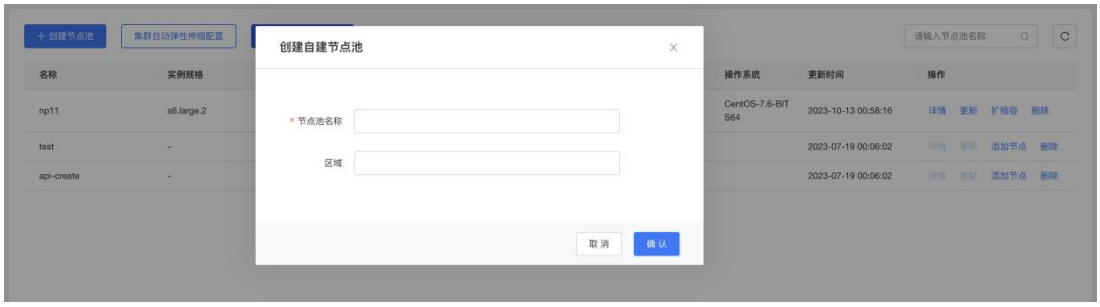
1. 登录边缘容器集群控制台。

2. 在控制台左侧导航栏中，单击**集群管理**。

3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

4. 在控制台左侧导航栏中，单击**节点管理>节点池**。

5. 在节点池列表，单击左上角的**创建自建节点池**，填写节点池名称和区域(可选)后点击点确认即可创建节点池。自建节点池可添加已有节点节点，但不可进行扩缩容。



6. 在自建节点池右侧单击的**添加节点**。

6. 选择待添加节点的系统架构、操作系统及希望使用的容器运行时，单击**下一步**生成纳管命令

8. 复制纳管命令在待添加节点的 SSH 终端运行， 即可将节点添加到自建的节点池中。

4.3.9 移除节点

前提条件

已创建 ECK 集群，集群状态为运行中。

Worker 节点池大于 1 个，且节点池中的节点大于 1 个。

移除方法

通过设置节点池期望的节点数移除节点

1.登录边缘容器集群控制台。

2.在控制台左侧导航栏中，单击**集群管理** 。

3.在**集群列表**页面，选择目标集群并单击右侧的**节点池** 。

4.选择目标节点池，单击列表右侧**操作**中的**扩缩容** ，填入期望的节点数量，点击**确认** ，即可完成缩容。

通过节点管理移除节点

1.登录边缘容器集群控制台。

2.在控制台左侧导航栏中，单击 **集群管理** 。

3.在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情** 。

4.在控制台左侧导航栏中，单击**节点管理>节点** 。

5.在节点列表页面中，单击目标节点右侧操作列下的**移除。**

4.在弹出对话框中选择是否释放掉智能边缘云 ECX 的 EVM 资源和是否自动排空节点(将节点上的 Pod 迁移到其余节点 ），点击**确认**，即可完成节点移除。

4.4 镜像管理

4.4.1 使用限制

配额

为保证资源的有效利用率，我们对于用户的镜像设置了配额。容器镜像服务对单个组织可承载的镜像数量及大小没有限制，只对单个用户可添加的组织数量限定了配额，如下表所示。如果您需要添加更多组织，请提交工单申请。

资源类型	配额(单位/个)
组织	15

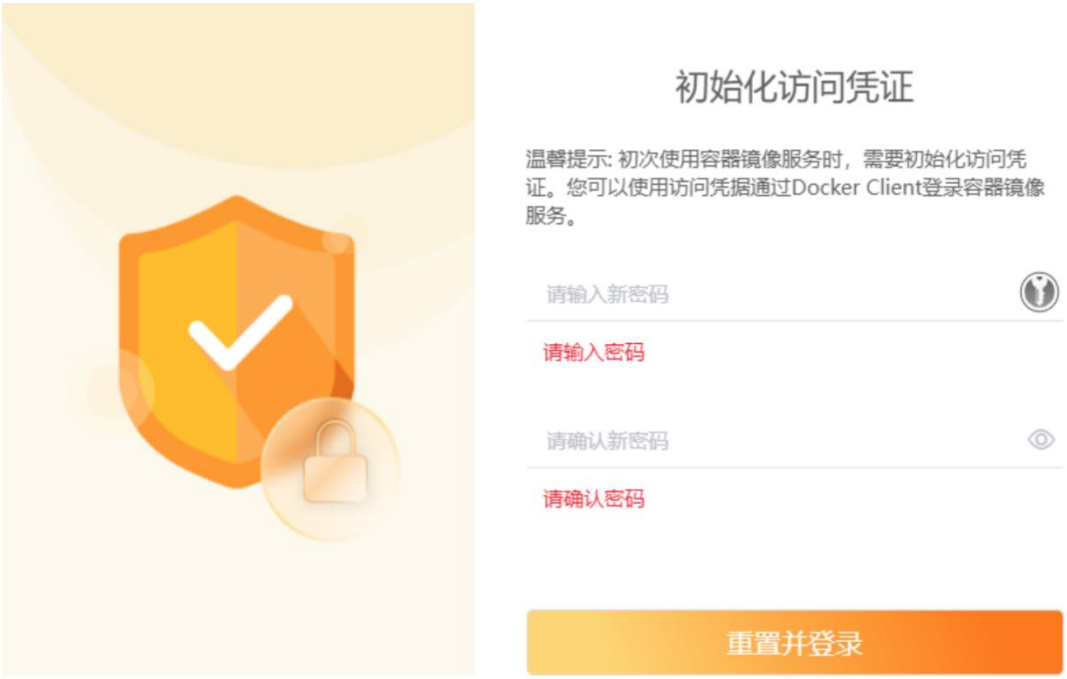
上传镜像限制

使用客户端上传镜像，镜像的每个 layer 大小不能超过 10G。

4.4.2 首次使用服务设置密码

1. 登录容器镜像服务控制台。

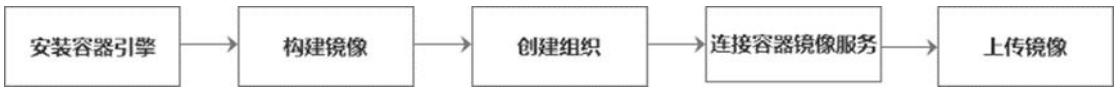
2. 首次进入容器镜像服务，在弹窗中按照要求设置登录容器镜像服务的密码。



类型	描述
密码	密码是用户登录到镜像仓库的凭证。密码创建规则：8-20 位字符，包含大写字符小写字符和数字。
确认密码	再试输入相同的密码。

4.4.3 客户端上传容器镜像

客户端上传容器镜像的流程如下图所示：



本文以一个 2048 应用为例，讲述根据该应用编写 Dockerfile 文件构建镜像并上传至容器镜像服务的操作。

步骤一：安装容器引擎

1. 准备 1 台 Linux 服务器。
2. 以 root 用户登录服务器。
3. 安装和配置容器引擎，安装的 Docker 版本必须为 1.12 及以上。

步骤二：构建镜像

1. 在安装容器引擎的服务器上执行以下命令，下载 2048 应用的源码。

   git clone https://gitee.com/jorgensen/2048.git

2. 下载成功后，进入“2048”目录。

   cd 2048

3. 修改 Docker file 文件。

   vim Dockerfile

```
FROM nginx
COPY . /usr/share/nginx/html

EXPOSE 80
CMD ["nginx", "-g", "daemon off;"]
```

a. FROM：指定基础镜像 nginx。

b. COPY：将 2048 源码拷贝到容器内的“/usr/share/nginx/html”目录。

c. EXPOSE：暴露容器的 80 端口。

d. CMD：指定容器运行时的默认命令。

按“Esc”，输入**:wq**，保存并退出。

4. 使用 docker build 命令构建镜像。

docker build -t 2048 .

其中，

a. -t 表示给镜像加一个标签，也就是给镜像取名，这里镜像名为 2048。

b. . 表示上下文路径，镜像构建命令将该路径下的所有内容打包给容器引擎帮助构建镜像。

5. 执行以下命令，查看已成功构建的 2048 镜像，版本为默认的 latest。

docker images

```
# docker images
REPOSITORY   TAG       IMAGE
ID       CREATED              SIZE
2048         latest    8d421c503ed0   About a minute
ago   134MB
nginx        latest    dd34e67e3371   6 days
ago           133MB
```

您还可以看到一个 nginx 镜像，这个镜像是从镜像仓库下载下来，作为 2048 镜像的基础镜像使用的。

6. 运行容器镜像（可选）。

   镜像构建成功后，您可以执行 docker run 命令运行容器镜像。

   docker run -p 8080:80 2048

   docker run 命令会启动一个容器，命令中-p 是将服务器的 8080 端口映射到容器的 80 端口，即服务器的 8080 端口的流量会映射到容器的 80 端口，当您在本地机器的浏览器访问“https:// IP:8080”时，就会访问到容器中，此时浏览器返回的内容就是 2048 应用页面。

步骤三：创建组织

1. 登录容器镜像服务 CRS 管理控制台。
2. 在控制台左侧导航栏中，选择【组织管理】。
3. 在【组织管理】页面中，单击【创建自定义组织】。

   注意：每个用户最多允许创建 3 个组织。
4. 在【创建自定义组织】页面，输入组织名称并设置访问基本。
5. 单击【确认】。

步骤四：连接容器镜像服务

在安装容器引擎的服务器执行以下指令，连接容器镜像服务。

docker login -u {完整邮箱名} -p {password} ehub.ctcdn.cn

步骤五：上传镜像

在安装容器引擎的服务器上执行以下命令，将镜像上传到指定组织。

$ sudo docker tag {镜像名称}:{版本名称} ehub.ctcdn.cn/组织名称/{镜像名称}:{版本名称}

$ sudo docker push ehub.ctcdn.cn/组织名称/{镜像名称}:{版本名称}

4.5 网络管理

4.5.1 网络概述

容器网络 CNI

容器化应用会在同一个节点上部署多个业务，而每个业务都需要自己的网络空间。为避免与其他业务网络冲突，需要 Pod 有自己独立的网络空间，而 Pod 中应用需要和

其他网络进行通信，就需要 Pod 能够跟不同的网络互相访问。可见容器网络特点如下：

* 每个 Pod 都拥有自己独立的网络空间和 IP 地址。不同 Pod 的应用可以监听同样的端口而不冲突。Pod 可以通过各自的 IP 地址互相访问。
* 集群中 Pod 可以通过它独立的 IP 地址与其他应用互通：

同一个集群中，Pod 之间相互访问。

Pod 直接访问同一个 VPC 下的边缘虚拟机。

同一个 VPC 下的边缘虚拟机直接访问 Pod。

Flannel 网络模式

Flannel 网络模式中 Pod 的网段独立于 VPC 的网段。Pod 网段会按照掩码均匀划分给每个集群中的节点，每个节点上的 Pod 会从节点上划分的网段中分配 IP 地址。Flannel 网络模式的特点是：

* 支持 VXLAN 模式。
* Pod 网段是独立于 VPC 的虚拟网段。

Service

由于云原生的应用，通常需要敏捷的迭代和快速的弹性， 且单一容器和其相关的网络资源的生命周期非常短暂，所以需要固定的访问地址，以及自动负载均衡实现快速的业务弹性。ECK 采用 Service 方式为一组容器提供固定的访问入口，并对这一组容器进行负载均衡。实现原理如下：

创建 Service 对象时，ECK 会分配一个相对固定的 Service IP 地址。

通过字段 selector 选择一组容器，以将这个 Service 的 IP 地址和端口负载均衡到这一组容器 IP 和端口上。

* ClusterIP

ClusterIP 类型的 Service 用于集群内部的应用间访问，如果您的应用需要暴

露到集群内部提供服务，需使用 ClusterIP 类型的 Service 进行暴露。

说明：创建 Service 时默认的 Service 类型为 ClusterIP。

* NodePort

NodePort 类型的 Service 将集群中部署的应用向外暴露，通过集群节点上的一个固定端口暴露出去，这样在集群外部就可以通过节点 IP 和这个固定端口来访问。

* LoadBalancer

LoadBalancer 类型的 Service 同样是将集群内部部署的应用向外暴露，不过它是通过智能边缘云的负载均衡进行暴露的，相对于 NodePort 方式，有更高的可用性和性能。

* Headless Service

Headless Service 类型的 Service 是在 Service 属性中指定 clusterIP 字段为 None 类型。采用 Headless Service 类型后，Service 将没有固定的虚拟 IP 地址，客户端访问 Service 的域名时会通过 DNS 返回所有的后端 Pod 实例的 IP 地址，客户端需要采用 DNS 负载均衡来实现对后端的负载均衡。

* ExternalName

ExternalName 类型的 Service 将集群外部的域名映射到集群内部的 Service 上，例如将外部的数据库域名映射到集群内部的 Service 名，那么就能在集群内部通过 Service 名直接访问。

Ingress

在 ECK 集群中，与 Service 的 4 层负载均衡不同，Ingress 是集群内 Service 对外暴露 7 层的访问接入点。

您可以通过 Ingress 资源来配置不同的 7 层的转发规则，例如通过域名或者访问路径来路由到不同的 Service 上，从而达到 7 层的负载均衡作用。

服务发现 DNS

ECK 使用 DNS 来实现应用的服务发现能力，例如客户端应用可以通过 Service 的服务名解析出它的 ClusterIP 访问，可以通过 StatefulSet 的 Pod 名解析出 Pod 的 IP 地址。采用 DNS 服务发现的能力让集群中应用间的调用与 IP 地址和部署环境解耦。

集群的 Coredns 会将 Service 名自动转换成对应的 Service 的 IP 地址，来实现不同部署环境中相同的访问入口。

4.5.2 Kubernetes 集群网络规划

创建 ECK 集群时需要规划以下地址：

* VPC 和虚拟子网：选择用于部署集群的 VPC 和虚拟子网。

* Pod 网络 CIDR：为集群中的 Pod 分配 IP 地址的范围。

* Service CIDR：为集群中的 Service 分配 IP 地址的范围。

需确保这些地址范围不重叠或冲突，并且足够大以满足集群扩展的需求。

虚拟私有云 VPC 网段和 Kubernetes 网段关系

虚拟私有云 VPC（下文简称为 VPC）的网段规划包含 VPC 自身网段和虚拟子网网段，Kubernetes 网段规划包含 Pod 地址段和 Service 地址段。ECK 网络支持 Flannel 网络模式。配置 Flannel 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：

虚拟私有云

您在创建 VPC 时需要选择网段，可以从 10.0.0.0/8-28、172.16.0.0/12-28、192.168.0.0/16-28 三者当中选择一个。

子网

在 VPC 里创建子网的指定的网段， 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：子网是 VPC 子网。子网下边缘虚拟机所分配到的地址，就是从这个子网网段内获取的。一个 VPC 下，可以创建多个子网，但子网网段不

能重叠。

Pod 网络 CIDR

Pod 网络 CIDR，Pod 地址从该地址段分配，用于 Pod 网络通信。Pod 是 Kubernetes 内的概念，每个 Pod 具有一个 IP 地址。配置网段时，请注意：

* 非 VPC 子网，为虚拟网段。

* 该地址段不能和虚拟子网网段重叠。

* 该地址段不能和 Service CIDR 网段重叠。

例如，VPC 网段用的是 172.16.0.0/12，Kubernetes 的 Pod 地址段就不能使用 172.16.0.0/16、172.17.0.0/16 等，因为这些地址都包含在 172.16.0.0/12 里。

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 地址段不能和虚拟子网地址段重叠。

Service 地址段不能和 Pod 网络 CIDR 地址段重叠。

4.5.3 服务 Service 管理

4.5.3.1 Service 的负载均衡配置注意事项

ECK 的 CCM（Cloud Controller Manager）组件会为 Type=LoadBalancer 的 Service 配置负载均衡器，本文介绍配置 Service 负载均衡的注意事项以及 CCM 的资源更新策略。

SLB 更新策略

资源对象	CCM 管理 SLB 动作

资源对象	CCM 管理 SLB 动作
SLB	CCM 会根据 Service 的配置，自动创建和配置 SLB、弹性公网 IP、监听器、虚拟服务器组资源，所有资源由 CCM 管理。
监听器	CCM 会根据 Service 的配置，自动创建和配置监听策略。
后端服务器组	当 Service 对应的后端 Endpoint 或者集群节点发生变化时，CCM 会自动更新 SLB 的后端虚拟服务器组，更新规则如下： CCM 不会将 Master 节点作为 SLB 的后端。 如节点状态从 Ready 变成 NotReady ,CCM 会将该节点从后端服务器组中移除。 如果节点状态从 NotReady 变成 Ready ,CCM 会将该节点重新加入后端服务器组。

注意事项

- CCM 只为 Type=LoadBalancer 类型的 Service 配置 SLB，对于非 LoadBalancer 类型的 Service 则不会为其配置负载均衡。
- 当 Type=LoadBalancer 的 Service 变更为 Type!=LoadBalancer 时，CCM 会删除为该 SLB 添加的配置，从而造成无法通过该 SLB 访问 Service。
- 请勿在 SLB 控制台上手动修改 ECK 创建并维护的 SLB 的任何配置，否则有配置丢失的风险，造成 Service 不可访问。

4.5.3.2 创建服务

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**网络>服务**。
5. 在**服务**列表，在左上角选择**命名空间**并单击**创建服务**。
6. 完成服务的参数配置，单击**确认**添加服务。

配置项	描述
名称	输入服务的名称。
类型	选择服务访问的方式 **虚拟集群 IP**：即 ClusterIP，指通过集群的内部 IP 暴露服务，选择该值，服务只能够在集群内部可以访问，这也是默认的 ServiceType。 **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，该 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
实例间发现服务	服务类型为**虚拟集群 IP** 时，才能设置**实例间发现服务（Headless Service）**。
外部流量策略	**Local**：流量只发给本机的 Pod。 **Cluster**：流量可以转发到其他节点上的 Pod。 服务类型为**节点端口**或**负载均衡**时，才能设置**外部流量策略**。
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。
标签	为该服务添加一个标签，标识该服务

注解	为该服务添加一个注解（Annotation），配置负载均衡的参数

4.5.3.3 使用 SLB 负载均衡暴露应用

通过设置服务（service）的 spec.type=LoadBalancer ，ECK 会为该服务创建 SLB 负载均衡器来暴露该服务（Service）。在集群外可通过 SLB 的 <IP:服务端口>的方式访问服务，在集群内可通过 <服务名:服务端口>的方式访问服务。本文以 Nginx 应用为例，介绍如何通过 SLB 负载均衡公开应用。

注意 本示例会创建公网 IP 以及负载均衡器，产生一定的费用。

步骤一：创建 Nginx 应用

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击 **集群管理** 。
3. 在集群列表页面中，单击目标集群右侧操作列下的 **详情** 。
4. 在控制台左侧导航栏中，单击 **工作负载>无状态** 。
5. 在无状态列表，在左上角选择命名空间并单击 **创建无状态** 。
6. 完成服务的参数配置，单击确认添加无服务应用。无状态应用的创建操作请参见“使用镜像创建 Deployment” 。

步骤二：创建 Service

1. 接步骤一， 在控制台左侧导航栏中，单击**网络>服务**。
2. 在服务列表，在左上角选择与步骤一相同的命名空间并单击**创建服务**。
3. 完成服务的参数配置，单击确认添加 Service。

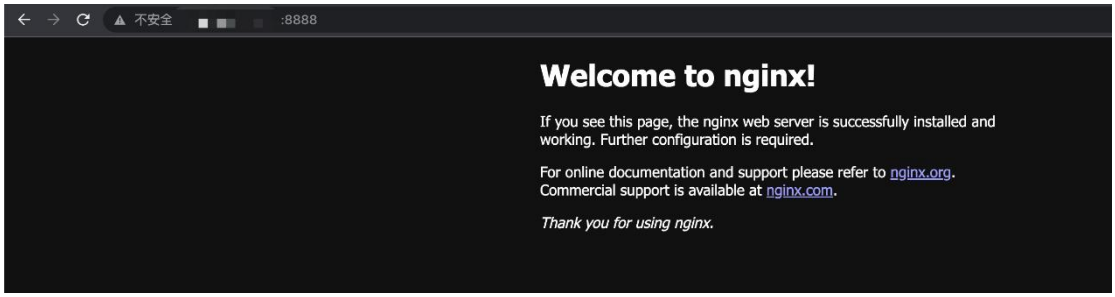
配置项	描述
名称	服务名称，例如：nginx-test-loadbalancer

配置项	描述
类型	服务类型，选择负载平衡器（LoadBalancer）
关联	关联步骤一创建的无状态负载
端口映射	示例：名称: http 服务端口：对公网暴露的端口。例：8888 容器端口：容器对应的端口，例：80 节点端口：NodePort 端口，可留空协议：容器服务协议，本示例为 TCP

示例：

步骤二完成后，部署在集群内的 CCM（cloud-controller-manager）控制器，会调用智能边缘云接口，申请负载均衡器，以及公网 IP，暴露集群内服务。服务创建完毕后，等待几秒钟，服务列表中会显示供访问的 <公网 IP:端口> 外部端点。

通过浏览器或者 curl 访问外部端点 <公网 IP:端口> 测试服务连通性：



使用命令行工具创建 Nginx 应用与负载均衡器 Service

1. 执行以下命令启动容器（本实例中为 Nginx Web 服务器）。

kubectl create deployment nginx-test-loadbalancer

-image=ehub.ctcdn.cn/eck/nginx

2. 执行以下命令，为该容器创建一个服务入口，指定--type=LoadBalancer 将会为您创建一个智能边缘云负载均衡路由到该 Nginx 容器。

kubectl expose deployment nginx-test-loadbalancer --port=8888 --target-port=80

--type=LoadBalancer

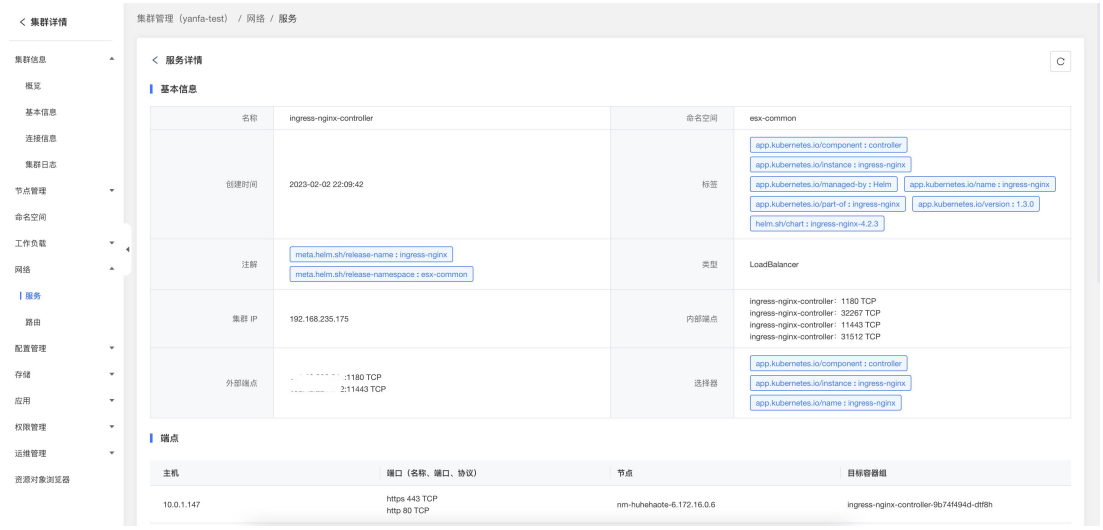
3.执行以下命令，查看 service 的公网 IP 地址。

kubectl get svc nginx-test-loadbalancer -ojsonpath='{.status.loadBalancer}'

4.curl 或浏览器访问 <公网 IP:8888> 地址，验证连通性。

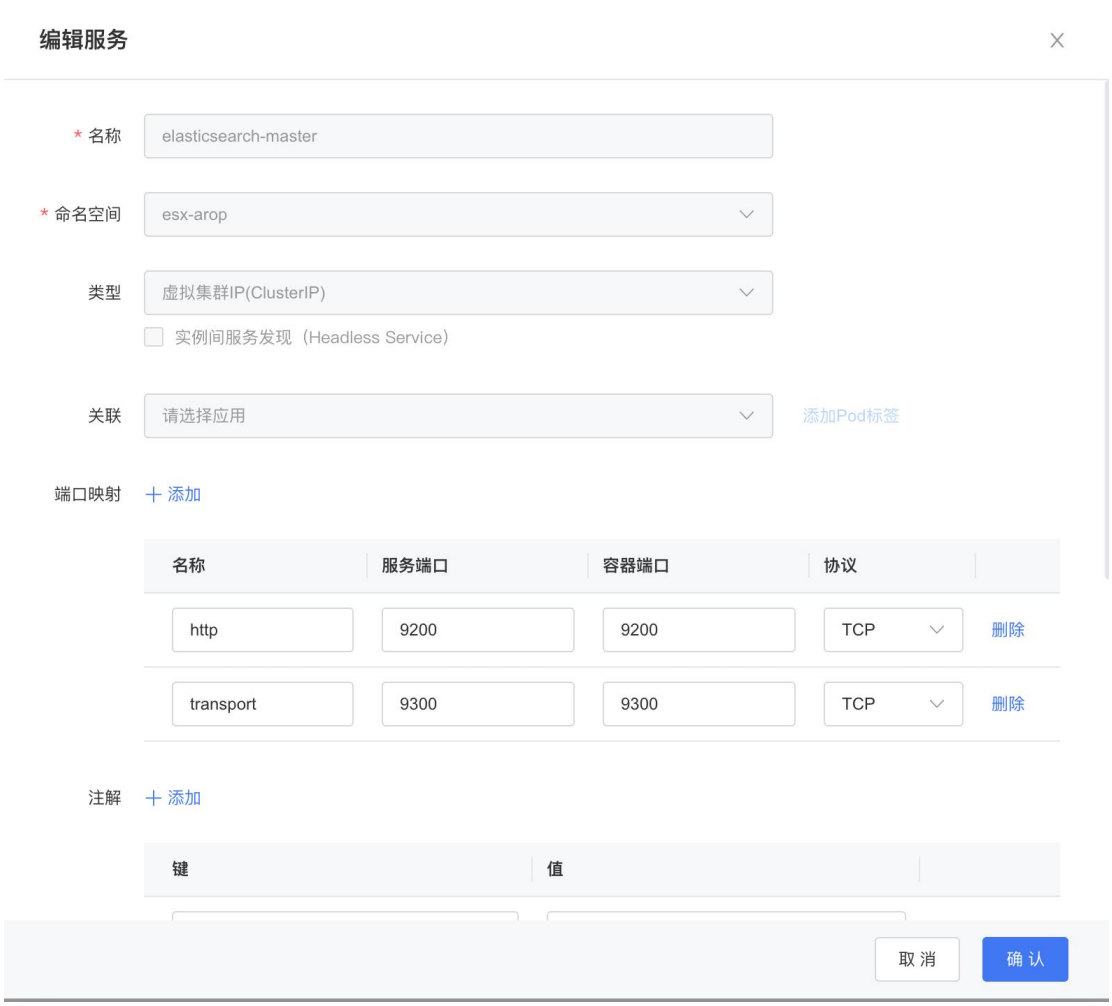
4.5.3.4 查看服务

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**网络>服务**。
5. 在**服务**列表，单击目标服务右则的**详情** 查看服务的详情

