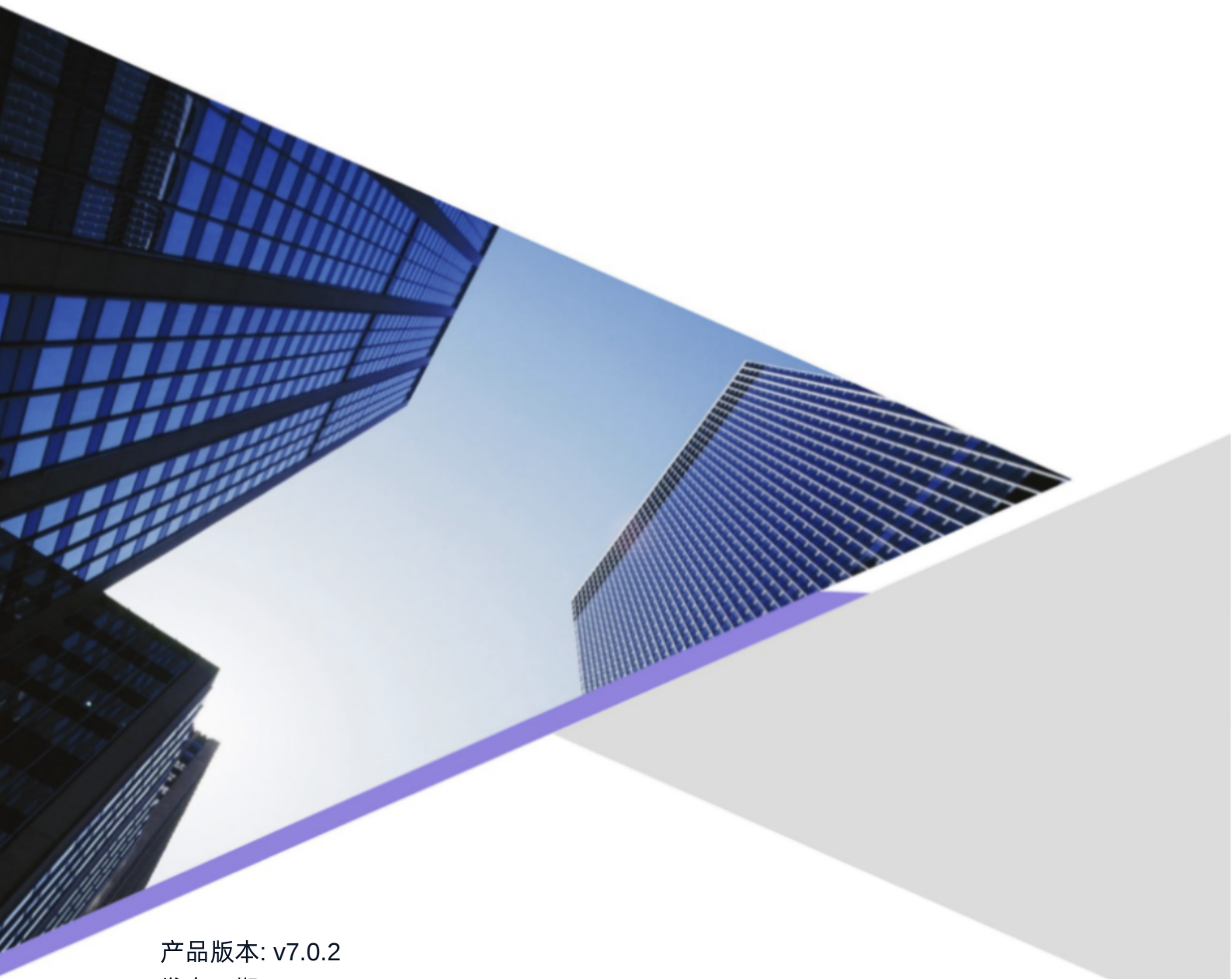
EasyStack
易捷行云

安全容器服务

使用手册



产品版本: v7.0.2

发布日期: 2026-10-09

© 北京易捷思达科技发展有限公司

目录

1 版本说明 ---- 1
1.1 版本说明书 ---- 1
2 产品介绍 ---- 2
2.1 什么是安全容器服务 ---- 2
2.2 使用场景 ---- 4
2.3 基本概念 ---- 5
2.4 产品获取 ---- 8
2.5 权限说明 ---- 9
2.6 使用限制 ---- 13
2.7 与其他服务的关系 ---- 15
3 快速入门 ---- 16
3.1 操作指引 ---- 16
3.2 前置条件准备 ---- 18
3.3 创建命名空间 ---- 20
3.4 创建工作负载 ---- 21
3.5 创建Ingress（可选） ---- 32
3.6 使用Yaml创建资源（可选） ---- 33
4 用户指南 ---- 35
4.1 集群管理 ---- 35

版权所有© 北京易捷思达科技发展有限公司

4.2 节点管理 ---- 36
4.3 命名空间 ---- 38
4.4 存储管理 ---- 39
4.5 业务概览 ---- 41
4.6 工作负载 ---- 42
4.7 持久卷声明 ---- 49
4.8 配置中心 ---- 51
4.9 网络管理 ---- 54
4.10 自定义资源管理 ---- 58
5 最佳实践 ---- 60
5.1 配置SR-IOV网络 ---- 60
5.1.1 前置条件准备 ---- 60
5.1.2 使用Yaml配置SR-IOV网络 ---- 69
5.1.3 runc容器和InfiniBand卡IB模式场景 ---- 76
5.1.4 runc容器和InfiniBand卡ETH模式场景 ---- 79
5.1.5 rune容器和InfiniBand卡IB模式场景 ---- 82
5.1.6 rune容器和InfiniBand卡ETH模式场景 ---- 85
5.2 创建GPU资源的容器实例 ---- 88
5.3 AI开发环境容器镜像的制作方法 ---- 94
6 常见问题 ---- 96
6.1 如何理解安全容器网络方案 ---- 96
6.2 容器状态为未知错误，如何排查解决 ---- 102

版权所有© 北京易捷思达科技发展有限公司

6.3 容器状态为失败，如何排查解决 ---- 103
7 部署指南 ---- 104
7.1 Kube-OVN部署指南 ---- 104
8 升级指南 ---- 115
8.1 示例 ---- 115
9 运维指南 ---- 116
9.1 运维指南 ---- 116
10 API参考 ---- 118
10.1 API文档模板 ---- 118

版权所有© 北京易捷思达科技发展有限公司

1 版本说明

1.1 版本说明书

版本信息

产品名称	产品版本	发布日期
安全容器服务	V7.0.2	2026-09-30

更新说明

优化功能

- 节点管理模块新增GPU，NPU，HCU异构算力设备信息展示及配置管理能力
- 工作负载-普通容器新增GPU，NPU，HCU异构算力资源选择
- 工作负载-容器组支持使用的异构算力资源展示
- 移除工作负载-安全容器使用GPU能力
- 异构算力配额管理

依赖说明

> - 平台版本至少为v7.0.2。
> - SDN网络服务版本至少为v7.0.2。

版权所有© 北京易捷思达科技发展有限公司

2 产品介绍

2.1 什么是安全容器服务

安全容器服务基于成熟、轻量的安全容器运行时和SDN网络服务，提供卓越的不可信应用隔离、故障隔离、性能隔离以及多租户应用网络隔离等能力，以便用户轻松高效地在云端运行安全容器化应用。

产品优势

- **安全且故障隔离**

  基于安全容器运行时，提供超强的不可信应用隔离、故障隔离等能力。

- **云资源网络互通**

  安全容器与计算、存储、网络等资源内网互通，以便容器可以分配到虚拟网卡、公网IP和负载均衡等资源，同时也方便传统云主机应用与容器应用之间网络互通。

- **标准适配**

  在网络、日志、监控、存储等方面有着和普通容器一样的用户体验，并具备极速启动和优秀的兼容性、稳定性等特点。

- **网络隔离**

  基于SDN网络服务，在安全容器运行时上增加多租户应用网络隔离能力。

- **统一权限管理**

  为防止资源误操作，将资源的操作能力和云平台的授权管理服务结合，一体化实现。

- **统一配额管理**

  为防止资源滥用，将资源的配额管理能力和云平台的配额管理结合，一体化管理。

主要功能

版权所有© 北京易捷思达科技发展有限公司

- **配额管理**

  为防止资源滥用，云平台支持设置安全容器相关资源的配额，对各项目的可用资源数量和容量做出限制，配额项包括CPU、内存、存储容量、GPU等。

- **容器负载**

  支持对部署容器实例的全生命周期管理，包括启动/停止、重新部署、配置更新、历史版本回滚、终端操作、查看监控与日志、删除等操作。

- **弹性伸缩**

  基于HPA（Horizontal Pod Autoscaler）能力，根据容器当前使用的CPU与内存压力自动扩缩容。

- **滚动升级**

  当通过控制器部署多副本的工作负载时，支持设置自定义滚动更新策略。

- **负载均衡**

  当用户服务是通过控制器部署时，使用负载均衡可将传入流量分配到部署中的各个容器实例，当部署发生变化时，云平台会自动从负载均衡器中添加和删除实例。

- **GPU调度**

  提供NVIDIA GPU设备的发现与管理能力，云平台在创建容器时将依据指定GPU使用需求自动调度GPU资源。

- **持久化存储**

  提供数据持久化存储满足容器运行过程中需要保存数据的需求，并支持普通容量型以及高性能型两种存储类型（使用高性能存储类型时需要搭配高性能云存储产品）。在创建工作负载时支持添加多个存储卷，以及为每个存储卷指定存储类型和容量，并支持挂载存储卷到容器的指定路径。

版权所有© 北京易捷思达科技发展有限公司

2.2 使用场景

- **替换传统虚拟机业务**

  传统虚拟机部署应用，虽然安全性较高，但无法享用到镜像和容器带来的技术红利，并且传统虚拟机的损耗开销较大，交付效率分钟级，难以脱离虚拟机镜像，网络自建和交付不统一等难题，安全容器为解决以上痛点，通过精简的虚拟机，启动在秒级内，复用容器管理平台上的多种资源，包括CNI，CSI等通用化网络，存储方案。

- **隔离不可信应用与故障**

  由于在同一节点中，普通容器通常都混部着不同的业务和租户应用，这些容器都共享同一内核。所以，当内核或者运行时出现漏洞时，恶意代码将会逃逸到对宿主机产生不可逆影响，甚至会导致系统瘫痪。安全容器服务提供超强的不可信应用隔离、故障隔离、性能隔离以及多租户应用网络隔离等能力，保障用户轻松高效地在云端运行安全容器化应用。

- **业务应用运行独占操作系统内核**

  安全容器服务提供内核级的进程隔离机制，天然满足容器独占内核的需求。

版权所有© 北京易捷思达科技发展有限公司

2.3 基本概念

集群（Cluster）

一个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、存储、网络等基础资源。

节点（Node）

安全容器集群中的节点包括Master节点和Worker节点两种类型，每一个节点对应一个云主机。Master节点是安全容器集群的管理者，运行着一些用于保证集群正常工作的组件，如 kube-apiserver、kube-scheduler等。Worker节点是 安全容器集群中承担工作负载的节点，承担实际的 Pod 调度以及与管理节点的通信等。一个Worker节点上运行的组件包括containerd运行时组件、kubelet、Kube-Proxy等。

命名空间（Namespace）

在同一个集群内可以创建不同的命名空间，不同命名空间中的数据彼此隔离，使它们既可以共享同一个集群的服务，也能够互不干扰，为集群提供资源逻辑隔离作用。

容器组（Pod）

容器组即Pod，是安全容器服务部署应用或服务的最小的基本单位。一个容器组封装多个容器（也可以只有一个容器）、存储资源、网络资源以及管理控制容器运行方式的策略选项。

工作负载

工作负载是安全容器服务对一组Pod的抽象模型，用于描述业务的运行载体，包括部署（Deployment）、有状态副本集（StatefulSet）、守护进程集（DaemonSet）、任务（Job）、定时任务（CronJob）。

- 部署：即Kubernetes中的“Deployment”，部署支持弹性伸缩与滚动升级，适用于容器组完全独立、功能相同的场景，如nginx。
- 有状态副本集：即Kubernetes中的“StatefulSet”，有状态副本集支持容器组有序部署和删除，支持持久化存储，适用于实例间存在互访的场景，如ETCD等。
- 守护进程集：即Kubernetes中的“DaemonSet”，守护进程集确保全部（或者某些）节点都运行一个容器组，支持容器组动态添加到新节点，适用于容器组在每个节点上都需要运行的场景，如fluentd、Prometheus

版权所有© 北京易捷思达科技发展有限公司

Node Exporter等。

- 任务：即Kubernetes中的“Job”，任务是一次性运行的短任务，部署完成后即刻执行。
- 定时任务：即Kubernetes中的“CronJob”，定时任务是按照指定时间周期运行的任务。

安全工作负载

安全工作负载拥有独立的操作系统内核以及安全隔离的虚拟化层。通过安全工作负载，不同容器之间的内核、计算和网络资源均相互隔离，保护Pod的资源和数据不被其他Pod抢占和窃取。

服务（Service)

由于每个容器组都有自己的IP地址，并且可能随时被删除重建，如果这个容器组要为其它容器组提供服务，则如何找出并跟踪要连接的IP地址会非常麻烦。安全容器服务针对这个问题给出的方案是服务（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。使用安全容器服务，您无需修改应用程序即可使用不熟悉的服务发现机制。 安全容器服务为Pods提供自己的IP地址和一组Pod的单个DNS名称，并且可以在它们之间进行负载平衡。

路由（Ingress）

Ingress是一组将集群内服务暴露给集群外服务的路由规则集合。一个ingress对象能够配置具备为服务提供外部可访问的URL、负载均衡流量、卸载 SSL/TLS，以及提供基于名称的虚拟主机等能力。

持久化存储

- **持久卷（PV）**

  持久卷描述的是持久化存储卷，主要定义的是一个持久化存储在宿主机上的目录，独立于容器组生命周期。具体到本平台，一个持久卷对应一个云硬盘。

- **持久卷声明（PVC）**

  持久卷是存储资源，而持久卷声明（PVC) 是对持久卷的请求。持久卷声明跟容器组类似：容器组消费节点资源，而持久卷声明消费持久卷资源；容器组能够请求CPU和内存资源，而持久卷声明请求特定大小和访问模式的持久卷。

- **存储类（StorageClass）**

版权所有© 北京易捷思达科技发展有限公司

存储类可以实现动态供应持久卷，即能够按照用户的需要，自动创建其所需的存储。

配置（ConfigMap）

ConfigMap用于将非机密性的数据保存到键值对中。使用时，容器组可以将其用作环境变量、命令行参数或者存储卷中的配置文件。ConfigMap将环境配置信息和容器镜像解耦，便于应用配置的修改。

密钥（Secret）

密钥（Secret）是一种包含认证信息、密钥等敏感信息的资源类型，可以用作工作负载的环境变量、加密配置文件。将数据放在密钥对象中，可以更好地控制它的用途，并降低意外暴露的风险。

标签（Label）

标签是一对 key/value，被关联到对象上，比如节点、容器组。通过标签可以方便地标识及筛选对象。

版权所有© 北京易捷思达科技发展有限公司

2.4 产品获取

前提条件

在执行下述产品获取操作步骤前，请确保以下条件均已满足：

- 已成功获取并安装“计算服务”、“块存储”、“SDN网络服务”和“容器镜像服务”云产品，如果要使用LB service功能，还需要安装“独享型负载均衡服务”。获取并安装云产品的具体操作说明，请参考“产品与服务管理”帮助中的相关内容。
- 如需获取正式版云产品，请提前将已获取的许可文件准备就绪。

操作步骤

1. 获取并安装“安全容器服务”云产品。

   在顶部导航栏中，依次选择[产品与服务]-[产品与服务管理]-[云产品]，进入“云产品”页面获取并安装“安全容器服务”云产品。具体的操作说明，请参考“产品与服务管理”帮助中“云产品”的相关内容。

2. 访问安全容器服务。

   在顶部导航栏中，依次选择[产品与服务]-[安全容器服务]-[任意子菜单]，即可访问该服务的各项功能。

版权所有© 北京易捷思达科技发展有限公司

2.5 权限说明

本章节主要用于说明安全容器服务各功能的用户权限范围。其中，√ 代表该类用户可对云平台内所有项目的操作对象执行此功能，**XX项目** 代表该类用户仅支持对XX项目内的操作对象执行此功能，未标注代表该类用户无权限执行此功能。

功能		云管理员	部门管理员/项目管理员	普通用户
集群管理	信息展示	√		
节点管理	信息展示	√		
	开始/停止调度			
	标签管理			
	污点管理			
命名空间	信息展示	√	仅已加入项目	
	创建命名空间			
	删除			
存储管理	信息展示	√	仅已加入项目	
	查看存储类Yaml			
	查看持久卷Yaml			
	删除持久卷			
工作负载	信息展示	√	仅已加入项目	仅已加入项目
	创建部署			
	创建有状态副本集			
	创建守护进程集			
	创建任务			

版权所有© 北京易捷思达科技发展有限公司

功能		云管理员	部门管理员/项目管理员	普通用户
	创建定时任务			
	容器配置			
	手动伸缩			
	版本回滚			
	升级策略			
	伸缩策略			
	调度策略			
	网络设置			
	标签设置			
	编辑Yaml			
	启动/停止			
	重新部署			
	删除			
	运行/停止定时任务			
	查看容器组Yaml			
	容器日志			
	容器终端			
	删除容器组			
持久卷声明	信息展示	√	仅已加入项目	仅已加入项目
	创建持久卷声明			
	编辑Yaml			
	删除			

版权所有© 北京易捷思达科技发展有限公司

功能		云管理员	部门管理员/项目管理员	普通用户
配置中心	信息展示	√	仅已加入项目	仅已加入项目
	创建配置			
	更新配置			
	编辑配置Yaml			
	删除配置			
	创建密钥			
	更新密钥			
	编辑密钥Yaml			
	删除密钥			
网络管理	信息展示	√	仅已加入项目	仅已加入项目
	创建服务			
	更新服务			
	编辑服务Yaml			
	删除服务			
	创建Ingress			
	更新Ingress			
	编辑Ingress Yaml			
	删除Ingress			
自定义资源管理	信息展示	√	仅已加入项目	仅已加入项目
	使用Yaml导入自定义资源描述			

版权所有© 北京易捷思达科技发展有限公司

	功能	云管理员	部门管理员/项目管理员	普通用户
	使用Yaml导入自定义资源		仅已加入项目	仅已加入项目
	删除自定义资源		仅已加入项目	仅已加入项目

版权所有© 北京易捷思达科技发展有限公司

2.6 使用限制

GPU相关

- 在Arm架构的云平台中，容器不支持使用GPU。
- 目前云平台支持在安全容器中使用英伟达GPU。

英伟达（NVIDIA）GPU

英伟达（NVIDIA）GPU采用 **580.82.07** 版本的 NVIDIA GPU 驱动，兼容 **CUDA 13.0.1**。该驱动及 CUDA 版本支持的 GPU 设备如下表所示：

类型	型号
英伟达（NVIDIA）GPU设备	Tesla T4
	RTX 8000、RTX 6000、NVIDIA RTX A6000、NVIDIA RTX A5000、NVIDIA RTX A4000、NVIDIA T1000、NVIDIA T600、NVIDIA T400
	HGX H800、HGX H100、HGX A800、HGX A100、HGX-2
	NVIDIA AX800、NVIDIA A800、NVIDIA A100、NVIDIA A40、NVIDIA A30、NVIDIA A16、NVIDIA A10、NVIDIA A2
	NVIDIA H100 PCIe、NVIDIA H100 NVL、NVIDIA H800 PCIe、NVIDIA H800 NVL、L40、L40S、L20、L4、L2
CUDA版本	CUDA 13.0.1

> **说明：** CUDA 13.0 起不再支持 Maxwell、Pascal 和 Volta 架构的离线编译和相关库支持。因此，Tesla M 系列、Tesla P 系列以及 Tesla V100 不再属于 CUDA 13.0.1 的支持范围。

网络相关

版权所有© 北京易捷思达科技发展有限公司

- 安全运行时容器 Pod 不支持 `hostNetwork` 。
- 不能同时为工作负载设置 EIP 和 SNAT。
- EIP 只能分配给一个 Pod，设置了 EIP 的工作负载，滚动策略需设置为先删除旧 Pod 再创建新 Pod。

版权所有© 北京易捷思达科技发展有限公司

2.7 与其他服务的关系

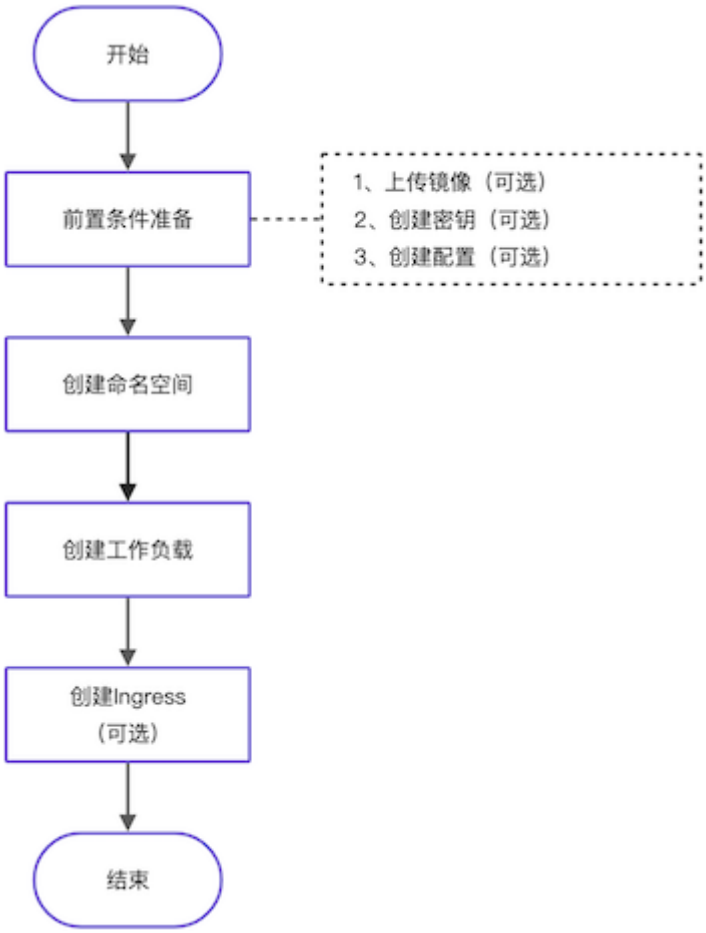
服务	关系说明
容器镜像服务	创建工作负载时需要为容器指定所使用的容器镜像。
块存储	块存储为容器集群提供持久化存储资源。
SDN网络服务	为安全容器服务提供网络、公网IP、负载均衡等网络资源及相关服务。

