华为云 UCS

最佳实践

文档版本 01

发布日期 2025-01-14

华为云计算技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为云计算技术有限公司

地址： 贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址： https://www.huaweicloud.com/

目 录

1 权限配置

1.1 IAM 用户配置 UCS 服务权限

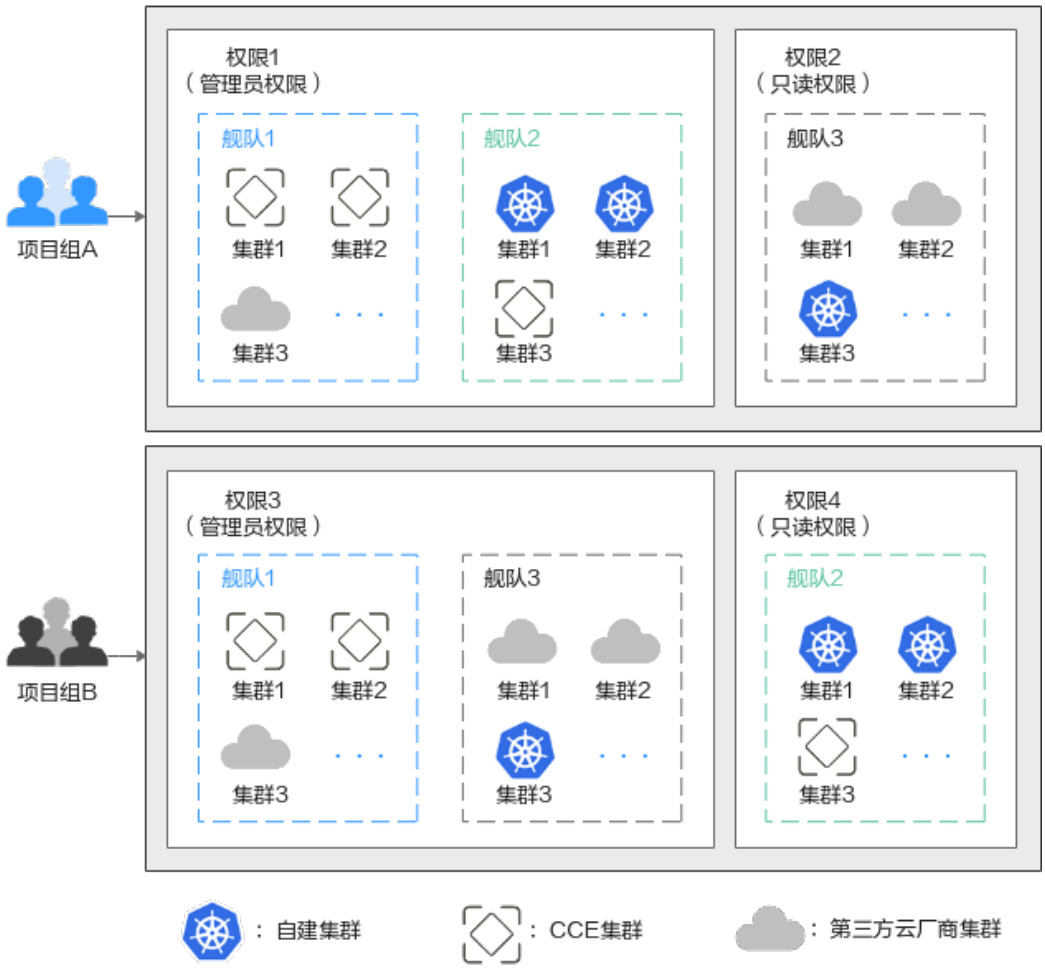
应用场景

UCS在统一身份认证服务（IAM）能力基础上，为用户提供细粒度的权限管理功能，帮助用户灵活便捷地对租户下的IAM用户设置不同的UCS资源权限，结合权限策略和舰队设计，可实现企业不同部门或项目之间的权限隔离。

例如，某公司同时推进两个项目组，每个项目组中有多名成员，权限分配如**图1 权限设计**所示。

- 项目组A在开发过程中需要舰队1、2的管理员权限以及舰队3的只读权限。
- 项目组B在开发过程中需要舰队1、3的管理员权限以及舰队2的只读权限。

图 1-1 权限设计



方案介绍

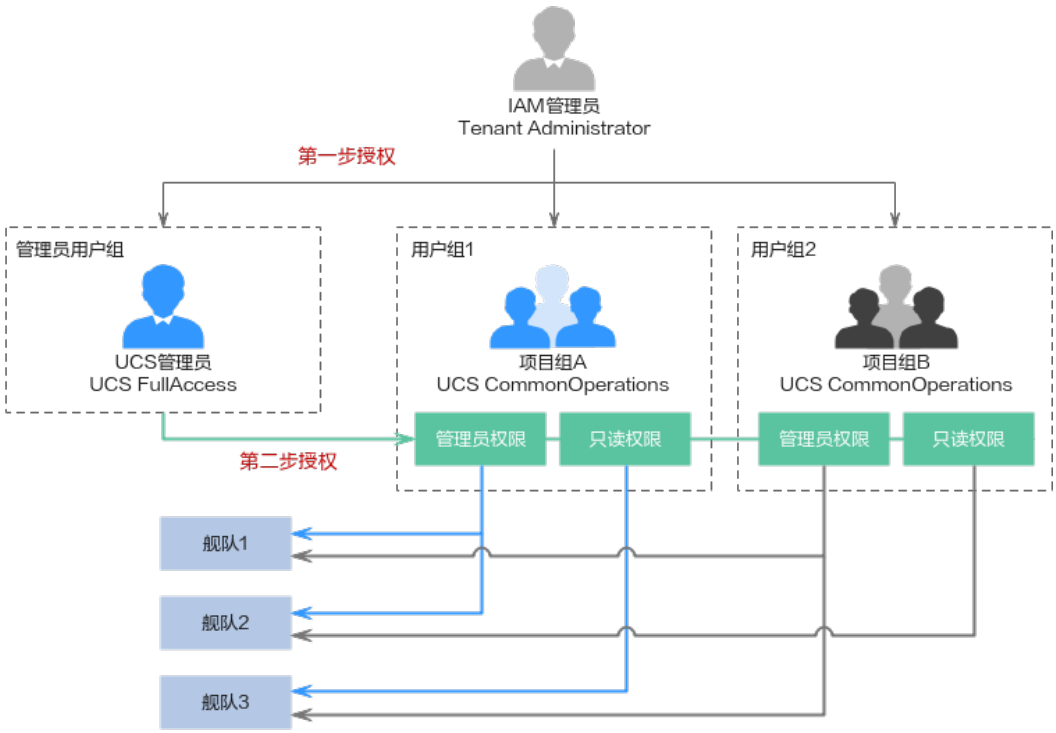
要想实现上述的权限隔离，必须结合使用IAM系统策略和UCS权限管理功能，IAM系统策略控制用户可操作哪些UCS控制台的功能，UCS权限管理控制用户可操作哪些舰队和集群资源。