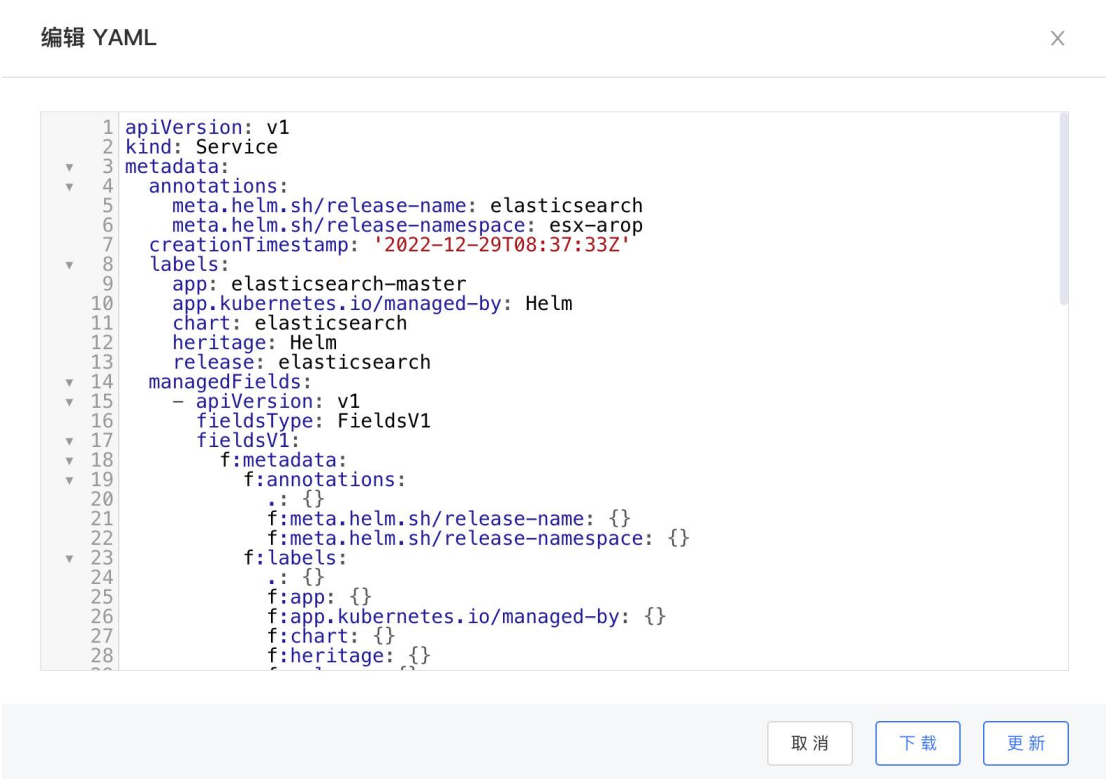


4.5.3.5 更新服务

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**网络>服务**。
5. 在**服务**列表，单击目标服务右则的**更新**



6. 也可以通过单击目标服务右则的 **查看 YAML** 在 yaml 文件中更新服务



4.5.3.6 删除服务

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**网络>服务**。
5. 在**服务**列表，单击目标服务右则的**删除**
6. 在弹出的确认框中，单击**确认**即可删除服务

4.5.4 路由 Ingress 管理

4.5.4.1 Ingress 概述

Ingress 是 Kubernetes 中的 API 对象，用于管理集群内服务的外部访问，实现 HTTP 和 HTTPS 路由规则的配置与管理。

Ingress 基本概念

在 Kubernetes 集群中，Ingress 作为管理集群内服务对外暴露的访问接入点的资源对象，负责承载大部分集群内服务访问流量。通过配置不同的转发规则，Ingress 可以实现对集群内不同 Service 的访问控制和路由管理。

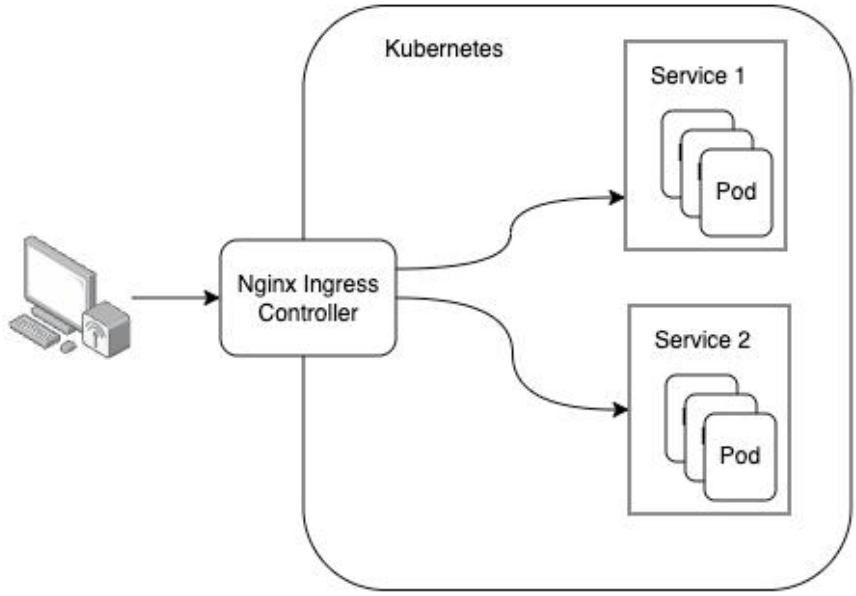
Nginx Ingress Controller

工作原理

Nginx Ingress 资源依赖 Nginx Ingress Controller 组件，该组件用于解析和应用 Ingress 资源的转发规则。

当收到请求时，Nginx Ingress Controller 会基于 Ingress 规则，匹配合适的 Service 服务，将请求以负载均衡的方式转发到 Service 服务对应的后端应用 Pod。

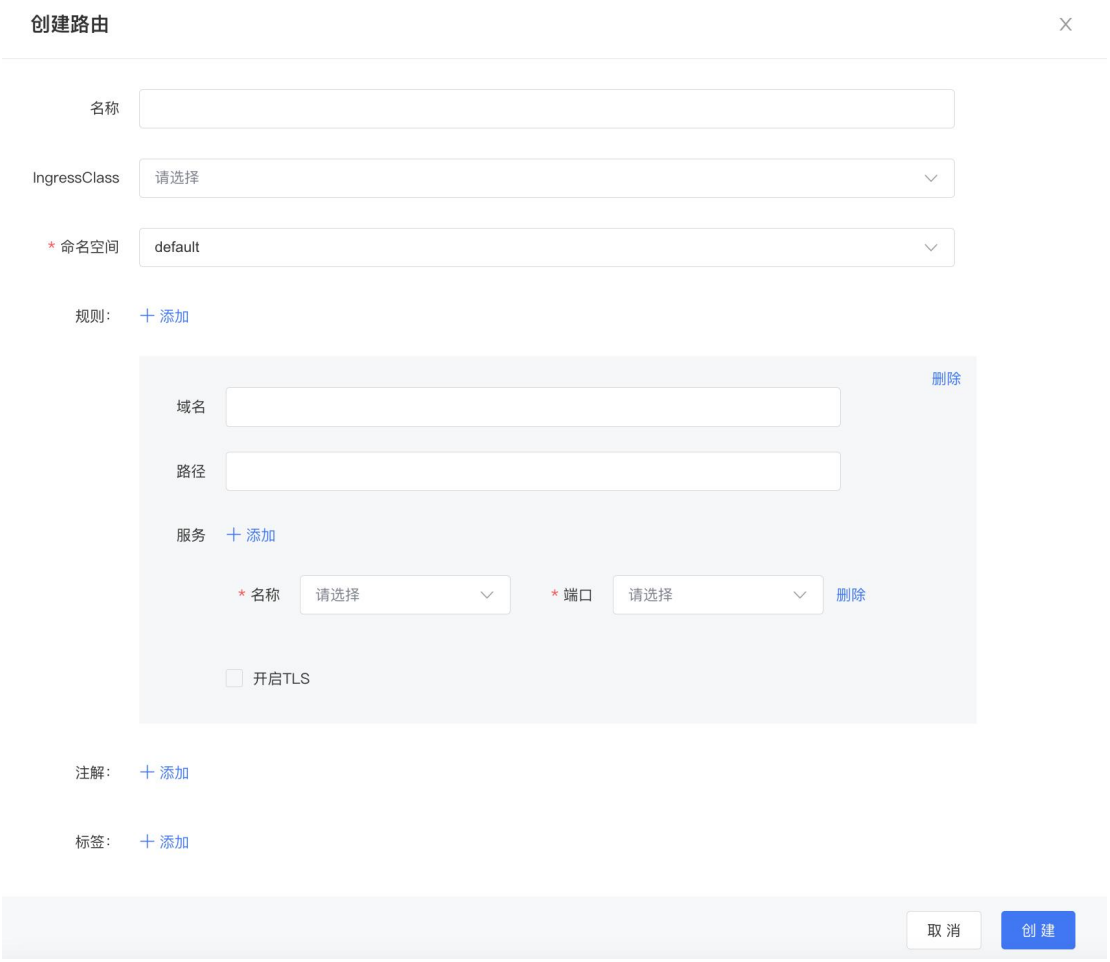
当 Ingress 资源发生变更时，Nginx Ingress Controller 会监听资源变更，动态的生成 Nginx 配置，然后重新加载 Nginx 来生成新的路由转发规则。



Nginx Ingress Controller 可通过配置 LoadBalancer 类型的 Service 对外暴露，因此外部可以通过 SLB 访问到 Kubernetes 集群的内部服务。根据 Nginx Ingress 配置的不同规则来访问不同的服务。

4.5.4.2 创建路由

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**网络>路由**。
5. 在**路由**列表，在左上角选择**命名空间**并单击**创建路由**



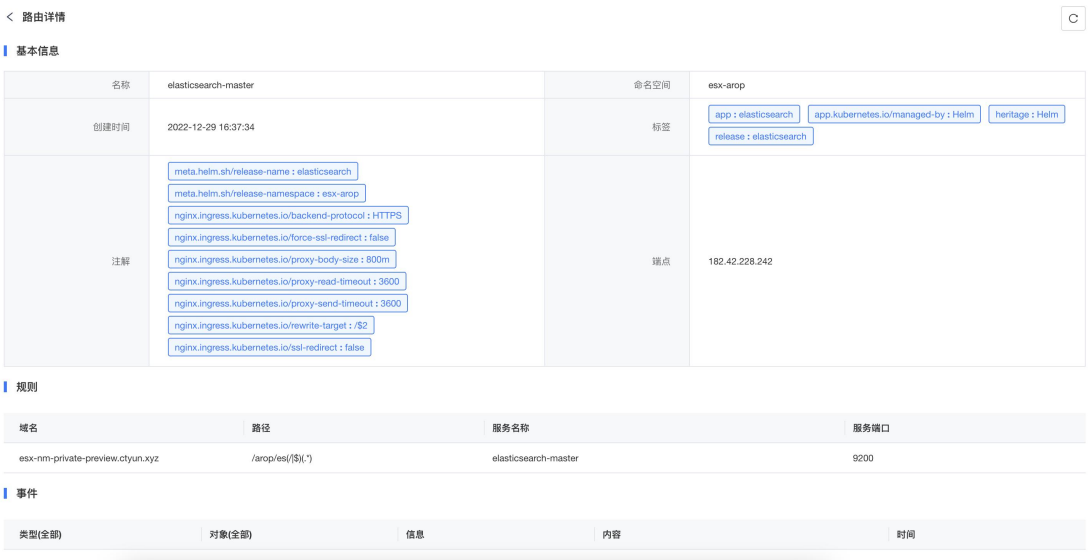
6. 完成参数配置，单击**确认**添加路由

配置项	描述
名称	输入路由的名称。
域名	通过域名对外提供访问服务
路径	指定服务访问的 URL 路径，每个路径都关联一个服务，流量转发到

	服务之前，所有的入站请求都要先匹配域名和路径
服务	选择关联的服务名称及服务端口
开启 TLS	可以选择开启 TLS，配置安全的路由服务，目前支持使用已有密钥
标签	为该路由添加一个标签，标识该路由
注解	为该路由添加一个注解（Annotation）

4.5.4.3 查看路由

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**网络>路由**。
5. 在**路由**列表，单击目标路由右则的**详情** 查看路由的详情

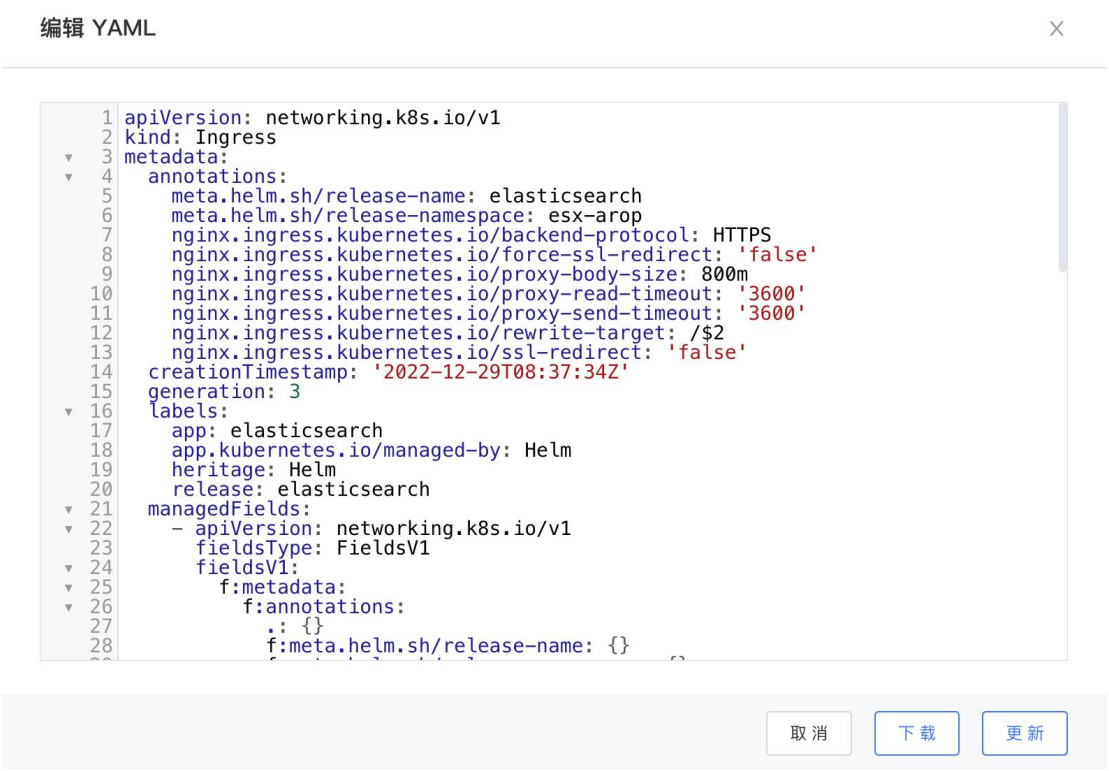


4.5.4.4 更新路由

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

4. 在控制台左侧导航栏中，单击**网络>路由。**

5. 在**路由**列表，单击目标路由右则的**查看 YAML** ，修改 YAML 点击更新



4.5.4.5 删除路由

1. 登录边缘容器集群控制台。

2. 在控制台左侧导航栏中，单击**集群管理。**

3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情。**

4. 在控制台左侧导航栏中，单击**网络>路由。**

5. 在**路由**列表，单击目标路由右则的**删除。**

6. 在弹出的确认框中，单击**确认**即可删除路由。

4.5.5 云边协同版网络管理增强功能

4.5.5.1 为边缘应用配置 Service 流量拓扑

在原生 Kubernetes 集群中，Service 的后端端点扁平分布在集群中任意节点，节点之间 IP 可达。在云边协同版集群架构下，边缘节点分散在不同网络区域，Service 后端端点可能会分散在网络隔离的节点分组上。因此，跨越不同分组节点的 Service 流量，会大概率出现访问不可达、或者访问效率低下的问题。

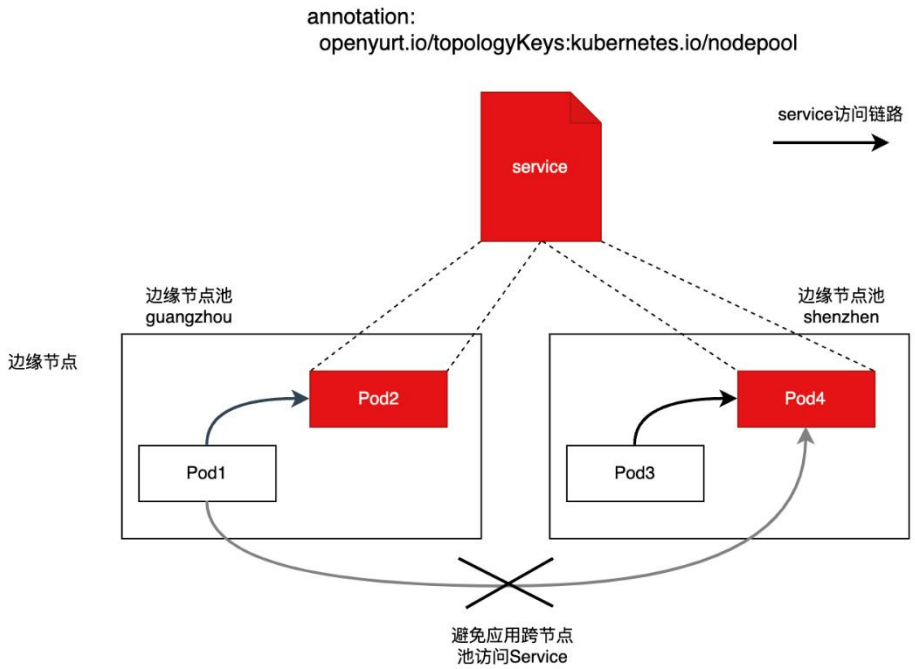
Service 流量拓扑支持边缘节点应用只能由相同节点池的节点访问 ,或者只能由本节点访问。本文为您介绍 Service 流量拓扑管理功能和配置 Service 流量拓扑方法。

背景信息

在边缘计算场景下，边缘节点通常具备很强的区域性、地域性、或者其他逻辑上的分组特性（比如具有相同的 CPU 架构、运营商或云提供商），不同分组的节点间往往存在网络不互通、资源不共享、资源异构、应用独立等明显的隔离属性。

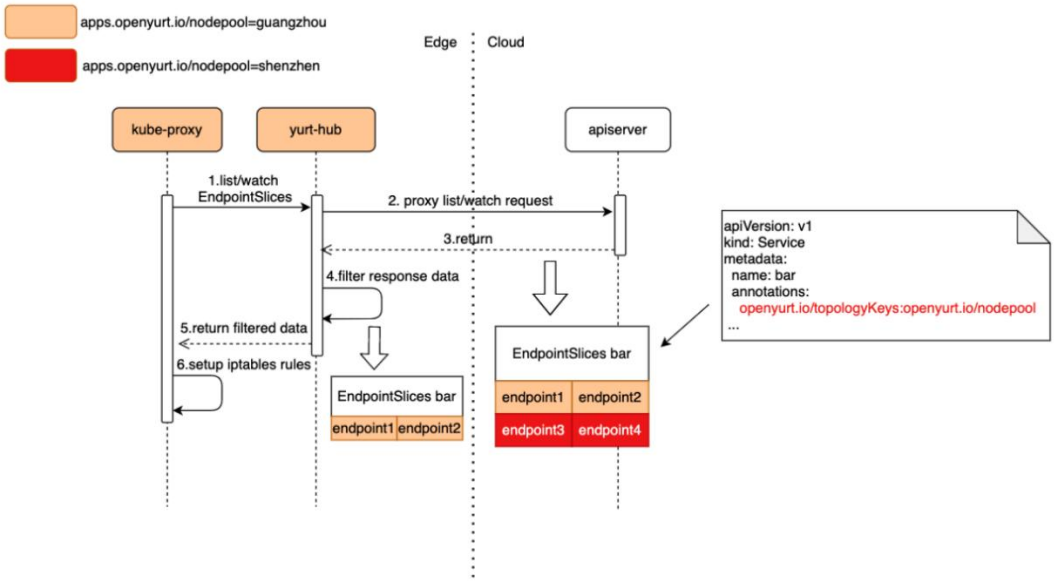
Service 流量拓扑实现原理

为了解决上述问题，ECK 云边协同版在原生的 Service 之上，增加了 Service 流量拓扑管理功能，即通过简单配置来限制 Service 后端 Endpoint 的访问范围，使边缘节点应用只能由相同节点池的节点访问，或者只能由本节点访问。具体实现原理如下图所示。



- Service 关联后端两个实例（Pod2，Pod4），且 Service 通过 annotation："openyurt.io/topologyKeys: kubernetes.io/nodepool" 配置了其拓扑节点池范围。
- Pod2 和 Pod4 分别属于两个不同的节点池 guangzhou 和节点池 shenzhen。
- 因为 Pod1 和 Pod4 不在一个节点池，当 Pod1 访问 Service 时，流量只会转发到 Pod2 上，访问 Pod4 的流量被限制。

实现细节：



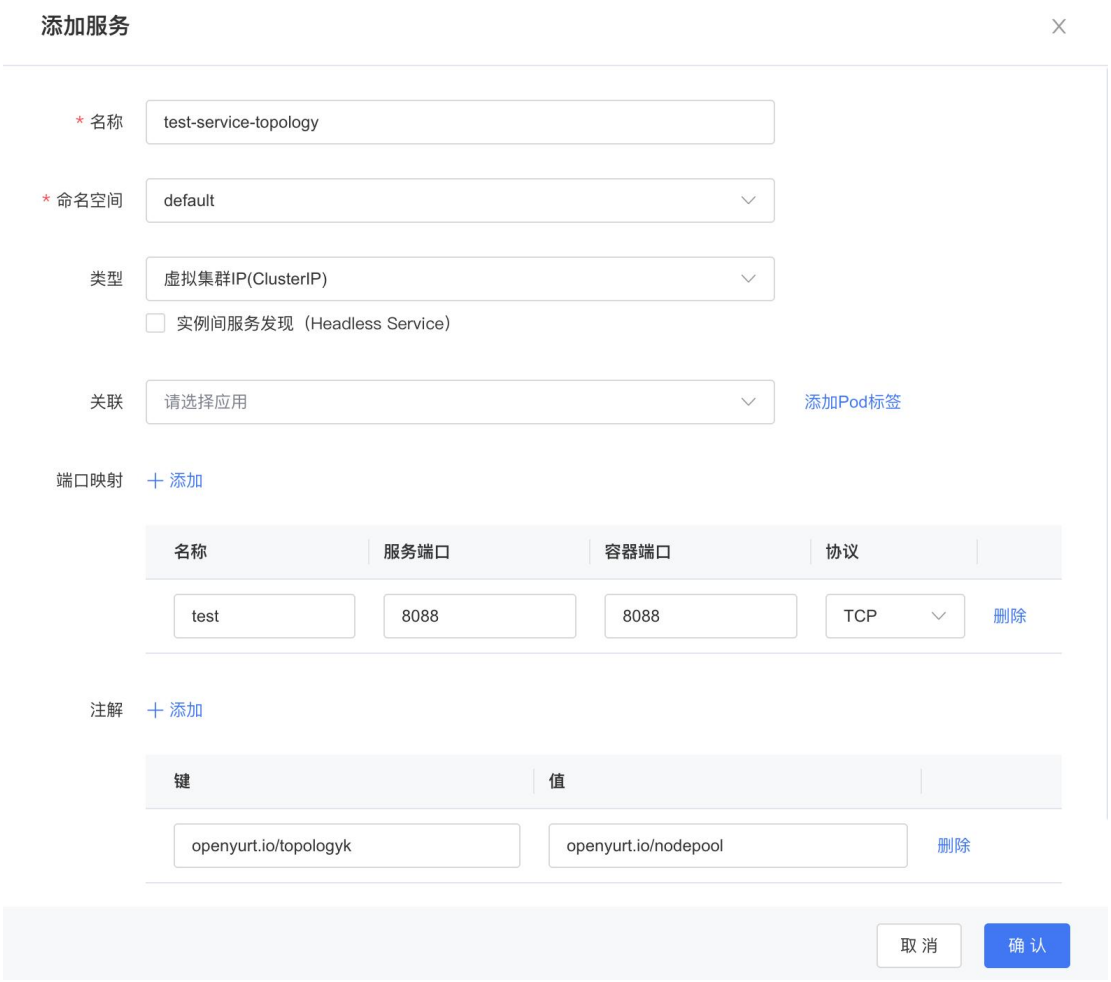
- kube-proxy list/watch EndpointSlices
- 请求经过本地 yurt-hub 代理组件访问集群 APIServer。
- yurt-hub 接收到响应后，基于 Service 中的 openyurt.io/topologyKeys 配置，修改响应，过滤其他节点池的 endpoint，只保本地节点池的 endpoint。
- kube-proxy 收到的响应中只包含本地节点池 endpoint，配置 service 转发规则。 因此转发规则不会跨节点池。

注意事项

创建 Service 时，需要同步配置 Service 的流量拓扑注解，流量拓扑功能才能生效。如果 Service 创建完成后，再增加注解配置，流量拓扑功能无法生效，此时需要删除该 Service，重新创建。

方式一：通过控制台配置 Service 拓扑

若您需要创建一个限制在本节点池内被访问的 Service，只需给 Service 添加注解即可。例如将名称配置为 openyurt.io/topologyKeys，值配置为 openyurt.io/nodepool。



方式二：通过命令行配置 Service 拓扑

新建一个使用节点池拓扑的 Service，YAML 样例如下。

```
apiVersion: v1
kind: Service
metadata:
  name: test-service-topology
  namespace: default
  annotations:
    openyurt.io/topologyKeys: openyurt.io/nodepool
```

```
spec:
  ports:
    - name: test
      port: 8088
      targetPort: 8088
          protocol: TCP

  type: ClusterIP
```

注解说明 ：通过在原生的 Service 上添加 Annotation 实现流量的拓扑配置，相关参数如下所示。

annotation Key	annotation Value	说明
openyurt.io/topologyKeys	kubernetes.io/hostname	限制 Service 只能被本节点访问。
openyurt.io/topologyKeys	openyurt.io/nodepool	限制 Service 只能被本节点池的节点访问。
-	-	Service 不做任何拓扑限制。

4.5.5.2 为边缘应用配置 SLB 负载均衡

在标准版和云边协同版集群中 ,均支持通过设置服务(Service)的 spec.type=LoadBalancer，来配置负载均衡器暴露应用。

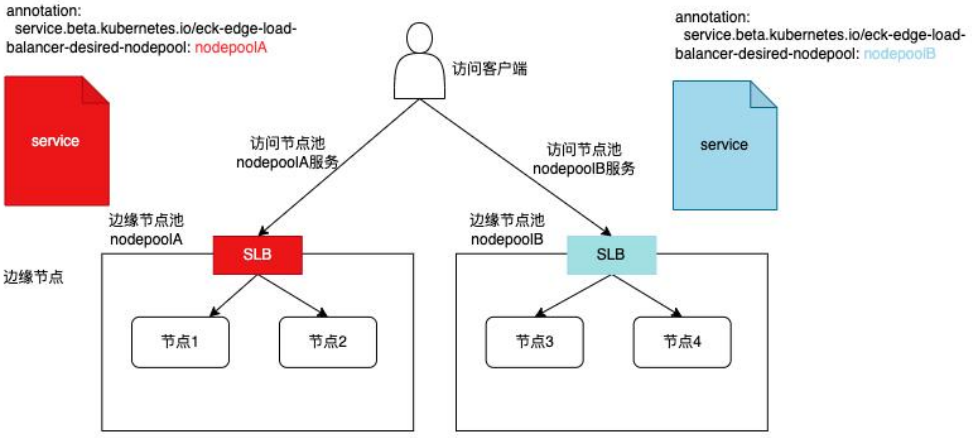
在云边协同版集群中，边缘节点池的节点和云端节点存在跨地域，跨 VPC 的可能。在云边协同版中，会默认为 LoadBalancer 类型 Service 在云端创建 SLB 负载均衡器，当云边节点跨 VPC 时，这种负载均衡器将无法服务于边缘应用。