版权所有© 北京易捷思达科技发展有限公司

3 快速入门

3.1 操作指引

安全容器服务云产品的主线使用流程及具体说明如下：



操作流程		描述
前置条件准备	上传镜像（可选）	预先上传工作负载的容器创建时所需要的镜像文件。 请根据客户实际业务需求酌情创建。如已有可用镜像或使用第三方镜像时，可跳过本步骤。

操作流程		描述
	创建密钥（可选）	预先创建工作负载创建时所需要的密钥。 请根据客户实际业务需求酌情创建。如不使用开启密钥认证的第三方镜像，且数据卷、环境变量都不选择“密钥”类型时，可跳过本步骤。
	创建配置（可选）	预先创建工作负载创建时所需要的配置。 请根据客户实际业务需求酌情创建。如数据卷和环境变量都不选择“配置”类型时，可跳过本步骤。
创建命名空间		通过命名空间实现同一集群内不同资源之间的隔离。
创建工作负载		工作负载是对一组Pod的逻辑抽象，用于承载业务运行。
创建Ingress（可选）		通过Ingress为工作负载的服务提供外部访问时所需的路由规则集合。 请根据客户实际业务需求酌情配置。当工作负载已添加服务，且该服务需要配置对外访问的路由规则时，才需执行此操作。

版权所有© 北京易捷思达科技发展有限公司

3.2 前置条件准备

在创建工作负载前，请先完成以下准备工作。

上传镜像（可选）

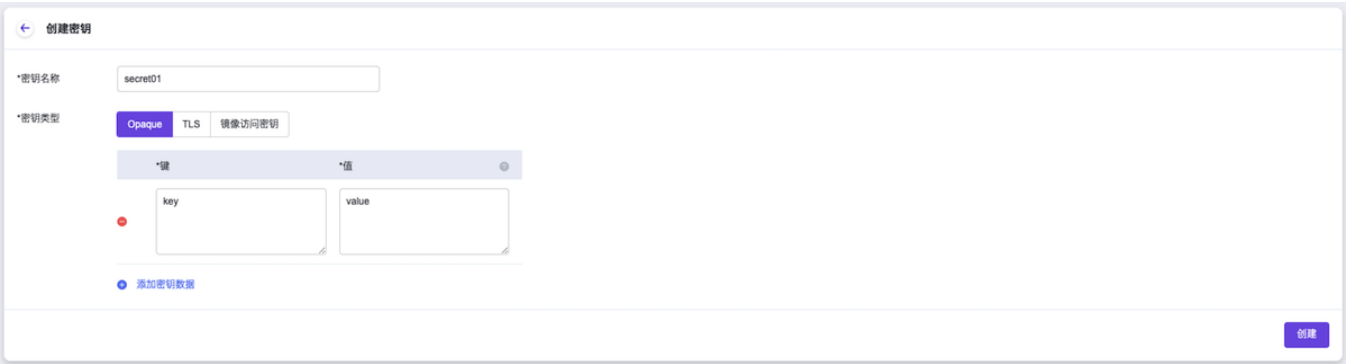
本操作用于预先上传工作负载的容器创建时所需要的镜像文件，请根据客户实际业务需求酌情创建。如已有可用镜像或使用第三方镜像时，可跳过本步骤。

1. 在云平台的顶部导航栏中，依次选择[产品与服务]-[容器镜像服务]-[镜像管理]，进入“镜像管理”页面。
2. 单击 `上传镜像` 或 `Push镜像` ，弹出对应的对话框。
3. 配置参数后，完成操作。各参数的具体说明，请参考“容器镜像服务”帮助中“镜像管理”的相关内容。

创建密钥（可选）

本操作用于预先创建工作负载创建时所需要的密钥，请根据客户实际业务需求酌情创建。如不使用开启密钥认证的第三方镜像，且数据卷、环境变量都不选择“密钥”类型时，可跳过本步骤。

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[安全容器服务]-[配置中心]，进入“配置中心”页面。
2. 在左侧导航栏选择[业务视图]，选择目标命名空间，选择[配置中心]-[密钥]，进入“密钥”页面。
3. 单击 `创建密钥` ，进入“创建密钥”页面。
4. 配置参数后，单击 `创建` 完成操作。



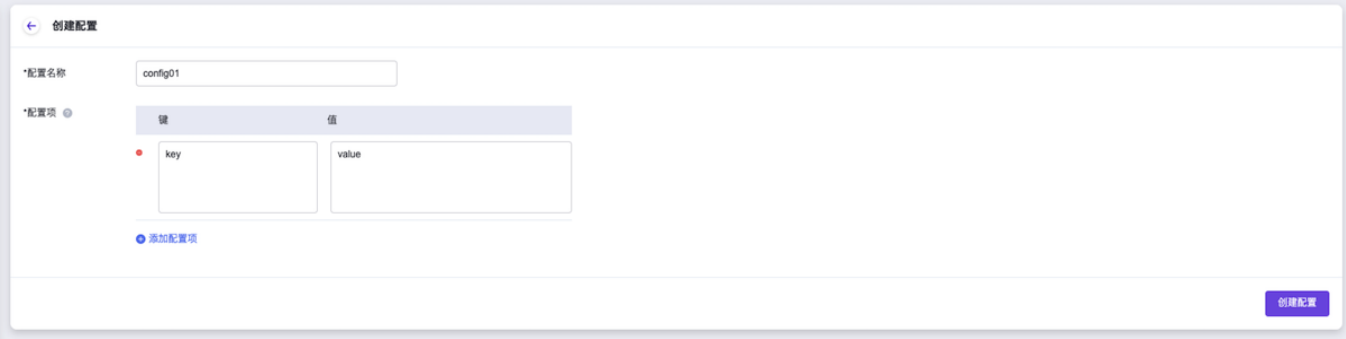
参数	说明

参数	说明
密钥类型	* Opaque：一般密钥类型。 * TLS：存放7层负载均衡服务所需的证书。 * 镜像访问密钥：存放拉取私有仓库镜像所需的认证信息。
密钥数据	* 当密钥类型为Opaque时，单击“添加密钥数据”，输入键、值。 * 当密钥类型为TLS时，上传证书和私钥文件。 * 当密钥类型为镜像访问密钥时，输入镜像仓库地址、用户名、密码和邮箱。

创建配置（可选）

本操作用于预先创建工作负载创建时所需要的配置，请根据客户实际业务需求酌情创建。如数据卷和环境变量都不选择“配置”类型时，可跳过本步骤。

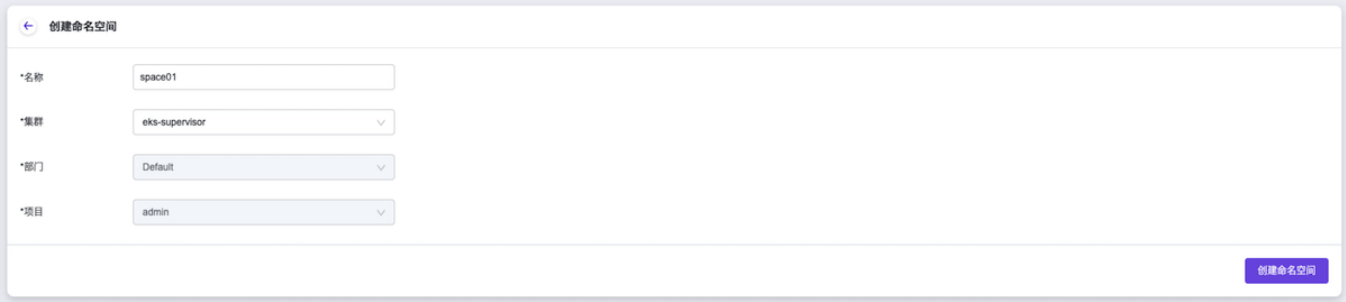
1. 在左侧导航栏选择[业务视图]，选择目标命名空间，选择[配置中心]-[配置]，进入“配置”页面。
2. 单击 `创建配置` ，进入“创建配置”页面。
3. 填写参数后，单击 `创建配置` ，完成操作。



3.3 创建命名空间

通过命名空间实现同一集群内不同项目资源之间的隔离。

1. 在顶部导航栏选择[产品与服务]-[安全容器服务]-[命名空间]，进入“命名空间”管理页面。
2. 单击 `创建命名空间` ，跳转至“创建命名空间”页面。
3. 配置参数，单击 `创建命名空间` 完成操作。



参数	说明
名称	选择命名空间的名称。
集群	选择命名空间所属集群。
部门/项目	用户所在部门和项目，不支持修改。

版权所有© 北京易捷思达科技发展有限公司

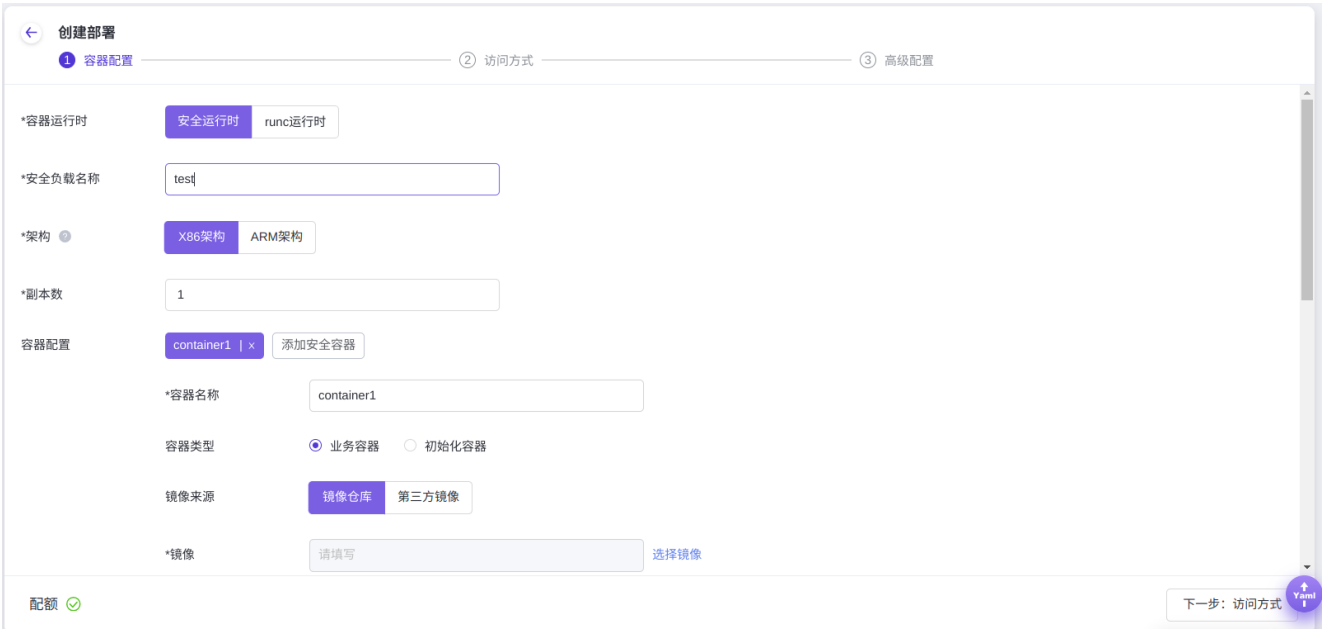
3.4 创建工作负载

工作负载是对一组Pod的逻辑抽象，用于承载业务运行。其类型包括部署（Deployment）、有状态副本集（StatefulSet）、守护进程集（DaemonSet）、任务（Job）、定时任务（CronJob），请根据客户实际业务需求酌情创建。创建方式支持界面创建和Yaml创建，本节将介绍界面创建方式，Yaml创建方式请参见 如何使用Yaml创建资源。

1. 在左侧导航栏选择[业务视图]页签-选择目标命名空间后，依据工作负载类型选择对应子菜单，进入对应页面。

2. 单击 `创建部署/有状态副本集/守护进程集/任务/定时任务` ，进入对应创建页面的“容器配置”页面。

3. 在“容器配置”页面中，配置参数后，单击 `下一步：访问方式` ，进入“访问方式”配置页面。其中，在“容器配置”区域框中，单击 `添加容器` ，可在该容器实例中添加多个容器，但是在容器添加过程中，请先确保已完成当前容器的配置。



参数	说明

参数		说明
容器运行时		该工作负载中安全容器的运行时类型。该参数值可选“安全运行时”或“runc运行时”。 运行安全运行时的工作负载与运行runc运行时的工作负载相比，其进程隔离机制为内核级隔离，安全容器间的计算资源、网络资源具有更为彻底的隔离性。 runc运行时不支持内核隔离，不支持使用GPU。
架构		工作负载CPU架构，可选“X86架构”和“ARM架构”。
副本数		仅当负载类型为“部署”或“有状态副本集”时可配置此参数。表示该工作负载包括的容器组个数。每个容器组都由相同的容器部署而成。设置多个容器组主要用于实现高可靠性，当某个实例故障时，工作负载还能正常运行。
容器配置	容器类型	包括业务容器和初始化容器。业务容器即真正运行业务的容器，初始化容器则运行于业务容器启动期间。若容器组中有多个初始化容器，这些容器会按顺序逐个运行，每个初始化容器必须运行成功，下一个才能够运行，当所有初始化容器运行完成时，集群才会正常运行业务容器。由于一个容器组中的存储卷是共享的，所以初始化容器中产生的数据可以被业务容器使用到。由于初始化容器提供了一种机制来阻塞或延迟业务容器的启动，可以应用于有启动顺序要求的容器组之间。
	镜像来源	包括镜像仓库和第三方镜像两种来源。选择镜像仓库则使用本集群对接的镜像仓库，选择第三方镜像则需要输入第三方镜像地址且保证网络可达。
	密钥认证	仅当镜像来源为“第三方镜像”时可配置。
	密钥	仅当镜像来源为“第三方镜像”且密钥认证为“是”时可配置。
	镜像	若镜像来源为“镜像仓库”，则单击 `选择镜像` ，弹出选择镜像对话框。选择目标镜像，单击 `确定` 完成操作。若镜像来源为“第三方镜像”，则输入格式为ip:port/path/name的镜像地址。

版权所有© 北京易捷思达科技发展有限公司

参数		说明
	镜像版本	若镜像来源为“镜像仓库”，则在下拉框中选择目标版本；若镜像来源为“第三方镜像”，则手动输入目标版本。
	拉取镜像策略	包括“本地不存在时拉取”和“总是拉取”两种策略。
	资源预留	保证容器成功调度到节点的最小资源。 当需要勾选“使用GPU”时，建议此参数值的CPU大于等于1，内存大于等于1024MiB。
	资源限制	容器运行中允许使用的最大资源。
	使用GPU	仅当容器运行时为“安全运行时”时可配置此参数。表示容器是否使用GPU资源。 “守护进程集（DaemonSet）”和“定时任务（CronJob）”类型的工作负载不支持使用GPU。
	环境变量（可选）	容器在启动过程中需要的一些配置信息如启动命令、证书等，这类信息需要在容器组故障重启后仍然存在并重新加载到新容器组中，这类信息可以通过环境变量的形式单独存储。当前支持以下类型： * 普通变量：普通变量不需提前创建，直接输入即可。 * 配置：选择已创建好的配置。 * 密钥：选择已创建好的密钥。 * Pod字段：直接选择具体字段即可。 * 容器资源：直接选择具体资源即可。

版权所有© 北京易捷思达科技发展有限公司

参数		说明
	数据卷（可选）	单击 `添加数据卷` ，弹出“添加数据卷”对话框。配置参数，单击 `确定` 完成操作。参数说明如下： * 类型： - 持久卷声明：仅工作负载类型为部署、任务、定时任务时可选择本类型。给容器挂载持久化存储，数据不会因容器的销毁或节点异常而消失。适用于需持久化存储、高磁盘IO等场景。持久卷声明需要事先创建，相关介绍请参考 创建持久卷声明。 - 存储类：仅工作负载类型为有状态副本集时可选择。不需事先创建持久卷声明，可直接通过指定存储类及所需存储容量创建持久卷，并挂载到指定的容器路径。各参数的具体说明，请参考 创建持久卷声明。 - 临时路径：将容器所在宿主机的临时目录挂载到容器的指定路径。 - 配置：选择已创建好的配置。 - 密钥：选择已创建好的密钥。 * 挂载路径(可选)：所选数据卷挂载至容器的绝对路径。
	健康检查（可选）	健康检查包括存活检查、就绪检查和启动检查功能。存活检查用于检测容器是否正常，如果容器的存活检查失败，集群会对该容器执行重启操作；若容器的存活检查成功则不执行任何操作。就绪检查用于检查用户业务是否就绪，如果容器的就绪检查失败，则不转发流量到当前容器组；若检查成功，则会开放对该容器组的访问。启动检查用于保护慢启动容器有充足时间完成启动，避免死锁状况发生。 * 检查方式： - HTTP/HTTPS方式：适用于提供HTTP/HTTPS服务的容器，集群周期性地对该容器发起HTTP/HTTPS GET请求，如果HTTP/HTTPS 返回状态码小于400，则证明检查成功、容器健康，否则检查失败。例如，方式选择HTTP，路径为/check，端口为80，则集群周期性向容器发起如下请求： `GET http://容器IP:80/check` 。 - TCP方式：适用于提供TCP通信服务的容器，集群周期性地检测端口是否为打开状态，若端口为打开状态，则检查成

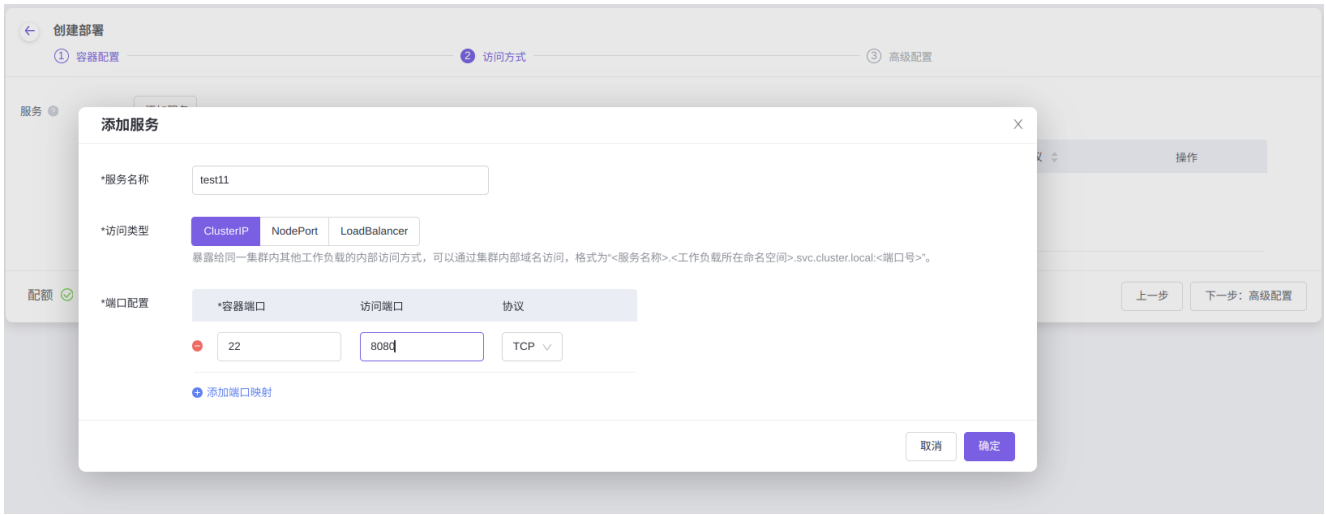
参数		说明
		功、容器健康；若端口为关闭或进程为停止状态，则检查失败。例如：一个提供nginx服务的容器，服务端口为80，则配置TCP检查端口为80，那么集群会周期性检测该容器的80端口打开状态。 - 容器命令方式：该方式要求用户指定一个容器内的可执行命令，集群会周期性地在容器内执行该命令，若进程退出状态码为 0则检查成功、容器健康，否则检查失败。 * 公共参数： - 首次检查延时：容器启动后第一次进行健康检查的延迟时间，这段时间为预留给业务程序正常启动。例如，设置为10，表明容器启动后10秒才开始健康检查。 - 检查间隔：执行健康检查的时间间隔。例如，设置为30，则每间隔30秒执行一次健康检查。 - 超时时间：检查超时后的等待时间。例如，设置为10，表明执行健康检查的超时等待时间为10秒，如果超过这个时间，本次健康检查就被视为失败。 - 健康认定（）次成功：假设本参数设置为N，健康检查失败后，至少连续成功N次会认为容器健康。 - 不健康认定（）次失败：假设本参数设置为X，健康检查失败后，集群将继续尝试X次健康检查，若仍不符合健康条件，则放弃该容器。对于存活检查，放弃意味着重启容器；对于就绪检查，放弃意味着容器组将被标记为未就绪。
	安全设置（可选）	* 非root用户运行：要求容器组具有非零runAsUser值，或在镜像中定义了USER环境变量。 * 只读root文件系统：是否必须使用一个只读的root文件系统。 * runAsUser：用户ID。容器中的进程都以该用户ID运行。 * runAsGroup：Group ID。容器中的进程都以该Group ID运行。