如**图2 授权方案**所示，授权包括如下两大步骤。

- 第一步授权（IAM控制台）：拥有Tenant Administrator权限的IAM管理员需要创建三个用户组，一个为管理员用户组，另外两个为项目组A、B所对应的用户组（用户组1、2），分别授予UCS FullAccess和UCS CommonOperations权限。
- 第二步授权（UCS控制台）：拥有UCS FullAccess权限的UCS管理员分别为用户组1、用户组2创建各自的管理员权限、只读权限，然后关联到舰队上。

  具体的关联策略如下：用户组1的管理员权限关联至舰队1、舰队2，只读权限关联至舰队3；用户组2的管理员权限关联至舰队1、舰队3，只读权限关联至舰队2。

图 1-2 授权方案



前提条件

- 账号已开通UCS服务，并且按照图1-1完成舰队、集群资源的准备工作。
- 按照图1-2完成权限数据的准备工作。

表 1-1 IAM 控制台数据准备

用户组	用户	权限
管理员用户组：UCS_Group_admin	UCS_Group_admin_User1	UCS FullAccess
用户组1：UCS_Group_1	UCS_Group_1_User1、UCS_Group_1_User2 ...	UCS CommonOperations
用户组2：UCS_Group_2	UCS_Group_2_User1、UCS_Group_2_User2 ...	UCS CommonOperations

表 1-2 UCS 控制台数据准备

用户组	用户	权限类型	权限名称
用户组1	UCS_Group_1_User1、UCS_Group_1_User2 ...	管理员权限	ucs-group-1-admin
		只读权限	ucs-group-1-readonly

用户组	用户	权限类型	权限名称
用户组2	UCS_Group_2_User1、UCS_Group_2_User2 ...	管理员权限	ucs-group-2-admin
		只读权限	ucs-group-2-readonly

步骤一：IAM 管理员授权

步骤1 使用IAM管理员账号登录IAM控制台。

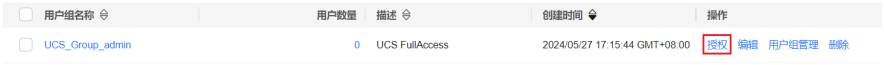
步骤2 左侧导航栏选择“用户组”，单击右上角“创建用户组”。

步骤3 在“创建用户组”界面，输入管理员用户组的名称及描述，单击“确定”，完成用户组创建。

图 1-3 创建用户组

步骤4 在用户组列表中，单击目标用户组右侧的“授权”按钮。

图 1-4 为用户组授权



步骤5 搜索并选择权限策略UCS FullAccess。

图 1-5 选择权限策略

步骤6 单击“下一步”，选择授权范围方案。

选择“所有资源”，不设置最小授权范围，用户可根据权限使用账号中所有资源，包括企业项目、区域项目和全局服务资源。

步骤7 单击“确定”完成授权。

步骤8 左侧导航栏选择“用户”，单击右上角“创建用户”，新建一个IAM用户。

填写用户名及初始密码，其余参数说明请参见**创建IAM用户**。

步骤9 单击“下一步”，选择加入**步骤4**中已授权的用户组。

图 1-6 加入用户组

步骤10 单击“创建用户”。

步骤11 重复上述步骤，完成**表1-1**中其他用户组、用户的创建和授权。

----结束

步骤二：UCS 管理员授权

步骤1 使用UCS管理员登录UCS控制台，在左侧导航栏选择“权限管理”。

步骤2 单击右上角的“创建权限”按钮。

步骤3 在弹出页面中填写权限的参数项。

- 权限名称：自定义权限的名称，需以小写字母开头，由小写字母、数字、中划线（-）组成，且不能以中划线（-）结尾。
- 用户：选择权限关联的用户，即上一步创建的IAM用户。实际应用中，一个用户组会有多个用户，创建权限时，可以将这个用户组下的所有用户全部选中，以达到批量授权的目的。
- 权限类型：选择“管理员权限”。管理员权限表示对所有集群资源对象的读写权限。

步骤4 单击“确定”，创建权限。

步骤5 权限创建完成后，可前往“容器舰队”页面，单击目标舰队右上角  按钮。

图 1-7 为舰队关联权限

步骤6 在弹出的页面单击“关联权限”，打开“修改权限”页面，将**步骤3**中创建的权限和舰队的全部命名空间关联起来。

图 1-8 修改权限

步骤7 单击“确定”。完成后，使用该IAM用户登录UCS控制台可使用权限范围内的功能。

步骤8 重复以上步骤，完成表1-2中其他权限的创建，以及权限和舰队的关联。

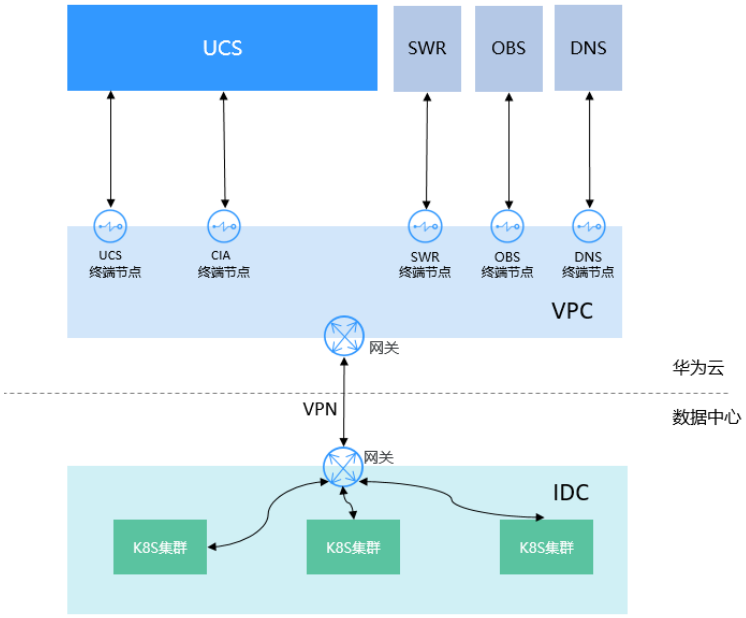
----结束

2 本地集群

2.1 创建终端节点以私网接入本地集群

应用场景

用户在线下IDC有kubernetes集群，接入到UCS开启容器智能分析服务，能够与SWR、OBS通信，在无法通过公网连接的情况下，可以先通过VPN与华为云VPC连接，然后通过VPC终端节点服务，让VPC能够在内网访问UCS、SWR、DNS、OBS、CIA。