为了解决上述问题，ECK 云边协同版本支持为边缘应用配置边缘负载均衡。

边缘负载均衡实现原理

SLB 负载均衡器实例需绑定 VPC 使用，且只允许关联同 VPC 下的服务器组（在 ECK 云边协同版中，同一边缘节点池中的节点同属一个 VPC ）。ECK 云边协同版支持通过 Service 上的 annotation 配置，指定 LoadBalancer 类型 Service 所属的节点池，从而指定 SLB 负载均衡器所绑定的 VPC。

集群中的 CCM（cloud-controller-manager）组件能够识别对应配置，为对应 Service 在指定的节点池上开通 SLB 负载均衡器，暴露节点池上的服务。



在上图示例中：

- 节点 1 ,节点 2 和节点 3 ,节点 4 分别属于两个边缘节点池 nodepoolA 和 nodepoolB。

- 用户通过创建两个 LoadBalancer 类型的 Service ，并通过 service.beta.kubernetes.io/eck-edge-load-balancer-desired-nodepool annotation 配置来指定该 Service 所属的节点池。
- CCM 组件基于该 Service 的配置，分别在两个节点池中创建各自的 SLB 负载均衡器，暴露应用。

创建边缘负载均衡服务示例

本示例将向您展示如何创建边缘负载服务。

注意 本示例会创建公网 IP 以及负载均衡器，产生一定的费用。

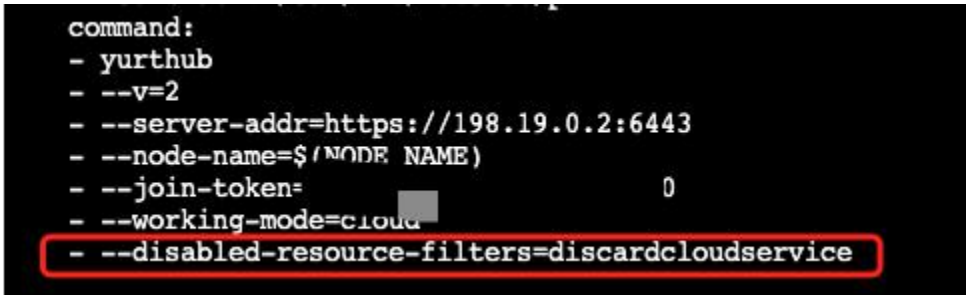
前置准备

运行本示例需要一个至少有 2 个节点池的 ECK 云边协同版本集群。如果您还没有集群，请参见【创建集群】指引，创建云边协同版集群。

如果集群的 kube-proxy 配置的是 ipvs 模式，需要修改对应节点池中所有节点的 yurthub 配置，在启动命令中添加： --disabled-resource-filters=discardcloudservice 配置。

修改方式如下：

修改节点上的 `/etc/kubernetes/manifests/yurt-hub.yaml` 文件，增加 --disabled-resource-filters=discardcloudservice 配置。



注意

节点上的 kubelet, coredns, kube-proxy, flannel 组件通过 yurthub 代理与云端 APIServer 连接， 修改该配置会重启 yurthub 组件，导致连接闪断。

方式一：通过控制台创建边缘负载均衡

您可以通过设置 LoadBalancer 类型 Service 的 annotation 配置 service.beta.kubernetes.io/eck-edge-load-balancer-desired-nodepool 等于对应的节点池 ID，来指定负载均衡器对应的节点池。

您可以通过节点池详情页面查询节点池 ID：

方式二：通过命令行方式创建边缘负载均衡

新建一个使用边缘负载均衡暴露服务的 Servic 的 YAML 样例如下：

```
apiVersion: v1
kind: Service
metadata:
  annotations:
    service.beta.kubernetes.io/eck-edge-load-balancer-desired-nodepool:
c78b61a1-32f0-411f-b2c6-41e3212bc3f8
  name: test-edge-loadbalancer
  namespace: default
spec:
  ports:
    - name: http
      port: 8089
      protocol: TCP
      targetPort: 80
          selector:
    app: nginx-test
```

注解说明

通过在原生的 Service 上添加 Annotation 配置边缘负载均衡，相关参数如下所示。

annotation Key	annotation Value	说明
service.beta.kubernetes.io/eck-edge-load-balancer-desired-nodepool	kubernetes.io/hostname 节点池 ID	当 Service 类型为 LoadBalancer，且集群为云边协同版本时生效。表示在对应节点池上创建负载均衡器暴露 Service。
-	-	Service 不做任何拓扑限制。当 Service 类型为 LoadBalancer，且集群为云边协同版本时 ,表示在云端节点池创建负载均衡器暴露 Service。

4.5.5.3 为边缘节点池部署 Nginx Ingress Controller

基于 ECK 云边协同版本提供的边缘负载均衡能力，您可以通过在边缘节点池部署 Nginx Ingress Controller，为边缘节点池上的应用提供负载均衡能力。

注意事项

在边缘节点池部署 eck-Ingress-nginx 时，注意事项如下：

- 支持在一个边缘容器集群上多次部署，每次部署需要唯一的发布名称。
- 支持在一个边缘容器集群的同一命名空间下多次部署，但为了管理方便，建议每次部署选择独立的命名空间。
- 如果集群的 kube-proxy 配置是 ipvs 模式，请参见【 为边缘应用配置 SLB 负载均衡 】进行调整。

操作步骤

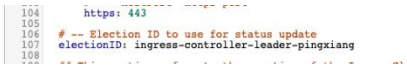
1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击集群管理
3. 在集群列表页面中，单击目标集群右侧操作列下的详情
4. 在控制台左侧导航栏中，单击应用->Helm
5. 在 Helm 应用列表中，单击左上角的的创建 Helm

6. 在应用基本信息配置页，填写应用名称，选择应用部署命名空间，选择 eck-ingress-nginx，点击下一步

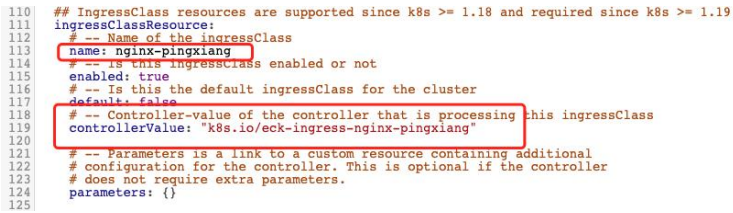
建议发布名称以 eck-ingress-nginx-{节点池名称} 命名，例如：eck-ingress-nginx-pingxiang

7. 在参数配置页面，调整以下参数：

1）修改 controller.electionID，确保每个节点池都有唯一的 controller.electionID。建议增加节点池名称后缀。如：



2）修改 controller.ingressClassResource.name 和 controller.ingressClassResource.controllerValue。建议命名规范：给默认值增加节点池后缀。如：



3）修改 controller.nodeSelector，配置 Label apps.openyurt.io/nodepool：节点池 ID。节点池 ID 可以在节点池详情页面中查看。如：

4）修改 service.annotations 配置，增加 service.beta.kubernetes.io/eck-edge-load-balancer-desired-nodepool：节点池 ID。如：

```
service:
  enabled: true

  # -- If enabled is adding an appProtocol option for Kubernetes service. An appProtocol field replacing annotations that
  # using for setting a backend protocol. Here is an example for AWS: service.beta.kubernetes.io/aws-load-balancer-backen
  # It allows choosing the protocol for each backend specified in the Kubernetes service.
  # See the following GitHub issue for more details about the purpose: https://github.com/kubernetes/kubernetes/issues/40
  # Will be ignored for Kubernetes versions older than 1.20
  ##
  appProtocol: true

  annotations:
    service.beta.kubernetes.io/eck-edge-load-balancer-desired-nodepool: "a1b34ca8b6839668848180cb8e01f5db"

  labels: {}
  # clusterIP: ""
```

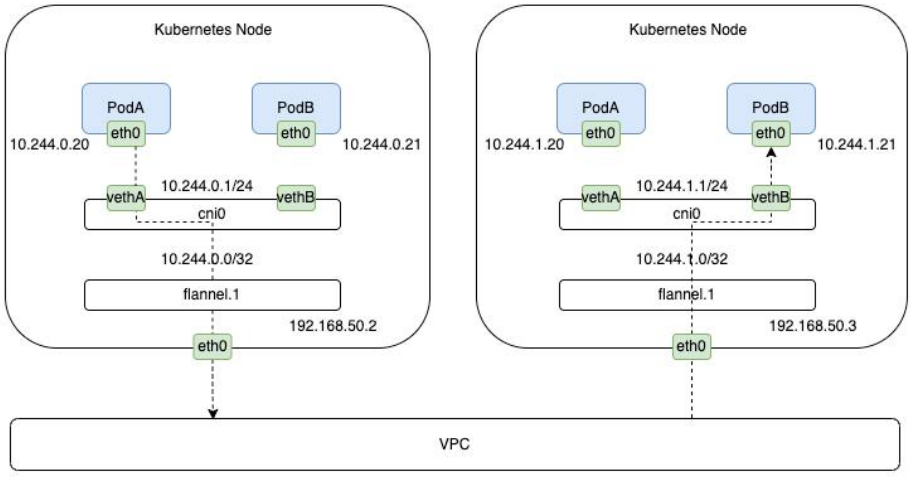
4.5.6 容器网络 CNI

4.5.6.1 Flannel 网络插件

Flannel 是为容器集群设计的一种简单易用的容器网络解决方案，将所有的容器都组织在同一个虚拟大二层网络中，实现集群内容器间、容器与节点间的通信。ECK 支持 Flannel VXLAN 模式。

网络流量路径说明

在 Flannel 网络模式中，Pod 的网段与 VPC 的网段相互独立。Pod 的网段会根据掩码（节点 IP 数量）在每个集群节点上均匀划分，每个节点上的 Pod 会从其所在节点的网段中分配 IP 地址。下图是对 Flannel 模式下，容器间网络流量路径的说明：



- 节点内容器间通信时，通过 cni0 网桥完成，无需进行 VXLAN 封装、解封装。

- 跨节点容器通信时，在源节点封装 VXLAN 报文，通过 VPC 发送到对端节点，对端节点对 VXLAN 报文解封装后，发送给接收容器。

4.6 配置管理

4.6.1 配置项

4.6.1.1 创建配置项

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**配置管理>配置项**。
5. 在**配置项**列表，在左上角选择**命名空间**并单击**创建配置项**
6. 在创建页面输入配置信息，然后单击**确定**即可添加配置项。

配置项	描述
配置项名称	指定配置项的文件名，名称可以包含小写字母、数字、连字符（-）或者半角句号（.），名称不能为空。其他资源对象需要引用配置文件名来获取

	配置信息。
命名空间	设置配置项所在的命名空间。
配置项	填写配置项名称和配置项的值。

4.6.1.2 修改配置项

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**配置管理>配置项**。
5. 在**配置项**列表，单击目标服务右则的**编辑**
6. 在编辑页面对数据进行编辑，然后单击**确定**。

4.6.1.3 删除配置项

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**配置管理>配置项**。
5. 在**配置项**列表，单击目标服务右则的**删除**
6. 在弹出的确认框中，单击**确认**即可删除配置项

4.6.2 保密字典

4.6.2.1 创建保密字典

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**配置管理>保密字典**。
5. 在**保密字典**列表，在左上角选择**命名空间**并单击**创建保密字典**
6. 输入保密字典名称，配置保密字典参数，参数如下：

配置项	描述
名称	设置保密字典的名称，名称长度 1-253 字符，只能包含小写字母、数字、中划线(-)和小数点〔.)
类型	包含 Opaque、私有镜像仓库登录密钥以及 TLS 证书。
Opaque	若类型选择为 Opaque，则需要配置以下参数： 可选：若您输入保密字典的明文数据，请选中对数据值进行 base64 编码。 配置保密字典的数据。单击+ 添加，在名称和值文本框中，输入保密字典名称和值。
私有镜像仓库登录密钥	若类型选择为私有镜像仓库登录密钥，则需要配置以下参数： **镜像仓库地址**：输入保密字典所在镜像仓库地址。 **用户名**：输入镜像仓库的用户名。 **密码**：输入镜像仓库的密码。
TLS 证书	若类型选择为 TLS 证书，则需要配置以下参数： **Cert**：输入 TLS 证书。 **Key**：输入 TLS 证书 Key。

8. 配置完成后，点击**确定**即可创建保密字典。

4.6.2.2 修改保密字典

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**配置管理>保密字典**。
5. 在**保密字典**列表，单击目标服务右则的**编辑**
6. 在编辑页面对数据进行编辑，然后单击**确定**。

4.6.2.3 删除保密字典

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**配置管理>保密字典**。
5. 在**保密字典**列表，单击目标服务右则的**删除**
6. 在弹出的确认框中，单击**确认**即可删除**保密字典**

4.7 存储管理

4.7.1 存储概述

ECK 存储目前支持两种方式创建存储，一种是使用 Kubernetes 的标准存储接口（CSI），一种使用 Kubernete 原生的文件存储（NFS）。

- 容器存储接口（CSI）：融合了 ECX（智能边缘云）的存储资源（云盘），并且兼容了 Kubernetes 的原生的存储服务，比如 EmptyDir、HostPath、Secret、ConfigMap 等存储。需要安装 CSI 组件。
- 文件存储（NFS）：使用原生 Kubernetes 访问 NFS 服务器，将服务器的共享文件挂载到容器中。这种方式不需要安装 CSI 组件。

以下是这两种存储方式的特点和应用场景：

存储驱动	**存储类型**	**特点**
原生驱动	NFS	无需安装 csi 插件，只需要标准 kubernetes 集群就可以使用。 支持多个节点，多个容器共享目录。 仅支持文件存储。 需要自行创建一个 NFS 的服务器，通过 ip/域名+端口的方式暴露出来。 需要所有节点都能够访问到响应的 nfs 服务器。
CSI	云盘	需要安装 eck-csi 组件。安装流程可以到组件管理查看相关帮助文档。 使用 kubernetes 标准的存储接口，融合 ecx 的云盘资源，实现的存储服务。 支持块存储和文件存储功能。 eck 标准专有版和 eck 云边协同版的存储服务。

存储驱动	存储类型	特点
		仅仅支持单个节点的同一个 deployment 的 pod 使用。

4.7.2 安装与升级 CSI 组件

安装组件

1. 登录边缘容器集群控制台。

2. 在控制台左侧导航栏中，单击集群管理。

3. 在集群列表页面中，单击目标集群右侧操作列下的详情。

4. 在控制台左侧导航栏中，单击运维管理>组件管理。

5. 找到 eck-csi 组件，点击安装。

升级组件

组件如果有新的版本上线，在组件管理界面可以看到**升级**按钮， 点击**升级**即可开始升级组件。

4.7.3 存储类管理

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧操作列下的**详情**。
4. 在控制台左侧导航栏中，单击**存储>存储类**。
5. 点击左上方的**创建存储类**按钮，填写对应参数

配置项	描述
存储驱动	仅仅在安装支持 CSI 的时候可以创建。如果没有安装 csi 插件，会有提示
存储类型	云盘，对应的是 ECX 边缘集群的边缘存储。
名称	填写自定义名称
区域	可选集群所有节点的区域。可以在指定区域创建存储。响应的容器也会部署到相应的区域。 • 如果是 ECK 标准版，区域便是 master 所在的区域； • 如果是 ECK 云边协同版本，区域便是所有节点池所在的区域。 • 无论哪种版本，都有一个随机区域的选项。表示会在调度容器的时候，才会确定到哪个区域创建存储。
回收策略	• Retain: 删除 pv 的时候，会保留底层云盘 • Delete: 删除 pv 的时候，会删除底层云盘
绑定策略	• Immediate： 创建 pvc 的时候，就会立即创建相应的 pv

	• WaitForFirstConsumer: 创建 pvc 的时候 ,需要等到 pod 具体调度到哪个节点，才会创建相应的 pv

6. 点击确定。就会看到创建好的存储类

4.7.4 存储声明管理

4.7.4.1 创建 NFS 存储声明

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理。**
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情。**
4. 在控制台左侧导航栏中，单击**存储>存储声明。**
5. 在存储声明列表，单击**创建存储声明。**
6. 在弹出的创建存储声明对话框选择**原生驱动**，并配置参数。

选择已有存储卷

配置项	描述
存储类型	目前支持 NFS 存储类型
名称	数据声明名称在集群内必须唯一，为 1~63 个字符,支持小写字母、数字、"-"以及"."等字符
分配模式	可选择已有存储卷或创建存储卷，创建存储卷可参考 存储卷管理-创建存储卷
存储卷	使用已有存储卷模式则选择一个在存储卷管理-创建存储卷 中创建的存储卷
总量	存储声明的容量，所创建存储卷声明的容量不能超过待挂载的存储卷容量

选择创建存储卷

配置项	描述
存储类型	目前支持 NFS 存储类型
名称	数据卷名在集群内必须唯一，为 1~63 个字符,支持小写字母、数字、"-"以及"."等字符
总量	存储卷的容量声明

配置项	描述
访问模式	支持 ReadWriteMany、ReadWriteOnce 和 ReadOnlyMany
服务器地址	存储服务器地址，如:192.168.0.1 或 server-name
共享目录	共享目录，如：/data
标签	为存储卷添加标签

7. 参数配置完成后，单击**创建。**创建成功后，可在创建工作负载时数据卷-云存储中使用。

4.7.4.2 创建云盘存储声明

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击 **集群管理** 。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的 **详情** 。
4. 在控制台左侧导航栏中，单击 **存储>存储声明** 。
5. 在存储声明列表，单击创**建存储声明。**
6. 存储驱动选择 **CSI。**
7. 在弹出的创建存储声明对话框配置参数

- 选择使用存储类动态创建

点击**选择存储类**按钮。选择已经创建好的区域的存储类。设置需要的存储容量。最小 20G。

- **选择已有存储卷**

点击选择已经释放的存储卷(删除 claimRef 相关内容)，总量会使用 pv 里面的大小回填

- **选择创建存储卷**

选择 没有被绑定的 ecx 云盘 ，选择文件系统类型，配置好之后点击确定。

4.7.4.1 查看存储声明是否可用

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击 **集群管理** 。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的 **详情** 。
4. 在控制台左侧导航栏中，单击 **存储>存储声明** 。
5. 在列表中，查看状态为 **Bound** 的 pvc，表示可以提供给容器使用。

如果是 **Pending** 的状态 ，并且存储类的绑定策略为 **WaitForFirstConsumer** ，这种情况也是可以提供给容器使用，但需要您确保容器调度到具有云盘的节点

注意：创建存储声明的时候，选择使用存储类动态创建会校验是否支持云盘，随机区域的存储类不支持检查云盘。

4.7.5 存储卷管理

4.7.5.1 创建 NFS 存储卷

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。

3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情。**

4. 在控制台左侧导航栏中，单击**存储>存储卷。**

5. 在**存储卷**列表，单击左上角的**创建存储卷**

6. 在弹出的创建存储卷对话框配置参数，存储驱动选择**原生驱动**，存储类型选择 **NFS**，其他参数根据需求配置。

配置项	描述
存储类型	目前支持 NFS 存储类型
名称	数据卷名在集群内必须唯一，为 1~63 个字符,支持小写字母、数字、"-"以及"."等字符
总量	存储卷的容量声明

访问模式	支持 ReadWriteMany、ReadWriteOnce 和 ReadOnlyMany
服务器地址	存储服务器地址，如:192.168.0.1 或 server-name
共享目录	共享目录，如：/data
标签	为存储卷添加标签

7. 参数配置完成后，单击**确定**。

4.7.5.2 创建云盘存储卷

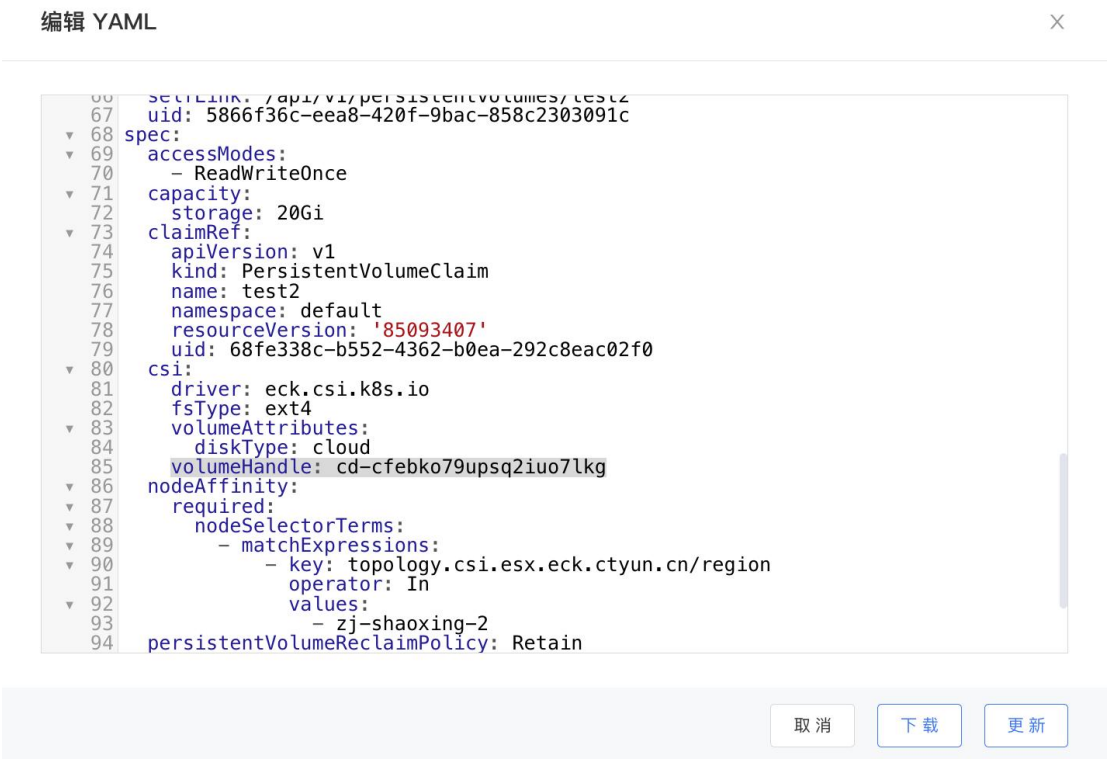
1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在集群列表页面中，单击目标集群右侧操作列下的**详情**。
4. 在控制台左侧导航栏中，单击**存储>存储卷**。
5. 在存储卷列表，单击左上角的**创建存储卷**。
6. 在弹出的创建存储卷对话框配置参数，存储驱动选择 **CSI**，存储类型选择**云盘**，其他参数根据需求配置。

配置项	描述
名称	存储卷名在集群内必须唯一 ,名称为 1~63 个字符,支持小写字母、数字、"-"以及"."等字符。
访问模式	支持 ReadWriteOnce。
云盘 ID	选择未被绑定可以使用云盘。
文件系统类型	支持 ext4、ext3、xfs、vfat 等类型。
标签	点击添加按钮，为存储卷添加标签。

7.参数配置完成后，单击**确定** 。

4.7.5.2 查看存储卷是否可用

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击 **集群管理** 。
3. 在集群列表页面中，单击目标集群右侧操作列下的 **详情** 。
4. 在控制台左侧导航栏中，单击 **存储>存储卷** 。
5. 在存储卷列表，单击左上角的**创建存储卷**
6. 点击查看 YAML，找到 **spec.csi. volumeHandle**，如果存在一个 cd-的值，并且能够在 ecx 的边缘存储的云硬盘列表能够找到对应的硬盘，就表示 pv 已经成功关联了 ecx 边缘存储。



4.7.6 CSI 存储操作

4.7.6.1 动态创建存储卷

本操作适用于需要使用特定区域的边缘存储，并且交给存储类去完成 PV 创建的场景，您只需要创建 PVC，就可以直接在容器里面使用存储，eck-csi 存储组件会帮助您完成底层存储和磁盘挂载的操作。

1.登录边缘容器集群控制台。

2.在控制台左侧导航栏中，单击 **集群管理** 。

3.在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情** 。

4.在控制台左侧导航栏中，单击**存储>存储类**，创建一个存储类，注意需要选择指定区域。这个区域能够帮助判断是否可以在特定区域使用边缘存储。

5.选择上一步创建好的存储类(比如 tests)，创建 PVC。

6.查看 PVC 列表可以看到 tests-pvc 处于 Pending 中，刷新列表之后，绑定了关联的存储类 pvc-0f3f1d9e-0acf-4a0e-a5d4-8fb1f43e62e5， 这就表示可以使用这个 PVC 了。


创建存储声明
存储驱动
原生驱动
CSI
存储类型
云盘
* 名称
tests-pvc
* 命名空间
default
分配模式
使用存储类动态创建
已有存储卷
创建存储卷
* 访问模式
ReadWriteOnce
* 已有存储类
选择存储类
tests
* 总量
20
Gi
取 消
转yaml
确 定


命名空间
default
+ 创建存储声明
+ YAML 创建资源
名称
容量
访问模式
状态
存储类型
关联的存储卷
创建时间
操作
test-random
20Gi
ReadWriteOnce
Pending
eck-csi
--
2023-04-11 09:35:55
详情 | 查看YAML | 删除
tests
20Gi
ReadWriteOnce
Bound
tests-2
pvc-0a787965-818a-4e24-bd93-5eefccc9ac60
2023-04-10 20:38:57
详情 | 查看YAML | 删除
tests-pvc
20Gi
ReadWriteOnce
Pending
tests
--
2023-04-11 11:45:00
详情 | 查看YAML | 删除

4.7.6.2 使用已有存储卷

当您需要复用一些未被绑定的 PV，同时又不想去创建新的存储资源时，可以使用已有存储卷。

情况一：您已经创建了 PV。

1.登录边缘容器集群控制台。

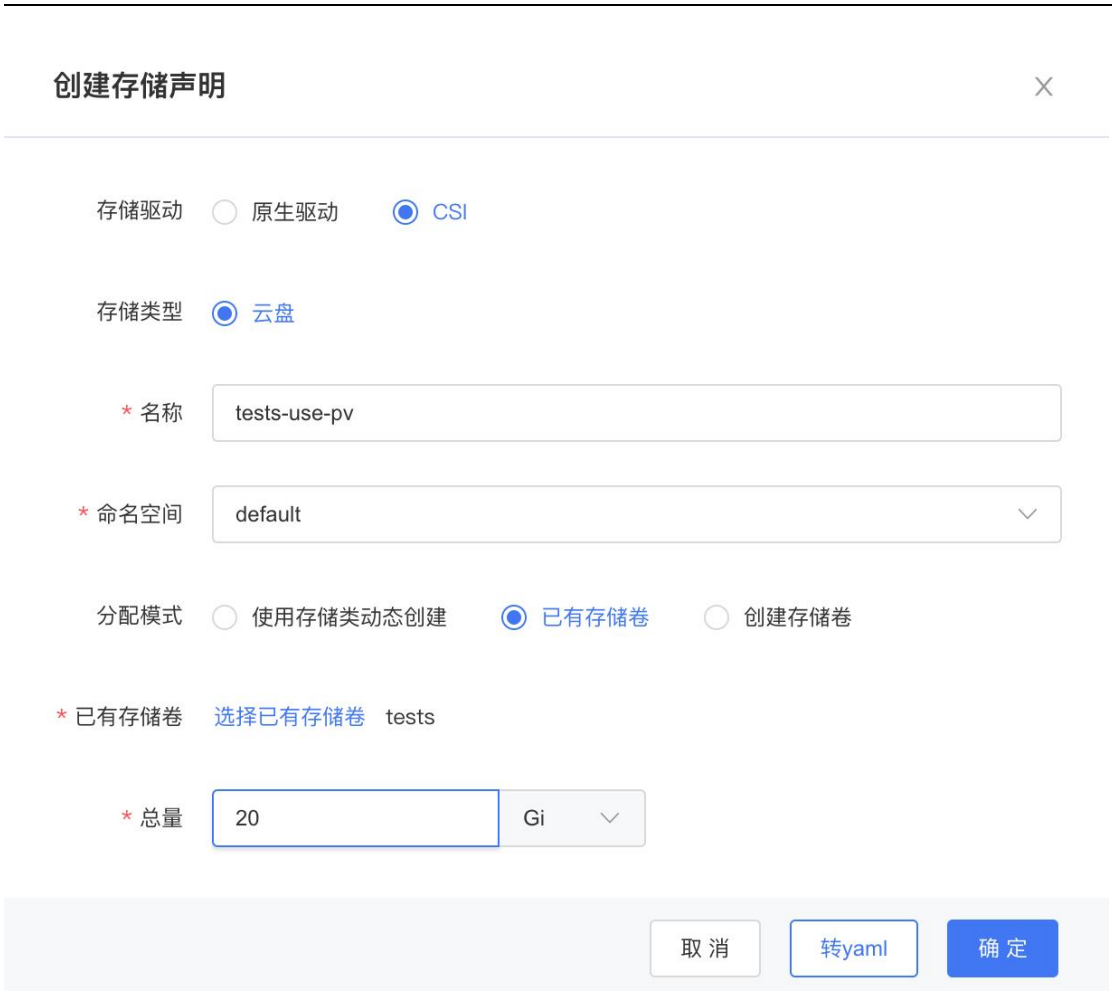
2.在控制台左侧导航栏中，单击 **集群管理** 。

3.在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情** 。

4.在控制台左侧导航栏中，单击**存储>存储声明**

5.在存储声明列表点击**创建存储声明**，并选择已有存储卷(如 tests)，点击**确定**完成创建。

6.创建后查看存储声明列表，看到了一个 test-use-pv 的状态未 Bound，表示 PVC 已经绑定成功，可以提供给容器使用。



情况二 :您已经绑定了 PVC 和 PV ,但是想换另一个 PVC 绑定相应之前已经绑定过的 PVC。

1.登录边缘容器集群控制台。

2.在控制台左侧导航栏中，单击 **集群管理** 。

3.在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情** 。

4.在控制台左侧导航栏中，单击**存储>存储卷**，在列表看到一个状态为 Released 的 PVC，点击**查看 YAML**， 可以将 spec.claimRef .直接删除，点击更新。