版权所有© 北京易捷思达科技发展有限公司

参数		说明
	命令（可选）	* 启动命令：容器启动时运行的第一条命令，将覆盖镜像中的Entrypoint指令。 * 启动命令参数：覆盖镜像中的CMD执行，如已设置了运行命令，该条指令将被附加到运行命令的参数中。 * 启动后执行命令：该命令在创建容器之后立即执行。 * 停止前执行命令：这个命令在停止容器前执行，是否立即调用此命令取决于 API 的请求或者管理事件。
	日志采集（可选）	采集应用的运行日志，实现平台内应用与组件日志的全量采集与查询。 * 日志源：选择需采集日志的资源，支持容器标准日志（默认）和应用日志两种。 * 日志文件路径：当“日志源”选择“应用日志”时需配置。日志文件存放路径，需注意，该路径要求已挂载数据卷。

4. 在“访问方式”页面中，配置参数后，单击 `下一步：高级配置` ，进入“高级配置”配置页面。

   当该工作负载需要提供对外访问的服务时，请单击 `添加服务` ，在弹出的对话框中配置参数后，单击 `确定` ，完成服务添加。否则，可直接跳过本步骤。



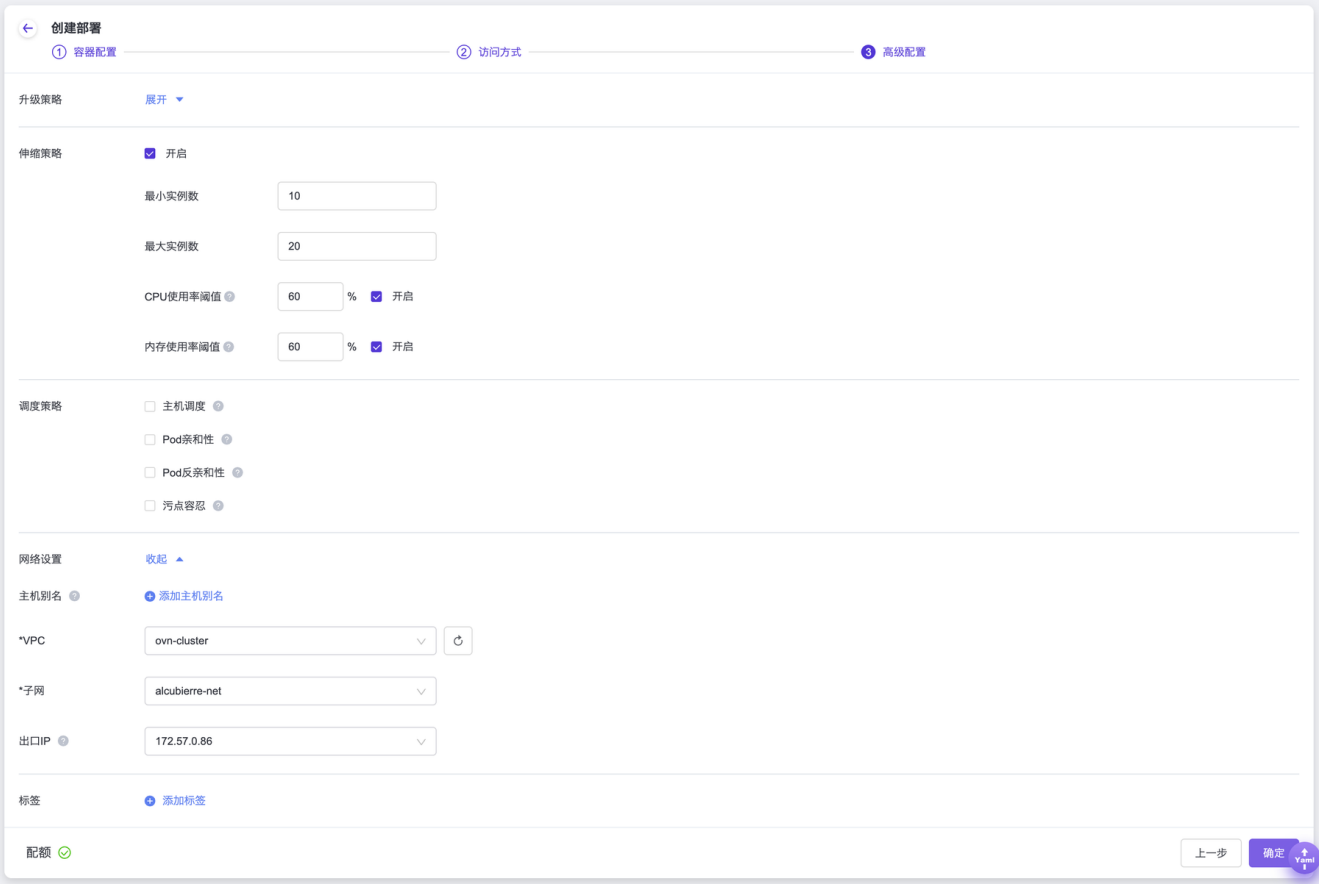
参数	说明

参数		说明
服务（可选）	服务名称	该工作负载中用于提供外部访问的服务的名称。
	访问类型	* ClusterIP：适用于集群内部访问场景，集群为服务分配一个固定的集群内虚拟IP，集群内其它pod可以通过集群内部域名访问，格式为“<服务名称>.<工作负载所在命名空间>.svc.cluster.local:<端口号>”。集群外无效。 * NodePort：适用于集群外部访问场景，集群除了会给服务分配一个内部的虚拟IP，还会在每个节点上为服务分配静态端口号，集群外部可通过集群任一节点IP和静态端口号访问服务。 * LoadBalancer：负载均衡器会随服务一起创建。若需从集群外部访问，需在独享型负载均衡页面绑定公网IP，并使用<公网IP>:<端口号> 进行访问。当删除服务时，负载均衡器也将自动删除。

版权所有© 北京易捷思达科技发展有限公司

参数		说明
	端口配置	当访问方式为ClusterIP或NodePort时： * 容器端口：容器镜像中工作负载实际监听的端口。 * 访问端口：容器端口映射到节点IP上的端口。当访问方式为“NodePort”时，支持随机生成。 * 协议：包括TCP、UDP，根据业务类型选择。 当访问方式为LoadBalancer时： * 可用区：负载均衡器所在可用区。 * 规格：负载均衡器的规格。 * CPU架构：负载均衡器的CPU架构，包括X86计算和ARM计算。 * 磁盘类型：负载均衡器的磁盘类型。 * 拓扑模式：负载均衡器的拓扑模式，包括主备模式和多活模式。 * VPC：负载均衡器关联的VPC。 * 子网：负载均衡器关联的子网，只可选择非容器网络类型的子网。 * VIP地址：负载均衡器的VIP地址，支持自动分配和手动指定。 * 协议：服务端口协议类型，只支持TCP。 * 容器端口：容器镜像中工作负载实际监听的端口。 * 访问端口：容器端口映射到节点IP上的端口。

5. 在“高级配置”页面中，配置参数后，单击 `确定` ，完成操作。



参数		说明
升级策略	工作负载类型为“部署”	* 先启动新Pod，再停止旧Pod/先停止旧Pod，再启动新Pod：可定义每次启动或停止Pod的数量。例如选择先启动新Pod，再停止旧Pod，批量大小设置为1，则每次先启动1个新的Pod，新的Pod成功后停止1个旧Pod，以此类推。 * 停止所有Pod，再启动新Pod：先停止所有老版本容器组，再启动新版本容器组，升级过程中业务会中断。 * 自定义：“最大超量”表示更新过程中容器组数量可以超过期望副本的数量或百分比。“最多不可用数”表示升级过程中允许的最多不可用容器组数量，如果等于期望副本数量有业务中断风险（最小存活容器组数量=期望副本数量-最多不可用数）。

参数		说明
	工作负载类型为“有状态副本集”或“守护进程集”	* 滚动：滚动升级将逐步用新版本的实例替换旧版本的实例，升级的过程中，业务流量会同时负载均衡分布到新老的实例上，因此业务不会中断。其中，“最多不可用数”表示升级过程中允许的最多不可用容器组数量，如果等于期望副本数量则有业务中断风险（最小存活容器组数量=期望副本数量-最多不可用数）。 * 手动删除时更新：集群不会自动更新工作负载中的容器组，需手动删除容器组以使集群创建新的容器组。
伸缩策略		仅当工作负载类型为“部署”时可配置。当达到设置的条件后自动扩展或收缩容器组数量。 * 最小实例数：期望容器组数量的最小值。 * 最大实例数：期望容器组数量的最大值。 * CPU使用率阈值：所有容器组平均cpu使用率超过阈值自动扩展，n-1（n为容器组总数）个容器组平均内存使用率低于阈值自动收缩。需勾选“开启”后才能输入阈值。 * 内存使用率阈值：所有容器组平均内存使用率超过阈值自动扩展，n-1（n为容器组总数）个容器组平均内存使用率低于阈值自动收缩。需勾选“开启”后才能输入阈值。
调度策略（可选）	主机调度	* 指定主机：可选择集群内任一节点，该工作负载内的容器将被调度到所选节点上。 * 自定义规则：包括必须满足条件和尽量满足条件。必须满足条件是硬性要求，必须满足才能成功调度，支持添加多条规则，多条规则间是“且”的关系，即需要满足所有规则才可以调度；尽量满足条件表示集群会尽量将容器调度到符合规则的主机上，支持添加多条规则，多条规则间是“或”的关系，不满足规则的主机也会进行调度，根据规则的权重值，权重值越高越会被优先调度。

参数		说明
	Pod亲和性/Pod反亲和性	Pod亲和性决定哪些工作负载的Pod部署在同一个拓扑域，可根据业务需求进行工作负载的就近部署，容器间通信就近路由，减少网络消耗。Pod反亲和性决定工作负载的Pod不和哪些工作负载的Pod部署在同一个拓扑域，互相干扰的工作负载反亲和部署，避免干扰，减少宕机影响。拓扑域是由一个或多个节点组成的，这些节点在所指定的属性上具有相同的值，例如拓扑域为kubernetes.io/hostname，则具有相同hostname的节点成为一个拓扑域（即同一节点）。必须满足条件是硬性要求，支持添加多条规则，多条规则间是“且”的关系，即需要满足所有规则才可以调度；尽量满足条件表示集群会尽量将容器调度到符合规则的主机上，多条规则间是“或”的关系，不满足规则的主机也会进行调度，根据规则的权重值，权重值越高越会被优先调度。
	污点容忍	调度工作负载时能够容忍具有指定污点的节点。支持添加多条污点规则，多条规则间是“或”的关系，即满足任一规则即可调度。
网络设置		* 主机别名：添加主机别名后即可通过域名访问对应IP地址的主机。 * VPC：该工作负载的私有网络名称。支持选择系统默认生成的ovn-cluster，也支持自定义VPC和各网络的子网，具体操作请参考 如何自定义VPC网络。 * 子网：该工作负载的子网名称。 * 出口IP：该工作负载提供对外访问服务时，所使用的公网IP地址。
标签（可选）		通过标签可以方便地标识及筛选对象。

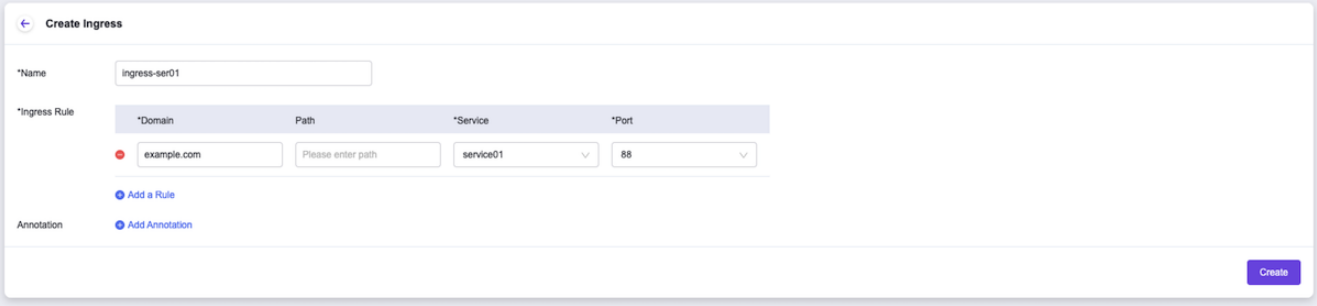
3.5 创建Ingress（可选）

通过Ingress可以为工作负载的服务提供外部访问时所需的路由规则集合，请根据客户实际业务需求酌情创建。当工作负载已添加服务，且该服务需要配置对外访问的路由规则时，才需执行此操作。

1. 在左侧导航栏选择[业务视图]，选择目标命名空间，选择[网络管理]-[Ingresses]，进入“Ingress”页面。

2. 单击 `创建Ingress` ，进入“创建Ingress”页面。

3. 配置参数后，单击 `创建` 完成操作。



参数	说明
Ingress规则	是一种HTTP方式的路由转发机制。例如域名填写为example.com，路径填写为/path，服务选择已创建的名称为“app”的服务，则外部可通过 `http://example.com/path` 访问名称为“app”的服务。
注解	Ingress经常使用注解（annotations）来配置一些选项，具体取决于Ingress控制器。

版权所有© 北京易捷思达科技发展有限公司

3.6 使用Yaml创建资源（可选）

本章节主要描述如何使用Yaml创建安全容器服务资源。工作负载、持久卷声明和配置、密钥、服务、Ingress、自定义资源描述等资源不仅可以通过云平台页面进行创建，还可以通过Yaml文件进行创建。用户可以根据实际的业务场景，灵活选择适合的资源创建方案。

1. 在顶部导航栏选择[产品与服务]-[安全容器服务]任意一个子菜单，进入“安全容器服务”页面。
2. 在左侧导航栏选择[业务视图]页签-选择目标命名空间后，选择任意一个子菜单，进入对应页面。
3. 单击页面右下角的“Yaml”图标，进入“导入Yaml”页面。



4. 直接粘贴Yaml文件内容，或单击编辑区域右上角的“导入”图标，选择本地存储的Yaml文件。

> 说明：
> 请关注调试结果。该调试主要针对格式校验，若有错误可点击错误信息，跳至目标行进行修改。

为了更清晰地说明，下面以创建名为"demo"的"部署"类型的工作负载为例，介绍如何使用Yaml创建资源。以下是一个示例Yaml内容:

版权所有© 北京易捷思达科技发展有限公司

```
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: demo
  name: demo
spec:
  replicas: 1
  selector:
    matchLabels:
      app: demo
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: demo
    spec:
      containers:
      - image: nginx:latest
        name: nginx
        resources: {}
status: {}
```

> 说明：
> 如果是一云多芯环境，请在yaml文件中根据节点架构添加如下内容： nodeSelector: kubernetes.io/arch: amd64 或 nodeSelector: kubernetes.io/arch: arm64

5. 待调试通过后，单击 `导入` ，完成操作。

4 用户指南

4.1 集群管理

本章节主要介绍在“集群管理”页面中，针对集群的运维管理操作。“集群管理”页面进入路径如下：

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[安全容器服务]-[集群管理]，进入“集群管理”页面。

查看集群详情

可以查看集群内资源使用情况及当前运行情况和集群事件等。

1. 进入“集群管理”页面。
2. 在集群列表中单击目标集群名称链接，进入集群详情页面。
3. 选择[概览]页签，查看集群概览信息；选择[集群事件]页签，查看集群中对象的错误或警示信息。

版权所有© 北京易捷思达科技发展有限公司

4.2 节点管理

本章节主要介绍在“节点管理”页面中，针对节点的运维管理操作。“节点管理”页面进入路径如下：

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[安全容器服务]-[节点管理]，进入“节点管理”页面。

> 说明：
> 云平台控制平面的节点，不支持开始调度、停止调度、标签管理和污点管理操作。

查看节点详情

1. 进入“节点管理”页面。
2. 单击节点名称链接，进入节点详情页面，查看详细信息。

开始调度/停止调度

开始调度后，新创建的容器组可以调度到节点上，停止调度后不可以。

1. 进入“节点管理”页面。
2. 单击目标节点操作栏的 `开始调度` 或 `停止调度` ，弹出“开始调度”或“停止调度”提示框。
3. 单击 `确定` 完成操作。

标签管理

本功能用于增加或删除节点标签，系统标签不支持编辑和删除。

1. 进入“节点管理”页面。
2. 单击目标节点操作栏的 `更多` - `标签管理` ，弹出“标签管理”对话框。
3. 添加或移除标签。
4. 单击 `确定` 完成操作。

污点管理

版权所有© 北京易捷思达科技发展有限公司

节点设置上污点之后就和容器组之间存在了相斥的关系，可以让节点拒绝容器组的调度，甚至将节点上已经存在的容器组驱逐出去。例如，当已知某个节点资源不足且无法为其扩容时，可以给节点打上污点标记，使容器组不再调度到该节点或者驱逐节点上的容器组，隔离该节点。

1. 进入“节点管理”页面。
2. 单击目标节点操作栏的 `更多` - `污点管理` ，弹出“污点管理”对话框。
3. 添加或删除污点。
4. 单击 `确定` 完成操作。

参数		说明
调度策略	不允许调度（NoSchedule）	新创建的容器组不会调度到该节点。
	尽量不调度（PreferNoSchedule）	新创建的容器组尽量不调度到该节点。
	不允许并驱逐已有容器组（NoExecute）	新创建的容器组不会调度到该节点，已经运行在节点上的容器组也会被驱逐。

版权所有© 北京易捷思达科技发展有限公司

4.3 命名空间

本章节主要介绍在“命名空间”页面中，针对命名空间的运维管理操作。“命名空间”页面进入路径如下：

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[安全容器服务]-[命名空间]，进入“命名空间”页面。

创建命名空间

1. 进入“命名空间”页面。
2. 单击 `创建命名空间` ，进入“创建命名空间”页面。
3. 配置参数，单击 `创建命名空间` 完成操作。

参数	说明
名称	选择命名空间的名称。
集群	选择命名空间所属集群。
部门/项目	用户所在部门和项目，不支持修改。

删除命名空间

> 警告：
> 删除命名空间会同时删除该命名空间下的所有资源。

1. 进入“命名空间”页面。
2. 选择目标命名空间，单击 `删除` ，弹出“删除命名空间”提示框。
3. 单击 `删除` 完成操作。

版权所有© 北京易捷思达科技发展有限公司

4.4 存储管理

存储类

存储类可以实现动态供应持久卷，即能够按照用户的需要，自动创建其所需的存储。本章节主要介绍在“存储类”页面中，针对存储的运维管理操作。“存储类”页面的进入路径如下：

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[安全容器服务]-[存储管理]，进入“存储管理”页面。
2. 在左侧导航栏选择[存储管理]-[存储类]，进入“存储类”页面。

查看Yaml

1. 进入“存储类”页面。
2. 单击目标存储类操作栏的 `查看Yaml` ，弹出“查看Yaml”提示框。
3. 查看信息后，单击 `关闭` 完成操作。

持久卷

持久卷描述的是持久化存储卷，主要定义的是一个持久化存储在宿主机上的目录，独立于容器组生命周期。具体到本平台，一个持久卷对应一个云硬盘。本章节主要介绍在“持久卷”页面中，针对存储的运维管理操作。“持久卷”页面的进入路径如下：

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[安全容器服务]-[存储管理]，进入“存储管理”页面。
2. 在左侧导航栏选择[存储管理]-[持久卷]，进入“持久卷”页面。

查看Yaml

1. 进入“持久卷”页面。
2. 单击目标持久卷操作栏的 `查看Yaml` ，弹出“查看Yaml”提示框。
3. 查看信息后，单击 `关闭` 完成操作。

删除持久卷

> 说明：
> 已被绑定至持久卷声明的持久卷无法删除。

版权所有© 北京易捷思达科技发展有限公司

1. 进入“持久卷”页面。
2. 单击目标持久卷操作栏的 `删除` ，弹出“删除持久卷”提示框。
3. 单击 `删除` 完成操作。

批量删除持久卷

> 说明：
> 已被绑定至持久卷声明的持久卷无法删除。

1. 进入“持久卷”页面。
2. 勾选目标持久卷后，单击 `删除` ，弹出“删除持久卷”提示框。
3. 单击 `删除` 完成操作。

版权所有© 北京易捷思达科技发展有限公司

4.5 业务概览

本功能用于查看各命名空间下的业务运行状况，如容器组状态、容器配额使用情况等。

1. 在顶部导航栏单击[产品与服务]-[安全容器服务]进入“安全容器服务”页面。
2. 在左侧导航栏选择[业务视图]页签，进入业务视图页面。
3. 在左侧导航栏选择目标命名空间，切换至目标命名空间视图。
4. 在左侧导航栏选择[概览]，进入概览页面即可查看信息。

版权所有© 北京易捷思达科技发展有限公司

4.6 工作负载

本章节主要介绍在相应类型工作负载页面中，针对工作负载的运维管理操作。相应类型工作负载页面的进入路径如下：

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[安全容器服务]-[工作负载]，进入“工作负载”页面。
2. 在左侧导航栏选择目标命名空间后，选择对应子菜单，进入对应页面。