接入前准备

服务	域名	IP（如涉及）	端口
SWR	swr.cn-north-4.myhuaweicloud.com	从VPCEP中获取。	443
OBS	op-svc-swr-b051-10-38-19-62-3az.obs.cn-north-4.myhuaweicloud.com	不涉及	443、80
CIA	cie-{容器智能分析实例instanceid前八位数字}{当前选择接入的VPC子网ID前八位数字}.cn-north-4.myhuaweicloud.com	从VPCEP中获取。	443
DNS	不涉及	创建VPCEP，选择DNS Endpoint对应的地址。	53

其他区域的SWR及依赖OBS的域名信息。

Region	SWR域名	OBS域名
华北-北京四	swr.cn-north-4.myhuaweicloud.com	op-svc-swr-b051-10-38-19-62-3az.obs.cn-north-4.myhuaweicloud.com
华东-上海二	swr.cn-east-2.myhuaweicloud.com	obs.cn-east-2.myhuaweicloud.com
华东-上海一	swr.cn-east-3.myhuaweicloud.com	op-svc-swr-b051-10-147-7-14-3az.obs.cn-east-3.myhuaweicloud.com
华南-广州	swr.cn-south-1.myhuaweicloud.com	op-svc-swr-b051-10-230-33-197-3az.obs.cn-south-1.myhuaweicloud.com

Region	SWR域名	OBS域名
西南-贵阳一	swr.cn-southwest-2.myhuaweicloud.com	op-svc-swr-b051-10-205-14-19-3az.obs.cn-southwest-2.myhuaweicloud.com
华北-乌兰察布一	swr.cn-north-9.myhuaweicloud.com	obs.cn-north-9.myhuaweicloud.com
亚太-新加坡	swr.ap-southeast-3.myhuaweicloud.com	op-svc-swr-b051-10-38-34-172-3az.obs.ap-southeast-3.myhuaweicloud.com
香港	swr.ap-southeast-1.myhuaweicloud.com	obs.ap-southeast-1.myhuaweicloud.com
拉美-墨西哥一	swr.na-mexico-1.myhuaweicloud.com	obs.na-mexico-1.myhuaweicloud.com
拉美-墨西哥二	swr.la-north-2.myhuaweicloud.com	obs.la-north-2.myhuaweicloud.com

操作步骤

步骤1 设置虚拟专用网络（VPN）方案：请参见**通过VPN连接云下数据中心与云上VPC**。

如已设置VPN网络可跳转至**在华为云侧创建VPCEP**。

说明

- 数据中心的私网网段与华为云上连接VPN使用的VPC网段不能有重叠冲突。
- 该VPC子网网段不能与IDC中已使用的网络网段重叠，否则将无法接入集群。例如，IDC中已使用的VPC子网为192.168.1.0/24，那么华为云VPC中不能使用192.168.1.0/24这个子网。

步骤2 在华为云**创建VPN网关**。

登录到华为云控制台，选择服务“虚拟专用网络 VPN”进入，左侧导航栏选择“虚拟专用网络 > 企业版-VPN网关”，单击“站点入云VPN网关”进入“站点入云VPN网关”页面，然后单击“创建站点入云VPN网关”。

表 2-1 规划数据

类别	规划项	规划值
VPC	待互通子网	10.188.1.0/24,100.64.0.0/10（该网段是云上的SWR、OBS等服务所在网段）
VPN网关	互联子网	用于VPN网关和VPC通信，不能和VPC已有子网重叠
		10.188.2.0/24
	EIP地址	EIP地址在购买EIP时由系统自动生成，无需填写，VPN网关默认使用2个EIP。本示例假设EIP地址生成如下：
		主EIP：11.xx.xx.11
		备EIP：11.xx.xx.12
VPN连接	Tunnel接口地址	用于VPN网关和对端网关建立IPSec隧道，配置时两边需要互为镜像。
		VPN连接1：169.254.70.1/30
		VPN连接2：169.254.71.1/30

步骤3 VPN创建完成后，**设置对端网关**。

左侧导航栏，选择“虚拟专用网络 > 企业版-对端网关”，在“对端网关”界面，单击“创建对端网关”。

标识选择IP Address，公网IP是数据中心侧的公网IP。

步骤4 **创建VPN连接**。

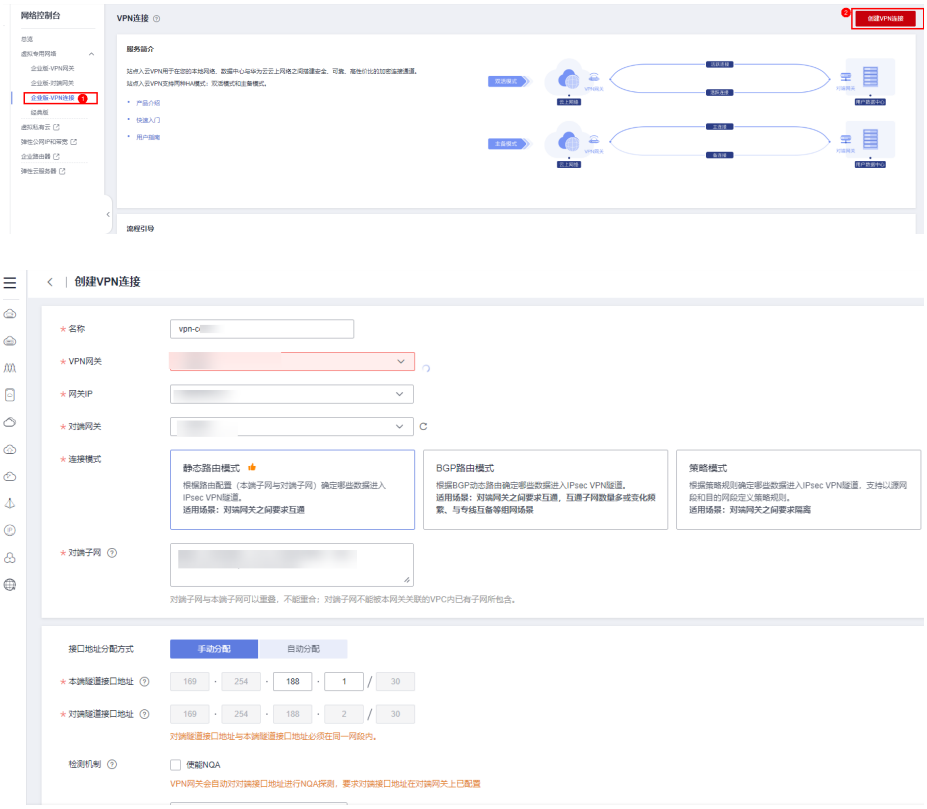


表 2-2 VPN 连接参数说明

参数	说明	参数取值
名称	输入VPN连接的名称。	vpn-xxx
VPN网关	选择**步骤**创建的VPN网关	vpngw-xxx
网关IP	选择VPN网关的主EIP。	11.xx.xx.11
对端网关	选择**步骤**创建的对端网关	cgw-xxx