5.查看列表，状态变为了 Available，这时候就可以绑定了。

6.执行情况一，创建 PVC，查看列表，可以看到已经绑定成功的 test-bound。

4.7.6.3 使用已有边缘存储

ECX 边缘存储平台有一个未绑定的云硬盘，您想要复用到 ECK 集群中。

情况一：您想通过 pv 绑定云硬盘

先创建 pv，然后选择云盘绑定

情况二： 直接创建 pvc，同时也创建 pv。当您看到 pvc 和 pv 的状态都是 bound 的时候，就可以使用。

4.7.7 容器绑定存储

1.使用 deployment 绑定存储：其中 persistentVolumeClaim 需要填写已经创建好的 pvc

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

```
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: deployment-eck-ttt
spec:
  replicas: 1
  selector:
    matchLabels:
      name: deployment-eck
  template:
    metadata:
      name: deployment-eck
      labels:
        name: deployment-eck
    spec:
      nodeSelector:
        "kubernetes.io/os": linux
      containers:
        - name: deployment-eck
          image: ehub.ctcdn.cn/eck/nginx
          command:
            - "/bin/bash"
            - "-c"
            - set -euo pipefail; while true; do echo $(hostname) $(date) >>
/mnt/eck/outfile; sleep 1; done
          volumeMounts:
            - name: eck
              mountPath: "/mnt/eck"
      volumes:
        - name: eck
                persistentVolumeClaim:
          claimName: pvc-eck-dynamic
```

2.使用 statefulset 绑定存储：

有状态的存储也是有状态的，最好使用 volumeClaimTemplates，去创建 pvc。没有容器实例会绑定一个特定 pvc

```
---
```

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: statefulset-eck
  labels:
    app: nginx
spec:
  serviceName: statefulset-eck
  replicas: 1
  template:
    metadata:
      labels:
        app: nginx
    spec:
      nodeSelector:
        "kubernetes.io/os": linux
      containers:
        - name: statefulset-eck
          image: ehub.ctcdn.cn/eck/nginx
          command:
            - "/bin/bash"
            - "-c"
            - set -euo pipefail; while true; do echo $(date) >> /mnt/eck/outfile;
sleep 1; done
          volumeMounts:
            - name: persistent-storage
              mountPath: /mnt/eck
  updateStrategy:
    type: RollingUpdate
  selector:
    matchLabels:
      app: nginx
  volumeClaimTemplates:
    - metadata:
        name: persistent-storage
        annotations:
          volume.beta.kubernetes.io/storage-class: eck-csi
      spec:
        accessModes: ["ReadWriteOnce"]
        resources:
                requests:

            storage: 20Gi
```

4.8 命名空间管理

4.8.1 创建命名空间

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**命名空间**。
5. 在**命名空间**列表，单击左上角的的**创建命名空间**
6. 在创**建命名空间**对话框配置命名空间，然后单击**确定**。

配置项	描述
名称	设置保您需要设置命名空间的名称，长度为 1 ~ 63 个字符，只能包含数字、小写字母和中划线（-），且首尾只能是字母或数字
标签	可为命名空间添加多个标签。标签用于标识该命名空间的特点

4.8.2 编辑命名空间

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**命名空间**。
5. 在**命名空间**页面单击目标命名空间右侧**操作**列的**编辑**。
6. 在弹出的对话框中，单击**编辑**，对命名空间的标签进行修改，然后单击**确定**。

4.8.3 删除命名空间

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。

3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情。**

4. 在控制台左侧导航栏中，单击**命名空间。**

5. 单击目标命名空间右侧**操作**列的**删除。**

6. 在**提示**对话框中，单击**确定。**

4.9 应用管理

4.9.1 工作负载管理

4.9.1.1 无状态 Deployment

4.9.1.1.1 使用镜像创建 Deployment

1. 登录边缘容器集群控制台。

2. 在控制台左侧导航栏中，单击**集群管理。**

3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情。**

4. 在控制台左侧导航栏中，单击**工作负载>无状态。**

5. 在**无状态**负载列表，单击左上角的**创建无状态负载。**

6. 在应用配置页面，设置应用的配置信息

应用基本信息

配置项	**描述**
应用名称	设置应用的名称，名称为 1~63 个字符，支持小写字母、数字、"-"以及"."等字符
命名空间	设置应用所在的命名空间。默认命名空间为 default
副本数量	应用包含的 Pod 数量，默认数量为 2
类型	定义资源对象的类型，目前可选择**无状态**
标签	为应用添加标签，标识该应用
注解	为应用添加标签

容器配置

配置项	描述
容器镜像	镜像类型：可以选择公有镜像或私有镜像 公有镜像：填入需要使用的公有镜像地址 私有镜像：选择上传到镜像仓库的私有镜像 支持以下三种镜像拉取策略（imagePullPolicy）： **优先使用本地镜像（IfNotPresent）**：如果本地有该镜像（之前拉取过该镜像至宿主机中），则使用本地镜像，本地不存在时拉取镜像。 **总是拉取镜像（Always）**：表示每次部署或扩容都会从容器镜像服务重新拉取镜像，而不会从本地拉取镜像。 **仅使用本地镜像（Never）**：仅使用本地镜像。 单击**设置镜像密钥**，您可以实现免密拉取镜像
资源限制	可指定该应用所能使用的资源上限，包括 CPU、内存和 GPU 三种资源，防止占用过多资源。
所需资源	即为该应用预留资源额度，包括 CPU、内存和 GPU 三种资源，即容器独占该资源，防止因资源不足而被其他服务或进程争夺资源，导致应用不可用
容器启动项	stdin：表示为该容器开启标准输入 tty：表示为该容器分配一个虚拟终端，以便于向容器发送信号。 通常这两个选项是一起使用的，表示将终端（tty）绑定到容器的标准输入（stdin）上。例如，一个交互式的程序从用户获取标准输入，并显示到终端中
特权容器	选择特权容器，则 privileged=true，开启特权模式。 不选择特权容器，则 privileged=false，关闭特权模式。

Init Container	勾选该项，表示创建一个 Init Container

可选：在**端口设置**区域，单击**新增**设置容器的端口。

配置项	描述
名称	设置容器端口名称。
容器端口	设置暴露的容器访问端口或端口名，端口号必须介于 1~65535。
协议	支持 TCP 和 UDP。

可选：在**环境变量**区域，单击**新增**设置环境变量。

支持通过键值对的形式为 Pod 配置环境变量。用于给 Pod 添加环境标志或传递配置等。

配置项	描述
类型	设置环境变量的类型，支持： **自定义** **配置项** **保密字典** **变量/变量引用** **资源引用** 配置项、保密字典支持全部文件的引用，以保密字典为例。选择**保密字典**类型，只选择目标保密字典，则默认引用全部文件。
变量名称	设置环境变量名称。
变量/变量引用	设置变量引用的值。

可选：在**健康检查**区域，根据需要开启**存活检查**、**就绪检查**及**启动探测**。

存活检查（Liveness）：用于检测何时重启容器。

就绪检查（Readiness）：确定容器是否已经就绪，且可以接受流量。

启动探测（Startup Probes）：用于检测何时启动容器。

请求类型	描述
HTTP 请求	即向容器发送一个 HTTP　Get 请求，支持的参数包括： **协议**：HTTP/HTTPS。 **路径**：访问 HTTP Server 的路径。 **端口**：容器暴露的访问端口或端口名，端口号必须介于 1~65535。 **HTTP 头**：即 HTTP Headers，HTTP 请求中自定义的请求头，HTTP 允许重复的 Header。支持键值对的配置方式。 **延迟探测时间（秒）**：即 initialDelaySeconds，容器启动后第一次执行探测时需要等待多少秒，默认为 3 秒。 **执行探测频率（秒）**：即 periodSeconds，指执行探测的时间间隔，默认为 10 秒，最小为 1 秒。 **超时时间（秒）**：即 timeoutSeconds，探测超时时间。默认 1 秒，最小 1 秒。 健康阈值：探测失败后，最少连续探测成功多少次才被认定为成功。默认是 1，最小值是 1。对于存活检查（liveness）必须是 1。 **不健康阈值**：探测成功后，最少连续探测失败多少次才被认定为失败。默认是 3，最小值是 1。
TCP 连接	即向容器发送一个 TCP Socket，Kubelet 将尝试在指定端口上打开容器的套接字。　如果可以建立连接，容器被认为是健康的，如果不能就认为是失败的。支持的参数包括： **端口**：容器暴露的访问端口或端口名，端口号必须介于 1~65535。 **延迟探测时间（秒）**：即 initialDelaySeconds，容器启动后第一次执行探测时需要等待多少秒，默认为 15 秒。

	执行探测频率（秒）：即 periodSeconds，指执行探测的时间间隔，默认为 10 秒，最小为 1 秒。 **超时时间（秒）**：即 timeoutSeconds，探测超时时间。默认 1 秒，最小 1 秒。 **健康阈值**：探测失败后，最少连续探测成功多少次才被认定为成功。默认是 1，最小值是 1。对于存活检查（liveness）必须是 1。 **不健康阈值**：探测成功后，最少连续探测失败多少次才被认定为失败。默认是 3，最小值是 1。
命令行	通过在容器中执行探针检测命令，来检测容器的健康情况。支持的参数包括：**命令行**：用于检测容器健康情况的探测命令。 **延迟探测时间（秒）**：即 initialDelaySeconds，容器启动后第一次执行探测时需要等待多少秒，默认为 5 秒。 **执行探测频率（秒）**：即 periodSeconds，指执行探测的时间间隔，默认为 10 秒，最小为 1 秒。 **超时时间（秒）**：即 timeoutSeconds，探测超时时间。默认 1 秒，最小 1 秒。 **健康阈值**：探测失败后，最少连续探测成功多少次才被认定为成功。默认是 1，最小值是 1。对于存活检查（liveness）必须是 1。 **不健康阈值**：探测成功后，最少连续探测失败多少次才被认定为失败。默认是 3，最小值是 1。

可选：在**生命周期**区域，设置容器的生命周期。

您可以为容器的生命周期配置启动执行、启动后处理和停止前处理。

配置项	**描述**
启动执行	为容器设置预启动命令和参数。

启动后处理	为容器设置启动后的命令。
停止前处理	为容器设置预结束命令。

可选：在**数据卷**区域，增加本地存储或云存储声明 PVC（Persistent Volume Claim）。

配置项	描述
本地存储	本地存储：支持主机目录（HostPath）、配置项（ConfigMap）、保密字典（Secret）和临时目录，将对应的挂载源挂载到容器路径中。
云存储声明	支持通过 PVC 挂载云存储卷。在选择目标挂载源前，您需要创建云存储声明。

高级配置。

在**高级配置**配置向导页面中设置访问、伸缩、调度和标签注解。

在**访问设置**区域，通过设置**服务**暴露后端 Pod

配置服务（Service）：

配置项	描述
名称	输入服务的名称。
类型	选择服务访问的方式 **虚拟集群 IP**：即 ClusterIP，指通过集群的内部 IP 暴露服务，选择该值，服务只能够在集群内部可以访问，这也是默认的 ServiceType。 **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，该 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

实例间发现服务	服务类型为**虚拟集群 IP** 时，才能设置**实例间发现服务（Headless Service）**。
外部流量策略	**Local**：流量只发给本机的 Pod。 **Cluster**：流量可以转发到其他节点上的 Pod。 服务类型为**节点端口**或**负载均衡**时，才能设置**外部流量策略。**
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。
标签	为该服务添加一个标签，标识该服务
注解	为该服务添加一个注解（Annotation），配置负载均衡的参数

可选：在**伸缩配置**区域，配置是否开启**指标伸缩。**

指标伸缩：根据容器 CPU 和内存 占用情况自动调整容器组数量。

配置项	**描述**
指标	支持 CPU 和内存，需要和设置的所需资源类型相同。
触发条件	资源使用率的百分比，超过该使用量，容器开始扩容。
最大副本数量	该负载类型可扩容的容器数量上限。
最小副本数量	该负载类型可缩容的容器数量下限。

可选：在**调度设置**区域，设置**升级方式**、**节点亲和性**、**应用亲和性**、**应用非亲和性 和 调度容忍**

配置项	**描述**
升级方式	升级方式包括滚动升级（rollingupdate）和替换升级（recreate）

节点亲和性	设置节点亲和性，通过 Worker 节点的 Label 标签进行设置。节点调度支持硬约束和软约束（Required/Preferred），以及丰富的匹配表达式（In, NotIn, Exists, DoesNotExist. Gt, and Lt）： **必须满足**，即硬约束，一定要满足，对应 requiredDuringSchedulingIgnoredDuringExecution，效果与 NodeSelector 相同。本例中 Pod 只能调度到具有对应标签的 Worker 节点。您可以定义多条硬约束规则，但只需满足其中一条。 **尽量满足**，即软约束，不一定满足，对应 preferredDuringSchedulingIgnoredDuringExecution。调度会尽量调度 Pod 到具有对应标签的 Node 节点。您还可为软约束规则设定权重，具体调度时，若存在多个符合条件的节点，权重最大的节点会被优先调度。您可定义多条软约束规则，但必须满足全部约束，才会进行调度。
应用亲和性	决定应用的 Pod 可以和哪些 Pod 部署在同一拓扑域。例如，对于相互通信的服务，可通过应用亲和性调度，将其部署到同一拓扑域（如同一个主机）中，减少它们之间的网络延迟。 根据节点上运行的 Pod 的标签（Label）来进行调度，支持硬约束和软约束，匹配的表达式有：In, NotIn, Exists, DoesNotExist。 **必须满足**，即硬约束，一定要满足，对应 requiredDuringSchedulingIgnoredDuringExecution，Pod 的亲和性调度必须要满足后续定义的约束条件 **命名空间**：该策略是依据 Pod 的 Label 进行调度，所以会受到命名空间的约束。 **拓扑域**：即 topologyKey，指定调度时作用域，这是通过 Node 节点的标签来实现的，例如指定为 kubernetes.io/hostname，那就

	是以 Node 节点为区分范围；如果指定为 beta.kubernetes.io/os，则以 Node 节点的操作系统类型来区分。 **选择器**：单击选择器右侧的加号按钮，您可添加多条硬约束规则。 **查看应用列表**：单击应用列表，弹出对话框，您可在此查看各命名空间下的应用，并可将应用的标签导入到亲和性配置页面。 **硬约束条件**：设置已有应用的标签、操作符和标签值。本例中，表示将待创建的应用调度到该主机上，该主机运行的已有应用具有 app:nginx 标签。 **尽量满足**，即软约束，不一定满足，对应 preferredDuringSchedulingIgnoredDuringExecution。Pod 的亲和性调度会尽量满足后续定义的约束条件。对于软约束规则，您可配置每条规则的权重，其他配置规则与硬约束规则相同 **权重**：设置一条软约束规则的权重，介于 1~100，通过算法计算满足软约束规则的节点的权重，将 Pod 调度到权重最大的节点上
应用非亲和性	决定应用的 Pod 不与哪些 Pod 部署在同一拓扑域。应用非亲和性调度的场景包括： 将一个服务的 Pod 分散部署到不同的拓扑域（如不同主机）中，提高服务本身的稳定性。 给予 Pod 一个节点的独占访问权限来保证资源隔离，保证不会有其它 Pod 来分享节点资源。 把可能会相互影响的服务的 Pod 分散在不同的主机上。
调度容忍	容忍被应用于 Pod，允许这个 Pod 被调度到相对应的污点上。

可选：在**标签和注释**区域，单击**添加**设置容器组的标签和注释。

配置项	**描述**

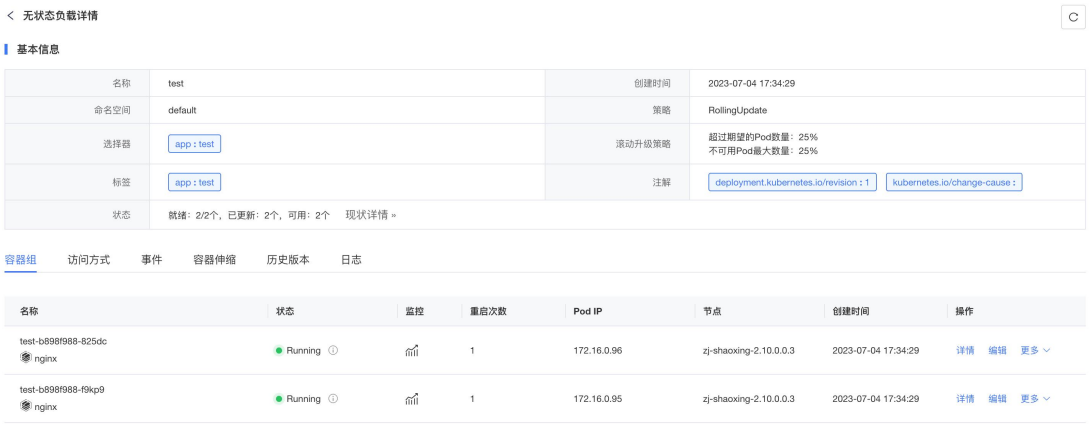
Pod 标签	为该 Pod 添加一个标签，标识该应用。
Pod 注解	为该 Pod 添加一个注解（Annotation）。

单击**创建**。

4.9.1.1.2 查看 Deployment 详情

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>无状态**。
5. 在**无状态**负载列表，单击目标负载右则操作列中的**详情**。

在应用详情页面可以基本信息、容器组、访问方式、事件、容器伸缩配置、历史版本及日志

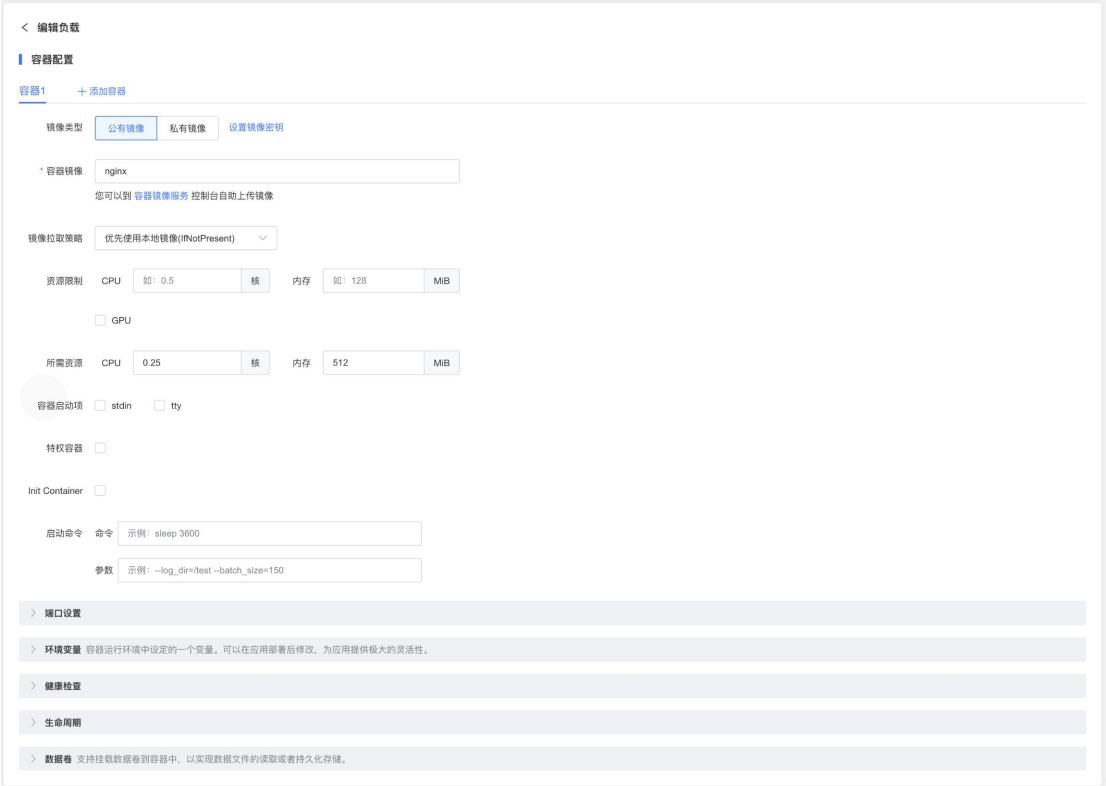


4.9.1.1.3 修改 Deployment

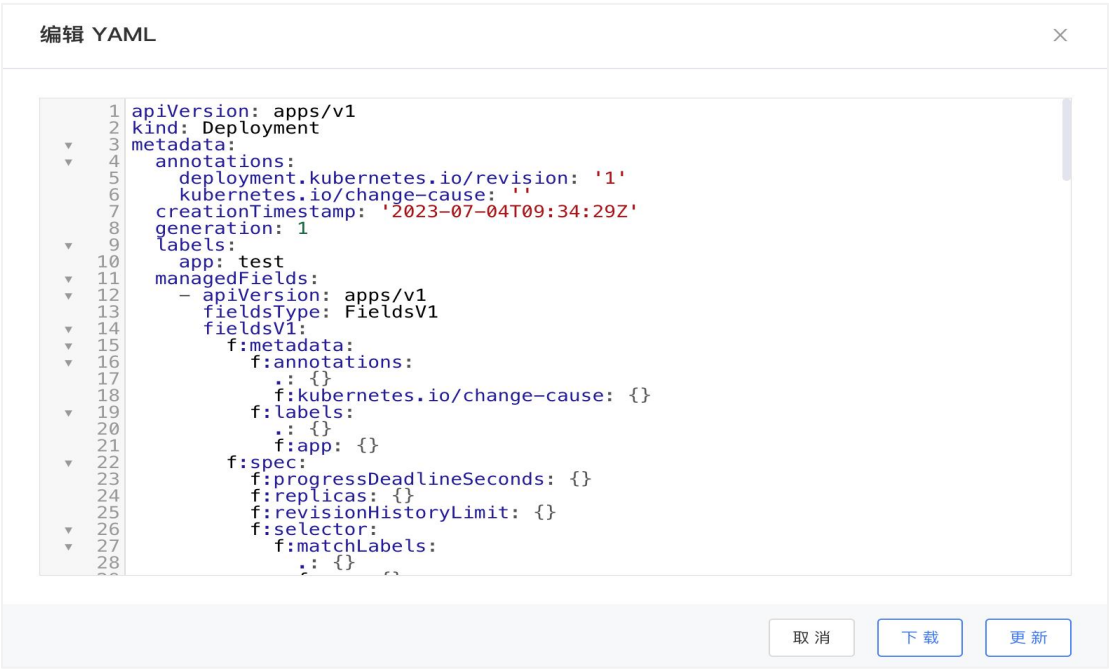
1. 登录**边缘容器集群控制台**。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

4. 在控制台左侧导航栏中，单击**工作负载>无状态。**

5. 在**无状态**负载列表，单击目标负载右侧操作列中的**编辑。**



单击目标负载右侧操作列中的 **YAML 编辑**，可通过 yaml 方式修改 Deployment 的配置。



4.9.1.1.4 删除 Deployment

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>无状态**。
5. 在**无状态**负载列表，单击目标负载右则操作列中的...>**删除**
6. 弹出确认对话框，单击**确认**即可删除 Deployment。

4.9.1.2 有状态 StatefulSet

4.9.1.2.1 使用镜像创建 StatefulSet

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>无状态**。
5. 在**有状态**负载列表，单击左上角的**创建有状态负载**。
6. 在应用配置页面，设置应用的配置信息。

应用基本信息

配置项	描述
应用名称	设置应用的名称，名称为 1~63 个字符，支持小写字母、数字、"-"以及"."等字符
命名空间	设置应用所在的命名空间。默认命名空间为 default
副本数量	应用包含的 Pod 数量，默认数量为 2

类型	定义资源对象的类型，选择**有状态**
标签	为应用添加标签，标识该应用
注解	为应用添加标签

容器配置

配置项	**描述**
容器镜像	镜像类型：可以选择公有镜像或私有镜像 公有镜像：填入需要使用的公有镜像地址 私有镜像：选择上传到镜像仓库的私有镜像 支持以下三种镜像拉取策略（imagePullPolicy）： **优先使用本地镜像（IfNotPresent）**：如果本地有该镜像（之前拉取过该镜像至宿主机中），则使用本地镜像，本地不存在时拉取镜像。 **总是拉取镜像（Always）**：表示每次部署或扩容都会从容器镜像服务重新拉取镜像，而不会从本地拉取镜像。 **仅使用本地镜像（Never）**：仅使用本地镜像。 单击**设置镜像密钥**，您可以实现免密拉取镜像
资源限制	可指定该应用所能使用的资源上限，包括 CPU、内存和 GPU 三种资源，防止占用过多资源。
所需资源	即为该应用预留资源额度，包括 CPU、内存和 GPU 三种资源，即容器独占该资源，防止因资源不足而被其他服务或进程争夺资源，导致应用不可用
容器启动项	stdin：表示为该容器开启标准输入 tty：表示为该容器分配一个虚拟终端，以便于向容器发送信号。 通常这两个选项是一起使用的，表示将终端（tty）绑定到容器的标准输入（stdin）上。例如，一个交互式的程序从用户获取标准输入，并显示到终端中

特权容器	选择特权容器，则 privileged=true，开启特权模式。 不选择特权容器，则 privileged=false，关闭特权模式。
Init Container	勾选该项，表示创建一个 Init Container

可选：在**端口设置**区域，单击**新增**设置容器的端口。

配置项	**描述**
名称	设置容器端口名称。
容器端口	设置暴露的容器访问端口或端口名，端口号必须介于 1~65535。
协议	支持 TCP 和 UDP。

可选：在**环境变量**区域，单击**新增**设置环境变量。

支持通过键值对的形式为 Pod 配置环境变量。用于给 Pod 添加环境标志或传递配置等。

配置项	**描述**
类型	设置环境变量的类型，支持： **自定义** **配置项** **保密字典** **变量/变量引用** **资源引用** 配置项、保密字典支持全部文件的引用，以保密字典为例。选择**保密字典**类型，只选择目标保密字典，则默认引用全部文件。
变量名称	设置环境变量名称。
变量/变量引用	设置变量引用的值。

可选：在**健康检查**区域，根据需要开启**存活检查**、**就绪检查**及**启动探测**。

存活检查（Liveness）：用于检测何时重启容器。

就绪检查（Readiness）：确定容器是否已经就绪，且可以接受流量。

启动探测（Startup Probes）：用于检测何时启动容器。

请求类型	描述
HTTP 请求	即向容器发送一个 HTTP Get 请求，支持的参数包括： **协议**：HTTP/HTTPS。 **路径**：访问 HTTP Server 的路径。 **端口**：容器暴露的访问端口或端口名，端口号必须介于 1~65535。 **HTTP 头**：即 HTTP Headers，HTTP 请求中自定义的请求头，HTTP 允许重复的 Header。支持键值对的配置方式。 **延迟探测时间（秒）**：即 initialDelaySeconds，容器启动后第一次执行探测时需要等待多少秒，默认为 3 秒。 **执行探测频率（秒）**：即 periodSeconds，指执行探测的时间间隔，默认为 10 秒，最小为 1 秒。 **超时时间（秒）**：即 timeoutSeconds，探测超时时间。默认 1 秒，最小 1 秒。 健康阈值：探测失败后，最少连续探测成功多少次才被认定为成功。默认是 1，最小值是 1。对于存活检查（liveness）必须是 1。 **不健康阈值**：探测成功后，最少连续探测失败多少次才被认定为失败。默认是 3，最小值是 1。
TCP 连接	即向容器发送一个 TCP Socket，Kubelet 将尝试在指定端口上打开容器的套接字。 如果可以建立连接，容器被认为是健康的，如果不能就认为是失败的。支持的参数包括： **端口**：容器暴露的访问端口或端口名，端口号必须介于 1~65535。 **延迟探测时间（秒）**：即 initialDelaySeconds，容器启动后第一次执行

	探测时需要等待多少秒，默认为 15 秒。 **执行探测频率（秒）**：即 periodSeconds，指执行探测的时间间隔，默认为 10 秒，最小为 1 秒。 **超时时间（秒）**：即 timeoutSeconds，探测超时时间。默认 1 秒，最小 1 秒。 **健康阈值**：探测失败后，最少连续探测成功多少次才被认定为成功。默认是 1，最小值是 1。对于存活检查（liveness）必须是 1。 **不健康阈值**：探测成功后，最少连续探测失败多少次才被认定为失败。默认是 3，最小值是 1。
命令行	通过在容器中执行探针检测命令，来检测容器的健康情况。支持的参数包括：**命令行**：用于检测容器健康情况的探测命令。 **延迟探测时间（秒）**：即 initialDelaySeconds，容器启动后第一次执行探测时需要等待多少秒，默认为 5 秒。 **执行探测频率（秒）**：即 periodSeconds，指执行探测的时间间隔，默认为 10 秒，最小为 1 秒。 **超时时间（秒）**：即 timeoutSeconds，探测超时时间。默认 1 秒，最小 1 秒。 **健康阈值**：探测失败后，最少连续探测成功多少次才被认定为成功。默认是 1，最小值是 1。对于存活检查（liveness）必须是 1。 **不健康阈值**：探测成功后，最少连续探测失败多少次才被认定为失败。默认是 3，最小值是 1。

可选：在**生命周期**区域，设置容器的生命周期。

您可以为容器的生命周期配置启动执行、启动后处理和停止前处理。

配置项	**描述**

启动执行	为容器设置预启动命令和参数。
启动后处理	为容器设置启动后的命令。
停止前处理	为容器设置预结束命令。

可选：在**数据卷**区域，增加本地存储或云存储声明 PVC（Persistent Volume Claim）。

配置项	描述
本地存储	本地存储：支持主机目录（HostPath）、配置项（ConfigMap）、保密字典（Secret）和临时目录，将对应的挂载源挂载到容器路径中。
云存储声明	支持通过 PVC 挂载云存储卷。在选择目标挂载源前，您需要创建云存储声明。

高级配置。

在**高级配置**配置向导页面中设置访问、伸缩、调度和标签注解。

在**访问设置**区域，通过设置**服务**暴露后端 Pod

配置服务（Service）：

配置项	描述
名称	输入服务的名称。
类型	选择服务访问的方式 **虚拟集群 IP**：即 ClusterIP，指通过集群的内部 IP 暴露服务，选择该值 ,服务只能够在集群内部可以访问 ,这也是默认的 ServiceType。 **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，该 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

实例间发现服务	服务类型为**虚拟集群 IP** 时，才能设置**实例间发现服务（Headless Service）**。
外部流量策略	**Local**：流量只发给本机的 Pod。 **Cluster**：流量可以转发到其他节点上的 Pod。 服务类型为**节点端口**或**负载均衡**时，才能设置**外部流量策略**。
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。
标签	为该服务添加一个标签，标识该服务
注解	为该服务添加一个注解（Annotation），配置负载均衡的参数

可选：在**伸缩配置**区域，配置是否开启**指标伸缩**。

指标伸缩：根据容器 CPU 和内存 占用情况自动调整容器组数量。

配置项	**描述**
指标	支持 CPU 和内存，需要和设置的所需资源类型相同。
触发条件	资源使用率的百分比，超过该使用量，容器开始扩容。
最大副本数量	该负载类型可扩容的容器数量上限。
最小副本数量	该负载类型可缩容的容器数量下限。

可选：在**调度设置**区域，设置**升级方式**、**节点亲和性**、**应用亲和性**、**应用非亲和性 和 调度容忍**

配置项	**描述**
升级方式	升级方式包括滚动升级（rollingupdate）和替换升级（recreate）

节点亲和性	设置节点亲和性，通过 Worker 节点的 Label 标签进行设置。节点调度支持硬约束和软约束（Required/Preferred），以及丰富的匹配表达式（In, NotIn, Exists, DoesNotExist. Gt, and Lt）： **必须满足**，即硬约束，一定要满足，对应 requiredDuringSchedulingIgnoredDuringExecution，效果与 NodeSelector 相同。本例中 Pod 只能调度到具有对应标签的 Worker 节点。您可以定义多条硬约束规则，但只需满足其中一条。 **尽量满足**，即软约束，不一定满足，对应 preferredDuringSchedulingIgnoredDuringExecution。调度会尽量调度 Pod 到具有对应标签的 Node 节点。您还可为软约束规则设定权重，具体调度时，若存在多个符合条件的节点，权重最大的节点会被优先调度。您可定义多条软约束规则，但必须满足全部约束，才会进行调度。
应用亲和性	决定应用的 Pod 可以和哪些 Pod 部署在同一拓扑域。例如，对于相互通信的服务，可通过应用亲和性调度，将其部署到同一拓扑域（如同一个主机）中，减少它们之间的网络延迟。 根据节点上运行的 Pod 的标签（Label）来进行调度，支持硬约束和软约束，匹配的表达式有：In, NotIn, Exists, DoesNotExist。 **必须满足**，即硬约束，一定要满足，对应 requiredDuringSchedulingIgnoredDuringExecution ,Pod 的亲和性调度必须要满足后续定义的约束条件 **命名空间**：该策略是依据 Pod 的 Label 进行调度，所以会受到命名空间的约束。 **拓扑域**：即 topologyKey，指定调度时作用域，这是通过 Node 节点的标签来实现的，例如指定为 kubernetes.io/hostname，那就是以

	Node 节点为区分范围；如果指定为 beta.kubernetes.io/os，则以 Node 节点的操作系统类型来区分。 **选择器**：单击选择器右侧的加号按钮，您可添加多条硬约束规则。 **查看应用列表**：单击应用列表，弹出对话框，您可在此查看各命名空间下的应用，并可将应用的标签导入到亲和性配置页面。 **硬约束条件**：设置已有应用的标签、操作符和标签值。本例中，表示将待创建的应用调度到该主机上，该主机运行的已有应用具有 app:nginx 标签。 **尽量满足**，即软约束，不一定满足，对应 preferredDuringSchedulingIgnoredDuringExecution。Pod 的亲和性调度会尽量满足后续定义的约束条件。对于软约束规则，您可配置每条规则的权重，其他配置规则与硬约束规则相同 **权重**：设置一条软约束规则的权重，介于 1~100，通过算法计算满足软约束规则的节点的权重，将 Pod 调度到权重最大的节点上
应用非亲和性	决定应用的 Pod 不与哪些 Pod 部署在同一拓扑域。应用非亲和性调度的场景包括： 将一个服务的 Pod 分散部署到不同的拓扑域（如不同主机）中，提高服务本身的稳定性。 给予 Pod 一个节点的独占访问权限来保证资源隔离，保证不会有其它 Pod 来分享节点资源。 把可能会相互影响的服务的 Pod 分散在不同的主机上。
调度容忍	容忍被应用于 Pod，允许这个 Pod 被调度到相对应的污点上。

可选：在**标签和注释**区域，单击**添加**设置容器组的标签和注释。

配置项	**描述**

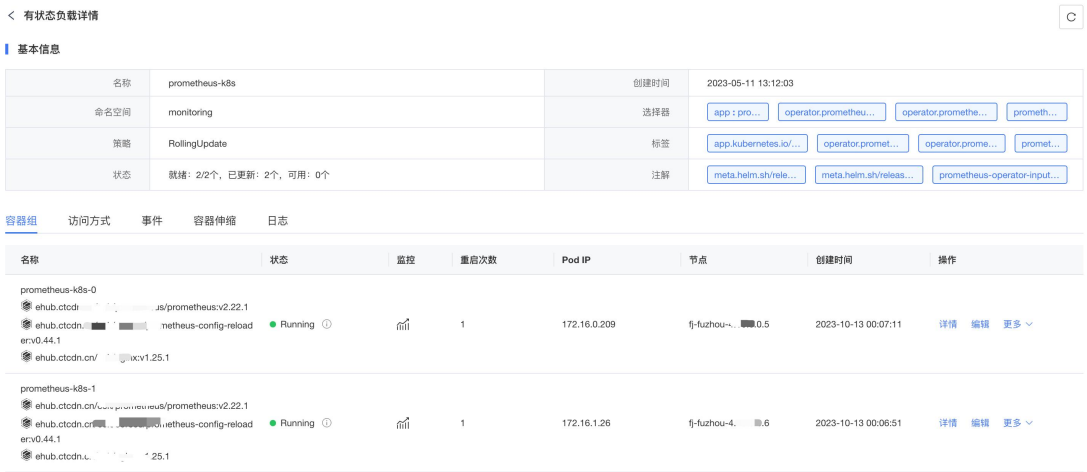
Pod 标签	为该 Pod 添加一个标签，标识该应用。
Pod 注解	为该 Pod 添加一个注解（Annotation）。

单击**创建**。

4.9.1.2.2 查看 StatefulSet 详情

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>有状态**。
5. 在**有状态**负载列表，单击目标负载右则操作列中的**详情**。

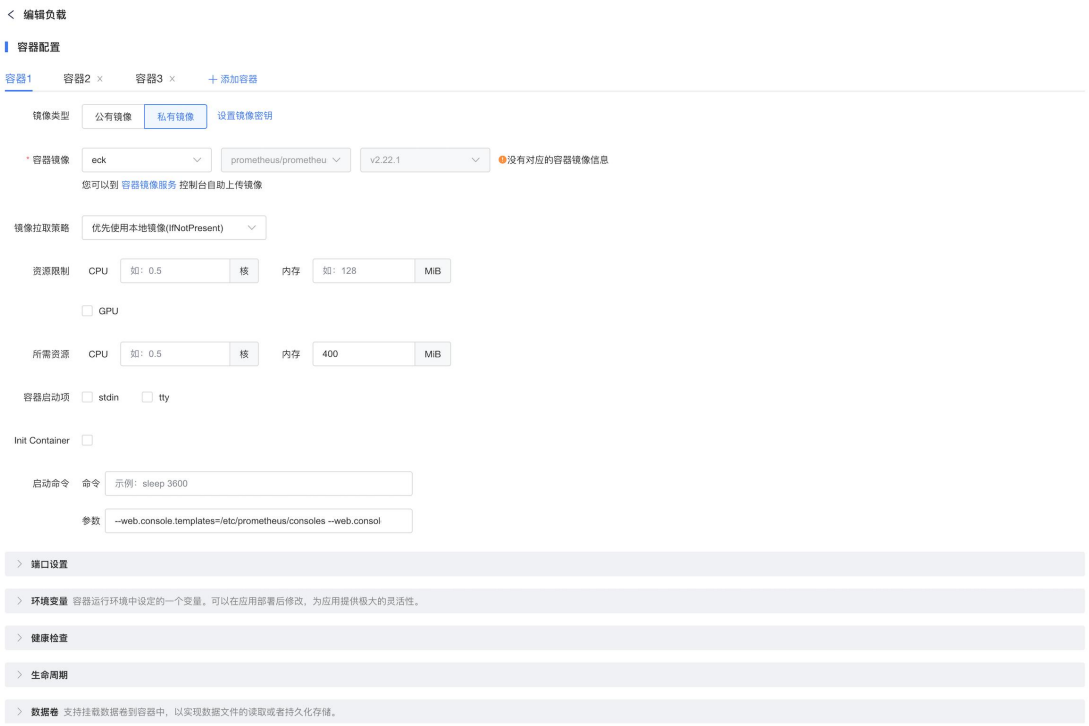
在应用详情页面可以基本信息、容器组、访问方式、事件、容器伸缩及日志。



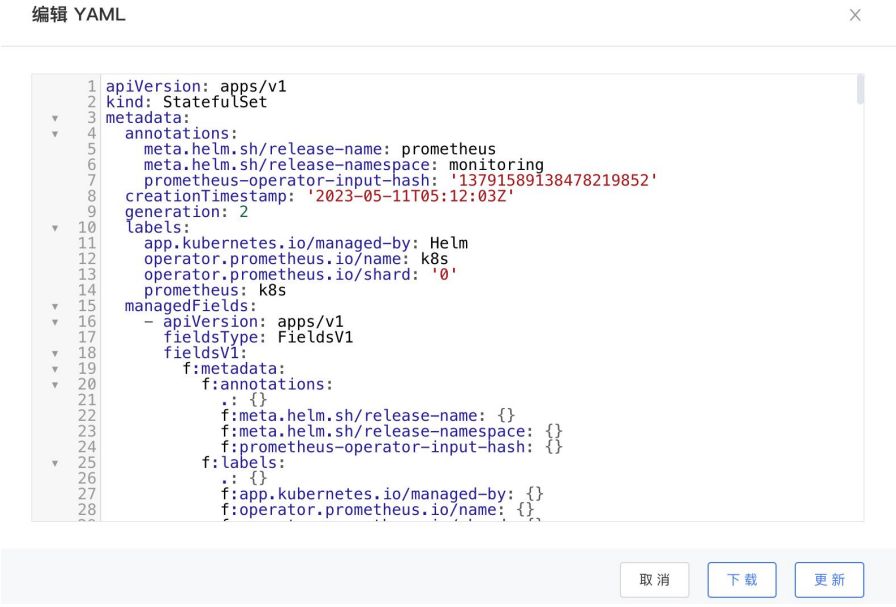
4.9.1.2.3 修改 StatefulSet

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>无状态**。

5. 在**无状态**负载列表，单击目标负载右则操作列中的**编辑**。



单击目标负载右侧操作列中的 **YAML 编辑** ，您可通过 YAML 方式修改 StatefulSet 的配置。



4.9.1.2.4 删除 StatefulSet

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>有状态**。
5. 在**有状态**负载列表，单击目标负载右则操作列中的**删除**。
6. 弹出确认对话框，单击**确认**即可删除。

4.9.1.3 守护进程集 DaemonSet

4.9.1.3.1 使用镜像创建 DaemonSet

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>守护进程集**。
5. 在**守护进程集**负载列表，单击左上角的**创建守护进程集**。
6. 在应用配置页面，设置应用的配置信息

应用基本信息

配置项	描述
应用名称	设置应用的名称，名称为 1~63 个字符，支持小写字母、数字、"-"以及"."等字符

命名空间	设置应用所在的命名空间。默认命名空间为 default
副本数量	应用包含的 Pod 数量，默认数量为 2
类型	定义资源对象的类型，目前选择**守护进程集**
标签	为应用添加标签，标识该应用
注解	为应用添加标签

容器配置

配置项	**描述**
容器镜像	镜像类型：可以选择公有镜像或私有镜像 公有镜像：填入需要使用的公有镜像地址 私有镜像：选择上传到镜像仓库的私有镜像 支持以下三种镜像拉取策略（imagePullPolicy）： **优先使用本地镜像（IfNotPresent）**：如果本地有该镜像（之前拉取过该镜像至宿主机中），则使用本地镜像，本地不存在时拉取镜

	像。 **总是拉取镜像（Always）**：表示每次部署或扩容都会从容器镜像服务重新拉取镜像，而不会从本地拉取镜像。 **仅使用本地镜像（Never）**：仅使用本地镜像。 单击**设置镜像密钥**，您可以实现免密拉取镜像
资源限制	可指定该应用所能使用的资源上限，包括 CPU、内存和 GPU 三种资源，防止占用过多资源。
所需资源	即为该应用预留资源额度，包括 CPU、内存和 GPU 三种资源，即容器独占该资源，防止因资源不足而被其他服务或进程争夺资源，导致应用不可用
容器启动项	stdin：表示为该容器开启标准输入 tty：表示为该容器分配一个虚拟终端，以便于向容器发送信号。 通常这两个选项是一起使用的，表示将终端（tty）绑定到容器的标准输入（stdin）上。例如，一个交互式的程序从用户获取标准输入，并显示到终端中
特权容器	选择特权容器，则 privileged=true，开启特权模式。

	不选择特权容器，则 privileged=false，关闭特权模式。
Init Container	勾选该项，表示创建一个 Init Container

可选：在**端口设置**区域，单击**新增**设置容器的端口。

配置项	**描述**
名称	设置容器端口名称。
容器端口	设置暴露的容器访问端口或端口名，端口号必须介于 1~65535。
协议	支持 TCP 和 UDP。

可选：在**环境变量**区域，单击**新增**设置环境变量。