创建部署/有状态副本集/守护进程集

部署即kubernetes中的Deployment控制器，一个“部署”可以包含一个或多个容器组副本，这些容器组是无状态的（即完全相同、相互独立、可被替换），系统会自动为Deployment的多个Pod副本分发请求。通过定义期望的副本数、容器属性等，“部署”会保证实际状态与所需状态一致，即使发生意外情况也可以将容器组恢复到期望状态。通过“部署”可以实现上线部署、滚动升级（不停止旧服务的状态下升级）、回滚应用（将应用回滚到之前的版本）、平滑扩缩容功能。

有状态副本集即kubernetes中的StatefulSet控制器，一个“有状态副本集”可以包含一个或多个容器组副本，这些容器组是有状态的（运行过程中会保存数据或状态），支持有序部署和删除，支持持久化存储，适用于容器组间存在主从关系、主备关系、互相访问等关系的场景。

守护进程集即kubernetes中的DaemonSet控制器。守护进程集确保全部（或者某些）节点都运行一个容器组，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，例如在每个节点上运行日志收集程序、节点监视程序等。

1. 进入相应类型工作负载的页面。
2. 单击 `创建部署/有状态副本集/守护进程集` ，进入对应页面。
3. 配置参数，参数说明请参考 创建工作负载 。
4. 单击 `确认` 完成操作。

创建任务

任务会创建一个或者多个容器组，并将持续重试容器组的执行，直到指定数量的容器组成功终止。 随着容器组成功结束，任务跟踪记录成功完成的容器组个数。当数量达到指定的成功个数阈值时，任务（即 Job）结束。

1. 进入“任务”页面。

版权所有© 北京易捷思达科技发展有限公司

2. 单击 `创建任务` ，进入“创建任务”的“基础配置”页面。

3. 填写基础配置参数。

参数	说明
目标完成次数	当成功完成的容器组达到该值时认为任务完成。
并行实例数	每次创建的容器组数量。
失败重试次数	失败容器组的最大重试次数，超过这个次数不会继续重试。
超时时间	任务运行的超时时间。如果任务运行的时间超过了设定的时间，此任务将自动停止运行所有容器组。
重启策略	容器组内容器的重启策略，包括“不重启”和“失败时重启”。
调度策略	容器组内容器的调度策略。即调度工作负载时，是否能够容忍具有污点的节点。

4. 单击 `下一步：容器配置` ，进入“创建任务”的“容器配置”页面。

5. 填写容器配置参数，参数说明请参考 创建工作负载。

6. 单击 `创建` 完成操作。

创建定时任务

定时任务即Kubernetes中的CronJob，是基于时间的“任务”，在指定的时间周期运行指定的“任务”。

1. 进入“定时任务”页面。

2. 单击 `创建定时任务` ，进入“创建定时任务”的“基础配置”页面。

3. 填写基础配置参数。

参数	说明
定时规则	指定任务运行周期。

版权所有© 北京易捷思达科技发展有限公司

参数	说明
并发策略	* Forbid：在前一个任务未完成时，不创建新任务。 * Allow：当到达新任务创建时间点，而前一个任务未完成时，新的任务会取代前一个任务。 * Replace：定时任务不断创建新的任务，会抢占集群资源。
目标完成次数	当成功完成的容器组达到该值时认为任务完成。
并行实例数	每次创建的容器组数量。
失败重试次数	失败容器组的最大重试次数，超过这个次数不会继续重试。
超时时间	任务运行的超时时间。如果任务运行的时间超过了设定的时间，此任务将自动停止运行所有容器组。
重启策略	容器组内容器的重启策略，包括“不重启”和“失败时重启”。
调度策略	容器组内容器的调度策略。即调度工作负载时，是否能够容忍具有污点的节点。

4. 单击 `下一步：容器配置` ，进入“创建定时任务”的“容器配置”页面。

5. 填写容器配置参数，参数说明请参考 创建工作负载。

6. 单击 `创建` 完成操作。

管理工作负载

> 说明：
> 各类型工作负载支持的操作不尽相同，请根据实际页面显示和业务需求酌情配置。

查看工作负载详情

1. 进入相应类型工作负载的页面。
2. 找到目标工作负载，单击工作负载名称链接，进入工作负载详情页。
3. 查看工作负载详细信息。

版权所有© 北京易捷思达科技发展有限公司

容器配置

1. 进入相应类型工作负载的页面。
2. 找到目标工作负载，单击操作栏的 `容器配置` ，进入“容器配置”页面。
3. 配置参数，参数说明请参考 创建工作负载 。
4. 单击 `确认` 完成操作。

手动伸缩

> 说明：
> - 处于“已停止”状态的的工作负载不支持手动伸缩。
> - 针对部署类型的工作负载，若设置了弹性伸缩策略，则不支持进行手动伸缩。

1. 进入相应类型工作负载的页面。
2. 找到目标工作负载，单击操作栏的 `手动伸缩` ，弹出“手动伸缩”对话框。
3. 默认展示当前工作负载副本数量，可手动修改。此数量为目标值而非差值。
4. 单击 `确认` 完成操作。

版本回滚

1. 进入“部署”页面。
2. 找到目标工作负载，单击操作栏的 `更多` - `版本回滚` ，弹出“版本回滚”对话框。
3. 选择需要回滚到的历史版本。
4. 单击 `确认` 完成操作。

升级策略

1. 进入相应类型工作负载的页面。
2. 找到目标工作负载，单击操作栏的 `更多` - `升级策略` ，弹出“升级策略”对话框。
3. 配置参数，参数说明请参考 创建工作负载 。
4. 单击 `确认` 完成操作。

伸缩策略

版权所有© 北京易捷思达科技发展有限公司

1. 进入“部署”页面。
2. 找到目标工作负载，单击操作栏的 `更多` - `伸缩策略` ，弹出“伸缩策略”对话框。
3. 配置参数，参数说明请参考 创建工作负载 。
4. 单击 `确认` 完成操作。

调度策略

1. 进入相应类型工作负载的页面。
2. 找到目标工作负载，单击操作栏的 `更多` - `调度策略` ，弹出“调度策略“对话框。
3. 配置参数，参数说明请参考 创建工作负载 。
4. 单击 `确认` 完成操作。

网络设置

1. 进入相应类型工作负载的页面。
2. 找到目标工作负载，单击操作栏的 `更多` - `网络设置` ，弹出“网络设置”对话框。
3. 配置参数，参数说明请参考 创建工作负载 。
4. 单击 `确认` 完成操作。

标签设置

1. 进入相应类型工作负载的页面。
2. 找到目标工作负载，单击操作栏的 `更多` - `标签设置` ，弹出“标签设置”对话框。
3. 增加或移除标签。
4. 单击 `确认` 完成操作。

编辑Yaml

1. 进入相应类型工作负载的页面。
2. 找到目标工作负载，单击操作栏的 `更多` - `编辑Yaml` 或直接单击操作栏的 `编辑Yaml` ，弹出“编辑Yaml”对话框。
3. 修改信息。
4. 单击 `确认` 完成操作。

版权所有© 北京易捷思达科技发展有限公司

启动

1. 进入相应类型工作负载的页面。
2. 找到目标工作负载，单击操作栏的 `更多` - `启动` ，弹出对应提示框。
3. 单击 `启动` 完成操作。

停止

1. 进入相应类型工作负载的页面。
2. 找到目标工作负载，单击操作栏的 `更多` - `停止` ，弹出对应提示框。
3. 单击 `停止` 完成操作。

重新部署

1. 进入相应类型工作负载的页面。
2. 找到目标工作负载，单击操作栏的 `更多` - `重新部署` ，弹出对应提示框。
3. 单击 `重新部署` 完成操作。

删除

1. 进入相应类型工作负载的页面。
2. 找到目标工作负载，单击操作栏的 `更多` - `删除` 或直接单击操作栏的 `删除` ，弹出对应提示框。
3. 单击 `删除` 完成操作。

运行/停止定时任务

1. 进入“定时任务”页面。
2. 找到目标工作负载，单击操作栏的 `运行` 或 `停止` ，弹出对应提示框。
3. 单击 `运行` 或 `停止` 完成操作。

管理容器组

查看容器组详情

支持查看容器组基本信息、容器配置、状态、事件、监控、日志和终端。

版权所有© 北京易捷思达科技发展有限公司

1. 进入“容器组”页面。
2. 单击容器组名称链接，进入容器组详情页面，查看信息。

查看Yaml

1. 进入“容器组”页面。
2. 单击目标容器组操作栏的 `查看Yaml` ，查看信息。

查看日志

1. 进入“容器组”页面。
2. 单击目标容器组操作栏的 `日志` ，查看信息。

终端

1. 进入“容器组”页面。
2. 单击目标容器组操作栏的 `更多` - `终端` ，进入终端页面。

删除

1. 进入“容器组”页面。
2. 单击目标容器组操作栏的 `更多` - `删除` ，弹出“删除容器组”对话框。
3. 根据需要确认是否勾选“强制删除”。例如，目标容器组因所在节点已经停止或者无法连接API Server等异常情况无法被正常删除，此时可进行强制删除。
4. 单击 `删除` 完成操作。

版权所有© 北京易捷思达科技发展有限公司

4.7 持久卷声明

本章节主要介绍在“持久卷声明”页面中，针对持久卷声明的运维管理操作。“持久卷声明”页面进入路径如下：

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[安全容器服务]-[持久卷声明]，进入“持久卷声明”页面。
2. 在左侧导航栏选择目标命名空间。

创建持久卷声明

容器可通过持久卷声明请求使用持久化存储。

1. 进入“持久卷声明”页面。
2. 单击 `创建持久卷声明` ，弹出“创建持久卷声明”对话框。
3. 配置参数。
4. 单击 `创建` 完成操作。

参数	说明
存储类	即[管理视图]-[存储管理]中管理的存储类，详细介绍请参考 存储管理-存储类。
大小	所需存储卷的容量。
访问模式	包括三种模式，需根据存储类的能力选择其支持的模式： * 单节点读写（RWO）：卷可以被一个节点以读写方式挂载。 * 多节点读写（RWX）：卷可以被多个节点以读写方式挂载。 * 多节点只读（ROX）：卷可以被多个节点以只读方式挂载。若“部署”类型的工作负载需挂载单节点读写（RWO）模式的卷，其副本数需为1；若“任务”、“定时任务”类型的工作负载需挂载单节点读写（RWO）模式的卷，其并行实例数需为1。

编辑Yaml

1. 进入“持久卷声明”页面。
2. 单击目标持久卷声明操作栏的 `编辑Yaml` ，弹出“编辑Yaml”对话框。
3. 修改信息。

版权所有© 北京易捷思达科技发展有限公司

4. 单击 `确认` 完成操作。

删除

> 说明：
> 已关联容器组的持久卷声明不支持删除。

1. 进入“持久卷声明”页面。
2. 单击目标持久卷声明操作栏的 `删除` ，弹出“删除持久卷声明”提示框。
3. 单击 `删除` 完成操作。

版权所有© 北京易捷思达科技发展有限公司

4.8 配置中心

配置

配置用于保存配置数据，可以用作工作负载的环境变量、命令行参数或者存储卷中的配置文件。使用配置实现容器化应用的配置管理，可以使配置与镜像内容分离，保持容器化应用的可移植性。本章节主要介绍在“配置”页面中，针对“配置”的运维管理操作。“配置”页面进入路径如下：

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[安全容器服务]-[配置中心]，进入“配置中心”页面。
2. 在左侧导航栏选择目标命名空间，选择[配置中心]-[配置]，进入“配置”页面。

创建配置

1. 进入“配置”页面。
2. 单击 `创建配置` ，进入“创建配置”页面。
3. 填写配置名称和配置项内容。
4. 单击 `创建配置` 完成操作。

编辑Yaml

1. 进入“配置”页面。
2. 单击目标配置操作栏的 `编辑Yaml` ，弹出“编辑Yaml”对话框。
3. 修改信息。
4. 单击 `确认` 完成操作。

更新

1. 进入“配置”页面。
2. 单击目标配置操作栏的 `更新` ，进入“更新”页面。
3. 填写配置名称和配置项内容。
4. 单击 `更新` 完成操作。

删除

版权所有© 北京易捷思达科技发展有限公司

1. 进入“配置”页面。
2. 单击目标配置操作栏的 `删除` ，弹出“删除配置”提示框。
3. 单击 `删除` 完成操作。

密钥

密钥（Secret）是一种包含认证信息、密钥等敏感信息的资源类型，可以用作工作负载的环境变量、加密配置文件。将数据放在密钥对象中，可以更好地控制它的用途，并降低意外暴露的风险。本章节主要介绍在“密钥”页面中，针对“密钥”的运维管理操作。“密钥”页面进入路径如下：

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[安全容器服务]-[配置中心]，进入“配置中心”页面。
2. 在左侧导航栏选择目标命名空间，选择[配置中心]-[密钥]，进入“密钥”页面。

创建密钥

1. 进入“密钥”页面。
2. 单击 `创建密钥` ，进入“创建密钥”页面。
3. 配置参数。
4. 单击 `创建` 完成操作。

参数	说明
密钥类型	* Opaque：一般密钥类型。 * TLS：存放7层负载均衡服务所需的证书。 * 镜像访问密钥：存放拉取私有仓库镜像所需的认证信息。
密钥数据	* 当密钥类型为Opaque时，单击“添加密钥数据”，输入键、值。 * 当密钥类型为TLS时，上传证书和私钥文件。 * 当密钥类型为镜像访问密钥时，输入镜像仓库地址、用户名、密码和邮箱。

编辑Yaml

1. 进入“密钥”页面。
2. 单击目标密钥操作栏的 `编辑Yaml` ，弹出“编辑Yaml”对话框。

版权所有© 北京易捷思达科技发展有限公司

3. 修改信息。
4. 单击 `确认` 完成操作。

更新

1. 进入“密钥”页面。
2. 单击目标密钥操作栏的 `更新` ，进入“更新”页面。
3. 修改信息。
4. 单击 `更新` 完成操作。

删除

1. 进入“密钥”页面。
2. 单击目标密钥操作栏的 `删除` ，弹出“删除密钥”提示框。
3. 单击 `删除` 完成操作。

版权所有© 北京易捷思达科技发展有限公司

4.9 网络管理

服务

服务（Service）是容器服务的基本操作单元，是将请求进行负载分发到后端的各个容器应用上的控制器。对外表现为一个单一访问接口，外部不需要了解后端如何运行，这给扩展或维护后端带来很大的好处。本章节主要介绍在“服务”页面中，针对“服务”的运维管理操作。“服务”页面进入路径如下：

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[网络]-[服务]，进入“服务”页面。
2. 在左侧导航栏选择目标命名空间，选择[网络管理]-[服务]，进入“服务”页面。

创建服务

1. 进入“服务”页面。
2. 单击 `创建服务` ，进入“创建服务”页面。
3. 配置参数。
4. 单击 `创建` 完成操作。

参数		说明
类型	ClusterIP	适用于集群内部访问场景，集群为服务分配一个固定的集群内虚拟IP，集群内其它pod可以通过集群内部域名访问，格式为“<服务名称>.<工作负载所在命名空间>.svc.cluster.local:<端口号>”。集群外无效。
	NodePort	适用于集群外部访问场景，集群除了会给服务分配一个内部的虚拟IP，还会在每个节点上为服务分配静态端口号，集群外部可通过集群任一节点IP和静态端口号访问服务。
	LoadBalancer	负载均衡器会随服务一起创建。若需从集群外部访问，需在独享型负载均衡页面绑定公网IP，并使用 <公网IP<:<端口号> 进行访问。当删除服务时，负载均衡器也将自动删除。
	ExternalName	用于将服务请求指向一个自定义的域名。

版权所有© 北京易捷思达科技发展有限公司

参数	说明
以下是ClusterIP，NodePort和LoadBalancer公共参数	
容器端口	容器镜像中工作负载实际监听的端口。
访问端口	容器端口映射到节点IP上的端口。当访问方式为“NodePort”时，支持随机生成。
协议	包括TCP、UDP，根据业务类型选择。
关联工作负载	选择服务需关联的工作负载。当服务类型为“ExternalName”无此参数。
以下是LoadBalancer独有参数	
可用区	负载均衡器所在可用区。
规格	负载均衡器的规格。
CPU架构	负载均衡器的CPU架构，包括X86计算和ARM计算。
磁盘类型	负载均衡器的磁盘类型。
拓扑模式	负载均衡器的拓扑模式，包括主备模式和多活模式。
VPC	负载均衡器关联的VPC。
子网	负载均衡器关联的子网，只可选择非容器网络类型的子网。
VIP地址	负载均衡器的VIP地址，支持自动分配和手动指定。
以下是ExternalName独有参数	
目标	服务请求指向的目标域名。

编辑Yaml

版权所有© 北京易捷思达科技发展有限公司

1. 进入“服务”页面。
2. 单击目标服务操作栏的编辑Yaml，弹出“编辑Yaml”对话框。
3. 修改信息。
4. 单击 `确认` 完成操作。

更新

1. 进入“服务”页面。
2. 单击目标服务操作栏的 `更新` ，进入“更新”页面。
3. 修改参数。
4. 单击 `保存` 完成操作。

删除

1. 进入“服务”页面。
2. 单击目标服务操作栏的 `删除` ，弹出“删除服务”提示框。
3. 单击 `删除` 完成操作。

Ingresses

Ingress是一组将集群内服务暴露给集群外服务的路由规则集合。一个Ingress对象能够配置具备为服务提供外部可访问的URL、负载均衡流量、卸载SSL/TLS，以及提供基于名称的虚拟主机等能力。本章节主要介绍在“Ingresses”页面中，针对“Ingress”的运维管理操作。“Ingresses”页面进入路径如下：

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[安全容器服务]-[网络管理]，进入“网络管理”页面。
2. 在左侧导航栏选择目标命名空间，选择[网络管理]-[Ingresses]，进入“Ingresses”页面。

创建Ingress

1. 进入“Ingresses”页面。
2. 单击 `创建Ingress` ，进入“创建Ingress”页面。
3. 配置参数。
4. 单击 `创建` 完成操作。

版权所有© 北京易捷思达科技发展有限公司

参数	说明
Ingress规则	是一种HTTP方式的路由转发机制。例如域名填写为example.com，路径填写为/path，服务选择已创建的名称为“app”的服务，则外部可通过 `http://example.com/path` 访问名称为“app”的服务。
注解	Ingress经常使用注解（annotations）来配置一些选项，具体取决于Ingress控制器。

编辑Yaml

1. 进入“Ingresses”页面。
2. 单击目标Ingress操作栏的 `编辑Yaml` ，弹出“编辑Yaml”对话框。
3. 修改信息。
4. 单击 `确认` 完成操作。

更新

1. 进入“Ingresses”页面。
2. 单击目标Ingress操作栏的 `更新` ，进入“更新”页面。
3. 修改参数。
4. 单击 `保存` 完成操作。

删除

1. 进入“Ingresses”页面。
2. 单击目标Ingress操作栏的 `删除` ，弹出“删除Ingress”提示框。
3. 单击 `删除` 完成操作。

版权所有© 北京易捷思达科技发展有限公司

4.10 自定义资源管理

本章节主要介绍在“自定义资源管理”页面中，针对自定义资源的运维管理操作。“自定义资源管理”页面进入路径如下：

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[安全容器服务]-[自定义资源CRD管理]或[自定义资源CRD]，进入“自定义资源管理”页面。
2. 在左侧导航栏选择目标命名空间。

导入自定义资源描述/自定义资源

1. 进入“自定义资源管理”页面。
2. 单击页面右下角的“Yaml”图标，进入“导入Yaml”页面。
3. 直接粘贴Yaml文件内容，或单击编辑区域右上角的“导入”图标，选择本地存储的Yaml文件。

> 说明：
> - 请关注调试结果。该调试主要针对格式校验，若有错误可点击错误信息，跳至目标行进行修改。
> - 针对自定义资源描述，只能云管理员在[管理视图]中导入。

4. 待调试通过后，单击 `导入` ，完成操作。

查看自定义资源描述详情

1. 进入“自定义资源管理”页面。
2. 在自定义资源描述列表中单击目标资源描述的名称链接，进入资源描述详情页面。
3. 选择[Yaml]页签，查看其Yaml信息；选择[自定义资源]页签，查看自定义资源的信息。

删除自定义资源

1. 进入“自定义资源管理”页面。
2. 在自定义资源描述列表中单击目标资源描述的名称链接，进入资源描述详情页面。
3. 选择[自定义资源]页签后，勾选目标资源，单击 `删除` ，弹出“删除自定义资源”提示框。

版权所有© 北京易捷思达科技发展有限公司

4. 单击 `删除` 完成操作。

版权所有© 北京易捷思达科技发展有限公司

5 最佳实践

5.1 配置SR-IOV网络

5.1.1 前置条件准备

为了使用SR-IOV功能，需要满足以下前提条件：

1. 服务器BIOS中已开启SR-IOV功能。
2. 启用IOMMU功能，并以Pass-Through模式进行配置。
3. 已安装SR-IOV网卡驱动。
4. 网卡固件启用SR-IOV功能。
5. 网卡驱动启用SR-IOV。

这些前提条件的满足将确保系统具备使用SR-IOV功能所需的硬件支持和驱动程序。请确保在使用SR-IOV功能之前，在执行后续的配置操作之前，先完成以下准备工作：

操作步骤

1.服务器BIOS中已开启SR-IOV功能

请确保服务器BIOS中已开启SR-IOV功能。每个服务器都有不同的虚拟化BIOS配置选项。BIOS configuration examples可参考：BIOS configuration examples