参数	说明	参数取值
连接模式	选择“静态路由模式”。	静态路由模式
对端子网	输入数据中心待和VPC互通的子网。 **说明** • 对端子网可以和本端子网重叠，但不能重合。 • 对端子网不能被VPN网关关联的VPC内已有子网所包含；不能作为被VPN网关关联的VPC自定义路由表的目的地址。 • 对端子网不能是VPC的预留网段，例如100.64.0.0/10、214.0.0.0/8。 • 如果互联子网关联了ACL规则，则需要确保ACL规则中已放通所有本端子网到对端子网的TCP协议端口。	172.16.0.0/16
接口分配方式	支持“手动分配”和“自动分配”两种方式。	手动分配
本端接口地址	配置VPN网关的Tunnel隧道IP地址。 **说明** 对端网关需要对此处的本端接口地址/对端接口地址做镜像配置。	169.254.70.2/30
对端隧道接口地址	配置在用户侧设备上的tunnel接口地址。	169.254.70.1/30
检测机制	用于多链路场景下路由可靠性检测。 **说明** 功能开启前，请确认对端网关支持ICMP功能，且对端接口地址已在对端网关上正确配置，否则会导致VPN流量不通。	勾选“使能NQA”
预共享密钥、确认密钥	VPN连接协商密钥。 VPN连接和对端网关配置的预共享密钥需要一致。	Test@123
策略配置	包含IKE策略和IPsec策略，用于指定VPN隧道加密算法。 VPN连接和对端网关配置的策略信息需要一致。	默认配置

步骤5 **配置对端网关设备**。

步骤6 验证网络互通情况：

1. 登录管理控制台。
2. 单击管理控制台左上角的，选择区域和项目。
3. 单击“服务列表”，选择“计算 > 弹性云服务器”。
4. 登录弹性云服务器。
   弹性云服务器有多种登录方法，具体请参见**登录弹性云服务器**。
   本示例是通过管理控制台远程登录（VNC方式）。

5. 在弹性云服务器的远程登录窗口，执行以下命令，验证网络互通情况。

   ping 172.16.0.100

   其中，172.16.0.100为数据中心服务器的IP地址，请根据实际替换。

   回显如下信息，表示网络已通。

```
来自 xx.xx.xx.xx 的回复: 字节=32 时间=28ms TTL=245
来自 xx.xx.xx.xx 的回复: 字节=32 时间=28ms TTL=245
来自 xx.xx.xx.xx 的回复: 字节=32 时间=28ms TTL=245
来自 xx.xx.xx.xx 的回复: 字节=32 时间=27ms TTL=245
```

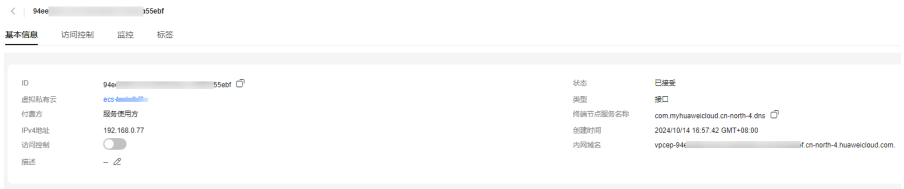
步骤7 在华为云侧创建VPCEP。

数据中心IDC访问华为云上各服务需要在与数据中心互通的VPC中创建VPCEP。需要在华为云终端节点页面分别创建DNS、SWR、OBS、UCS的终端节点：

创建DNS终端节点

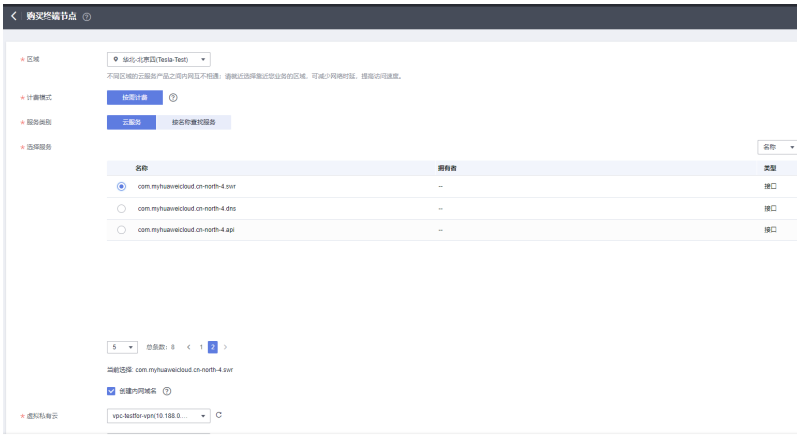
在“服务列表”中，选择“网络 > VPC终端节点 VPCEP”，进入终端节点页面。

1. 在左侧导航栏，选择“VPC终端节点 > 终端节点”。
2. 在终端节点界面，单击“购买终端节点”，创建连接DNS服务的终端节点。
3. 购买终端节点时，“服务类型”和“服务”选择“云服务 > com.myhuaweicloud.cn-north-4.dns”。
4. 虚拟私有云选择**步骤2 在华为云创建VPN网关**中进行VPN打通的VPC。
5. 单击生成的终端节点名称详情，查看生成的IP，记录。

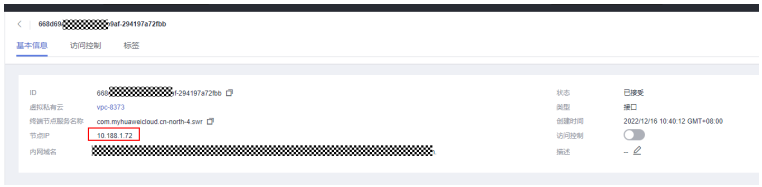


创建SWR终端节点

1. 在“服务列表”中，选择“网络 > VPC终端节点”，进入终端节点页面。
2. 在左侧导航栏，选择“VPC终端节点 > 终端节点”。
3. 在终端节点界面，单击“购买终端节点”，创建连接SWR服务的终端节点。
4. 购买终端节点时，“服务类型”和“服务”选择“云服务 > com.myhuaweicloud.cn-north-4.swr”。
5. 虚拟私有云选择**步骤2 在华为云创建VPN网关**中进行VPN打通的VPC。