支持通过键值对的形式为 Pod 配置环境变量。用于给 Pod 添加环境标志或传递配置等。

配置项	**描述**
类型	设置环境变量的类型，支持：

	自定义 **配置项** **保密字典** **变量/变量引用** **资源引用** 配置项、保密字典支持全部文件的引用，以保密字典为例。选择**保密字典**类型，只选择目标保密字典，则默认引用全部文件。
变量名称	设置环境变量名称。
变量/变量引用	设置变量引用的值。

可选：在**健康检查**区域，根据需要开启**存活检查**、**就绪检查**及**启动探测。**

存活检查（Liveness）：用于检测何时重启容器。

就绪检查（Readiness）：确定容器是否已经就绪，且可以接受流量。

启动探测（Startup Probes）：用于检测何时启动容器。

请求类型	**描述**

HTTP 请求	即向容器发送一个 HTTP　Get 请求，支持的参数包括： **协议**：HTTP/HTTPS。 **路径**：访问 HTTP Server 的路径。 **端口**：容器暴露的访问端口或端口名，端口号必须介于 1~65535。 **HTTP 头**：即 HTTP Headers，HTTP 请求中自定义的请求头，HTTP 允许重复的 Header。支持键值对的配置方式。 **延迟探测时间（秒）**：即 initialDelaySeconds，容器启动后第一次执行探测时需要等待多少秒，默认为 3 秒。 **执行探测频率（秒）**：即 periodSeconds，指执行探测的时间间隔，默认为 10 秒，最小为 1 秒。 **超时时间（秒）**：即 timeoutSeconds，探测超时时间。默认 1 秒，最小 1 秒。 健康阈值：探测失败后，最少连续探测成功多少次才被认定为成功。默认是 1，最小值是 1。对于存活检查（liveness）必须是 1。 **不健康阈值**：探测成功后，最少连续探测失败多少次才被认定为失败。默认是 3，最小值是 1。
TCP 连接	即向容器发送一个 TCP Socket，Kubelet 将尝试在指定端口上打开容

	器的套接字。　如果可以建立连接，容器被认为是健康的，如果不能就认为是失败的。支持的参数包括： **端口**：容器暴露的访问端口或端口名，端口号必须介于 1~65535。 **延迟探测时间（秒）**：即 initialDelaySeconds，容器启动后第一次执行探测时需要等待多少秒，默认为 15 秒。 **执行探测频率（秒）**：即 periodSeconds，指执行探测的时间间隔，默认为 10 秒，最小为 1 秒。 **超时时间（秒）**：即 timeoutSeconds，探测超时时间。默认 1 秒，最小 1 秒。 **健康阈值**：探测失败后，最少连续探测成功多少次才被认定为成功。默认是 1，最小值是 1。对于存活检查（liveness）必须是 1。 **不健康阈值**：探测成功后，最少连续探测失败多少次才被认定为失败。默认是 3，最小值是 1。
命令行	通过在容器中执行探针检测命令，来检测容器的健康情况。支持的参数包括：**命令行**：用于检测容器健康情况的探测命令。 **延迟探测时间（秒）**：即 initialDelaySeconds，容器启动后第一次执行探测时需要等待多少秒，默认为 5 秒。 **执行探测频率（秒）**：即 periodSeconds，指执行探测的时间间隔，

	默认为 10 秒，最小为 1 秒。 **超时时间（秒）**：即 timeoutSeconds，探测超时时间。默认 1 秒，最小 1 秒。 **健康阈值**：探测失败后，最少连续探测成功多少次才被认定为成功。默认是 1，最小值是 1。对于存活检查（liveness）必须是 1。 **不健康阈值**：探测成功后，最少连续探测失败多少次才被认定为失败。默认是 3，最小值是 1。

可选：在**生命周期**区域，设置容器的生命周期。

您可以为容器的生命周期配置启动执行、启动后处理和停止前处理。

配置项	**描述**
启动执行	为容器设置预启动命令和参数。
启动后处理	为容器设置启动后的命令。
停止前处理	为容器设置预结束命令。

可选：在**数据卷**区域，增加本地存储或云存储声明 PVC（Persistent Volume Claim）。

配置项	描述
本地存储	本地存储：支持主机目录（HostPath）、配置项（ConfigMap）、保密字典（Secret）和临时目录，将对应的挂载源挂载到容器路径中。
云存储声明	支持通过 PVC 挂载云存储卷。在选择目标挂载源前，您需要创建云存储声明。

高级配置。

在**高级配置**配置向导页面中设置访问、伸缩、调度和标签注解。

在**访问设置**区域，通过设置**服务**暴露后端 Pod

配置服务（Service）：

配置项	描述
名称	输入服务的名称。
类型	选择服务访问的方式 **虚拟集群 IP**：即 ClusterIP，指通过集群的内部 IP 暴露服务，选择

	该值 ,服务只能够在集群内部可以访问 ,这也是默认的 ServiceType。 **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，该 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
实例间发现服务	服务类型为**虚拟集群 IP** 时，才能设置**实例间发现服务（Headless Service）**。
外部流量策略	**Local**：流量只发给本机的 Pod。 **Cluster**：流量可以转发到其他节点上的 Pod。 服务类型为**节点端口**或**负载均衡**时，才能设置**外部流量策略。**
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。
标签	为该服务添加一个标签，标识该服务
注解	为该服务添加一个注解（Annotation），配置负载均衡的参数

可选：在**调度设置**区域，设置**升级方式**、**节点亲和性**、**应用亲和性**、**应用非亲和性 和 调度**

容忍

配置项	描述
升级方式	升级方式包括滚动升级（rollingupdate）和替换升级（recreate）
节点亲和性	设置节点亲和性，通过 Worker 节点的 Label 标签进行设置。节点调度支持硬约束和软约束（Required/Preferred），以及丰富的匹配表达式（In, NotIn, Exists, DoesNotExist. Gt, and Lt）： **必须满足**，即硬约束，一定要满足，对应 requiredDuringSchedulingIgnoredDuringExecution，效果与 NodeSelector 相同。本例中 Pod 只能调度到具有对应标签的 Worker 节点。您可以定义多条硬约束规则，但只需满足其中一条。 **尽量满足**，即软约束，不一定满足，对应 preferredDuringSchedulingIgnoredDuringExecution。调度会尽量调度 Pod 到具有对应标签的 Node 节点。您还可为软约束规则设定权重，具体调度时，若存在多个符合条件的节点，权重最大的节点会被优先调度。您可定义多条软约束规则，但必须满足全部约束，才会进行调度。

应用亲和性	决定应用的 Pod 可以和哪些 Pod 部署在同一拓扑域。例如，对于相互通信的服务，可通过应用亲和性调度，将其部署到同一拓扑域（如同一个主机）中，减少它们之间的网络延迟。 根据节点上运行的 Pod 的标签（Label）来进行调度，支持硬约束和软约束，匹配的表达式有：In, NotIn, Exists, DoesNotExist。 **必须满足**，即硬约束，一定要满足，对应 requiredDuringSchedulingIgnoredDuringExecution，Pod 的亲和性调度必须要满足后续定义的约束条件 **命名空间**：该策略是依据 Pod 的 Label 进行调度，所以会受到命名空间的约束。 **拓扑域**：即 topologyKey，指定调度时作用域，这是通过 Node 节点的标签来实现的，例如指定为 kubernetes.io/hostname，那就是以 Node 节点为区分范围；如果指定为 beta.kubernetes.io/os，则以 Node 节点的操作系统类型来区分。 **选择器**：单击选择器右侧的加号按钮，您可添加多条硬约束规则。 **查看应用列表**：单击应用列表，弹出对话框，您可在此查看各命名空间下的应用，并可将应用的标签导入到亲和性配置页面。 **硬约束条件**：设置已有应用的标签、操作符和标签值。本例中，表示将

	待创建的应用调度到该主机上，该主机运行的已有应用具有 app:nginx 标签。 **尽量满足**，即软约束，不一定满足，对应 preferredDuringSchedulingIgnoredDuringExecution。Pod 的亲和性调度会尽量满足后续定义的约束条件。对于软约束规则，您可配置每条规则的权重，其他配置规则与硬约束规则相同 **权重**：设置一条软约束规则的权重，介于 1~100，通过算法计算满足软约束规则的节点的权重，将 Pod 调度到权重最大的节点上
应用非亲和性	决定应用的 Pod 不与哪些 Pod 部署在同一拓扑域。应用非亲和性调度的场景包括： 将一个服务的 Pod 分散部署到不同的拓扑域（如不同主机）中，提高服务本身的稳定性。 给予 Pod 一个节点的独占访问权限来保证资源隔离，保证不会有其它 Pod 来分享节点资源。 把可能会相互影响的服务的 Pod 分散在不同的主机上。
调度容忍	容忍被应用于 Pod，允许这个 Pod 被调度到相对应的污点上。

可选：在**标签和注释**区域，单击**添加**设置容器组的标签和注释。

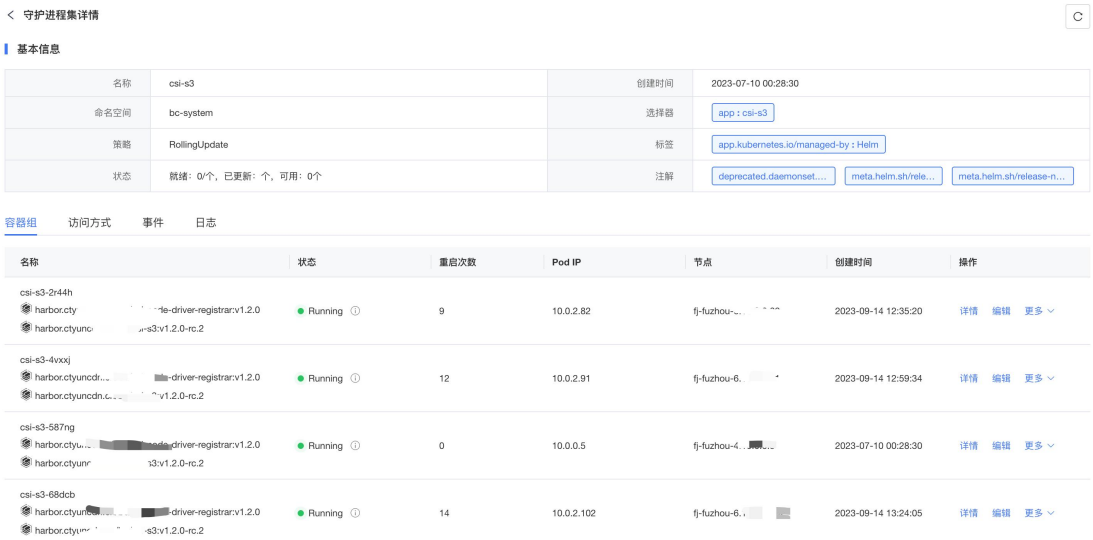
配置项	描述
Pod 标签	为该 Pod 添加一个标签，标识该应用。
Pod 注解	为该 Pod 添加一个注解（Annotation）。

单击**创建**。

4.9.1.3.2 查看 DaemonSet 详情

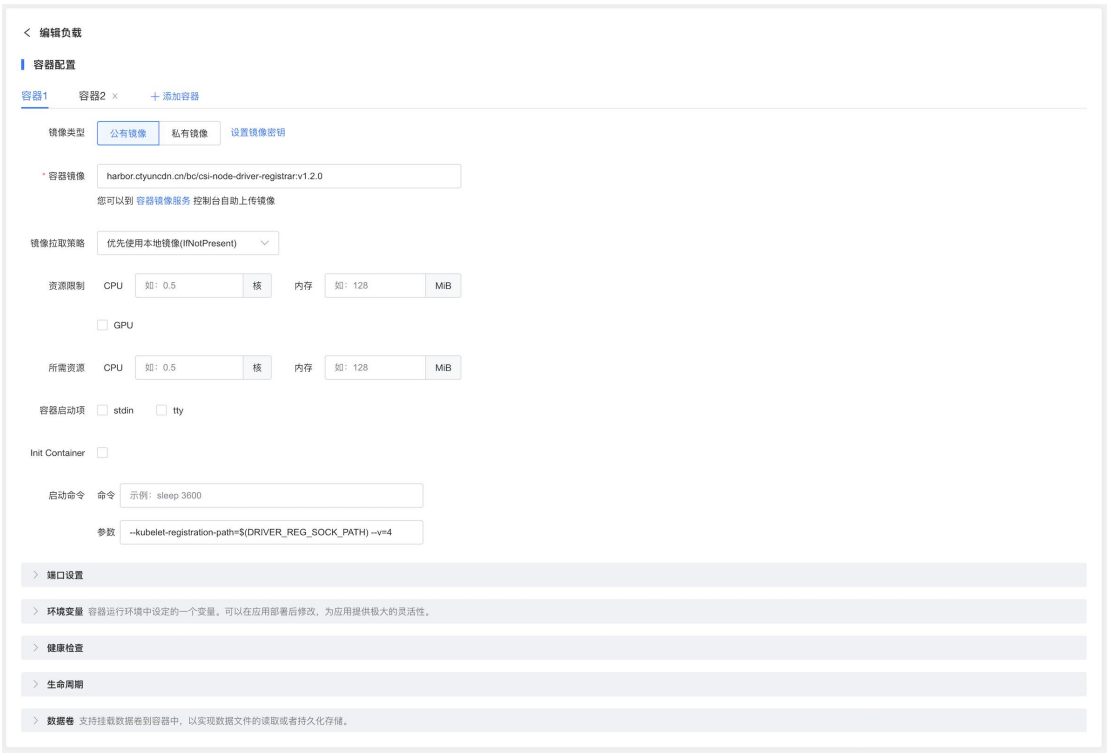
1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>守护进程集**。
5. 在**守护进程集**负载列表，单击目标负载右则操作列中的**详情**。

在应用详情页面可以基本信息、容器组、访问方式、事件及日志。

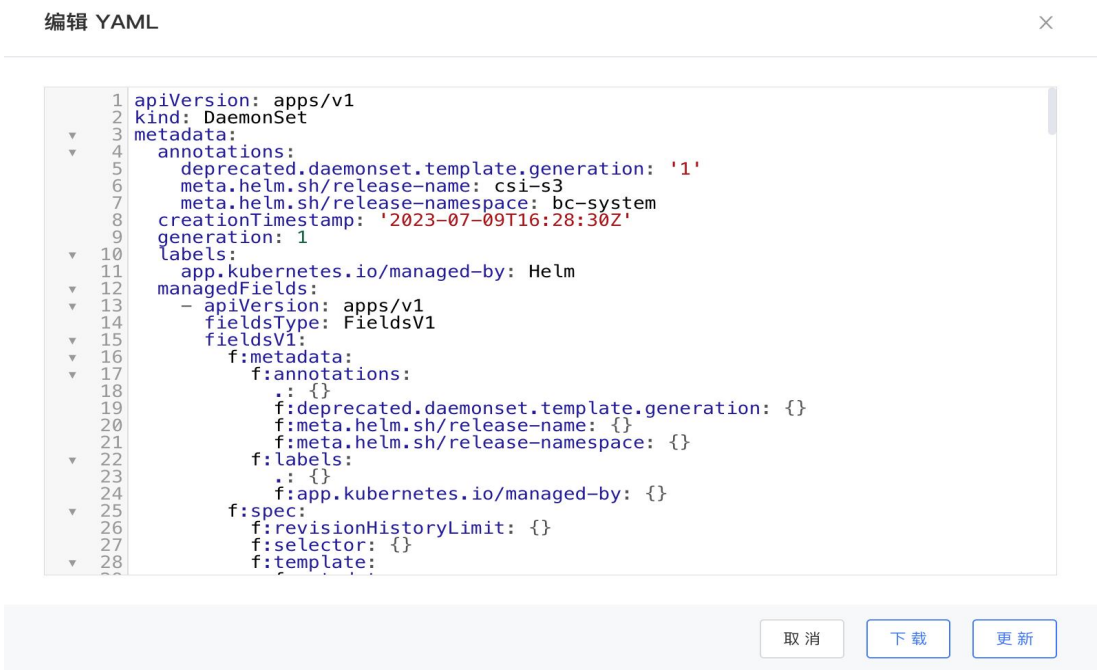


4.9.1.3.3 修改 DaemonSet

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>守护进程集**。
5. 在**守护进程集**负载列表，单击目标负载右则操作列中的**编辑**。



您也可以通过编辑 YAML 方式修改 DaemonSet，单击目标负载右侧操作列中的 **YAML 编辑**，即可通过 YAML 方式修改 DaemonSet 的配置。



4.9.1.3.4 删除 DaemonSet

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>守护进程集**。
5. 在**守护进程集**负载列表，单击目标负载右则操作列中的**删除**
6. 弹出确认对话框，单击**确认**即可删除 DaemonSet。

4.9.1.4 任务 Job

4.9.1.4.1 使用镜像创建 Job

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>任务**。
5. 在**任务**负载列表，单击左上角的**创建任务**。
6. 在应用配置页面，设置应用的配置信息

应用基本信息

配置项	描述
应用名称	设置应用的名称，名称为 1~63 个字符，支持小写字母、数字、"-"以及"."等字符

命名空间	设置应用所在的命名空间。默认命名空间为 default
副本数量	应用包含的 Pod 数量，默认数量为 2
类型	定义资源对象的类型，选择**任务**
标签	为应用添加标签，标识该应用
注解	为应用添加标签

容器配置

配置项	**描述**
容器镜像	镜像类型：可以选择公有镜像或私有镜像 公有镜像：填入需要使用的公有镜像地址 私有镜像：选择上传到镜像仓库的私有镜像 支持以下三种镜像拉取策略（imagePullPolicy）： **优先使用本地镜像（IfNotPresent）**：如果本地有该镜像（之前拉取过该镜像至宿主机中），则使用本地镜像，本地不存在时拉取镜像。 **总是拉取镜像（Always）**：表示每次部署或扩容都会从容器镜像服务重新拉取镜像，而不会从本地拉取镜像。 **仅使用本地镜像（Never）**：仅使用本地镜像。 单击**设置镜像密钥**，您可以实现免密拉取镜像
资源限制	可指定该应用所能使用的资源上限，包括 CPU、内存和 GPU 三种资源，防止占用过多资源。
所需资源	即为该应用预留资源额度，包括 CPU、内存和 GPU 三种资源，即容

	器独占该资源，防止因资源不足而被其他服务或进程争夺资源，导致应用不可用
容器启动项	stdin：表示为该容器开启标准输入 tty：表示为该容器分配一个虚拟终端，以便于向容器发送信号。 通常这两个选项是一起使用的，表示将终端（tty）绑定到容器的标准输入（stdin）上。例如，一个交互式的程序从用户获取标准输入，并显示到终端中
特权容器	选择特权容器，则 privileged=true，开启特权模式。 不选择特权容器，则 privileged=false，关闭特权模式。
Init Container	勾选该项，表示创建一个 Init Container

可选：在**端口设置**区域，单击**新增**设置容器的端口。

配置项	**描述**
名称	设置容器端口名称。
容器端口	设置暴露的容器访问端口或端口名，端口号必须介于 1~65535。
协议	支持 TCP 和 UDP。

可选：在**环境变量**区域，单击**新增**设置环境变量。

支持通过键值对的形式为 Pod 配置环境变量。用于给 Pod 添加环境标志或传递配置等。

配置项	**描述**
类型	设置环境变量的类型，支持：

	自定义 **配置项** **保密字典** **变量/变量引用** **资源引用** 配置项、保密字典支持全部文件的引用，以保密字典为例。选择**保密字典**类型，只选择目标保密字典，则默认引用全部文件。
变量名称	设置环境变量名称。
变量/变量引用	设置变量引用的值。

可选：在**生命周期**区域，设置容器的生命周期。

您可以为容器的生命周期配置启动执行、启动后处理和停止前处理。

配置项	描述
启动执行	为容器设置预启动命令和参数。
启动后处理	为容器设置启动后的命令。
停止前处理	为容器设置预结束命令。

可选：在**数据卷**区域，增加本地存储或云存储声明 PVC（Persistent Volume Claim）。

配置项	描述
本地存储	本地存储：支持主机目录（HostPath）、配置项（ConfigMap）、保密字典（Secret）和临时目录，将对应的挂载源挂载到容器路径中。

云存储声明	支持通过 PVC 挂载云存储卷。在选择目标挂载源前，您需要创建云存储声明。

高级配置。

在**高级配置**配置向导页面中设置任务和标签注解。

可选：在**任务设置**区域，通过设置任务 Pod 数、超时时间、重新次数、重启策略

配置项	描述
成功运行的 Pod 数	即 completions，指定 job 需要成功运行 Pods 的数量。默认值为 1
并行运行的 Pod 数	即 parallelism ,指定 job 在任一时刻应该并发运行 Pod 的数量。默认值为 1
超时时间	即 activeDeadlineSeconds，指定 job 可运行的时间期限，超过时间还未结束，系统将会尝试进行终止。
重新次数	Pod 执行失败后，尝试重建 Pod 的次数。默认是 6 次，每次失败后重试会有延迟时间，该时间是指数级增长，最长时间是 6min。
重启策略	仅支持不重启（Never）和失败时（OnFailure）

可选：在**标签和注释**区域，单击**添加**设置容器组的标签和注释。

配置项	描述

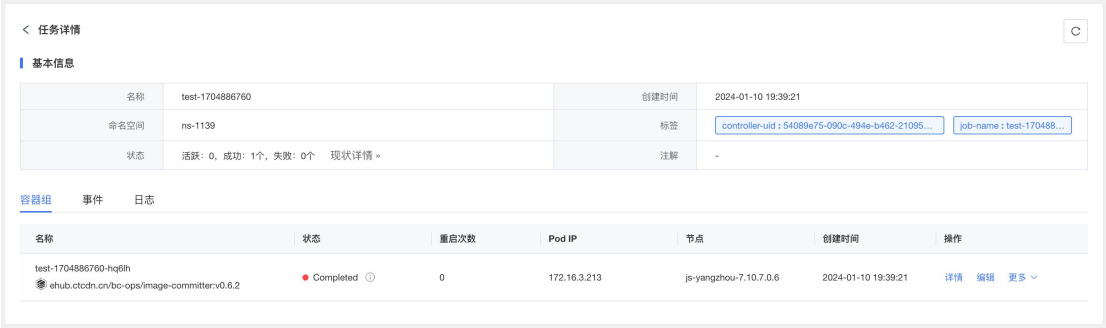
Pod 标签	为该 Pod 添加一个标签，标识该应用。
Pod 注解	为该 Pod 添加一个注解（Annotation）。

单击**创建**。

4.9.1.4.2 查看 Job 详情

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>任务**。
5. 在**任务**负载列表，单击目标负载右则操作列中的**详情**。

在应用详情页面可以基本信息、容器组、访问方式、事件、容器伸缩配置、历史版本及日志

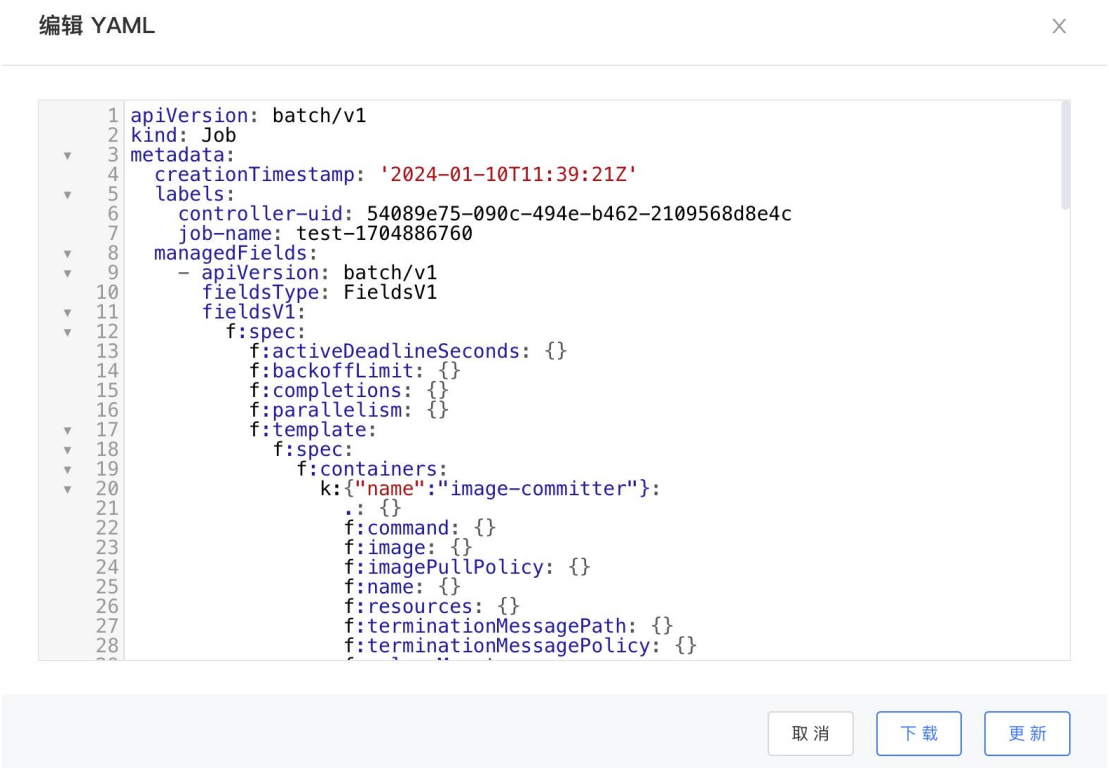


4.9.1.4.3 修改 Job

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>任务**。

5. 在**任务**负载列表，单击目标负载右则操作列中的**编辑**。

通过 yaml 方式修改 Job 的配置



4.9.1.4.4 Job 伸缩

1. 登录边缘容器集群控制台。

2. 在控制台左侧导航栏中，单击**集群管理**。

3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

4. 在控制台左侧导航栏中，单击**工作负载>任务**。

5. 在**任务**负载列表，单击目标负载右则操作列中的**伸缩**。

6. 弹出确认对话框，设置 Job 并行运行的 Pod 数，单击**确认**。

4.9.1.4.5 删除 Job

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>任务**。
5. 在**任务**负载列表，单击目标负载右则操作列中的**删除**。
6. 弹出确认对话框，单击**确认**即可删除 Job。

4.9.1.5 定时任务 CronJob

4.9.1.5.1 使用镜像创建 CronJob

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>定时任务**。
5. 在**定时任务**负载列表，单击左上角的**创建定时任务**。
6. 在应用配置页面，设置应用的配置信息

应用基本信息

配置项	描述
应用名称	设置应用的名称，名称为 1~63 个字符，支持小写字母、数字、"-"以及"."等字符。
命名空间	设置应用所在的命名空间。默认命名空间为 default。
副本数量	应用包含的 Pod 数量，默认数量为 2。
类型	定义资源对象的类型，选择**定时任务。**
标签	为应用添加标签，标识该应用。
注解	为应用添加标签。

容器配置

配置项	描述
容器镜像	镜像类型：可以选择公有镜像或私有镜像。 公有镜像：填入需要使用的公有镜像地址。 私有镜像：选择上传到镜像仓库的私有镜像。 支持以下三种镜像拉取策略（imagePullPolicy）： **优先使用本地镜像（IfNotPresent）**：如果本地有该镜像（之前拉取过该镜像至宿主机中），则使用本地镜像，本地不存在时拉取镜像。 **总是拉取镜像（Always）**：表示每次部署或扩容都会从容器镜像服务重新拉取镜像，而不会从本地拉取镜像。 **仅使用本地镜像（Never）**：仅使用本地镜像。 单击**设置镜像密钥**，您可以实现免密拉取镜像

资源限制	可指定该应用所能使用的资源上限，包括 CPU、内存和 GPU 三种资源，防止占用过多资源。
所需资源	即为该应用预留资源额度，包括 CPU、内存和 GPU 三种资源，即容器独占该资源，防止因资源不足而被其他服务或进程争夺资源，导致应用不可用
容器启动项	stdin：表示为该容器开启标准输入 tty：表示为该容器分配一个虚拟终端，以便于向容器发送信号。 通常这两个选项是一起使用的，表示将终端（tty）绑定到容器的标准输入（stdin）上。例如，一个交互式的程序从用户获取标准输入，并显示到终端中
特权容器	选择特权容器，则 privileged=true，开启特权模式。 不选择特权容器，则 privileged=false，关闭特权模式。
Init Container	勾选该项，表示创建一个 Init Container

可选：在**端口设置**区域，单击**新增**设置容器的端口。

配置项	**描述**
名称	设置容器端口名称。
容器端口	设置暴露的容器访问端口或端口名，端口号必须介于 1~65535。
协议	支持 TCP 和 UDP。

可选：在**环境变量**区域，单击**新增**设置环境变量。

支持通过键值对的形式为 Pod 配置环境变量。用于给 Pod 添加环境标志或传递配置等。

配置项	描述
类型	设置环境变量的类型，支持： **自定义** **配置项** **保密字典** **变量/变量引用** **资源引用** 配置项、保密字典支持全部文件的引用，以保密字典为例。选择**保密字典**类型，只选择目标保密字典，则默认引用全部文件。
变量名称	设置环境变量名称。
变量/变量引用	设置变量引用的值。

可选：在**生命周期**区域，设置容器的生命周期。

您可以为容器的生命周期配置启动执行、启动后处理和停止前处理。

配置项	描述
启动执行	为容器设置预启动命令和参数。
启动后处理	为容器设置启动后的命令。
停止前处理	为容器设置预结束命令。

可选：在**数据卷**区域，增加本地存储或云存储声明 PVC（Persistent Volume Claim）。

配置项	描述
本地存储	本地存储：支持主机目录（HostPath）、配置项（ConfigMap）、保密字典（Secret）和临时目录，将对应的挂载源挂载到容器路径中。
云存储声明	支持通过 PVC 挂载云存储卷。在选择目标挂载源前，您需要创建云存储声明。

高级配置。

在**高级配置**配置向导页面中设置定时任务、任务和标签注解。

可选：在定时**任务设置**区域，通过设置定时规则、并发策略及任务记录

配置项	描述
定时规则	即 completions，指定 job 需要成功运行 Pods 的数量。默认值为 1
并发策略	即 parallelism ,指定 job 在任一时刻应该并发运行 Pod 的数量。默认值为 1
任务记录	即 activeDeadlineSeconds，指定 job 可运行的时间期限，超过时间还未结束，系统将会尝试进行终止。

可选：在**任务设置**区域，通过设置任务 Pod 数、超时时间、重新次数、重启策略

配置项	描述
成功运行的 Pod 数	即 completions，指定 job 需要成功运行 Pods 的数量。默认值为 1

并行运行的 Pod 数	即 parallelism ,指定 job 在任一时刻应该并发运行 Pod 的数量。默认值为 1
超时时间	即 activeDeadlineSeconds，指定 job 可运行的时间期限，超过时间还未结束，系统将会尝试进行终止。
重新次数	Pod 执行失败后，尝试重建 Pod 的次数。默认是 6 次，每次失败后重试会有延迟时间，该时间是指数级增长，最长时间是 6min。
重启策略	仅支持不重启（Never）和失败时（OnFailure）

可选：在**标签和注释**区域，单击**添加**设置容器组的标签和注释。

配置项	**描述**
Pod 标签	为该 Pod 添加一个标签，标识该应用。
Pod 注解	为该 Pod 添加一个注解（Annotation）。

单击**创建**。

4.9.1.5.2 查看 CronJob 详情

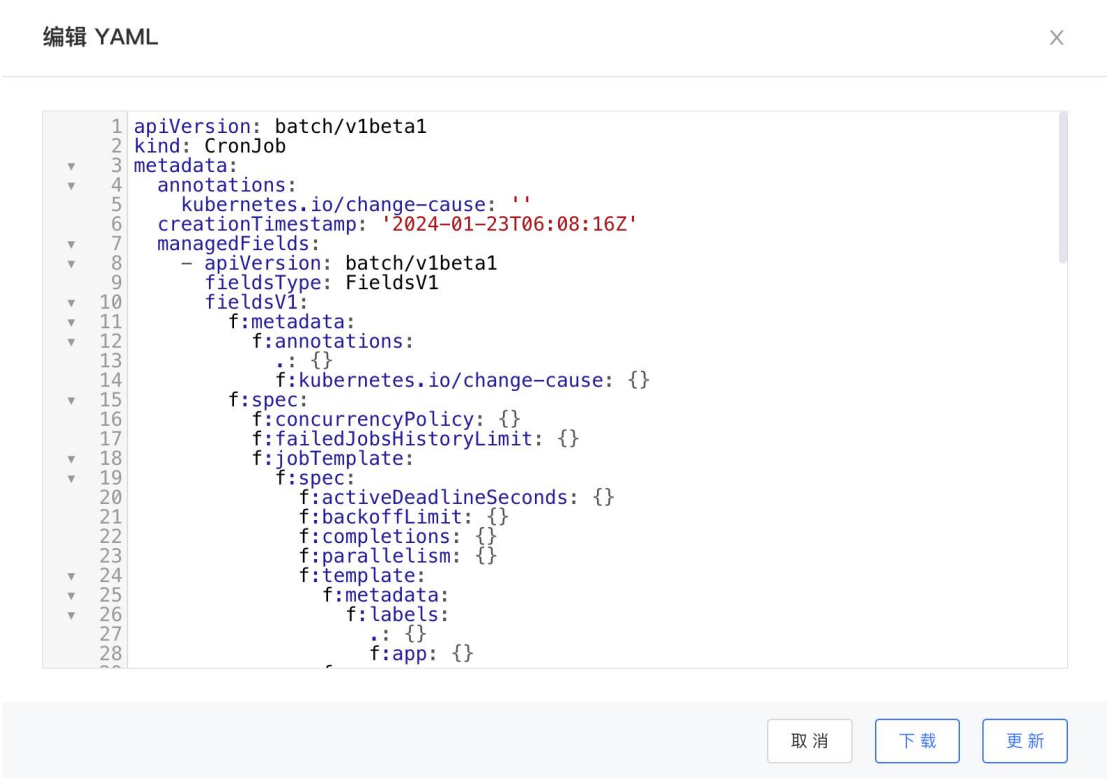
1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>定时任务**。
5. 在**定时任务**负载列表，单击目标负载右则操作列中的**详情**。

在应用详情页面可以基本信息、任务列表及事件。

4.9.1.5.3 修改 CronJob

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>定时任务**。
5. 在**定时任务**负载列表，单击目标负载右则操作列中的**编辑**。

通过 yaml 方式修改 CronJob 的配置。



4.9.1.5.4 停用 CronJob

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>定时任务**。
5. 在**定时任务**负载列表，单击目标负载右则操作列中的**停用**。
6. 弹出确认对话框，单击**确认**。

4.9.1.5.5 删除 CronJob

1. 登录边缘容器集群控制台。

2. 在控制台左侧导航栏中，单击**集群管理**。

3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

4. 在控制台左侧导航栏中，单击**工作负载>定时任务**。

5. 在**定时任务**负载列表，单击目标负载右则操作列中的**删除**。

6. 弹出确认对话框，单击**确认**即可删除 CronJob。

4.9.1.6 容器组(Pod)

4.9.1.6.1 查看容器组详情

查看容器组详细信息，包含基本信息、现状详情、容器、事件、创建者、初始化容器、存储及日志

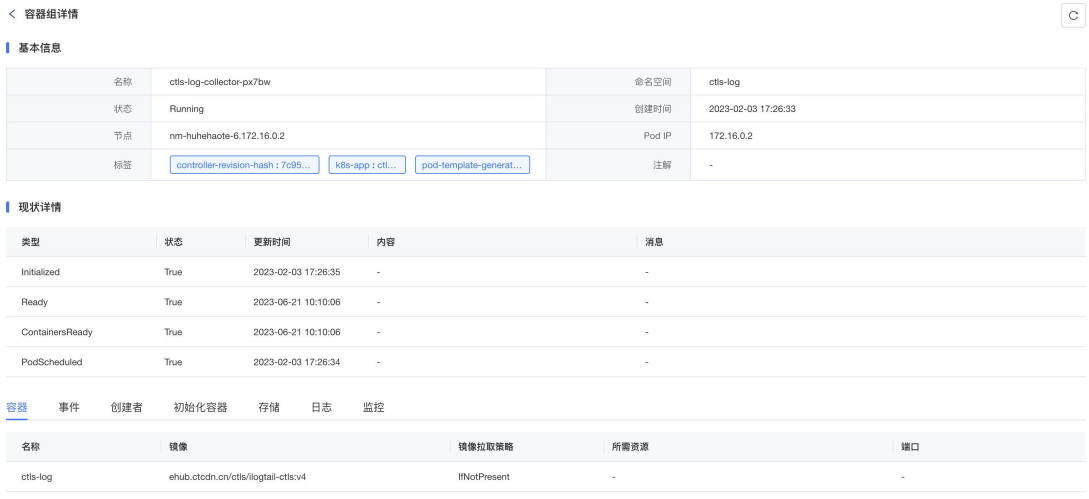
1. 登录边缘容器集群控制台。

2. 在控制台左侧导航栏中，单击**集群管理**。

3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

4. 在控制台左侧导航栏中，单击**工作负载>容器组**。

5. 在**容器组**列表，单击目标容器组右则的**详情**。



以下为容器组现状详情的说明：

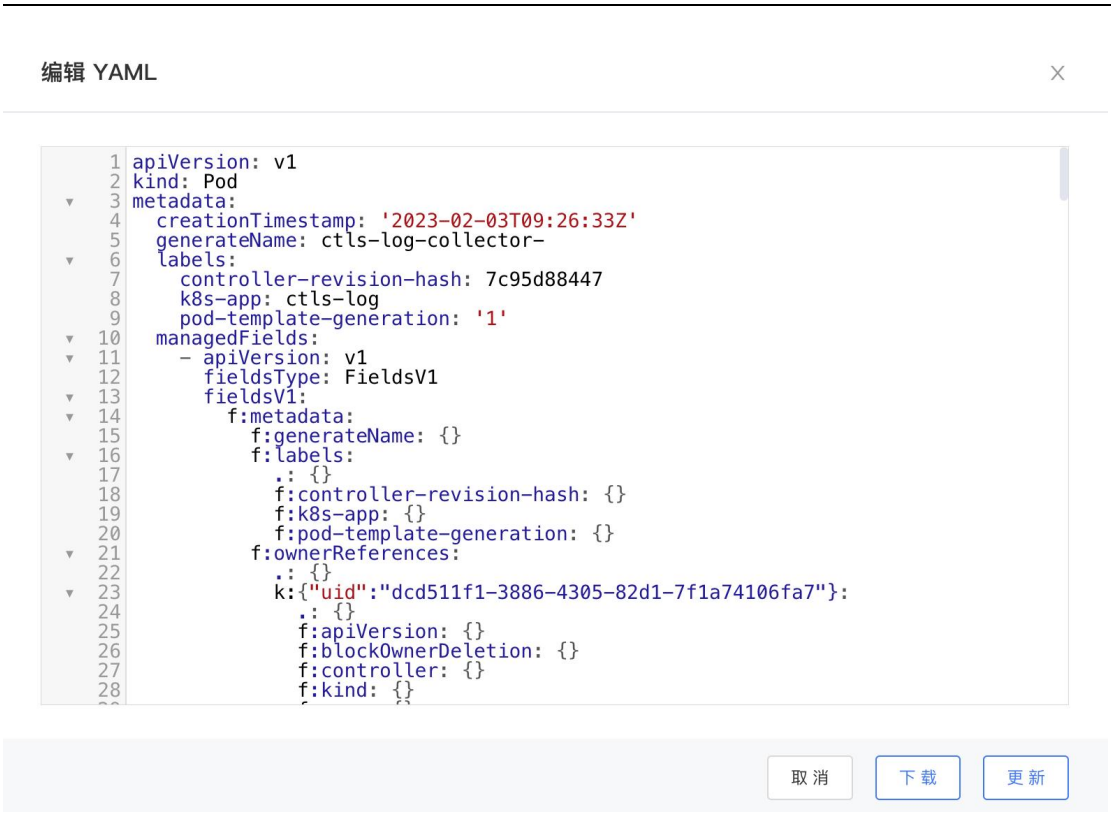
类型	描述
Initialized	所有的 Init 容器都已成功启动。
Ready	Pod 可以为请求提供服务 ,并且应该被添加到对应服务的负载均衡池中。
ContainersReady	Pod 中所有容器都已就绪。
PodScheduled	Pod 已经被调度到某节点。

4.9.1.6.2 编辑容器组

在容器组列表页面，您可对容器组进行编辑操作，对于通过部署（例如 Deployment）创建的容器组，建议您通过 Deployment 进行管理。

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>容器组**。
5. 在**容器组**列表，单击目标容器组右则的**编辑**

通过 yaml 文件编辑容器组。



4.9.1.6.3 删除容器组

在容器组列表页面，您可对容器组进行删除操作。对于通过部署（例如 Deployment）创建的容器组，建议您通过 Deployment 进行管理。

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**工作负载>容器组**。
5. 在**容器组**列表，单击目标容器组右则的**删除**。
6. 弹出对话框，单击**确认**删除容器组。

4.9.1.6.4 容器组终端

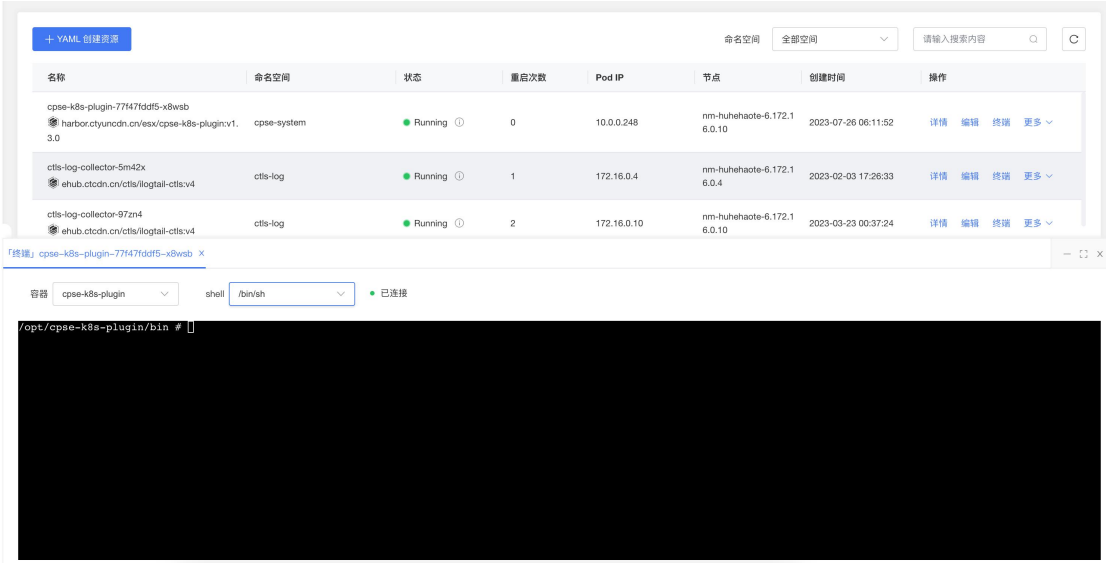
1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。

3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情。**

4. 在控制台左侧导航栏中，单击**工作负载>容器组。**

5. 在**容器组**列表，单击目标容器组右则的**终端。**

14. 60



4.9.1.6.5 查看容器组日志

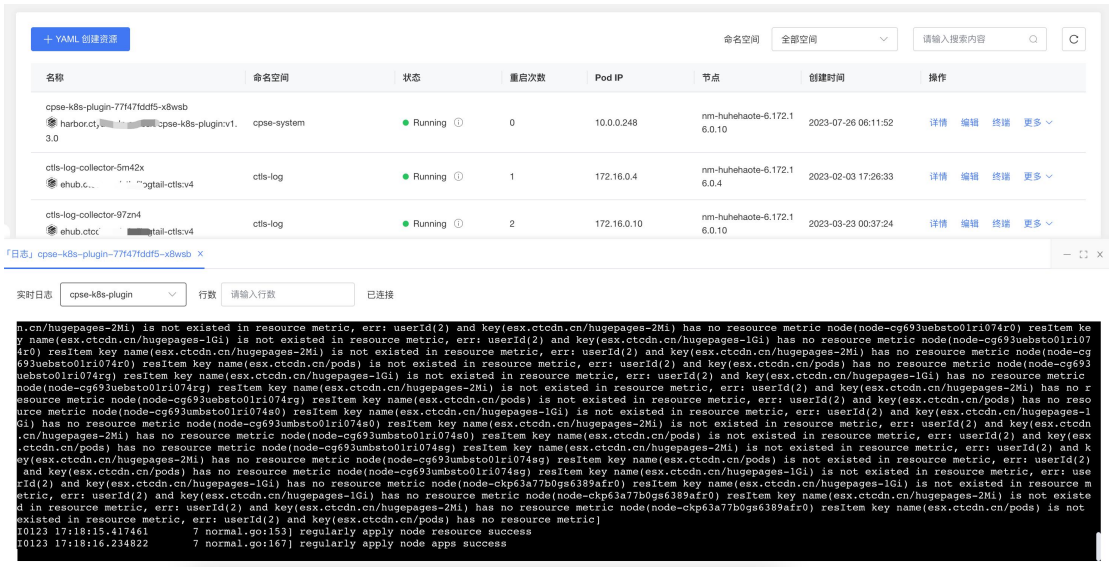
1. 登录边缘容器集群控制台。

2. 在控制台左侧导航栏中，单击**集群管理。**

3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情。**

4. 在控制台左侧导航栏中，单击**工作负载>容器组。**

5. 在**容器组**列表，单击目标容器组右则的**日志。**



4.9.1.6 管理自定义资源

Kubernetes 集群支持通过自定义资源定义（Custom Resource Definition）扩展 Kubernetes API，并允许您添加自定义资源（Custom Resources）。您可以查看集群中的所有 API 组和其包含的资源类型，查看各个资源类型的对象列表并进行管理操作。本文介绍如何管理自定义资源。

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理** 。
3. 在**集群列表**页面，选择目标集群并单击**详情**。
4. 在控制台左侧导航栏中选择**资源对象浏览器**。
5. 进入资源对象浏览器页面，您可以看到集群内的所有资源对象（包含自定义资源）。
6. 在左侧列表选择特定的 API 组 ,您可以查询对应的 Kubernetes 资源 ,并可以删除它们。

4.9.2 应用发布

4.9.2.1 通过 Nginx Ingress 实现灰度发布和蓝绿发布

前提条件

使用 ECK 应用管理功能安装 eck-ingress-nginx Helm Chart。

基本概念

灰度及蓝绿发布是为新版本创建一个与老版本完全一致的生产环境，在不影响老版本的前提下，按照一定的规则把部分流量切换到新版本，当新版本试运行一段时间没有问题后，将用户的全量流量从老版本迁移至新版本。

其中蓝绿发布就是一种灰度发布方式，一部分用户继续使用老版本的服务，将一部分用户的流量切换到新版本，如果新版本运行稳定，则逐步将所有用户迁移到新版本。

场景说明

基于客户端请求的流量切分场景

假设当前线上环境，您已经有一套服务 Service A 对外提供 7 层服务，此时上线了一些新的特性，需要发布上线一个新的版本 Service A'。但又不想直接替换 Service A 服务，而是希望将请求头中包含 foo=bar 或者 Cookie 中包含 foo=bar 的客户端请求转发到 Service A'服务中。待运行一段时间稳定后，可将所有的流量从 Service A 切换到 Service A'服务中，再平滑地将 Service A 服务下线。

基于服务权重的流量切分场景

假设当前线上环境，您已经有一套服务 Service B 对外提供 7 层服务，此时修复了一些问题，需要发布上线一个新的版本 Service B'。但又不想将所有客户端流量切换到新版本 Service B'中，而是希望将 20%的流量切换到新版本 Service B'中。待运行一段时间稳定后，再将所有的流量从 Service B 切换到 Service B'服务中，再平滑地将 Service B 服务下线。

针对以上的应用发布场景，ECK 的 Ingress Controller 支持下面的流量切分方式：

基于 Request Header 的流量切分，适用于灰度发布。

基于 Cookie 的流量切分，适用于灰度发布。

基于服务权重的流量切分，适用于蓝绿发布。

使用 canary-*注解实现灰度发布和蓝绿发布

注解说明

Annotation	说明
nginx.ingress.kubernetes.io/canary	是否开启灰度发布机制。取值：true *：启用* canary *功能* false：关闭 canary 功能
nginx.ingress.kubernetes.io/canary-by-header	表示基于请求头的名称进行灰度发布，取值：always *：无论什么情况下，流量均会进入灰度服务。* never：无论什么情况下，流量均不会进入灰度服务
nginx.ingress.kubernetes.i	表示基于请求头的值进行灰度发布。需要与

Annotation	说明
o/canary-by-header-value	canary-by-header 头配合使用。
nginx.ingress.kubernetes.io/canary-by-header-pattern	表示基于请求头的值进行灰度发布，并对请求头的值进行正则匹配。需要与 canary-by-header 头配合使用。取值为用于匹配请求头的值的正则表达式。
nginx.ingress.kubernetes.io/canary-by-cookie	表示基于 Cookie 进行灰度发布。例如，nginx.ingress.kubernetes.io/canary-by-cookie: foo。Cookie 内容的取值：* always：当 foo=always，流量会进入灰度服务。* never：当 foo=never，流量不会进入灰度服务。
nginx.ingress.kubernetes.io/canary-weight	表示基于权重进行灰度发布，权限取值：0~100。

不同灰度方式的优先级由高到低为：

canary-by-header>canary-by-cookie>canary-weight

步骤一：部署旧版本服务

1.创建 Deployment 和 Service。