2.启用IOMMU功能，并以Pass-Through模式进行配置

以Intel服务器为例，请确保将'intel_iommu=on'和'iommu=pt'添加到grub文件的配置中。

```
# cat /boot/grub2/grub.cfg

# GRUB Environment Block
saved_entry=0
kernelopts=root=/dev/mapper/os-root ro edd=off kvm.halt_poll_ns=400000
cgroup.memory=nokmem intel_iommu=on iommu=pt pci=realloc
```

版权所有© 北京易捷思达科技发展有限公司

```
ixgbe.allow_unsupported_sfp=1 rootdelay=90 nomodeset intel_idle.max_cstate=0
processor.max_cstate=0 crashkernel=300M rd.lvm.lv=os/root biosdevname=0
net.ifnames=1
boot_success=0
```

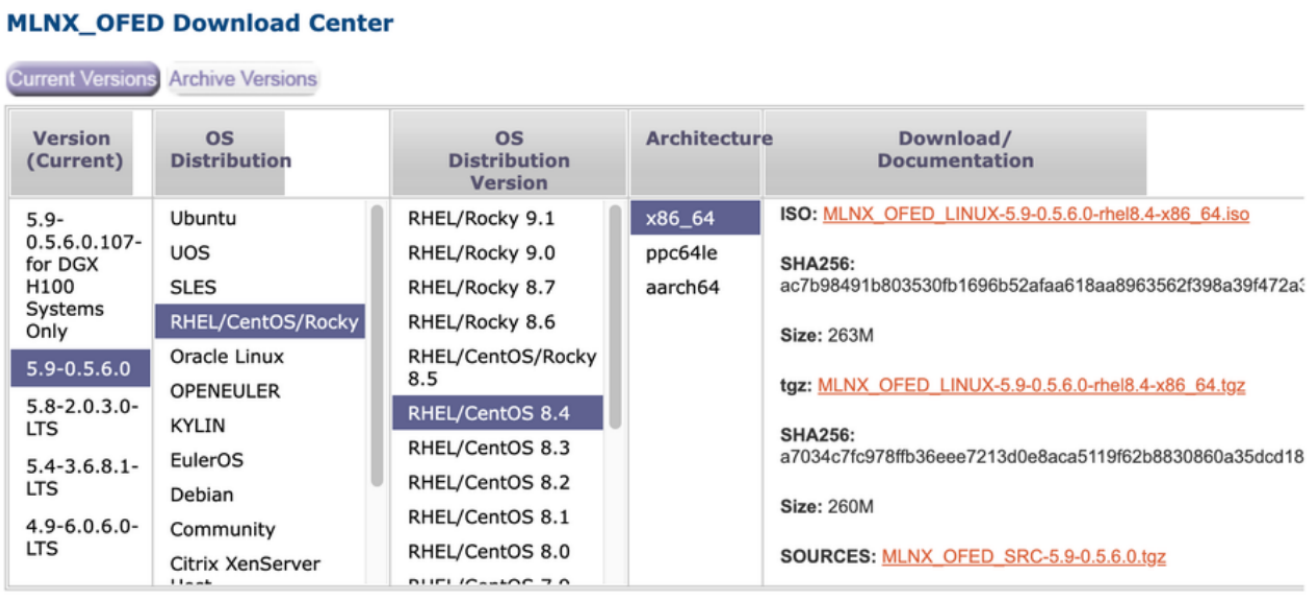
要了解更多关于iommu grub参数的信息，请参阅:Understanding the iommu Linux grub File Configuration

3.安装SR-IOV网卡驱动

如果使用产品提供的网卡驱动，请联系云原生团队获取和安装，下面给出安装NVIDIA驱动的步骤。

3.1 下载驱动包

NVIDIA驱动官方下载：Linux InfiniBand Drivers 请根据您所选择的操作系统和版本，以及CPU架构，下载相应的tar包。例如，选择MLNX_OFED_LINUX-5.9-0.5.6.0-rhel8.4-x86_64.tgz进行下载。



3.2 执行安装操作

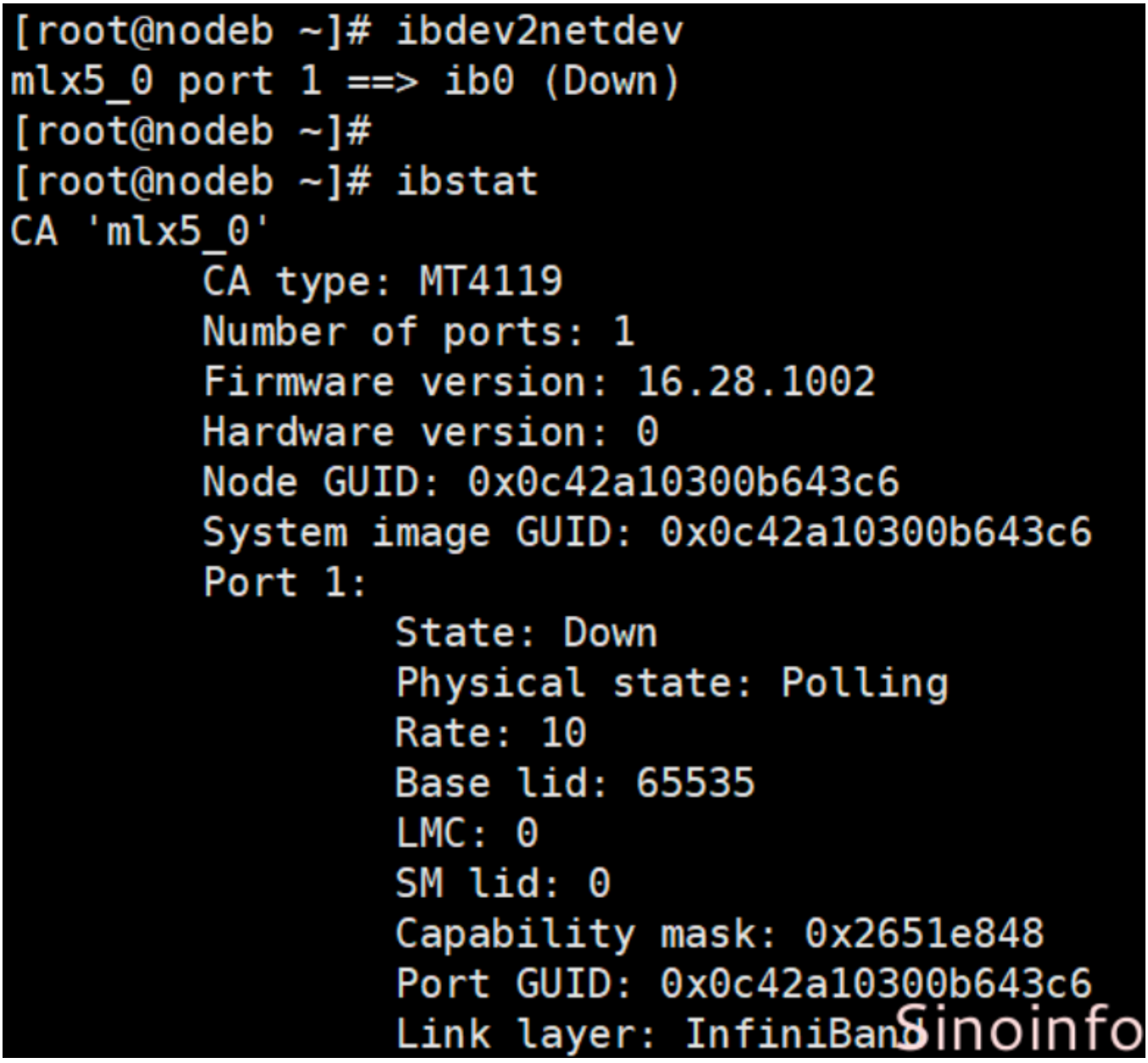
```
# ./mlnxofedinstall
```

3.3 查看驱动版本信息

```
# ofed_info -s
MLNX_OFED_LINUX-5.9-0.5.6.0:
```

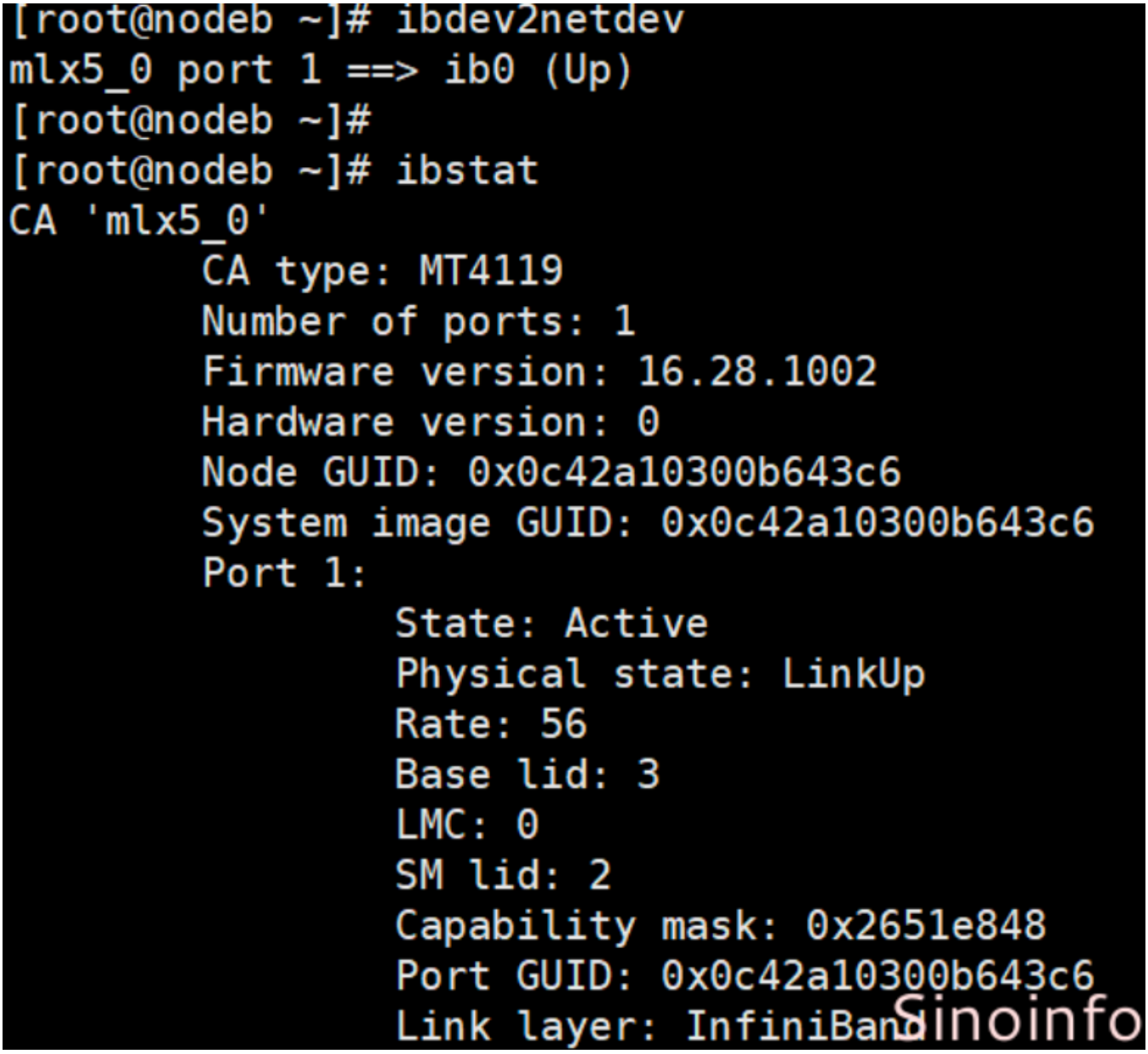
3.4 网卡状态

3.4.1 首次安装网卡时，如果未连接到交换机，网卡的状态将如下所示：



（如上图，ib0状态Down（双口卡还有ib1），网卡的SM lid为0，Base lid为65535，Link layer为IB模式）

3.4.2 使用线缆将网卡连接到交换机，大概30秒，端口UP，如下所示：



（如上图，网卡ib0物理状态LinkUp, 逻辑状态Active，子网管理器SM lid为 2，网卡Base lid为 3）

> 说明：
> 本案例交换机SB7800已经开启子网管理器功能，交换机端口速率为EDR 100Gb/s，网卡CX5也是 EDR 100Gb/s，由于线缆是FDR 56G，所以上图的Rate：56，并没有达到100

3.4.3 启动opensmd服务（视交换机型号选装）

如果交换机是SB7890或者是其它不带子网管理器功能的交换机（SB7890，SB7790，SX6025，IS5025），必须在服务器上开启子网管理器， 安装好网卡驱动后就可以使用下面命令：

```
# systemctl start opensmd
# systemctl enable opensmd
```

3.4.4 网卡运行环境监测

完成以上步骤，使用命令'mlnx_tune'检查，所有状态都正常，则网卡运行环境正常。

```
# mlnx_tune
```

4.网卡固件启用SR-IOV功能

4.1 下载Mellanox Management Tools (MFT)工具

1.联系云原生团队获取和安装（推荐）。 2.官方下载安装：Mellanox Management Tools (MFT)

4.2 使用Mellanox Firmware Tools包在固件中启用和配置SR-IOV

```
# mst start
Starting MST (Mellanox Software Tools) driver set
Loading MST PCI module - Success
Loading MST PCI configuration module - Success
Create devices
```

4.2.1 在需要的PCI插槽上找到Connect-IB设备

```
# mst status
MST modules:
------------
    MST PCI module loaded
    MST PCI configuration module loaded
MST devices:
------------
/dev/mst/mt4115_pciconf0         - PCI configuration cycles access.
...
```

4.2.2 查询设备状态

```
# mlxconfig -d /dev/mst/mt4115_pciconf0 q
```

4.2.3 开启SR-IOV，设置可以切分的vf数量上限

```
# mlxconfig -d /dev/mst/mt4115_pciconf0 set SRIOV_EN=1 NUM_OF_VFS=8
...
Apply new Configuration? ? (y/n) [n] : y
Applying... Done!
-I- Please reboot machine to load new configurations.
```

4.2.4 设置网卡的工作模式。

Infiniband卡支持两种工作模式：IB模式和Ethernet模式

```
# mlxconfig -d /dev/mst/mt4115_pciconf0 q

Configurations:                                Next Boot
         ...
         LINK_TYPE_P1                          IB(1)    #当前工作模式 IB
         ...
```

修改网卡的工作模式：

```
Ethernet模式:  mlxconfig -d /dev/mst/mt4115_pciconf0 set LINK_TYPE_P1=2
IB模式:  mlxconfig -d /dev/mst/mt4115_pciconf0 set LINK_TYPE_P1=1
```

重启机器生效

```
# reboot
```

> 此时，通过lspci无法看到vf。只有当SR-IOV在MLNX_OFED驱动上启用时，您才能看到它们。

4.2.5 查看节点物理网卡是否支持SR-IOV

```
查看所有的pci设备
# lspci -nnn |grep -i mellanox
```

版权所有© 北京易捷思达科技发展有限公司

```
61:00.0 Infiniband controller [0207]: Mellanox Technologies MT27700 Family
[ConnectX-4] [15b3:1013]

查看设备详情
# lspci -vvs
61:00.0 Infiniband controller: Mellanox Technologies MT27700 Family
[ConnectX-4]
    ...
    Capabilities: [180 v1] Single Root I/O Virtualization (SR-IOV)
        IOVCap: Migration-, Interrupt Message Number: 000
        IOVCtl: Enable- Migration- Interrupt- MSE- ARIHierarchy+
        IOVSta: Migration-
        Initial VFs: 3, Total VFs: 3, Number of VFs: 0, Function Dependency
Link: 00
        VF offset: 1, stride: 1, Device ID: 1014
        Supported Page Size: 000007ff, System Page Size: 00000001
        Region 0: Memory at 000005464e000000 (64-bit, prefetchable)
        VF Migration: offset: 00000000, BIR: 0
    Capabilities: [1c0 v1] Secondary PCI Express
        LnkCtl3: LnkEquIntrruptEn- PerformEqu-
        LaneErrStat: 0
    Kernel driver in use: mlx5_core
    Kernel modules: mlx5_core
```

显示如下代码段，说明固件已启动SR-IOV功能



5.网卡驱动启用SR-IOV

5.1 定位设备（通常为 'mlx5_0'）

```
# ibstat
CA 'mlx5_0'
    CA type: MT4115
    Number of ports: 1
    Firmware version: 12.28.2006
    Hardware version: 0
    Node GUID: 0xe41d2d03006768fa
    System image GUID: 0xe41d2d03006768fa
    Port 1:
```

```
        State: Down
        Physical state: Disabled
        Rate: 10
        Base lid: 65535
        LMC: 0
        SM lid: 0
        Capability mask: 0x2650e848
        Port GUID: 0xe41d2d03006768fa
        Link layer: InfiniBand
```

5.2 获取此设备上的当前vf数

```
# cat /sys/class/infiniband/mlx5_0/device/mlx5_num_vfs
0
```

> 如果命令失败，这可能意味着驱动程序没有加载。

5.3 设置所需的vf数量

```
# echo 2 > /sys/class/infiniband/mlx5_0/device/mlx5_num_vfs
```

> 更改mlx5_num_vfs不是持久的，并且不能在服务器重新启动后继续存在。

5.4 检查PCI总线

```
# lspci -nnn |grep -i mellanox
61:00.0 Infiniband controller [0207]: Mellanox Technologies MT27700 Family
[ConnectX-4] [15b3:1013]
61:00.1 Infiniband controller [0207]: Mellanox Technologies MT27700 Family
[ConnectX-4 Virtual Function] [15b3:1014]
61:00.2 Infiniband controller [0207]: Mellanox Technologies MT27700 Family
[ConnectX-4 Virtual Function] [15b3:1014]
```

5.5 恢复vf数量为0

```
# echo 0 > /sys/class/infiniband/mlx5_0/device/mlx5_num_vfs
```

版权所有© 北京易捷思达科技发展有限公司

版权所有© 北京易捷思达科技发展有限公司

5.1.2 使用Yaml配置SR-IOV网络

背景描述

SR-IOV (Single Root I/O Virtualization) 是一种允许物理设备如网络接口卡(NIC)在没有软件的情况下将其资源分割成多个虚拟设备的技术。使用SR-IOV可以在性能和资源隔离方面提供显著的优势，因为它减少了网络数据在主机和虚拟机之间传输的开销。安全容器服务支持SR-IOV网络设备插件，用于发现、公告和分配SR-IOV网络虚拟功能 (VF) 资源。本章节通过使用Yaml创建功能，配置SR-IOV网络。

前置条件

为了使用SR-IOV功能，需要满足以下前提条件：

1. 服务器BIOS中已开启SR-IOV功能。
2. 启用Intel IOMMU功能，并以Pass-Through模式进行配置。
3. 已安装NVIDIA MLNX_OFED驱动。
4. 网卡固件启用SR-IOV功能。
5. 在MLNX_OFED驱动上启用SR-IOV。 详细操作步骤请参见前置条件。这些前提条件的满足将确保系统具备使用SR-IOV功能所需的硬件支持和驱动程序。请确保在使用SR-IOV之前，你的系统已按照要求进行相应的设置和安装。

操作步骤

前置操作完成后，需重启环境中sriov-config-daemon的Pod

因为 `sriov-config-daemon` 是 `daemonSet` 部署，则删除 `namespace` 为 `eks-managed` ，label为 `app=sriov-network-config-daemon` 的Pod即可。

创建SriovNetworkNodePolicy对象：

SriovNetworkNodePolicy是SR-IOV Network Operator的一部分，用于定义如何配置SR-IOV网络。它是一个Kubernetes自定义资源(CR)，用于指定SR-IOV网络配置策略。 **可以通过定义SriovNetworkNodePolicy对象来指定节点的SR-IOV网络设备配置：**

版权所有© 北京易捷思达科技发展有限公司

1.查看节点SR-IOV设备信息 节点sriov-config-daemon上报上来的SR-IOV设备情况会更新到节点对应的 `sriovnetworknodestates.status` ，即查看 `sriovnetworknodestates.status` 内容

```
# kubectl get sriovnetworknodestates.sriovnetwork.openshift.io  -n eks-managed -o yaml
```

```
status:
  interfaces:
  - deviceID: "1013"
    driver: mlx5_core
    linkType: IB
    mac: 00:00:07:ff:fe:80:00:00:00:00:00:00:e4:1d:2d:03:00:67:68:fa
    mtu: 2048
    name: ib0
    pciAddress: 0000:61:00.0
    totalvfs: 3
    vendor: 15b3
 syncStatus: Succeeded
```

2.根据节点SR-IOV设备情况，创建SriovNetworkNodePolicy资源。

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy
metadata:
  name:  1
  namespace: eks-managed
spec:
  resourceName:  2
  nodeSelector:
    : 3
  priority:  3
  mtu:  4
  numVfs:  5
  nicSelector: 6
    vendor: "" 7
    deviceID: "" 8
    pfNames: ["", ...] 9
    rootDevices: ["", "..."] 10
```

版权所有© 北京易捷思达科技发展有限公司