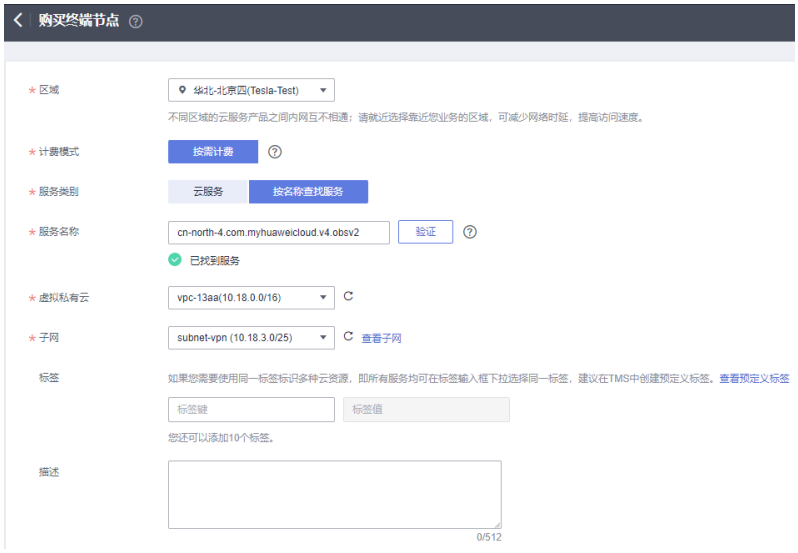


6. 单击创建出来的VPCEP节点名称，查看VPCEP的节点IP。



创建OBS终端节点

1. 在“服务列表”中，选择“网络 > VPC终端节点”，进入终端节点页面。
2. 在左侧导航栏，选择“VPC终端节点 > 终端节点”。
3. 在终端节点界面，单击“购买终端节点”，创建连接OBS服务的终端节点。
4. 购买终端节点时，“服务类型”和“服务”选择“按名称查找服务cn-north-4.com.myhuaweicloud.v4.obsv2 >”，并单击“验证”。
5. 虚拟私有云选择**步骤2**中进行VPN打通的VPC。



创建UCS终端节点

1. 在“服务列表”中，选择“网络 > VPC终端节点”，进入终端节点页面。
2. 在左侧导航栏，选择“VPC终端节点 > 终端节点”。
3. 在终端节点界面，单击“购买终端节点”，创建连接UCS服务的终端节点。
4. 购买终端节点时，“服务类型”和“服务”选择“按名称查找服务 > cn-north-4.open-vpcep-svc.29696ab0-1486-4f70-ab35-a3f6b1b37c02”，并单击“验证”。
5. 虚拟私有云选择**步骤2**中进行VPN打通的VPC。

步骤8 在IDC的DNS Server中增加华为云的DNS转发器。

1. 配置DNS转发器：在用户线下的DNS服务器配置相应的DNS转发规则，将解析华为云内网域名的请求转发到DNS终端节点。

   以常见的DNS软件Bind为例：/etc/named.conf内，增加DNS转发器的配置，forwarders为DNS终端节点IP地址。xx.xx.xx.xx是**步骤7**中DNS的终端节点IP。

   ```
   options
   {
     forward only;
     forwarders{ xx.xx.xx.xx;};
   }
   ```

2. 增加DNS静态配置，SWR与CIE实例地址，地址是从容器智能分析实例中获取到的。

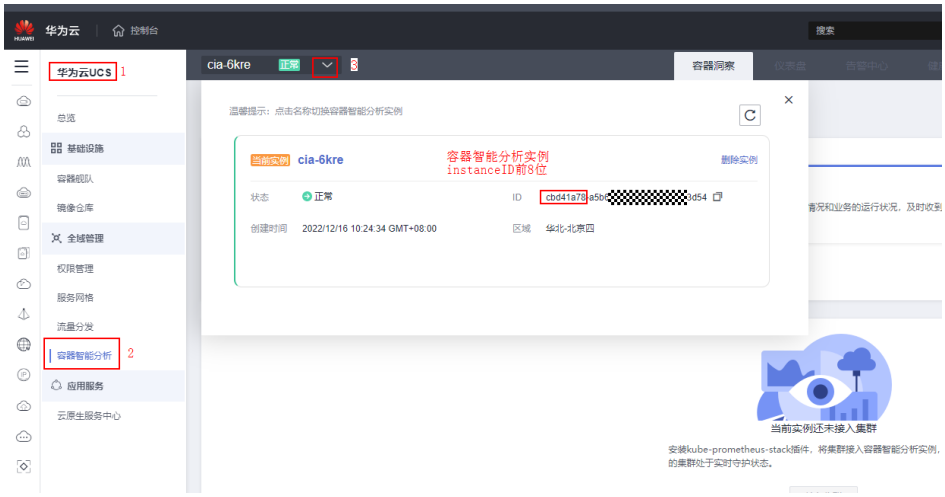
   以北京四为例，如使用dnsmasq为例，在/etc/dnsmasq.conf中添加以下静态解析。

   ```
   address=/swr.cn-north-4.myhuaweicloud.com/xx.xx.xx.xx
   ```

   *xx.xx.xx.xx*是**步骤7**中SWR的终端节点IP。

   address=/cie-{容器智能分析实例instanceid前八位数字}{当前选择接入的VPC子网ID前八位数字}.cn-north-4.myhuaweicloud.com

   获取容器智能分析实例instanceid前八位数字。

   获取当前选择接入的VPC子网ID前八位数字

步骤9 在UCS注册IDC的kubernetes集群：准备待接入集群的KubeConfig文件，请确保待接入集群的kubeconfig文件中的server字段是私网IP（非公网IP或者域名）。登录UCS控制台，在左侧树状导航栏，选择“容器舰队”。单击本地集群选项卡中的“注册集群”按钮。根据页面提示，选择集群服务商并填写集群参数。具体请参考**安装前准备**。

在完成集群添加后，集群需要终端节点来接入网络才能被UCS接管，单击私网接入，选择在与数据中心IDC打通VPN的VPC。

说明

该VPC只有在完成中的**在华为云侧创建VPCEP**配置才可以被选中。

下载集群代理agent的配置文件，上传到数据中心的kubernetes集群内，待接入集群中执行以下命令部署代理。

```
kubectl apply -f agent.yaml
```

查看集群代理部署状态。

```
kubectl -n kube-system get pod | grep proxy-agent
```

如果部署成功，预期输出如下：

```
proxy-agent-5f7d568f6-6fc4k 1/1 Running 0 9s
```

查看集群代理运行状态。

```
kubectl -n kube-system logs<Agent Pod Name>| grep "Start serving"
```

如果正常运行，日志预期输出如下：

```
Start serving
```

前往UCS控制台刷新集群状态，集群处于“运行中”。

步骤10 将在UCS下创建的待接入数据中心的kubernetes集群接入到容器智能分析服务。

1. 容器智能分析接入集群：登录UCS控制台，在左侧导航栏中单击“容器智能分析”。选择容器智能分析实例，并单击右上角“开启监控”。选择一个数据中心内的待接入附着集群，单击“下一步：接入配置”。
2. 接入方式选择“私网接入”。私网接入点：“虚拟私有云”选择已经与数据中心打通VPN的VPC。