```
apiVersion: apps/v1kind: Deploymentmetadata:
```

```
name: old-nginxspec:
replicas: 2
selector:
  matchLabels:
    run: old-nginx
template:
  metadata:
    labels:
      run: old-nginx
  spec:
    containers:
    - image: harbor.ctyuncdn.cn/eecdn/nginx:old
      imagePullPolicy: Always
      name: old-nginx
      ports:
      - containerPort: 80
        protocol: TCP
    restartPolicy: Always---apiVersion: v1kind: Servicemetadata:
```

```
name: old-nginxspec:
ports:
- port: 80
  protocol: TCP
  targetPort: 80
selector:
  run: old-nginx
sessionAffinity: None
type: NodePort
```

2.创建 Ingress。

```
apiVersion: networking.k8s.io/v1kind: Ingressmetadata:
  name: gray-releasespec:
  rules:
    - host: a.ctyun.cn
      http:
        paths:
          # 老版本服务。
          - path: /
```

```
            backend:
              service:
                name: old-nginx
                port:
                  number: 80
            pathType: ImplementationSpecific
```

步骤二：部署新版本服务

创建新版本的 Deployment 和 Service。

```
apiVersion: apps/v1kind: Deploymentmetadata:
  name: new-nginxspec:
  replicas: 1
  selector:
    matchLabels:
      run: new-nginx
  template:
    metadata:
      labels:
        run: new-nginx
```

```
  spec:
    containers:
    - image: harbor.ctyuncdn.cn/eecdn/nginx:new
      imagePullPolicy: Always
      name: new-nginx
      ports:
      - containerPort: 80
        protocol: TCP
    restartPolicy: Always---apiVersion: v1kind: Servicemetadata:
name: new-nginxspec:
ports:
- port: 80
  protocol: TCP
  targetPort: 80
selector:
  run: new-nginx
sessionAffinity: None
type: NodePort
```

步骤三：灰度发布验证

以设置满足特定规则 request header 的请求才能访问新版本服务为例（以下示例仅请求头中满足 foo=bar 的客户端请求才能路由到新版本服务）来验证：

1.创建新的 Ingress。

```
apiVersion: networking.k8s.io/v1kind: Ingressmetadata:
  name: gray-release-canary
  annotations:
    # 开启 Canary。
    nginx.ingress.kubernetes.io/canary: "true"
    # 请求头为 foo。
    nginx.ingress.kubernetes.io/canary-by-header: "foo"
    # 请求头 foo 的值为 bar 时，请求才会被路由到新版本服务 new-nginx 中。
    nginx.ingress.kubernetes.io/canary-by-header-value: "bar"
spec:
  rules:
    - host: a.ctyun.cn
      http:
        paths:
```

```
          # 新版本服务。
          - path: /
            backend:
              service:
                name: new-nginx
                port:
                  number: 80
            pathType: ImplementationSpecific
```

2.不携带特殊请求头验证。

```
curl -H "Host: a.ctyun.cn"    http://
```

期望返回：old

3.携带特殊请求头验证。

```
curl -H "Host: a.ctyun.cn" -H "foo: bar" http://
```

期望返回：new

4.9.3 应用部署

4.9.3.1 使用 YAML 部署和暴露 Nginx 服务

前提条件

已创建 ECK 集群，集群状态为运行中。

步骤一：部署 Nginx 应用

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击集群管理。
3. 在集群列表页面，选择所需的集群并单击详情，然后在控制台左侧导航栏中选择工作负载>无状态 。
4. 在无状态页面，单击左上角 YAML 创建资源 ，使用 ResourceBasicDeployment 公有模板创建一个 Nginx。
5. 创建成功后，在无状态列表中可以看到创建的 Nginx deployment。

步骤二：创建 Service 并发布 Nginx 应用

1. 在集群列表页面，选择所需的集群并单击详情，然后在控制台左侧导航栏中选择网络 > 服务 。
2. 在服务页面，单击左上角创建服务 ，然后在添加服务对话框，配置服务信息，然后单击确认。配置说明见 服务 Service 管理。

参数	说明	示例值
名称	服务名称	nginxservice
类型	设置为虚拟集群 IP	

参数	说明	示例值
关联	设置关联的 deployment	nginx-deployment-basic
端口映射	自定义端口名称，配置服务端口为 80，容器端口为 80，协议为 TCP。	

步骤三：创建 Nginx Ingress 路由对外暴露 Nginx 应用

1. 在集群列表页面，选择所需的集群并单击详情，然后在控制台左侧导航栏中选择网络 > 服务 。

2. 在路由页面，单击左上角创建路由 ，然后在创建路由对话框，配置路由信息，配置说明见 路由 ingress 管理，将 Ingress 关联至上述 Service，然后单击创建 。

3. 等待 1 分钟左右，路由列表的端点列将显示路由的 IP。在浏览器中输入端点 IP，显示 Nginx 欢迎页面，表明路由创建成功。

4.9.3.2 调度应用至指定节点

前提条件

1. 已创建 ECK 集群，集群状态为运行中。

2. 已创建无状态工作负载 Deployment 或创建有状态工作负载 StatefulSet。

操作步骤

1.设置节点标签。

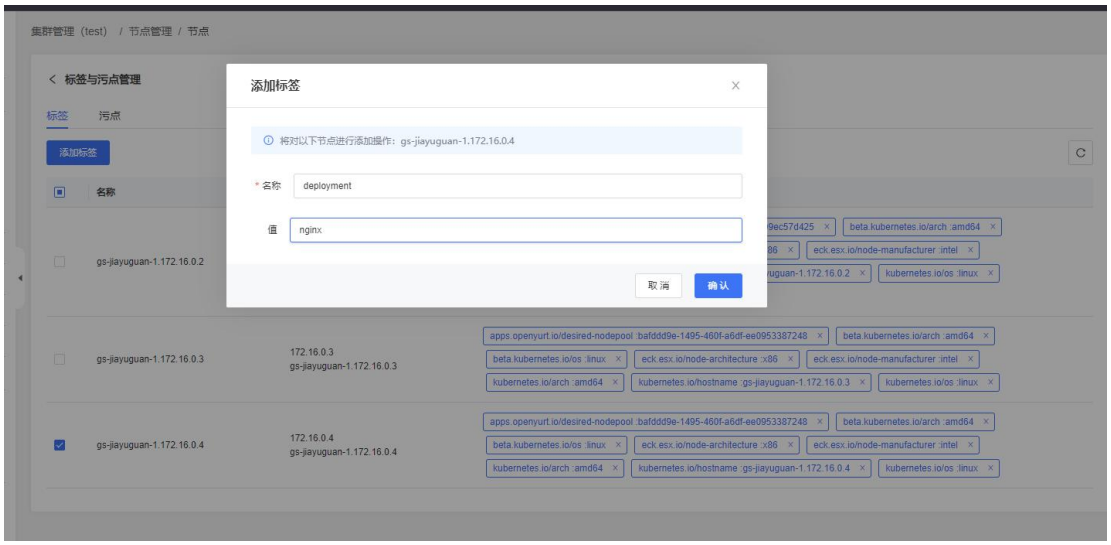
a. 登录边缘容器集群控制台。

b. 在控制台左侧导航栏中，单击集群管理 。

c. 在集群列表页面，选择所需的集群并单击详情，在控制台左侧导航栏中选择节点管理>节点 。

d. 进入节点列表页面，单击左上角的标签与污点管理 。

e. 勾选目标节点，单击左上角添加标签，在弹出的添加对话框中，输入标签的名称和值，然后单击确认。



2. 调度应用到指定节点。

a. 登录边缘容器集群控制台。

b. 在控制台左侧导航栏中，单击集群管理 。

c. 在集群列表页面，选择所需的集群并单击详情，在控制台左侧导航栏中选择工作负载>无状态 。

d. 进入无状态负载页面，单击列表右侧操作>Yaml 编辑，为应用设置 nodeSelector。本文以名称为 nginx-deployment-basic 的无状态工作负载为例进行说明。

说明

名称：标签名称由字母、数字、短划线（-）、下划线（_）、小数点（.）组成，且必须以字母或者数字开头和结尾。本文示例为 deployment。

值：标签值可以为空字符串或由字母、数字、短划线（-）、下划线（_）、小数点（.）组成，且必须以字母或者数字开头和结尾。本文示例为 nginx。

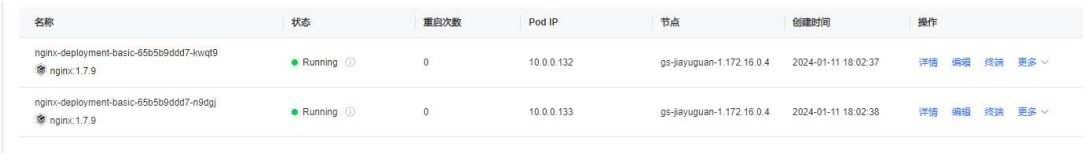
```
apiVersion: apps/v1 kind: Deploymentmetadata:
  name: nginx-deployment-basic
  labels:
    app: nginxspec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
```

```
spec:
  nodeSelector:
    deployment: nginx # #添加节点的标签，以保证您的应用只可以运行在目标节点上。请使用实际值。
  containers:
  - name: nginx
    image: nginx:1.7.9
    ports:
    - containerPort: 80
    resources:
      limits:
        cpu: "500m"
```

e. 单击**更新**，会提示部署状态信息。

结果验证

单击**工作负载>容器组**页面，查看对应名称的容器组，目标应用 Pod 全部被调度到了 gs-jiayuguan-1.172.16.0.4 的节点上，这个节点正是本文示例中在节点中打了标签（deployment: nginx）的节点。



4.9.3.3 调度应用至指定节点池

前提条件

已创建 ECK 集群，集群状态为运行中。

已创建无状态工作负载 Deployment 或创建有状态工作负载 StatefulSet。

操作步骤

1. 查看本节点池的特定标签。

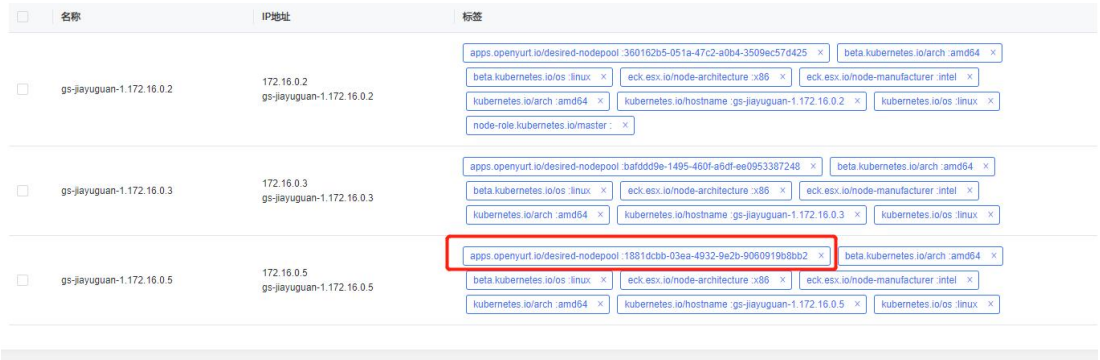
a. 登录边缘容器集群控制台。

b. 在控制台左侧导航栏中，单击**集群管理** 。

c. 在**集群列表**页面，选择所需的集群并单击左侧的**节点管理** > **节点** 。

d. 在节点列表页，点击左上角的**标签与污点管理**，进入节点的标签页面，找到名称为 apps.openyurt.io/desired-nodepool 的标签，此标签的值相同的节点在同一个节点池内。

1.



2. 为应用设置调度策略。

上述步骤已经获取了节点池特定标签 apps.openyurt.io/desired-nodepool:<特定值>。您可以利用 nodeSelector 或者 nodeAffinity 保证您的应用限定在指定节点池上运行。

- 为应用设置 **nodeSelector** 。**nodeSelector** 是 spec 中的一个字段。您只需要将上述的 apps.openyurt.io/desired-nodepool: <特定值>标签填充到 nodeSelector 中。示例如下：

```
apiVersion: apps/v1 kind: Deploymentmetadata:
  name: nginx-deployment-basic
  labels:
    app: nginxspec:
  replicas: 2
  selector:
```

```
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      nodeSelector:
        apps.openyurt.io/desired-nodepool: xxx # #添加节点池的特定标签，以保证您的应用只可以运行在目标节点节点池下的节点上。请使用实际值。
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
        resources:
          limits:
            cpu: "500m"
```

- 为应用设置 **nodeAffinity** 。
  - **requiredDuringSchedulingIgnoredDuringExecution** 表示 Pod 必须部署到满足条件的节点上。如果没有满足条件的节点，调度操作就会不停重试。其中 **IgnoreDuringExecution** 表示 Pod 部署之后运行时，如果节点标签发生了变化，不再满足 Pod 指定的条件，Pod 也会继续运行。
  - **preferredDuringSchedulingIgnoredDuringExecution** 表示优先部署到满足条件的节点上，如果没有满足条件的节点，则忽略这些条件，按照正常逻辑部署。

  本文示例使用 requiredDuringSchedulingIgnoredDuringExecution 策略保证应用始终运行在指定的节点池上。

```
apiVersion: apps/v1 kind: Deploymentmetadata:name: nginx-deployment-basiclabels:
```