```
  deviceType: vfio-pci
  isRdma: false 11
```

参数	说明
name	CR对象的名称。
resourceName	SR-IOV设备插件的资源名称。您可以为一个资源名称创建多个"SriovNetworkNodePolicy"对象。
priority（可选）	"0"到"99"之间的整数。较小的数值具有较高的优先权，优先级"10"高于优先级"99"。默认值为"99"。
mtu（可选）	为虚拟功能（VF）的最大传输单位 (MTU) 指定一个值。MTU的值必须是在"1"到"9000"之间。如果您不需要指定MTU，请指定一个"值。
numVfs	为SR-IOV物理网络设备指定要创建的虚拟功能 (VF) 的数量。对于Intel网络接口卡 (NIC)，VF的数量不能超过该设备支持的VF总数。对于Mellanox NIC，VF的数量不能超过"128"。
nicSelector	nicSelector 映射为Operator选择要配置的设备。您不需要为所有参数指定值。建议您以足够的准确度来识别网络设备，以便尽量减小意外选择其他网络设备的可能性。如果指定了rootDevices，则必须同时为vendor、deviceID或pfNames指定一个值。如果同时指定了"pfNames"和"rootDevices"，请确保它们指向同一个设备。
vendor（可选）	SR-IOV网络设备的厂商十六进制代码。允许的值只能是"8086"和"15b3"。
deviceID（可选）	SR-IOV网络设备的设备十六进制代码。允许的值只能是"158b"、"1015"和"1017"。
pfNames（可选）	该设备的一个或多个物理功能（PF）名称的数组。
rootDevices	用于该设备的PF的一个或多个PCI总线地址的数组。使用以下格式提供地址: "0000:02:00.1"。
isRdma（可选）	是否启用远程直接访问（RDMA）模式。默认值为"false"。

版权所有© 北京易捷思达科技发展有限公司

> 说明:
> 写**sriovNetworkNodePolicy**的 `spec` 的定义，要根据 `sriovNetworkNodeState.Status.interface` 去写。 若**sriovNetworkNodePolicy**的 `spec` 定义，不在 `sriovNetworkNodeState.Status.interface` 的设备范围内，则生成只有 `DpConfigVersion` 的 `spec` 。

创建 SriovIBNetwork资源:

您可以通过定义对象来配置 InfiniBand (IB) 网络设备"SriovIBNetwork"。

1.下方示例Yaml展示了如何通过 SriovIBNetwork 对象，配置 Infiniband network attachment

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovIBNetwork
metadata:
  name:  1
  namespace: eks-managed
spec:
  resourceName:  2
  networkNamespace:  3
  vlan:  4
  spoofChk: "" 5
  ipam: |- 6
    {}
  linkState:  7
  maxTxRate:  8
  minTxRate:  9
  vlanQoS:  10
  trust: "" 11
  capabilities:  12
```

参数	说明
name	对象的名称。SR-IOV Network Operator创建一个名称相同的"NetworkAttachment Definition"对象。

参数	说明
resourceName	定义附加网络的SR-IOV硬件"SriovNetworkNodePolicy"对象中的"spec.resourceName"参数的值。
networkNamespace	SriovIBNetwork对象的目标命名空间。只有目标命名空间中的Pod才能连接到网络设备。
vlan（可选）	额外网络的虚拟LAN（VLAN）ID。它需要是一个从"0"到"4095"范围内的一个整数值。默认值为"0"。
spoofChk（可选）	VF的"spoof"检查模式。允许的值是字符串"on"和"off"。 重要：指定的值必须由引号包括，否则SR-IOV Network Operator将拒绝对象。
ipam	为IPAM CNI插件指定一个配置对象做为一个Yaml块scalar。该插件管理网络附加定义的IP地址分配。
linkState（可选）	虚拟功能（VF）的链接状态。允许的值是"enable"、"disable"和"auto"。
maxTxRate（可选）	VF的最大传输率（以Mbps为单位）。
minTxRate（可选）	VF的最低传输率（以Mbps为单位）。这个值必须小于或等于最大传输率。 说明:Intel NIC不支持minTxRate参数。如需更多信息，请参阅 BZ#1772847
vlanQoS（可选）	VF的"IEEE 802.1p"优先级级别。默认值为"0"。
trust（可选）	VF的信任模式。允许的值是字符串"on"和"off"。 说明:您必须在引号中包含指定的值，或者SR-IOV Network Operator拒绝对象。
capabilities（可选）	为这个额外网络配置功能。您可以指定 "{ "ips": true }" 来启用IP地址支持，或指定 "{ "mac": true }" 来启用MAC地址支持。

2.使用Whereabouts进行动态IP地址分配配置

```
{
 "ipam": {
 "type": "whereabouts",
```

版权所有© 北京易捷思达科技发展有限公司

```
 "range": "", 1
 "exclude": [", ..."], 2
 }
}
```

参数	说明
range	以CIDR标记指定一个IP地址范围。IP地址是通过这个地址范围来分配的。
exclude（可选）	在CIDR标记中指定IP地址和范围列表。包含在排除地址范围中的IP地址。

创建业务容器（Pod）资源：

指定业务容器的规范，包括容器运行时（如runc、rune）、容器镜像、资源需求（如CPU和内存）、环境变量、命令等。

```
apiVersion: v1
kind: Pod
metadata:
  name: sriov-rune-pod
  annotations:
    io.katacontainers.config.runtime.enable_sriov: "true"
    k8s.v1.cni.cncf.io/networks: eks-managed/nics
    #v1.multus-cni.io/default-network: eks-managed/kube-ovn
spec:
  runtimeClassName: rune
  containers:
  - name: appcntr1
    image: hub.easystack.io/captain/nginx-ingress-controller:v0.49.3
    imagePullPolicy: IfNotPresent
    command: [ "/bin/bash", "-c", "--" ]
    args: [ "while true; do sleep 300000; done;" ]
    resources:
      requests:
        : '1'
      limits:
        : '1'
```

版权所有© 北京易捷思达科技发展有限公司

> 注意：业务Pod的resources.requests、resources.limits的资源名称必须与sriovNetworkNodePolicy.spec.resourceName、sriovNetwork.spec.resourceName相等。

版权所有© 北京易捷思达科技发展有限公司

5.1.3 runc容器和InfiniBand卡IB模式场景

基于runc运行时容器和InfiniBand卡（IB模式）的组合场景，主要原理是将容器运行时环境与高性能的InfiniBand网络卡相结合。通过利用runc容器的轻量级和可移植性优势，与InfiniBand卡（IB模式）相结合，适用于安全隔离的需求较低，对网络性能要求较高的应用场景。本文将通过Yaml配置信息和参数，演示如何定义SR-IOV网络节点的策略。

操作步骤

配置SriovNetworkNodePolicy对象

指定切分 `kubernetes.io/hostname=node-10` 节点上， `rootDevices: 0000:71:00.0` 的PF设备

> 警告:
> 创建SR-IOV SriovNetworkNodePolicy对象时，节点应用修改会重启。

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy
metadata:
  name: node-policy-10
  namespace: eks-managed
spec:
  resourceName: mlxnics
  nodeSelector:
    kubernetes.io/hostname: node-10
  nicSelector:
    vendor: "15b3"
    deviceID: "1017"
    rootDevices:
      - 0000:71:00.0
 deviceType: netdevice
 numVfs: 3
 priority: 50
 isRdma: true
 linkType: IB
```

版权所有© 北京易捷思达科技发展有限公司

配置SriovIBNetwork对象

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovIBNetwork
metadata:
  name: ibnics
  namespace: eks-managed
spec:
  ipam: |-
    {
    "type": "whereabouts",
    "range": "192.168.100.0/24",
    "gateway": "192.168.100.1",
    "exclude": [
      "192.168.100.0/26"
    ]
    }
  resourceName: mlxnics
  linkState: auto
```

配置runc运行时环境中的业务Pod

```
apiVersion: v1
kind: Pod
metadata:
  name: sriov-runc-pod-demo
  annotations:
    k8s.v1.cni.cncf.io/networks: eks-managed/ibnics
spec:
  containers:
  - name: app-demo
    image: hub.ecns.io/test/nginx:latest
    imagePullPolicy: Always
    command: [ "/bin/bash", "-c", "--" ]
    args: [ "while true; do sleep 300000; done;" ]
    resources:
      requests:
        ecnf.io/mlxnics: "1"
      limits:
```

版权所有© 北京易捷思达科技发展有限公司

```
        ecnf.io/mlxnics: "1"
  nodeName: node-10
```

版权所有© 北京易捷思达科技发展有限公司

5.1.4 runc容器和InfiniBand卡ETH模式场景

基于runc运行时容器和InfiniBand卡（ETH模式）的组合场景，主要原理是将容器运行时环境与高性能的InfiniBand网络卡相结合。通过利用runc容器的轻量级和可移植性优势，与InfiniBand卡（ETH模式）相结合，适用于安全隔离的需求较低，需要高性能数据传输和低延迟的应用场景。本文将通过Yaml配置信息和参数，演示如何定义SR-IOV网络节点的策略。

操作步骤

配置SriovNetworkNodePolicy对象

指定切分 `kubernetes.io/hostname=node-10` 节点上， `rootDevices: 0000:71:00.0` 的PF设备.

> 警告:
> 创建SR-IOV SriovNetworkNodePolicy对象时，节点应用修改会重启。

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy
metadata:
  name: node-policy-10
  namespace: eks-managed
spec:
  resourceName: mlxnics
  nodeSelector:
    kubernetes.io/hostname: node-10
  nicSelector:
    vendor: "15b3"
    deviceID: "1017"
    rootDevices:
      - 0000:71:00.0
 deviceType: netdevice
 numVfs: 3
 priority: 50
 isRdma: false
 linkType: ETH
```

版权所有© 北京易捷思达科技发展有限公司

配置SriovIBNetwork对象

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetwork
metadata:
  name: nics
  namespace: eks-managed
spec:
  ipam: |-
    {
    "type": "whereabouts",
    "range": "192.168.100.0/24",
    "gateway": "192.168.100.1",
    "exclude": [
      "192.168.100.0/26"
    ]
    }
 resourceName: mlxnics
```

配置runc运行时环境中的业务Pod

```
apiVersion: v1
kind: Pod
metadata:
  name: sriov-runc-pod-demo
  annotations:
    k8s.v1.cni.cncf.io/networks: eks-managed/nics
spec:
  containers:
  - name: app-demo
    image: hub.ecns.io/test/nginx:latest
    imagePullPolicy: Always
    command: [ "/bin/bash", "-c", "--" ]
    args: [ "while true; do sleep 300000; done;" ]
    resources:
      requests:
        ecnf.io/mlxnics: "1"
      limits:
```

版权所有© 北京易捷思达科技发展有限公司

```
        ecnf.io/mlxnics: "1"
  nodeName: node-10
```

版权所有© 北京易捷思达科技发展有限公司

5.1.5 rune容器和InfiniBand卡IB模式场景

基于rune（安全运行时）容器和InfiniBand卡（IB模式）的组合场景，主要原理是将容器运行时环境与高性能的InfiniBand网络卡相结合。通过利用rune容器的安全性和隔离性优势，与InfiniBand卡（IB模式）相结合，适用于对安全性和隔离性有一定需求的轻量级传输应用场景。本文将通过Yaml配置信息和参数，演示如何定义SR-IOV网络节点的策略。

操作步骤

配置SriovNetworkNodePolicy对象：

指定切分 `kubernetes.io/hostname=node-10` 节点上， `rootDevices: 0000:71:00.0` 的PF设备

> 警告：
> 创建SR-IOV SriovNetworkNodePolicy对象时，节点应用修改会重启。

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy
metadata:
  name: node-policy-5
  namespace: eks-managed
spec:
  resourceName: mlxnics
  nodeSelector:
    kubernetes.io/hostname: node-5
  nicSelector:
    vendor: "15b3"
    deviceID: "1017"
    rootDevices:
      - 0000:71:00.0
  deviceType: vfio-pci
  numVfs: 3
  priority: 50
  isRdma: false
  linkType: IB
```

版权所有© 北京易捷思达科技发展有限公司

配置SriovIBNetwork对象：

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovIBNetwork
metadata:
  name: ibnics
  namespace: eks-managed
spec:
  ipam: |-
    {
    "type": "whereabouts",
    "range": "192.168.100.0/24",
    "gateway": "192.168.100.1",
    "exclude": [
      "192.168.100.0/26"
    ]
    }
  resourceName: mlxnics
  linkState: auto
```

配置 rune(安全运行时) 环境中的业务 Pod：

如果需要对容器（Pod）进行资源限制（limit）的设置，您可以在Pod的request字段中设置limit值。为了实现这个需求，您需要为Pod添加以下的annotation配置：

`io.katacontainers.config.runtime.sandbox_cgroup_only: "false"`

```
apiVersion: v1
kind: Pod
metadata:
  name: sriov-rune-pod-demo
  annotations:
    k8s.v1.cni.cncf.io/networks: eks-managed/ibnics
    io.katacontainers.config.runtime.enable_sriov: "true"
spec:
  runtimeClassName: rune
  containers:
  - name: app-demo
    image: hub.ecns.io/test/nginx:latest
    imagePullPolicy: Always
```

版权所有© 北京易捷思达科技发展有限公司

```
    command: [ "/bin/bash", "-c", "--" ]
    args: [ "while true; do sleep 300000; done;" ]
    resources:
      requests:
        ecnf.io/mlxnics: "1"
      limits:
        ecnf.io/mlxnics: "1"
  nodeName: node-10
```

版权所有© 北京易捷思达科技发展有限公司

5.1.6 rune容器和InfiniBand卡ETH模式场景

基于rune（安全运行时）容器和InfiniBand卡（ETH模式）的组合场景，主要原理是将容器运行时环境与高性能的InfiniBand网络卡相结合。通过利用rune容器的安全性和隔离性优势，与InfiniBand卡（ETH模式）相结合，适用于对安全性和隔离性有一定需求的，需要快速数据传输和低延迟的应用场景。本文将通过Yaml配置信息和参数，演示如何定义SR-IOV网络节点的策略。

操作步骤

配置SriovNetworkNodePolicy对象：

指定切分 `kubernetes.io/hostname=node-10` 节点上， `rootDevices: 0000:71:00.0` 的PF设备

> 警告：
> 创建SR-IOV SriovNetworkNodePolicy对象时，节点应用修改会重启。

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetworkNodePolicy
metadata:
  name: node-policy-10
  namespace: eks-managed
spec:
  resourceName: mlxnics
  nodeSelector:
    kubernetes.io/hostname: node-10
  nicSelector:
    vendor: "15b3"
    deviceID: "1017"
    rootDevices:
      - 0000:71:00.0
  deviceType: vfio-pci
  numVfs: 3
  priority: 50
  isRdma: false
  linkType: ETH
```

版权所有© 北京易捷思达科技发展有限公司

配置SriovIBNetwork对象：

```
apiVersion: sriovnetwork.openshift.io/v1
kind: SriovNetwork
metadata:
  name: nics
  namespace: eks-managed
spec:
  ipam: |-
    {
    "type": "whereabouts",
    "range": "192.168.100.0/24",
    "gateway": "192.168.100.1",
    "exclude": [
    "192.168.100.0/26"
    ]
    }
  resourceName: mlxnics
```

配置 rune(安全运行时) 环境中的业务 Pod：

如果需要对容器（Pod）进行资源限制（limit）的设置，您可以在Pod的request字段中设置limit值。为了实现这个需求，您需要为Pod添加以下的annotation配置：

`io.katacontainers.config.runtime.sandbox_cgroup_only: "false"`

```
apiVersion: v1
kind: Pod
metadata:
  name: sriov-rune-pod-demo
  annotations:
  k8s.v1.cni.cncf.io/networks: eks-managed/nics
  io.katacontainers.config.runtime.enable_sriov: "true"
spec:
  runtimeClassName: rune
  containers:
  - name: app-demo
    image: hub.ecns.io/test/nginx:latest
    imagePullPolicy: Always
    command: [ "/bin/bash", "-c", "--" ]
```

版权所有© 北京易捷思达科技发展有限公司

```
    args: [ "while true; do sleep 300000; done;" ]
    resources:
      requests:
        ecnf.io/mlxnics: "1"
      limits:
        ecnf.io/mlxnics: "1"
  nodeName: node-10
```

版权所有© 北京易捷思达科技发展有限公司

5.2 创建GPU资源的容器实例

背景描述

GPU在工作负载中的主要作用是利用其出色的并行计算能力。在安全容器服务中，将GPU卡用于运行计算密集型工作负载对于某些特定的业务场景非常有价值，包括高性能计算、数据处理和分析加速、图形渲染和可视化优化等。通过将GPU与容器技术结合，可以实现灵活、高效且可扩展的计算环境，提供更好的用户体验和性能表现。本章节将详细介绍如何在部署工作负载时使用GPU卡能力。

前提条件

- **容器运行时：**当容器运行时为"安全运行时（rune）"时，支持使用GPU资源。（注意："守护进程集（DaemonSet）"和"定时任务（CronJob）"类型的工作负载不支持使用GPU。）
- **资源预留：**当需要勾选"使用GPU"时，建议设置CPU的参数值大于等于1，内存的参数值大于等于1024MiB。这样可以确保在使用GPU时，为工作负载分配足够的计算资源和内存资源，以获得良好的性能和稳定性。

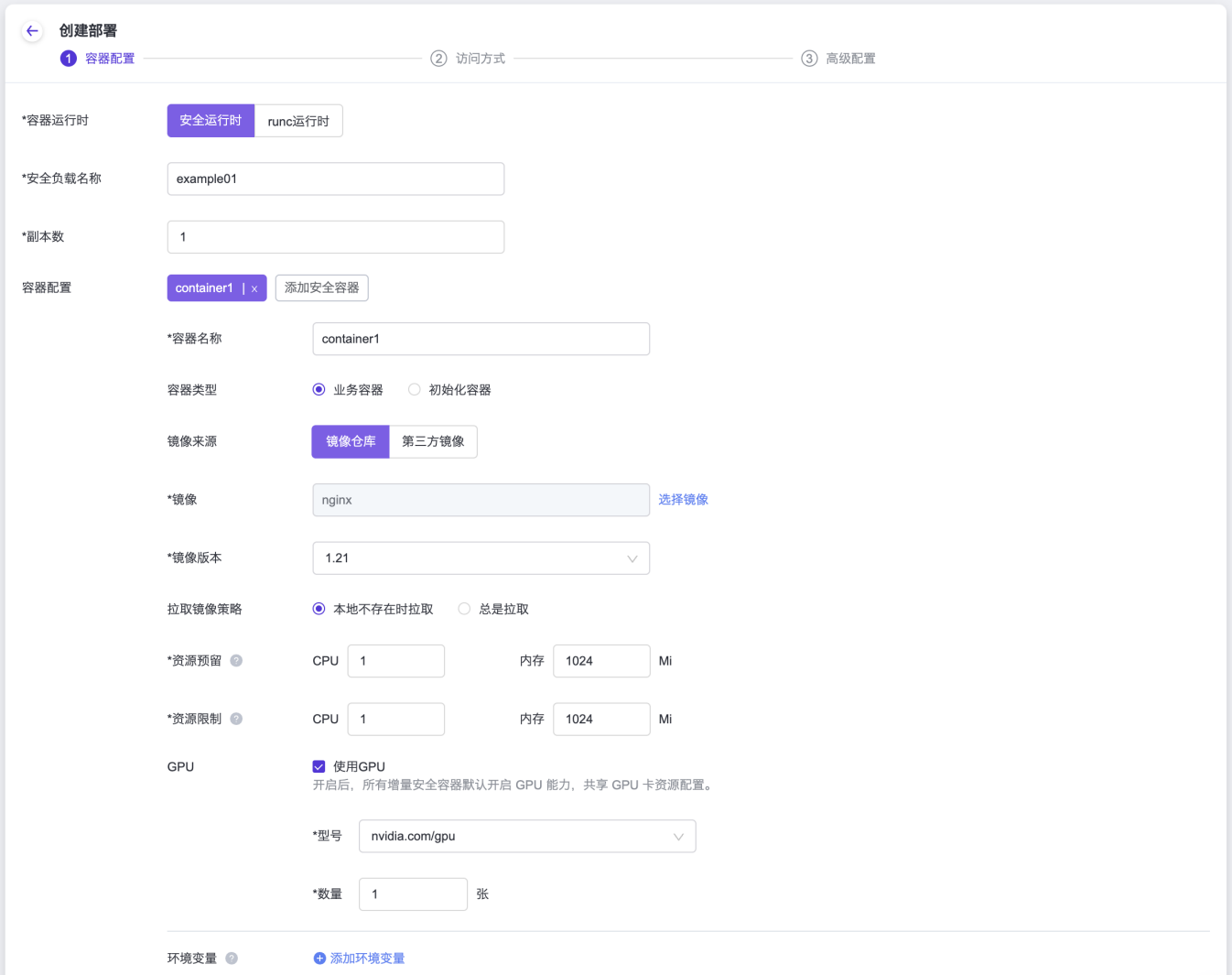
操作步骤

目前平台中支持在安全容器中使用英伟达GPU设备和百度昆仑XPU设备。具体支持型号参见使用限制。下文将分别列出不同品牌GPU的操作步骤和示例，以帮助用户正确配置和使用GPU资源。

英伟达（NVIDIA）GPU设备

在支持使用GPU能力的工作负载类型创建界面中，勾选"使用GPU"选项，并选择"nvidia.com/gpu"作为GPU资源的类型。这样配置后，系统将为工作负载分配相应的GPU资源，以提供所需的计算能力和性能。创建容器组配置时，配置好容器（Pod） GPU参数，所有增量容器（Pod）默认共享GPU卡资源。

版权所有© 北京易捷思达科技发展有限公司



如果需要对容器（Pod）进行资源限制（limit）的设置，您可以在Pod的request字段中设置limit值。为了实现这个需求，您需要为Pod添加以下的annotation配置：