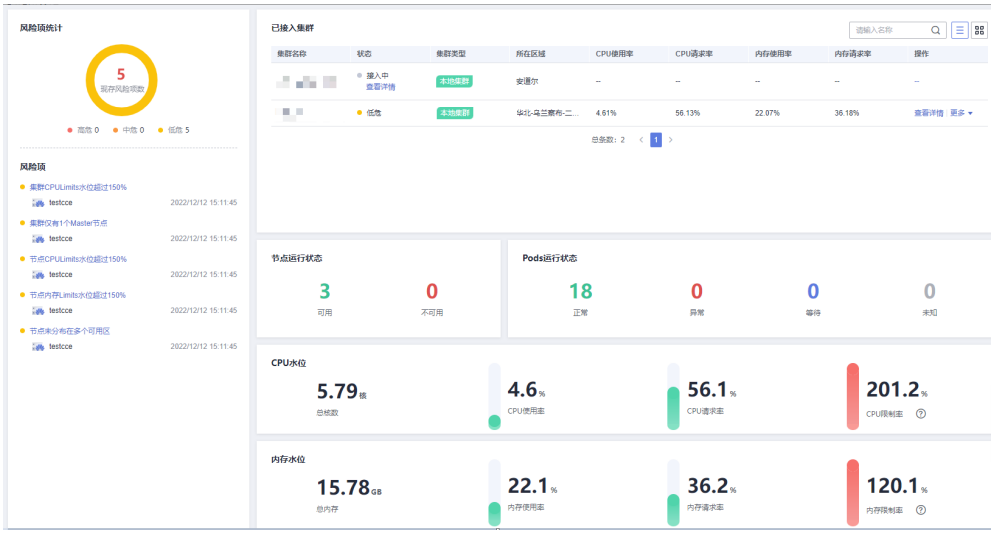
3. 完成插件配置。

   系统提供默认的插件配置，包括插件规格、采集周期和存储，如果您想修改这些默认值，请单击插件参数旁边的 ▼ 按钮，展开配置项。

   插件规格：包括演示规格（100容器以内）、不同规格对集群的CPU、内存等资源要求不同，UCS服务会对集群节点是否能够成功安装插件进行初步检测，当不满足时，会在页面给出提示。不同插件规格占用的资源配额可参考**不同规格的资源配额要求**。

   - 存储：用于普罗数据的临时存储。
   - 存储类型：附着集群支持Emptydir和Local Storage两种存储类型。
   - 使用Emptydir模式普罗数据将存储在Pod中，请确保prometheus-server-0调度到的节点上的容器存储挂载容量满足所输入的容量大小。
   - 使用本地存储将会在您的集群内创建monitoring命名空间（如果不存在），以及local-storage类型的PV及PVC，请保证您指定的节点上存在所输入的目录以及该目录满足所输入的容量大小。
   - 容量：为创建PVC时指定的容量大小或者选择Pod存储时的存储最大限制值。

等待集群接入，2-3min最终显示集群接入状态是低危/中危/高危，以及有监控数据。



----结束

2.2 使用工作负载 Identity 安全访问云服务

应用场景

工作负载Identity允许集群中的工作负载模拟IAM用户来访问云服务，从而无需直接使用IAM账号的 AK/SK 等信息，降低安全风险。

本文档介绍如何在UCS中使用工作负载Identity。

方案流程

使用工作负载Identity的流程如**图1 使用工作负载Identity流程**，具体流程如下：

步骤1 前置授权。

1. 在UCS**获取本地集群私钥签发的jwks**，该公钥用于验证集群签发的ServiceAccount Token。
2. 在 IAM **配置身份供应商**，标志当前集群在IAM侧的身份。
3. 为身份提供商配置集群签发的公钥，后续负载使用Token发送请求时，IAM使用该公钥验证Token。
4. 添加 ServiceAccount 与 IAM 账号的映射规则，配置后，当前 ServiceAccount 拥有对应用户的 IAM 权限。

步骤2 工作负载配置Token。

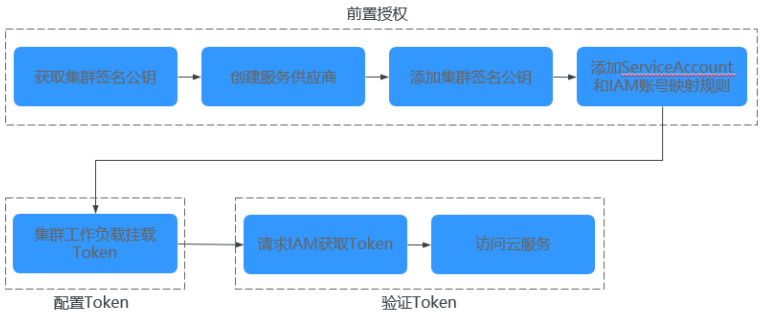
1. 部署应用并配置ServiceAccount。
2. 挂载对应ServiceAccount的Token。

步骤3 验证获取的Token能否正常进行访问。

1. 访问IAM接口获取IAM Token。
2. 使用IAMToken 访问云服务。

----结束

图 2-1 使用工作负载 Identity 流程



获取本地集群私钥签发的 jwks

步骤1 使用kubectl连接本地集群。

步骤2 执行如下命令获取公钥。

kubectl get --raw /openid/v1/jwks

返回结果为一个 json 字符串，是当前集群的签名公钥，用于访问身份提供商。

```
{
  "keys": [
    {
      "kty": "RSA",
      "e": "AQAB",
      "use": "sig",
      "kid": "Ew29q....",
```

```
        "alg": "RS256",
        "n": "peJdm...."
      }
   ]
}
```