```
app: nginxspec:replicas: 2selector:
matchLabels:
  app: nginxtemplate:
metadata:
  labels:
    app: nginx
spec:
  nodeAffinity:
    requiredDuringSchedulingIgnoredDuringExecution:
      nodeSelectorTerms:
        - matchExpressions:
            - key: apps.openyurt.io/desired-nodepool
              operator: In
              values:
                - xxx        #标签的值，需要根据实际情况填写。
  containers:
  - name: nginx
    image: nginx:1.7.9
    ports:
    - containerPort: 80
    resources:
      limits:
        cpu: "500m"
```

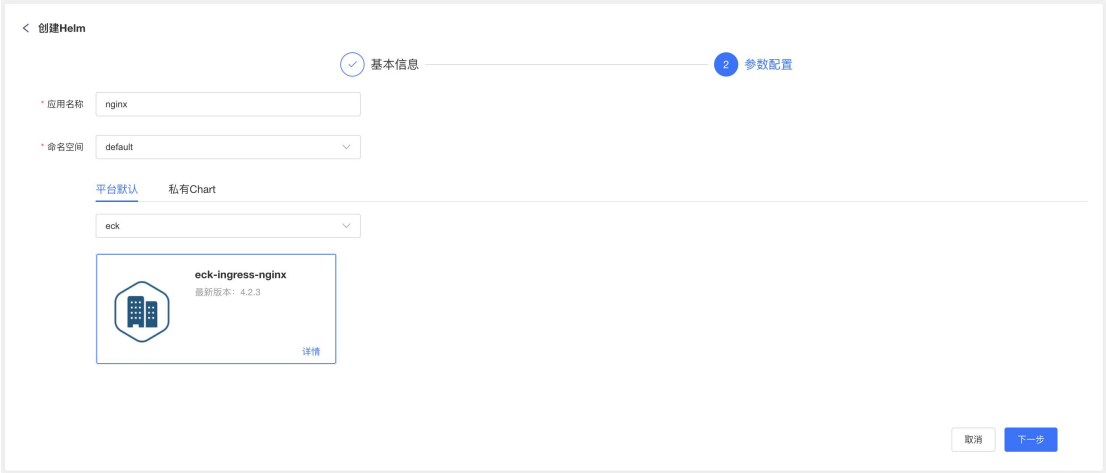
结果验证

单击 **工作负载** >**容器组**页面，查看对应名称的容器组，发现目标应用调度到了特定节点池下的某个节点上。

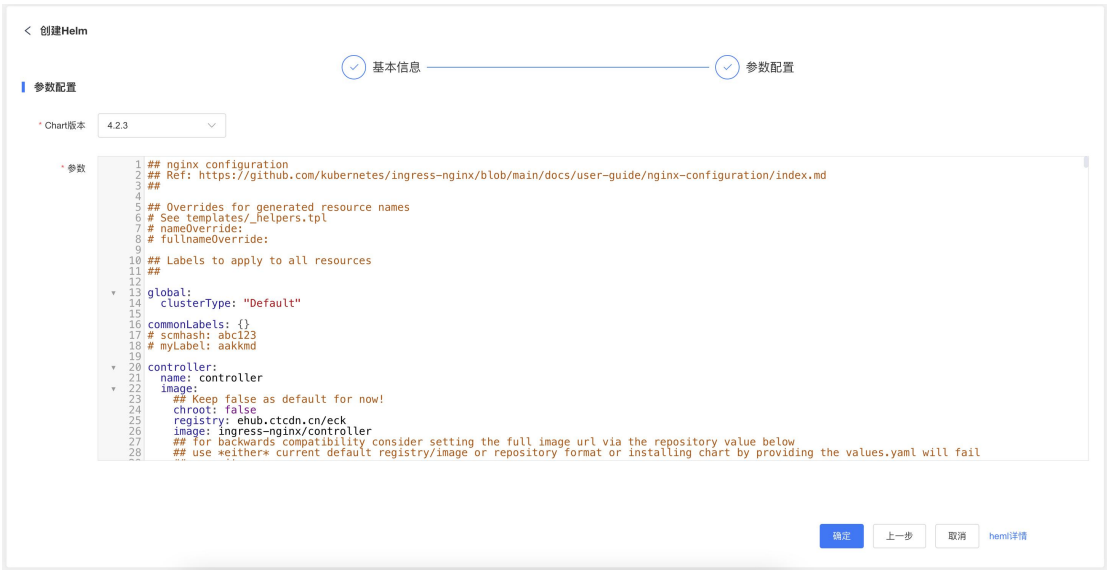
4.9.4 使用 Helm 管理应用

4.9.4.1 创建 Helm

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在集群列表页面中，单击目标集群右侧操作列下的**详情**。
4. 在控制台左侧导航栏中，选择**应用>Helm**。
5. 在应用列表，单击左上角的的**创建 Helm**。
6. 在基本信息页面，填写应用名称、选择命名空间，并选择应用，然后单击**下一步**。



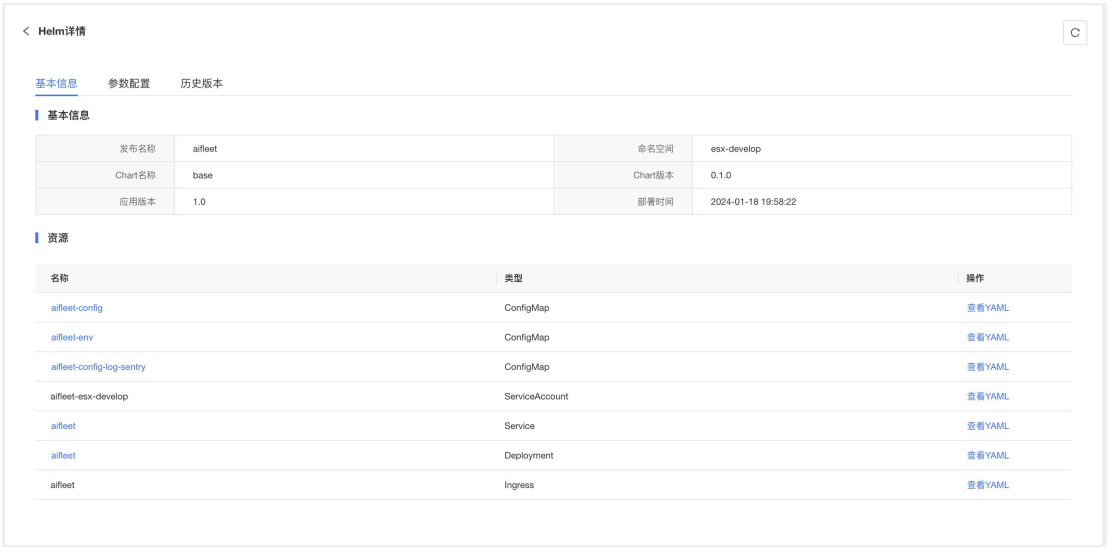
7. 在参数配置页面，选择 Chart 版本，并根据需求配置参数，然后单击**确定**，即可创建应用。



4.9.4.2 查看 Helm 详情

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**应用>Helm**。
5. 在 **helm** 列表，单击目标负载右则操作列中的**详情**。

在详情页面可以应用基本信息、配置参数及历史版本。



4.9.4.3 删除 Helm

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。
4. 在控制台左侧导航栏中，单击**应用>Helm**。
5. 在 **Helm** 列表，单击目标服务右则的**删除**。
6. 在弹出的确认框中，单击**确认**即可删除。

4.10 弹性伸缩

4.10.1 弹性伸缩概述

弹性伸缩是一种根据用户实际业务需求自动调整计算资源的能力。ECK 为用户提供了容器水平伸缩（HPA）与节点自动伸缩（cluster-autoscaler）两个组件。

组件名称	简介	适用场景
HPA	Kubernetes 的工作负载类型中，支持定义 Pod 的副本数，以适应不同程度的负载。但负载升高，可以调高 Pod 的副本数来调用更多的资源。对此，Kubernetes 官方提供了 HPA，来基于 CPU 或者内存等资源利用率	可以用于调整 ReplicationController、Deployment、ReplicaSet 和 StatefulSet 中 Pod 的副本数，但不适用于无法扩缩的对象，

组件名称	简介	适用场景
	调整工作负载的副本数。	比如 DaemonSet。
cluster-autoscaler	CA(cluster-autoscaler)是用来弹性伸缩 Kubernetes 集群的。我们在使用 Kubernetes 集群经常问到的一个问题是，我应该保持多大的节点规模来满足应用需求呢？ cluster-autoscaler 的出现解决了这个问题，它可以自动地根据部署的应用所请求的资源量来动态的伸缩集群。	适合所有场景。

4.10.1 设置容器水平伸缩（HPA）

设置方法

1.登录边缘容器集群控制台。

2.单击目标集群的**详情**按钮。

3.在控制台左侧导航栏中，单击左侧**工作负载**，再单击**无状态/有状态** 。

4.点击**创建无状态负载/创建有状态负载** 。

5.在**高级设置**-**指标伸缩**勾选**开启。**

6.设置参数。参数说明：

配置项	描述
指标	可选 **CPU 使用量**和**内存使用量** 。
触发条件	指定指标使用量的百分比。如果超过使用量，则会触发 Pod 扩容。
最小副本数	指定该工作负载可调整的副本数最小值。
最大副本数	指定该工作负载可调整的副本数最大值。

4.11 GPU

4.11.1 GPU 调度

ECK 支持对 GPU 卡虚拟化进行虚拟化并对 GPU 资源进行统一调度，一张卡可虚拟成 5 张卡，支持算力隔离、显存隔离，最小单位 1MB；支持虚拟显存，当显存不足时内存来凑。在使用 GPU 资源前您需先拥有带 GPU 的节点，并且安装好相应的 GPU 组件，然后就可以部署的您的应用让其使用 GPU 资源了。

4.11.1.1 创建 GPU 节点

支持在创建集群时或集群创建后，选择创建一个 GPU 型节点池来创建 GPU 类型工作节点

方法一：创建集群时创建 GPU 节点

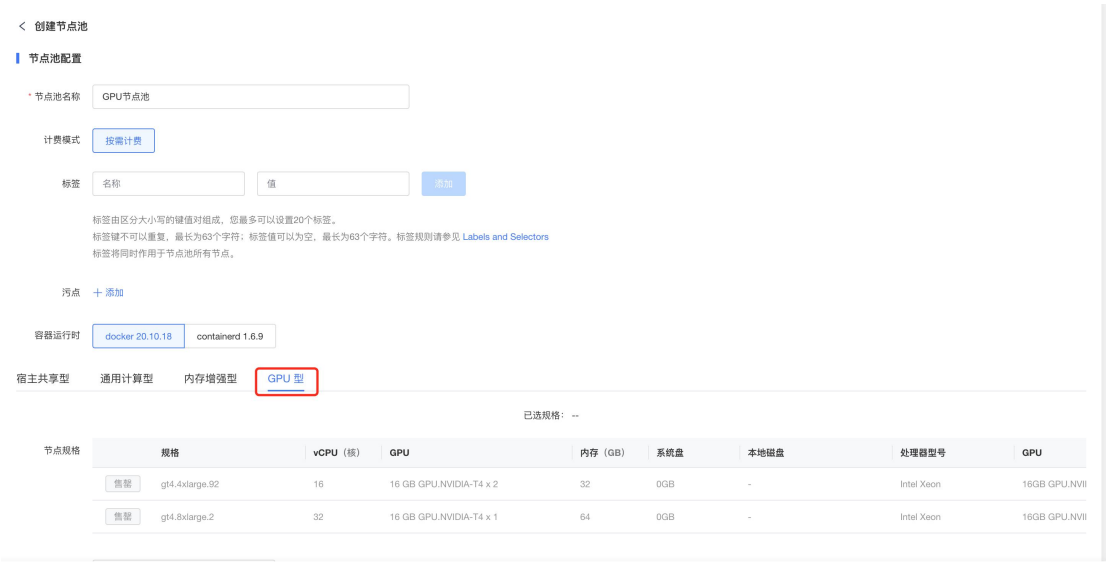
1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理。**
3. 在**集群列表**页面中，单击页面左上角的**创建专有集群。**

在“节点池配置”环节，节点规格选择 GPU 型，即可在创建集群后创建 GPU 类型的工作节点，创建集群操作详见 “创建专有版集群”。

方法二：通过新建节点池创建 GPU 节点

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理。**
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情。**
4. 在控制台左侧导航栏中，单击**节点管理>节点池。**
5. 在节点池列表页面，单击左上解的**创建节点池**

节点规格选择 GPU 型，即可在创建节点池后创建 GPU 类型的工作节点，创建节点池操作详见 “创建节点池”。



4.11.1.2 安装 GPU 组件

ECK 提供了 GPU 虚拟化组件，通过 GPU 组件可以对 GPU 卡进行虚拟化，支持多个容器共享 1 张 GPU 卡，安装 GPU 组件操作如下：

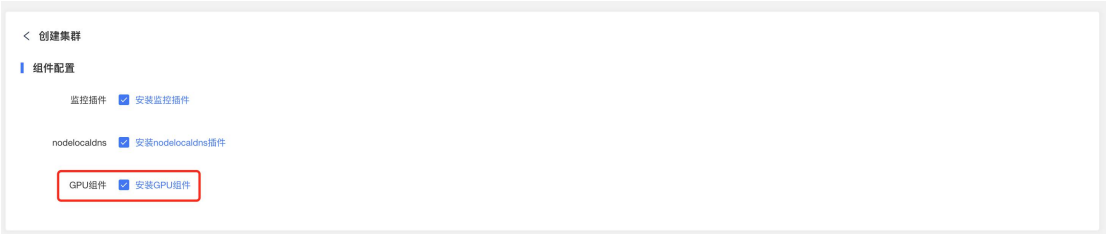
方法一：集群创建时安装 GPU 组件

1.登录边缘容器集群控制台。

2.在控制台左侧导航栏中，单击**集群管理** 。

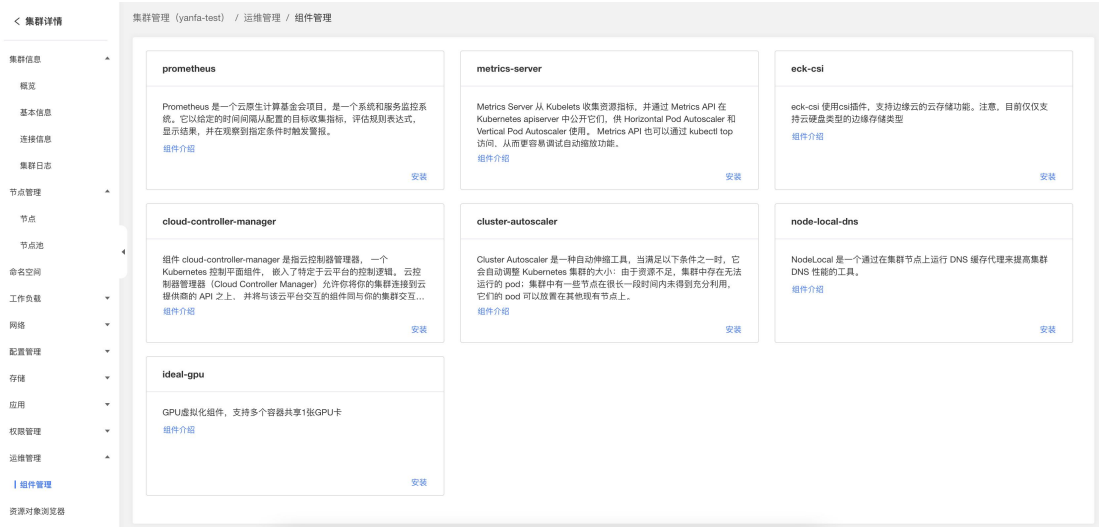
3.在**集群列表**页面中，单击页面左上角的**创建专有集群** 。创建集群操作详见 “创建专有版集群” 。

4.在“组件配置”环节，选择安装“GPU 组件”。



方法二：通过组件管理安装 GPU 组件

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群**。
3. 在**集群列表**页面，选择所需的集群并单击右侧的**... > 组件管理**。
4. 进入**组件管理**页面，选择 GPU 组件，单击**安装**即可安装组件



4.11.1.3 使用 GPU

前提条件

1.创建 GPU 类型节点

2.安装 GPU 组件

操作步骤

创建工作负载时，申请 GPU 资源，按如下方法配置，指定工作负载可使用 GPU 的数量

1. 登录边缘容器集群控制台。
2. 在控制台左侧导航栏中，单击**集群管理**。
3. 在**集群列表**页面中，单击目标集群右侧**操作**列下的**详情**。

4. 在控制台左侧导航栏中，单击**工作负载>无状态。**

5. 在**无状态**负载列表，单击左上角的**创建无状态负载。**

6. 在容器配置页面，勾选 GPU 配额，并指定使用的比例和大小，在调度时 ECK 会自动将负载调度到有 GPU 的节点。

配置项	描述
容器块数	支持配置容器最多可占用几 GPU 卡。
算力	容器可占用每块 GPU 算力的最大比例。
显存	容器可占用每块 GPU 算力的最大比例。

也可以通过在 yaml 文件添加以下内容实现 GPU 资源配置：

```
- apiVersion: apps/v1
  kind: Deployment
  metadata:
    name: ''
    labels:
      app: ''
    annotations:
      kubernetes.io/change-cause: ''
    namespace: default
  spec:
    replicas: 2
    selector:
      matchLabels:
        app: ''
    template:
      metadata:
        labels:
          app: ''
```

```
    annotations: {}
  spec:
    initContainers: []
    containers:
      - name: ''
        image: ''
        imagePullPolicy: IfNotPresent
        command: []
        args: []
        env: []
        envFrom: []
        stdin: false
        tty: false
        gpuChecked: true
        resources:
          requests:
            cpu: 0.25
            memory: 512Mi
            ideal.com/vcuda-core: 80
            ideal.com/vcuda-memory: 200
            ideal.com/gpu: 2
          limits:
            ideal.com/vcuda-core: 80
            ideal.com/vcuda-memory: 200
            ideal.com/gpu: 2
        volumeMounts: []
        ports: []
        livenessProbe: null
        readinessProbe: null
        startupProbe: null
        lifecycle: null
        securityContext:
          privileged: false
    imagePullSecrets: []
```

GPU 节点标签

创建 GPU 节点后，ECK 会给节点打上对应标签，不同类型的 GPU 节点有不同标签，利用 GPU 节点标签可以灵活地将应用调度到具有 GPU 设备的节点上。

选择一个 GPU 节点，执行以下命令，查看该 GPU 节点的标签。

```
kubectl describe node fj-xiamen-4.172.16.0.6
```

返回值：

```
Name:               fj-xiamen-4.172.16.0.6
Roles:              <none>
Labels:             apps.openyurt.io/desired-nodepool=06b59c8e-d0a4-44be-8f27-e8219f2a51f6
                    apps.openyurt.io/nodepool=06b59c8e-d0a4-44be-8f27-e8219f2a51f6
                    beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/os=linux
                    gpu-model=NVIDIA-GeForce-RTX-3060
                    kubernetes.io/arch=amd64
                    kubernetes.io/hostname=fj-xiamen-4.172.16.0.6
                    kubernetes.io/os=linux
                    node-type=gpu
                    nvidia-device-enable=enable
                    openyurt.io/is-edge-worker=false
```

在使用 GPU 时，可以根据标签让 Pod 与节点亲和，从而让 Pod 选择正确的节点，如下所示。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: gpu-test
  namespace: default
spec:
  replicas: 1
  selector:
    matchLabels:
      app: gpu-test
  template:
    metadata:
      labels:
        app: gpu-test
    spec:
      nodeSelector:
        gpu-model: NVIDIA-GeForce-RTX-3060
      containers:
      - image: nginx:perl
        name: container-0
        resources:
          requests:
            cpu: 250m
            memory: 512Mi
            nvidia.com/gpu: 1     # 申请 GPU 的数量
          limits:
            cpu: 250m
```

```
            memory: 512Mi
            nvidia.com/gpu: 1     # GPU 数量的使用上限
    imagePullSecrets:
    - name: default-secret
```

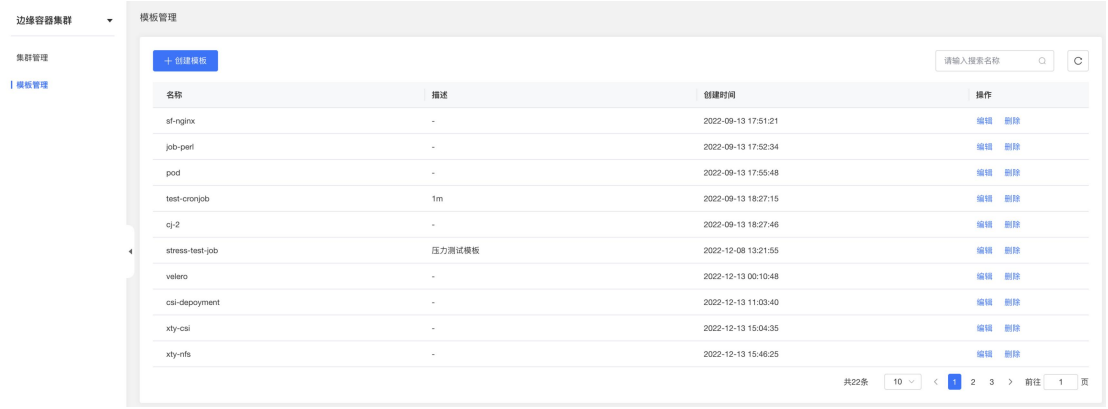
4.12 模板管理

使用自定义模板或管理平台提供的模板，快速通过 yaml 方式创建工作负载

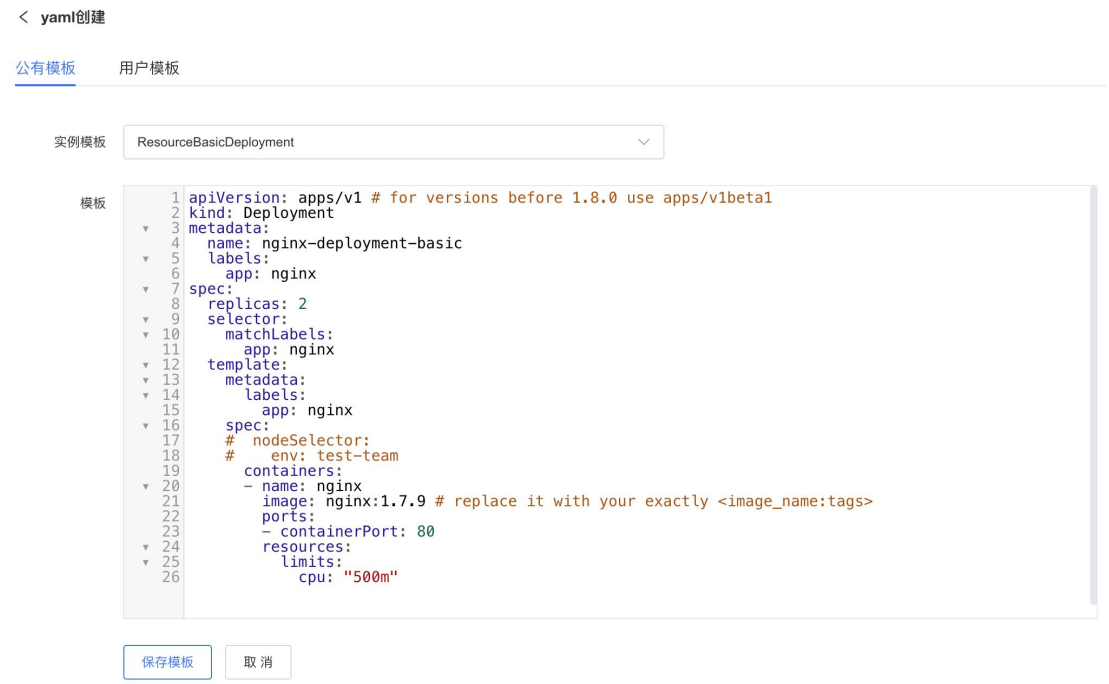
4.12.1 创建模板

1. 登录边缘容器集群控制台。

2. 在控制台左侧导航栏中，单击**模板管理**。



3. 在**模板列表**页面，单击左上角的**创建模板**。用户可以根据已经有模板或空白模板创建一个新的模板。



4.13 组件管理

4.13.1 组件概述

边缘容器集群 ECK 提供多种类型的组件，您可以根据业务需求部署、升级、卸载组件。本文从功能维度列举 ECK 管理的集群组件。

组件类型

边缘容器集群 ECK 管理的集群组件类型包括系统组件和可选组件：

- 系统组件：创建 ECK 集群时，默认安装的组件。
- 可选组件：创建 ECK 集群时，可选择性安装的组件，用于扩展集群功能。

主要组件介绍

组件名称	组件类型	功能概述
cloud-controller-manager	系统组件	云控制器管理器， 一个 Kubernetes 控制平面组件，嵌入了特定于云平台的控制逻辑。
metrics-server	系统组件	一个资源指标收集组件，通过 Metrics API 在 Kubernetes apiserver 中公开它们。
prometheus	可选组件	一个系统和服务监控组件。它以给定的时间间隔从配置的目标收集指标，提供指标自动发现和可视化能力。
eck-csi	可选组件	提供边缘云的云存储功能。注意，目前仅仅支持云硬盘类型的边缘存储类型。
cluster-autoscaler	可选组件	是一种自动伸缩组件，当满足特定条件自动调整 Kubernetes 集群的大小。
node-local-dns	可选组件	一个通过在集群节点上运行 DNS 缓存代理来提高集群 DNS 性能的工具。
ideal-gpu	可选组件	GPU 虚拟化组件，支持多个容器共享 1 张 GPU 卡。

4.13.1 管理组件

边缘容器集群 ECK 提供集群管理、监控管理等多种组件，便于您管理和运维集群。当 ECK

的组件更新时，您可以在控制台升级组件至最新版本。本文介绍如何管理组件，包括升级、安装、卸载等操作。

1.登录边缘容器集群控制台。

2.在控制台左侧导航栏中，单击**集群管理** 。

3.在**集群列表**页面，选择所需的集群并单击左侧的**更多** > **组件管理** 。

4.进入**组件管理**页面，您可选择需要安装的组件，单击**安装**即可安装组件。

5.安装成功的组件，可通过单击**卸载**按钮进行卸载。

6.安装成功的组件，如果有新版本会显示**升级**按钮，单击**升级**可进行升级。

5 常见问题

5.1 计费类

Q：边缘容器集群支持哪些计费方式？

A：目前暂支持按需求计费方式，后续将退出包年包月计费方式

Q：边缘容器集群如何收费？

A：边缘容器集群 专有版 集群本身不收费，但在使用过程中会创建相关智能边缘云产品资源，您需要为您使用的这些资源付费，如虚拟机、云硬盘、带宽等，智能边缘云产品资源详细价格请参见相应 产品价格表

Q：如何查看云产品资源的话单？

A :集群创建的相关智能边缘云产品资源的话单可到智能边缘云 ECX 控制台-总览-话单管理 查看

Q：集群节点对应虚拟机关机后是否会继续计费？

A：如果用户采用按需计费购买，虚机关机后将，相应的 CPU、内存、GPU 资源会被临时释放，这部分资源将不会被计费，但虚机占用的系统盘、数据盘资源仍然会被计费。如果您关机的时间太长，您的虚机资源可能会因为边缘集群资源不足而重启失败，您可以稍后重试。

Q：欠费冻结后的如何计费？

A：用户欠费后，ECK 集群创建的智能边缘云实例占用的计算资源将进入 1 小时的保留

期，1 小时内若用户未充值则会进入冻结期。进入保留期时，资源可以正常使用并正常计费。进入冻结期时，虚拟机将不能使用，虚拟机的 CPU、内存、GPU 资源将会被释放，但系统盘和挂载的数据盘将会保留并按原价继续收费；边缘存储实例将不能使用，实例资源及数据将会保留并按原价继续收费；网络将无法访问。冻结期内，若用户仍未充值，冻结期结束时系统会释放虚拟机、边缘存储实例、弹性 IP、NAT 网关、负载均衡等资源，VPC、专线、路由表、SSL 证书等资源或配置会被保留。

5.2 购买类

Q：边缘容器集群支持哪些资源节点？

A：目前开放的节点包括了福建-三明 1、福建-厦门 1、甘肃-张掖 1、海南-三亚 1、浙江-绍兴 2，后续会随业务需求和资源布局开放更多节点。如您有较大量的边缘资源节点需求，请联系您的客户经理，或线上咨询或拨打客服热线电话寻求帮助，热线话：400-810-9889 转 1。

Q：边缘容器集群是否支持试用？

A：边缘容器集群专有版 集群不收费，若需试用请按智能边缘云试用流程开通体验包试用，详细请参见 体验包开通

5.2 购买类

Q：创建集群 APIServer 选不到 IP

A：到 ECX 控制台-专有网络-公网 IP 页面确认对应的区域是否已经创建了公网 IP，没有则需要先创建。

Q：创建集群选不到虚拟私有云

A :到 ECX 控制台-专有网络-虚拟私有云页面确认对应的区域是否已经创建了虚拟私有云，没有则需要先创建。

Q：创建集群时节点规格售罄或无合适节点规格选择

A：更换区域创建集群，或联系客户经理申请。

Q：创建公网 IP 时，提示配额不足改如何处理

A：可能是当前区域的公网 IP 配额不足了，可以更换区域创建(需要根据集群创建的区域决定)，或联系客户经理申请。

Q：怎样登录集群节点对应的虚机

A：到 ECX 控制台-边缘虚拟机页面，找到节点 ID 对应的虚机，在右侧操作里面点击登录，使用创建节点时设置的账号密码登录。

Q：使用集群 KubeConfig 无法连接集群

A：在 ECK 客户控制台-集群管理-详情-连接信息页面点击吊销 KubeConfig，使用新生成的 KubeConfig 重新进行连接。

Q：创建 LoadBalancer 类型 Service 后，状态一直 Pending，没有分配公网 IP

A：您可以通过执行命令 kubectl -n {your-namespace} describe svc {your-svc-name}查看事件信息，查看具体原因。 如果错误信息中包含 Insufficient.IP ，表示节点 IP 已用完，可发起工单联系客服咨询。

6 相关协议

6.1 天翼云边缘容器集群服务协议

天翼云边缘容器集群服务协议

6.2 天翼云边缘容器集群服务等级协议

天翼云边缘容器集群服务等级协议