```
io.katacontainers.config.runtime.sandbox_cgroup_only: "false"
```

Yaml示例：

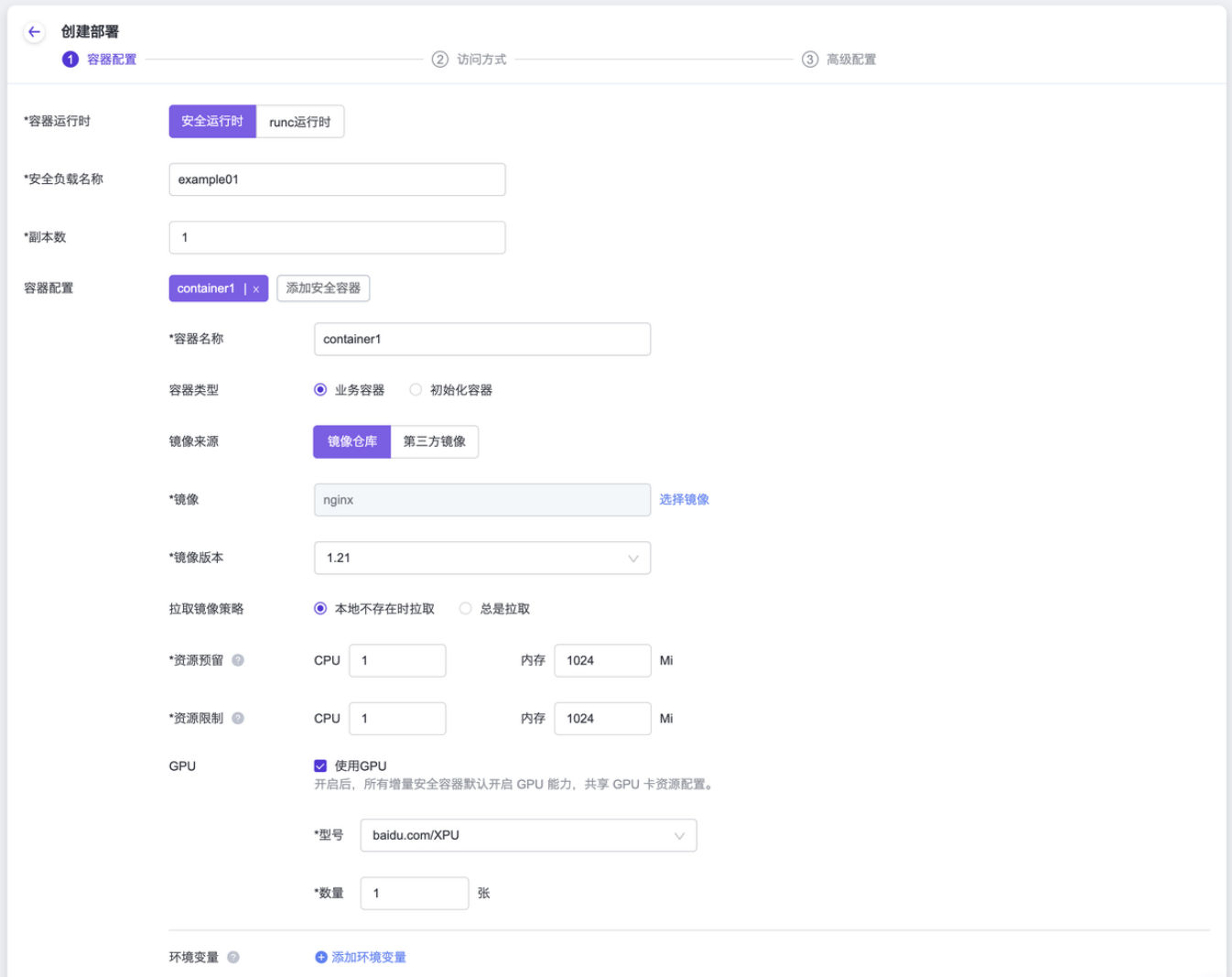
```
template:
    metadata:
      labels:
        my-app: perf-server0
      annotations:
        # 在limit中配置了memory时，需要添加如下注解
```

```
        io.katacontainers.config.runtime.sandbox_cgroup_only: "false"
    spec:
      runtimeClassName: rune
      containers:
      - name: perf-server0
        image: docker.io/test/perf:0.0.1
        command:
        - iperf3
        args:
        - -s
        resources:
          limits:
            cpu: "2"
            memory: 2Gi
            # 配置GPU资源数量
            nvidia.com/gpu: "1"
          requests:
            cpu: "2"
            memory: 2Gi
            nvidia.com/gpu: "1"
```

百度昆仑XPU

在支持使用GPU能力的工作负载类型创建界面中，勾选"使用GPU"选项，并选择"baidu.com/XPU"作为GPU资源的类型。这样配置后，系统将为工作负载分配相应的GPU资源，以提供所需的计算能力和性能。创建容器组配置时，配置好容器（Pod） GPU参数后，所有增量容器（Pod）默认共享GPU卡资源。

版权所有© 北京易捷思达科技发展有限公司



如果需要对容器（Pod）进行资源限制（limit）的设置，您可以在容器（Pod）的“request”字段中设置“limit”值。为了实现这个需求，您需要为容器（Pod）添加以下的“annotation”配置：

```
io.katacontainers.config.runtime.sandbox_cgroup_only: "false"
```

Yaml示例：

```
template:
   metadata:
     labels:
       my-app: perf-server0
     annotations:
       # 在limit中配置了memory时，需要添加如下注解
```

```
        io.katacontainers.config.runtime.sandbox_cgroup_only: "false"
    spec:
      runtimeClassName: rune
      containers:
      - name: perf-server0
        image: docker.io/test/perf:0.0.1
        command:
        - iperf3
        args:
        - -s
        resources:
          limits:
            cpu: "2"
            memory: 2Gi
            # 配置XPU资源数量
            baidu.com/XPU: "1"
          requests:
            cpu: "2"
            memory: 2Gi
            baidu.com/XPU: "1"
```

此外，百度昆仑XPU支持指定某个容器对象独享某个具体的GPU卡（其他容器不可见），可通过环境变量：`CXPU_VISIBLE_DEVICES: 1` 设置GPU卡隔离。当环境可用GPU卡未被指定完，配置后且未被隔离的GPU卡为所有容器共享资源。当环境可用GPU卡被指定完，剩余的容器无可用的GPU卡资源。

百度昆仑XPU在各个容器（Pod）中的可见性环境变量： `CXPU_VISIBLE_DEVICES` 可以设置值为 `all或者是卡序号（1, 2, 3）`

Yaml示例：

```
template:
    metadata:
      labels:
        my-app: perf-server0
      annotations:
        # 在limit中配置了memory时，需要添加如下注解
        io.katacontainers.config.runtime.sandbox_cgroup_only: "false"
    spec:
      runtimeClassName: rune
      containers:
```

版权所有© 北京易捷思达科技发展有限公司

```
    - name: perf-server0
      image: docker.io/test/perf:0.0.1
      command:
      - iperf3
      args:
      - -s
      env:
        - name: CXPU_VISIBLE_DEVICES
          value: ALL
      resources:
        limits:
          cpu: "2"
          memory: 2Gi
          # 配置GPU资源数量
          nvidia.com/gpu: "1"
        requests:
          cpu: "2"
          memory: 2Gi
          nvidia.com/gpu: "1"
```

附录

为了正确使用GPU资源，请确保准确指定GPU卡的型号对应的资源名称：

GPU卡型号	资源名称
英伟达（NVIDIA）GPU设备	nvidia.com/gpu
百度昆仑XPU设备	baidu.com/XPU

版权所有© 北京易捷思达科技发展有限公司

5.3 AI开发环境容器镜像的制作方法

EAF CUDA 开发容器基础镜像

这是一个面向 x86-64 GPU 开发环境的容器镜像，基于 CUDA 12.4、cuDNN 和 Ubuntu 22.04，预装 Python 3.10、JupyterLab、code-server 与 OpenSSH Server。

前置条件

- x86-64 Linux 主机
- Docker 及 NVIDIA Container Toolkit
- 与 CUDA 12.4 兼容的 NVIDIA 驱动

构建

```
docker build --platform linux/amd64 -t eaf-cuda-dev:12.4 .
```

可覆盖软件版本或开发用户 UID/GID：

```
docker build --platform linux/amd64 \
  --build-arg CODE_SERVER_VERSION=4.104.2 \
  --build-arg JUPYTERLAB_VERSION=4.4.7 \
  --build-arg DEV_UID="$(id -u)" \
  --build-arg DEV_GID="$(id -g)" \
  -t eaf-cuda-dev:12.4 .
```

运行

镜像默认只提供 `dev` 用户的交互式 Bash 环境，不会自动启动 JupyterLab、code-server、sshd 或 supervisord：

```
docker run --rm --gpus all \
  -it \
  --name eaf-cuda-dev \
```

版权所有© 北京易捷思达科技发展有限公司

```
  -v "$PWD/workspace:/workspace" \
  eaf-cuda-dev:12.4
```

需要时可在容器内手动启动工具：

```
# JupyterLab
jupyter lab --ip=0.0.0.0 --port=8888 --no-browser \
  --ServerApp.root_dir=/workspace --ServerApp.token= \
  --ServerApp.password= --ServerApp.allow_remote_access=True

# code-server
code-server --bind-addr 0.0.0.0:8080 --auth none \
  --disable-telemetry /workspace
```

如需使用预置的 SSH 配置和 supervisord 运行层，请以 root 用户启动入口脚本，再按需启动服务：

```
docker run --rm --gpus all -it --user root \
  -e DEV_PASSWORD='replace-with-a-strong-password' \
  -p 2222:22 -v "$PWD/workspace:/workspace" \
  eaf-cuda-dev:12.4 /usr/local/bin/eaf-entrypoint /bin/bash
```

入口脚本、`start-sshd`、`eaf-healthcheck` 和 supervisord 配置仍保留在镜像中，供手动调用。`EXPOSE` 仅声明端口，不会自动监听；启动服务时请按需发布对应端口。

验证 GPU 环境

```
docker exec eaf-cuda-dev nvidia-smi
docker exec eaf-cuda-dev nvcc --version
docker exec eaf-cuda-dev python3 --version
```

安全说明

手动启动 JupyterLab、code-server 或 SSH 时，它们会按上述配置监听容器网络接口；不要将 8888、8080 或 22 端口直接发布到公网。生产环境应通过防火墙、VPN 或带认证的反向代理限制访问。

版权所有© 北京易捷思达科技发展有限公司

6 常见问题

6.1 如何理解安全容器网络方案

问题描述

安全容器使用的CNI插件是什么？有什么重要特性？用户侧如何使用VPC、Subnet、FIP等各种网络资源？

解决方案

安全容器6.3.1版本CNI是基于易捷行云软SDN网络服务作为后端，以kube-ovn（v1.9.0版本）作为API接口，为不同算力(容器、云原生云主机、裸金属)提供统一网络方案；为实现这个设计目标，对kube-ovn 作为容器 CNI 做了一些改动和限制；和社区版本的主要区别体现在：

1、当前版本跨节点通信 不支持 underlay 模式，只支持geneve 隧道模式

2、VPC 功能增强 1）容器网络使用的VPC 依赖于软SDN的 router，所以默认VPC的更新配置，及 自定义VPC的创建 需要结合软SDN router页面操作完成，具体步骤见操作手册。

2）容器网络的VPC 网关出外网配置由软SDN router 自动完成，对应社区kube-ovn文档中 VPC 网关配置相关的操作，这部分无需用户再做配置。同时VPC只支持集中式网关(对应 VPC spec中 gatewaytype 字段)。

3）支持自定义VPC，且自定义VPC内容器网络也支持 nodeport、探针等功能(当前版本限制：不同vpc间 subnet cidr 不能有 overlap)，同时支持自定义VPC的网关节点的配置。

4）容器VPC与其他算力的互联需要在 软SDN router页面完成配置。

3、新增Floating IP（简称fip） 功能 1）fip 作为软SDN管理的统一资源，可以同时被 不同算力(vm、pod、裸金属) 使用；fip 在pods中的具体使用方式请参照用户手册。

2）产品中使用内部CRD fips.neutron.io 管理记录当前容器可用的fip 资源池，用户可以k8s API通过查看对应的 fip CR查看可用的 fip address 使用，fip的记录与回收会自动处理。

4、subnet功能增强 支持subnet 中定义projectIDs，表示该subnet只能被哪些projects使用；如果不配置表示该subnet为share subnet，所有用户可用。

版权所有© 北京易捷思达科技发展有限公司

如何启用kube-ovn 组件

通过安全容器服务页面创建的命名空间，默认所有负载都是使用kube-ovn作为CNI。

API方式，需要对命名空间打上标签 `managed.es.io/resource=namespace` ；例如：

```
apiVersion: v1
kind: Namespace
metadata:
  name: easystack
  managed.es.io/resource: namespace
```

如何自定义VPC网络

在创建工作负载时，其网络不仅支持使用系统默认生成的ovn-cluster和子网，还支持使用自定义的VPC网络或子网。用户可根据实际业务场景，酌情选择对应方案。

1. 配置外部网络。
    1. （可选）创建外部网络。

        本操作用于预创建外部网络，以便在自定义VPC网络时能够为其建立外部连接。如使用已有外部网络时，可跳过本步骤。

        1. 在云平台的顶部导航栏中，依次选择[产品与服务]-[网络]-[网络]，进入“网络”页面。
        2. 单击 `创建网络` ，进入“创建网络”页面。
        3. “网络类型”请选择“外部网络”，并配置其他参数后，单击 `创建网络` ，完成操作。其中，其他参数的具体参数说明，请参考“SDN网络服务”帮助中“网络”的相关内容。
    2. 查看外部网络的ID。

        在“网络”页面中，单击待操作网络名称，进入其详情页面。在该页面中，查看并记录该网络的ID（即UUID参数的值）。
2. 配置路由器。
    1. （可选）创建路由器。

版权所有© 北京易捷思达科技发展有限公司

本操作用于预创建路由器，以便在自定义VPC网络时能够为其建立外部连接。如使用已有路由器时，可跳过本步骤。

1. 在云平台的顶部导航栏中，依次选择[产品与服务]-[网络]-[路由器]，即可进入“路由器”页面。
2. 单击 `创建路由器` ，弹出“创建路由器”对话框。
3. 配置参数后，单击 `创建` ，完成操作。

2. （可选）设置路由器网关。

   本操作用于预设置路由器网关，以便在自定义VPC网络时能够为其建立外部连接。如路由器已设置网关时，可跳过本步骤。

   1. 在“路由器”页面中，勾选待操作路由器后，单击 `更多` - `设置网关` ，弹出“设置路由器网关”对话框。
   2. “分配外部IP”选择“手动选择”，“子网”选择上述外部网络子网，并配置其他参数后，单击 `设置` ，完成操作。其中，其他参数的具体参数说明，请参考“SDN网络服务”帮助中“路由器”的相关内容。

3. 查看路由器ID和外部网络IP地址。

   在“路由器”页面中，单击上述路由器名称，进入其详情页面。在该页面中，查看并记录该路由器的ID（即UUID参数的值）和外部网络IP地址（即外部IP参数的值）。

3. 导入VPC资源。

   1. 在云平台顶部导航栏中，依次选择[产品与服务]-[容器服务]-[安全容器服务]，进入“安全容器服务”页面。
   2. 在左侧导航栏选择[管理视图]-[自定义资源管理]，或在左侧导航栏选择[业务视图]，并选择目标命名空间后，选择[自定义资源管理]，进入“自定义资源管理”页面。
   3. 单击页面右下角的“Yaml”图标，进入“导入Yaml”页面。
   4. 依据实际业务情况，输入Yaml文件内容，或直接单击编辑区域右上角的“导入”图标，导入预先配置的Yaml文件。Yaml文件格式如下（其中，name为自定义输入的VPC名称，externalGatewayIp为外部网络的IP地址，externalNetworkID为外部网络的ID，neutronRouter为路由器的ID）：

      ```
      apiVersion: kubeovn.io/v1
      kind: Vpc
      metadata:
        name: test-vpc2
      ```

版权所有© 北京易捷思达科技发展有限公司

```
spec:
  externalGatewayIp: 172.110.0.160
  externalNetworkID: c9a831a0-6298-44c6-a664-2f362e60e419
  neutronRouter: c761887f-40bd-4df0-9b43-7c6bb009aab7
```

5. 待调试通过后，单击 `导入` ，完成操作。

6. spec关键字段说明：

   externalNetworkID：外部网络ID(必填)

   externalNetworkName：外部网络名字(选填)

   外部网关IP(选填)；注: 该ip为vpc下pod 访问外网的默认 snat ip, 允许pod访问外网有两种方式：vpc yaml中定义该字段，或者网络页面设置路由网关开启snat

   neutronRouter：路由ID(必填)

   gatewayNode(选填, ovn pod访问节点的出入口节点，默认为网络节点)

自定义子网

```
> 警告：
>
> * 在同一VPC内，请确保各子网不发生冲突。
> * 在不同VPC之间，如需配置多个用户路由器之间的云内对等连接，请确保各路由器的子网不发生冲突。
```

1. 在云平台顶部导航栏中，依次选择[产品与服务]-[容器服务]-[安全容器服务]，进入“安全容器服务”页面。

2. 在左侧导航栏选择[管理视图]-[自定义资源管理]，或在左侧导航栏选择[业务视图]，并选择目标命名空间后，选择[自定义资源管理]，进入“自定义资源管理”页面。

3. 单击页面右下角的“Yaml”图标，进入“导入Yaml”页面。

4. 依据实际业务情况，输入子网内容，或直接单击编辑区域右上角的“导入”图标，导入预先配置的Yaml文件。Yaml文件格式如下（其中，name为自定义输入的子网名称，vpc为该子网所属VPC的名称，cidrBlock为该子网的网段，natOutgoing为是否允许访问外部网络）：

版权所有© 北京易捷思达科技发展有限公司

```
  kind: Subnet
  apiVersion: kubeovn.io/v1
  metadata:
    name: net192
  spec:
    vpc: test-vpc-2
    cidrBlock: 192.168.100.0/24
    natOutgoing: true
```

5. 待调试通过后，单击 `导入` ，完成操作。

6. spec 关键字段说明

   vpc：属于哪个vpc

   cidrBlock：subnet cidr

   natOutgoing：是否可以访问外部网络

   projectIDs: 配置该subnet只能被哪些projects使用，如果不配置表示 该subnet为share subnet 所用用户可用

7. 如果通过页面部署工作负载时不选择Subnet子网，会使用vpc中默认子网进行部署；如果通过yaml定义，spec样例如下：

```
apiVersion: v1
kind: Pod
metadata:
  annotations:
    ovn.kubernetes.io/logical_switch: another-subnet
  namespace: default
  name: another-subnet-pod
```

Floating IP 使用

fip 在容器产品中使用包括两个场景：pods 指定使用 SNAT作为出口IP；允许kubevirt 云原生云主机 使用EIP。

使用方式有两种：页面操作，用户可在网络高级配置中可以选择正确的fip资源；或者yaml定义。

版权所有© 北京易捷思达科技发展有限公司

两种使用场景yaml spec定义样例如下： 1）SNAT出口IP

```
apiVersion: v1
kind: Pod
metadata:
  annotations:
    ovn.kubernetes.io/snat: 172.35.0.18
  name: snat-busybox
```

2）EIP

```
apiVersion: kubevirt.io/v1
kind: VirtualMachine
metadata:
  name: kubevirt-eip
spec:
  template:
    metadata:
      annotations:
        ovn.kubernetes.io/eip: 172.35.0.20
```

版权所有© 北京易捷思达科技发展有限公司

6.2 容器状态为未知错误，如何排查解决

问题描述

在云平台中，查看到容器的状态为“未知错误”。具体使用现象可能表现为：

- 在连接容器终端正常使用的过程中，断开连接后无法再次建立连接。
- 在云监控服务中，上报微服务管理停止告警。
- 维护此容器时，维护失败。

问题原因

容器发生未知错误。

解决方案

1. 在顶部导航栏选择[产品与服务]-[容器服务]-[安全容器服务]，进入“安全容器服务”页面。

2. 在左侧导航栏选择[业务视图]页签-选择目标命名空间后，选择“容器组”，进入“容器组”页面。

3. 单击目标容器所在容器组的名称链接，进入详情页面。

4. 在[终端]页签中，选择目标容器后，执行以下命令:

```
ctr -n k8s.io t ls | grep UNKNOWN |awk '{print $1}'|xargs -I % ctr -n
k8s.io t rm %
```

5. 确认问题已解决。具体命令如下（无返回数据，即表示问题已解决）：

```
ctr -n k8s.io t ls | grep UNKNOWN
```

版权所有© 北京易捷思达科技发展有限公司

6.3 容器状态为失败，如何排查解决

问题描述

在云平台中，查看到容器的状态为“失败”。

问题原因

容器命令执行错误。

解决方案

1. 在顶部导航栏选择[产品与服务]-[容器服务]-[安全容器服务]，进入“安全容器服务”页面。

2. 在左侧导航栏选择[业务视图]页签-选择目标命名空间后，选择“容器组”，进入“容器组”页面。

3. 单击目标容器所在容器组的名称链接，进入详情页面。

4. 在[事件]或[终端]页签中，排查具体执行错误的命令并解决。

版权所有© 北京易捷思达科技发展有限公司

7 部署指南

7.1 Kube-OVN部署指南

前言

安全容器云产品使用kube-ovn作为cni，底层调用的ovn，和proton底层的实现是一致的，并且一部分功能复用了proton。所以使用kube-ovn的pod可以类比为虚机，通过网络隔离加强了安全。

使用kube-ovn的pod以下场景是预期的： 1.pod内只通网络节点和安全容器节点的ovn0地址（br-roller，br-mgmt等地址是不通的）。 2.网络节点和安全容器节点可以访问所有的kube-ovn地址，非网络节点和安全容器节点角色，一般是不通pod的，要看云平台的网络规划。 3.不同vpc之间的pod默认是互相隔离的，包括service。 4.所有的kube-ovn网络的pod内都可以访问coredns服务和kubernetes服务（kubernetes.default.svc.cluster.local）。

另外，使用上有些限制需要注意： 1.所有subnet的网段不能重叠，即使跨vpc。 2.创建ns(namespace)时，如果没有设置全局默认subnet，namespace不会自动绑定subnet，如果没有手动给ns设置subnet，且此ns下的pod也不设置subnet，创建pod不会成功。 3.创建np(networkpolicy)时，如果np所在ns没有设置subnet，np的规则如果涉及到跨子网则会有问题。 4.已创建的subnet不能修改其vpc。 5.lic中指定的cluster-router参数指定的neutron router如果已存在，external-network参数则不会起作用。

云产品安装

使用场景的不同，云产品的一些参数需要灵活配置，可以通过license传入，目前kube-ovn相关的license是下面的参数： 值为"代表是有值的，只是值是空字符串，所以想用默认值，参数不需要写在license里。