----结束

配置身份提供商

步骤1 登录IAM控制台，创建身份提供商，协议选择OpenID Connect。

图 2-2 创建身份提供商

步骤2 单击“确定”，然后修改身份提供商信息，需要修改的信息如**表1 身份提供商配置参数说明**。若需要创建身份转换规则，单击“创建规则”进行创建。

图 2-3 修改身份提供商信息

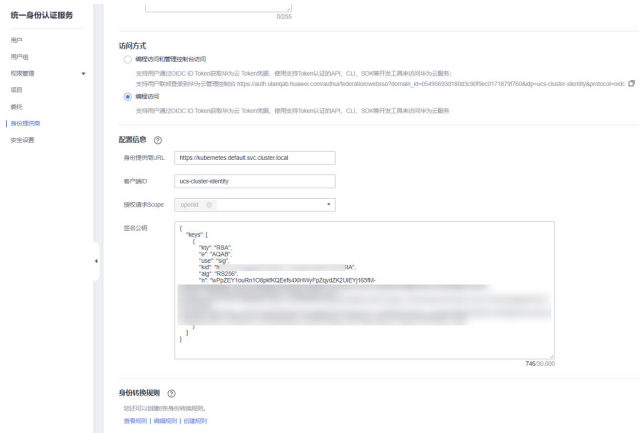


图 2-4 创建身份转换规则

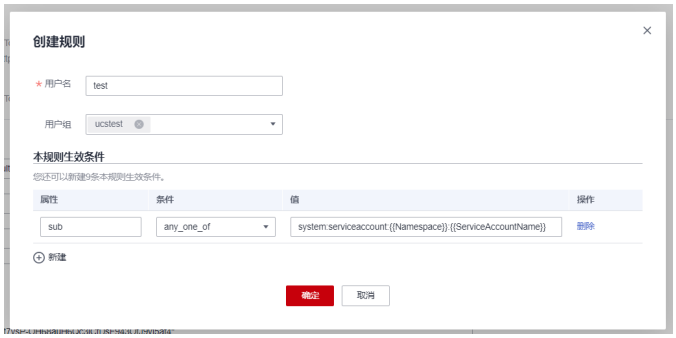


表 2-3 身份提供商配置参数说明

参数	说明
访问方式	选择“编程访问”
配置信息	• 身份提供商 URL：https://kubernetes.default.svc.cluster.local 。 • 客户端 ID：ucs-cluster-identity。 • 签名公钥：本地集群的jwks，获取方法请参见**获取本地集群私钥签发的jwks**。
身份转换规则	身份映射规则是将工作负载的 ServiceAccount 和 IAM 用户组做映射。 例如：在集群default命名空间下创建一个名为 XXX 的 ServiceAccount，映射到 demo 用户组（后续使用身份提供商 ID 访问云服务就具有 demo 用户组的权限）。 值的格式为：system:serviceaccount:*Namespace*:*ServiceAccountName*

步骤3 单击“确定”。

----结束

获取 IAM Token

步骤1 创建 ServiceAccount，此处 ServiceAccount 的名称需要与**步骤2**时填写的 ServiceAccountName 保持一致。

```
apiVersion: v1
kind: ServiceAccount
metadata:
 name: test_sa_name # 与配置身份转换规则处保持一致
```

步骤2 如下所示，在工作负载中新增 ServiceAccount 以及 Volume 相关配置。

```
apiVersion: apps/v1
kind: Deployment
```

```
metadata:
  name: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
      version: v1
  template:
    metadata:
      labels:
        app: nginx
        version: v1
    spec:
      containers:
      - name: container-1
        image: nginx:latest
        volumeMounts
        - mountPath: "/var/run/secrets/tokens" # 将Kubernetes生成的ServiceAccountToken 挂载到 /var/run/secrets/tokens/token_path 文件内
          name: token-volume
      imagePullSecrets:
      - name: default-secret
      serviceAccountName: test_sa_name   # 上一步创建的ServiceAccount的名称
      volumes:
      - name: token-volume
        projected:
          defaultMode: 420
          sources:
          - serviceAccountToken:
              audience: ucs-cluster-identity   # 此处取值必须为身份提供商的客户端ID
              expirationSeconds: 7200     # 过期时间
              path: token_path     # 路径名称，可自定义
```

步骤3 创建完成后，登录到容器中获取 Token。

步骤4 构造请求体数据，项目ID的获取请参见**获取项目ID**。

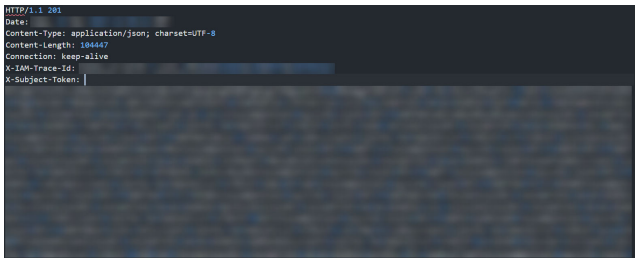
```
{
    "auth" : {
      "id_token" : {
        "id" : "eyJhbGciOiJSUzIXXXXX"   // 上一步获得的 token 内容
      },
      "scope": {
        "project" : {
          "id" : "05495693df80d3c92fa1c01795c2be02", // 项目 ID
          "name" : "cn-north-7"
        }
      }
    }
}
```

步骤5 请求IAM接口以获取IAM Token，IAM的Endpoint信息请参见**地区和终端节点**。

```
curl -i --location --request POST 'https://{{iam endpoint}}/v3.0/OS-AUTH/id-token/tokens'  --header 'X-Idp-Id: {{workload_identity}}'  --header 'Content-Type: application/json'  --data @token_body.json
```

- workload_identity为**步骤1**中注册的身份提供商名称，此示例内为 ucs-cluster-identity。
- token_body.json 为构造的请求体数据文件。

步骤6 返回体内获取IAM Token，响应消息头中 X-Subject-Token 字段即为 IAM Token。



----结束

使用 IAM Token 访问云服务

本小节以请求LTS服务为例，介绍如何使用IAM Token访问云服务。

步骤1 在使用IAM Token访问云服务前，应为用户组配置相应服务的权限。

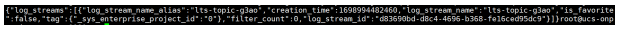
步骤2 请求LTS服务需要在用户组中加上 LTS FullAccess 权限，如图所示。

步骤3 执行如下命令，调用对应服务接口。

```
curl --location --request GET 'https://ltsperform.cn-north-7.myhuaweicloud.com/v2/{{项目 ID}}/groups/{{日志组 ID}}/streams' \--header 'Content-Type: application/json;charset=utf-8' \--header 'X-Auth-Token: {{上一步获得的 IAM token}}' \--data-raw ''
```