```
cloud-product:
- eks-managed: {
    ttl: '-1',
    values: {
       cluster-router: 'test-router', //默认vpc名字，填写手动创建的neutron
router name,如果不设置或者传的router不存在，会默认创建名字为ovn-cluster的router。邮储
应设置为手动创建的neutron router name
       disable-init-subnet: True,  //是否创建默认的全局subnet ovn-default，不设置
的话默认为False会创建。邮储应设置为True
```

```
        external-network: '', //默认vpc外部网关绑定的公网名字，如果传的为空字符串的话，代表kube-ovn默认创建的默认vpc不绑定外部网关，如果未设置此参数，默认值为pulic-net。邮储应设置为''，即不绑定默认网关。需要注意的是如果cluster-router指定的router为提前手动创建好的，此参数不起作用。
        node-switch-cidr: '100.64.0.0/16', //默认创建的join subnet的网段，不设置默认为100.64.0.0/16。邮储应根据规定填入此值
        default-cidr: '10.16.0.0/16', //默认subnet ovn-default的网段。如果设置了disable-init-subnet为true，此值不生效。可以不设置，默认值为10.16.0.0/16
        default-gateway: '10.16.0.1', //默认subnet ovn-default的网关地址，与default-cidr类似，默认值为10.16.0.1
        default-exclude-ips: '10.16.0.1,10.16.0.2', //默认subnet ovn-default的不可分配地址，与default-cidr，默认值为网关地址
        node-switch-gateway: '100.64.0.1',  //join子网默认网关，一般不用设置，不设置默认为node-switch-cidr的第一个ip
        service-cluster-ip-range: '10.222.0.0/16', //k8s集群svc网段，不设置默认值为10.222.0.0/16
        flannel-pod-ip-range: '10.232.0.0/14', //flannel 网段，不设置默认值为10.232.0.0/14
        network-type: 'geneve', //网络模式，不设置默认genve
        enable_lb: True, //是否开启ovn lb，不设置默认为True
        enable_np:  True, //是否开启networkpolicy，不设置默认为True
        keep_vm_ip:  True, //是否开启kubevirt vmip保持，不设置默认为False,不保持，一般建议设置为True
        shared_services: '', //所有的vpc的lb上负载的svc，默认为空，格式为namespace/svc-name，用,分割
        gateway_node_labels: 'node-role.kubernetes.io/master=true' //kube-ovn pod访问coredns和k8s服务经过的容器网关节点label，lic不写此值，默认值为node-role.kubernetes.io/master=true；要么就必须设置此值，不能写空字符串
    }
}
```

安装完成后，根据disable-init-subnet参数，创建的默认资源会不同，如果disable-init-subnet为False，代表创建默认全局subnet ovn-default。

```
[root@node-3 ~]# kubectl get vpc
NAME            STANDBY   SUBNETS                                 NAMESPACES
ovn-cluster     true      ["join","ovn-default"]
[root@node-3 ~]# kubectl get subnet
NAME            PROVIDER   VPC             PROTOCOL   CIDR
PRIVATE   NAT     DEFAULT   GATEWAYTYPE   V4USED   V4AVAILABLE   V6USED
V6AVAILABLE   EXCLUDEIPS
```

```
join            ovn         ovn-cluster     IPv4        100.64.0.0/16   false
false   false     distributed   11        65522          0         0
["100.64.0.1"]
ovn-default     ovn         ovn-cluster     IPv4        10.16.0.0/16    false
true    true      distributed   4         65527          0         0
["10.16.0.1"]
```

disable-init-subnet为True，不创建默认全局subnet ovn-default。

```
[root@node-3 ~]# kubectl get vpc
NAME            STANDBY   SUBNETS                                NAMESPACES
ovn-cluster     true      ["join"]
[root@node-3 ~]# kubectl get subnet
NAME            PROVIDER   VPC             PROTOCOL   CIDR
PRIVATE   NAT     DEFAULT   GATEWAYTYPE   V4USED   V4AVAILABLE   V6USED
V6AVAILABLE   EXCLUDEIPS
join            ovn         ovn-cluster     IPv4        100.64.0.0/16   false
false   false     distributed   11        65522          0         0
["100.64.0.1"]
```

另外还会创建ip资源，所有的网络和安全容器节点都会占用一个join网络的地址。

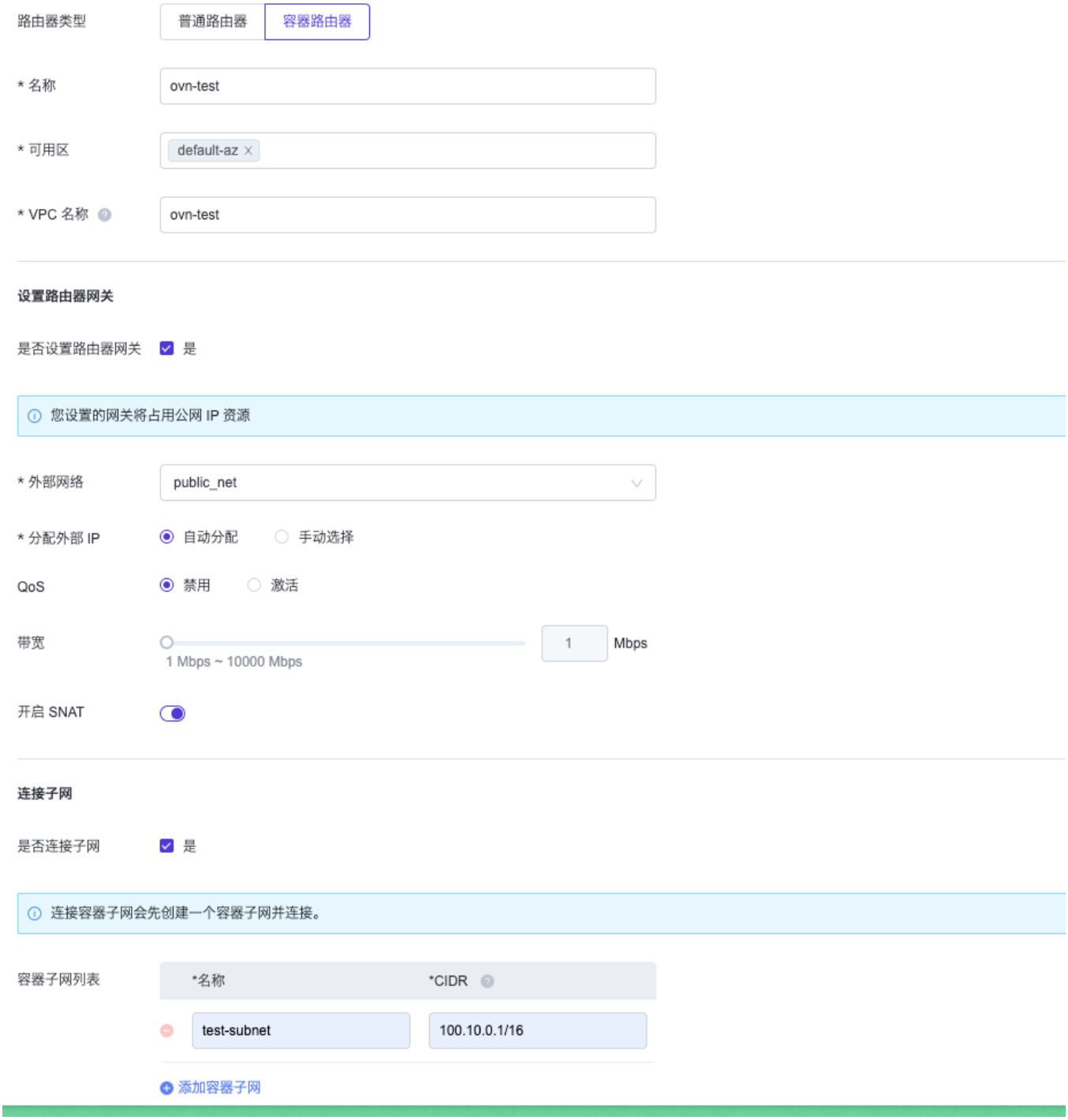
```
[root@node-3 ~]# kubectl get ip
NAME                              V4IP          V6IP   MAC
NODE                SUBNET
node-node-1                       100.64.0.3           00:00:00:74:BF:BB
node-1              join
node-node-2                       100.64.0.4           00:00:00:66:A6:AE
node-2              join
node-node-25                      100.64.0.2           00:00:00:E3:DF:5A
node-25             join
node-node-3                       100.64.0.5           00:00:00:10:AD:31
node-3              join
node-node-39                      100.64.0.9           00:00:00:4C:22:A5
node-39             join
```

创建VPC和Subnet

版权所有© 北京易捷思达科技发展有限公司

如果不使用默认创建的vpc，可以创建自定义vpc和subnet，两种方式: 1.通过网络页面创建容器路由器，选择容器路由器，如果不设置网关，vpc下资源不能出外网，选择连接子网，代表是否给vpc绑定subnet。这里也可以选择不连接子网，等容器路由器创建好后，在网络页面创建容器网络并连接到此路由器。

版权所有© 北京易捷思达科技发展有限公司





版权所有© 北京易捷思达科技发展有限公司



并会在后台创建vpc和subnet资源。

```
[root@node-9 ~]# kubectl get vpc new-vpc
NAME      STANDBY   SUBNETS              NAMESPACES
new-vpc   true      ["new-ovn-subnet"]
[root@node-9 ~]# kubectl get subnet new-ovn-subnet
NAME             PROVIDER   VPC       PROTOCOL   CIDR              PRIVATE
NAT     DEFAULT   GATEWAYTYPE   V4USED   V4AVAILABLE   V6USED   V6AVAILABLE
EXCLUDEIPS
new-ovn-subnet   ovn        new-vpc   IPv4       172.16.100.0/24   false
false   false     distributed   2        251           0        0
["172.16.100.1"]
```

2.yaml创建 先创建vpc:

- externalGatewayIp: 外部网关地址，选填,该ip为vpc下pod 访问外网默认 snat ip, 允许pod访问外网有两种方式：vpc yaml中定义该字段 或者 网络页面设置路由网关开启snat。
- externalNetworkID: 外部网络ID,如果vpc需要访问外部网络并要使用EIP和SNAT功能，此值为必填，如果不需要上述功能，此值不要设置。
- externalNetworkName：外部网络名字，选填。
- neutronRouter: proton router id，必填。
- gatewayNode：选填, kube-ovn pod访问节点的出入口节点，默认为网络节点。
- namespaces: 选填，指定哪些namespace可以使用此vpc，不可与其他vpc下的namespace重叠。

```
apiVersion: kubeovn.io/v1
kind: Vpc
metadata:
  name: ovn-cluster
spec:
  externalGatewayIp: 172.110.0.207
```

版权所有© 北京易捷思达科技发展有限公司

```
  externalNetworkID: c9a831a0-6298-44c6-a664-2f362e60e419
  externalNetworkName: public_net
  neutronRouter: c58a843a-b007-4fcf-949f-02836b71b7f3
  gatewayNode: node-1,node-2,node-3
  namespaces:
  - test1
  - test2
```

一般就设置neutronRouter和externalNetworkID即可。

```
apiVersion: kubeovn.io/v1
kind: Vpc
metadata:
  name: ovn-cluster
spec:
  externalNetworkID: c9a831a0-6298-44c6-a664-2f362e60e419
  neutronRouter: c58a843a-b007-4fcf-949f-02836b71b7f3 //proton router id, 必填
```

创建subnet：

- cidrBlock：subnet cidr，必填。
- default: true/false, 默认为false，vpc下哪个subnet为默认subnet，要避免一个vpc下多个subnet设置为true。如果一个vpc下多个subnet设置为true，其中一个subnet会被认为是默认subnet，不保证时间顺序，选填。
- excludeIps：不能被分配的地址，默认为网关地址，选填。
- gateway：网关地址，默认为子网的第一个地址，选填。
- namespaces：能使用此subnet的namespace，必须是关联vpc的namespaces子集，选填。
- private: 默认为false，如果设置成true，则该子网将和其他子网以及外部网络隔离，只能进行子网内部的通信。如需开白名单，可以通过 allowSubnets 进行设置。allowSubnets 内的网段和该子网可以双向互访，选填。
- allowSubnets: private设置为true后，允许访问的网段，单独设置无意义，选填。
- projectIDs: 配置该subnet只能被哪些projects使用，如果不配置表示 该subnet为share subnet 所用用户可用，目前页面上创建的容器网络不带有project id，后期可能会加上此选项，选填。
- vpc：属于哪个vpc，必填。

版权所有© 北京易捷思达科技发展有限公司

```
apiVersion: kubeovn.io/v1
kind: Subnet
metadata:
  name: custom-subnet
spec:
  cidrBlock: 192.168.233.0/24
  default: false
  excludeIps:
  - 192.168.233.1
  gateway: 192.168.233.1
  namespaces:
  - net2
  - test
  private: false
  allowSubnets:
  - 100.64.0.1/16
  projectIDs:
  - 3a4a2f23238d4851a64f79b4411bbee3
  vpc: custom-vpc
```

创建namespace

namespace会和subnet绑定，如果namespace没有绑定subnet，需要pod手动指定subnet。 namespace绑定subnet有多种情况： 1.假设subnet里namespaces指定了需要绑定subnet的ns，则此namespace会绑定此subnet，即使namespace手动指定了subnet也会被覆盖。 2.如果subnet没指定此ns，则会再去vpc的namespaces里找，如果找到了此ns，则会绑定此vpc的默认subnet，如果此vpc没有默认subnet，则会使用全局默认subnet，如果没有全局默认subnet，则此ns不绑定subnet。 3.如果没有vpc指定绑定此namespace，则会使用全局默认vpc（ovn-cluster）的默认subnet，如果此vpc没有默认subnet，则会使用全局默认subnet，如果没有全局默认subnet，则此ns不绑定subnet。 4.假设上述情况都不满足，则可以手动指定ns需要绑定的subnet，给ns添加注解ovn.kubernetes.io/logical_switch：

```
apiVersion: v1
kind: Namespace
metadata:
  annotations:
    ovn.kubernetes.io/logical_switch: yyx-yyx
  labels:
    managed.es.io/resource: namespace
```

版权所有© 北京易捷思达科技发展有限公司

```
    name: test123
  name: test123
```

namespace绑定subnet的优先级如下:

```
subnet指定ns > vpc指定ns > 全局默认subnet > ns手动指定subnet
```

创建负载

1.负载不手动指定subnet的情况下，会使用ns绑定的subnet。 2.负载手动指定subnet，在spec.template.metadata.annotations加上subnet的注解。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: test
  namespace: demo
spec:
  replicas: 1
  selector:
    matchLabels:
      workload.es.io: deployment.demo-2
    type: RollingUpdate
  template:
    metadata:
      annotations:
        ovn.kubernetes.io/logical_switch: ovn-ljj
      labels:
        workload.es.io: deployment.demo-2
    spec:
      containers:
        - image: 'hub.easystack.io/arm64v8/escloud-linux-source-
busybox:latest'
          imagePullPolicy: IfNotPresent
          name: container1
```

3.使用eip或snat 使用eip或snat，使用的subnet关联的vpc必须开启了外部网关。 使用eip，deployment，statefulset或daemonset的副本数必须为1。

版权所有© 北京易捷思达科技发展有限公司

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: test
  namespace: demo
spec:
  replicas: 1
  selector:
    matchLabels:
      workload.es.io: deployment.demo-2
    type: RollingUpdate
  template:
    metadata:
      annotations:
        ovn.kubernetes.io/logical_switch: ovn-ljj
        ovn.kubernetes.io/eip: 172.35.0.20
      labels:
        workload.es.io: deployment.demo-2
    spec:
      containers:
        - image: 'hub.easystack.io/arm64v8/escloud-linux-source-
busybox:latest'
          imagePullPolicy: IfNotPresent
          name: container1
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: test
  namespace: demo
spec:
  replicas: 1
  selector:
    matchLabels:
      workload.es.io: deployment.demo-2
    type: RollingUpdate
  template:
    metadata:
      annotations:
        ovn.kubernetes.io/logical_switch: ovn-ljj
```

版权所有© 北京易捷思达科技发展有限公司

```
        ovn.kubernetes.io/snat: 172.35.0.18
      labels:
        workload.es.io: deployment.demo-2
    spec:
      containers:
        - image: 'hub.easystack.io/arm64v8/escloud-linux-source-
busybox:latest'
          imagePullPolicy: IfNotPresent
          name: container1
```

版权所有© 北京易捷思达科技发展有限公司

8 升级指南

8.1 示例

待补充内容

版权所有© 北京易捷思达科技发展有限公司

9 运维指南

9.1 运维指南

一云多芯环境工作负载调度到不同CPU架构节点

问题描述

一云多芯环境在安全容器服务6.3.1之前版本创建的工作负载，可能会出现调度到不同CPU架构节点的问题。jira：https://easystack.atlassian.net/browse/EAS-127670。

问题原因

安全容器服务6.3.1支持一云多芯，创建工作负载会根据kubernetes.io/arch 的节点标签调度到指定CPU架构的节点上，而之前版本创建的工作负载，nodeSelector不包含这个标签，可能调度到不同架构的节点上。

解决办法

手动编辑工作负载yaml文件，nodeSelector增加kubernetes.io/arch 的节点标签，x86节点工作负载编辑如下： nodeSelector: kubernetes.io/arch: amd64 arm节点工作负载编辑如下： nodeSelector: kubernetes.io/arch: arm64

安全容器6.3.1环境安装DLB云产品后，创建LB服务失败

问题描述

在安全容器6.3.1环境安装DLB云产品，安全容器服务创建LB服务失败。jira：https://easystack.atlassian.net/browse/EAS-130275。

问题原因

安全容器服务先于DLB云产品安装，安全容器服务的openstack-cloud-controller-manager组件在DLB云产品安装后不会重新初始化endpoint连接，导致安全容器服务找不到DLB服务endpoint，从而创建LB服务失败。

解决办法

重启openstack-cloud-controller-manager pod。

版权所有© 北京易捷思达科技发展有限公司

10 API参考

10.1 API文档模板

一级规格（例：云主机）

二级规格（例：启动云主机）

功能介绍

说明该操作实现的功能或效果，例：启动已停止的云主机并将其状态更改为“ACTIVE”。

前提条件（可选）

若执行本操作前存在必要的前提条件，请说明；若无，则删除。

接口约束（可选）

若执行本操作存在限制或注意事项，请说明；若无，则删除。

> 注意**接口约束**与**前提条件**的区别：

> - 前提条件强调必须先做了什么才能执行本操作；
> - 接口约束强调与本操作相关的注意，例如本操作带来的重要影响，执行本操作时不宜进行的其它操作等。

URI

示例： `POST /v2.1/{project_id}/servers/{server_id}/action`

> 说明：需使用“行内代码”样式。

参数	是否必选	描述

请求消息

参数	参数类型	是否必选	描述

响应消息

参数	参数类型	描述

请求示例

```
POST https://{endpoint}/v1/{project_id}/cloudservers/delete
```

```
{
    "servers": [
        {
            "id": "616fb98f-46ca-475e-917e-2563e5a8cd19"
        }
    ],
    "delete_publicip": false,
    "delete_volume": false
   }
```

正常响应示例

```
{
    "user": {
        "name": "test_user",
        "links": {
```

版权所有© 北京易捷思达科技发展有限公司

```
            "self": "http://keystone-
api.openstack.svc.cluster.local:35357/v3/users/5df4ae79648b4d7e954382da88cc6
9ef"
        },
        "extra": {
            "user_type": "individual",
            "user_role": "domain_member"
        },
        "enabled": true,
        "user_type": "individual",
        "email": null,
        "user_role": "domain_member",
        "id": "5df4ae79648b4d7e954382da88cc69ef",
        "domain_id": "default",
        "password_expires_at": null
    }
}
```


正常响应代码

例：200

错误码

例：400，401

版权所有© 北京易捷思达科技发展有限公司

咨询热线：400-100-3070

北京易捷思达科技发展有限公司：
北京市海淀区西北旺东路10号院东区23号楼华胜天成科研大楼一层东侧120-123
南京分公司：
江苏省南京市雨花台区软件大道168号润和创智中心B栋一楼西101
上海office：
上海黄浦区西藏中路336号华旭大厦22楼2204
成都分公司：
成都市高新区天府五街168号德必天府五街WE602

邮箱：
contact@easystack.cn (业务咨询)
partners@easystack.cn(合作伙伴咨询)
marketing@easystack.cn (市场合作)
training@easystack.cn (培训咨询)
hr@easystack.cn (招聘咨询)

版权所有© 北京易捷思达科技发展有限公司