日志组ID可在LTS服务内进行查询。

期望的返回结果如图所示



----结束

3 集群联邦

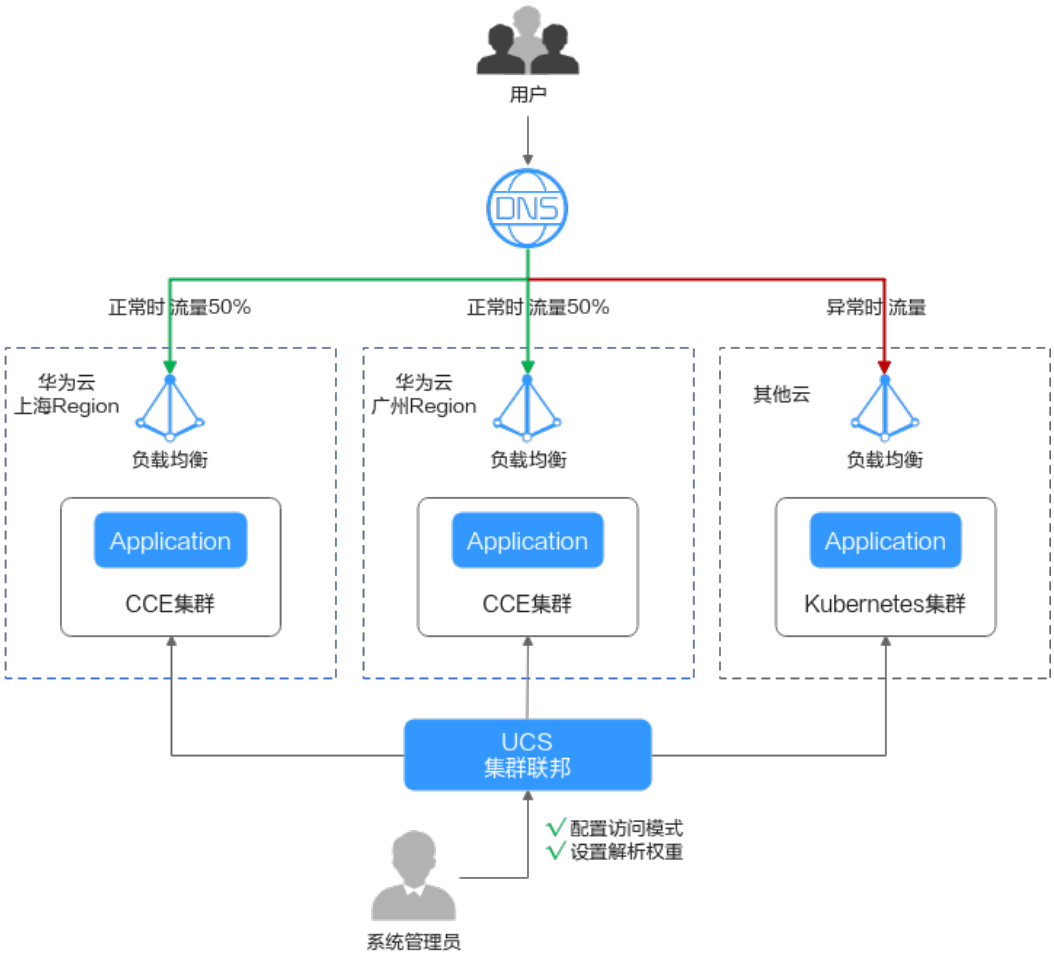
3.1 使用集群联邦实现应用多活容灾

应用场景

为了应对云单点宕机故障，UCS的集群联邦提供多云多活应用、秒级流量接管能力。业务应用的实例可以多云多活的部署在不同云上的容器服务中，当云单点宕机故障发生时，集群联邦可以秒级自动完成应用实例的弹性迁移以及流量的切换，业务的可靠性大大提升。

多活容灾方案示意如图3-1所示，通过创建域名访问规则，将应用分发到3个Kubernetes集群，包括两个华为云CCE集群（部署在不同Region）和一个其他云的Kubernetes集群，实现应用的多活容灾。

图 3-1 多云集群应用多活容灾示意图



准备工作

- 准备应用所运行的集群，本文以CCE集群为例进行演示，参考**购买CCE集群**在两个不同区域（如：华南-广州和华东-上海一）创建CCE集群，要求Kubernetes版本为1.19及以上，并且各个集群中至少拥有一个可用节点。

  说明

  在实际生产环境中，多个集群可位于不同区域、可用区，甚至不同云服务商，实现应用的多活容灾。

- 已购买公网域名，并添加至华为云云解析（DNS）服务，具体操作请参考**快速添加网站域名解析**。

基础环境搭建

步骤1 将集群注册到UCS并接入网络。具体操作请参见**注册集群**。

例如，将集群“ccecluster01”、“ccecluster02”注册到UCS的“ucs-group”容器舰队，并查看集群是否处于正常运行状态。

步骤2 为集群所在舰队开通集群联邦，并确保集群已成功接入集群联邦。具体操作请参见**集群联邦**。

图 3-2 集群管理

步骤3 创建联邦工作负载。

为展示流量切换的效果，本文中两个集群的容器镜像版本不同（实际生产环境中并不会存在此差异）。

- 集群ccecluster01：示例应用使用nginx:gz镜像，返回“ccecluster01 is in Guangzhou.”。
- 集群ccecluster02：示例应用使用nginx:sh镜像，返回“ccecluster02 is in Shanghai.”。

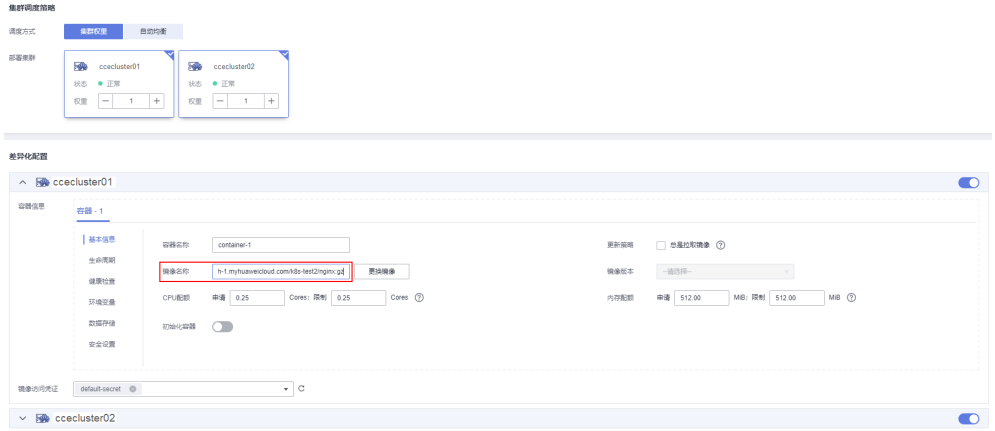
在开始操作之前，您需要将示例应用的镜像上传到对应集群所在区域的SWR镜像仓库中（也就是说，nginx:gz镜像需要上传至华南-广州，nginx:sh镜像上传至华东-上海一），否则联邦工作负载会因拉取不到镜像而异常。

说明

本文中的应用仅作示例，在实际生产环境中需替换为您的自有应用，且对集群的云服务商、区域、数量不作限制。

1. 登录UCS控制台，选择左侧导航栏中的“容器舰队”。
2. 单击已开通集群联邦的舰队名称，进入详情页面。
3. 在左侧导航栏选择“联邦管理 > 工作负载”，单击右上角“镜像创建”。
4. 填写基本信息并配置容器参数，镜像可以任意设置，单击“下一步：调度与差异化”。
5. 设置集群调度策略，完成集群差异化配置，单击“创建工作负载”。
   - 调度方式：选择“集群权重”，并设置两个集群的权重为1:1。
   - 差异化配置：单击集群左侧的图标开启差异化配置，设置集群ccecluster01的镜像名称为“swr.cn-south-1.myhuaweicloud.com/kubernetes-test2/nginx:gz”（nginx:gz镜像在SWR镜像仓库中的地址），集群ccecluster02的镜像名称为“swr.cn-east-3.myhuaweicloud.com/kubernetes-test2/nginx:sh”。

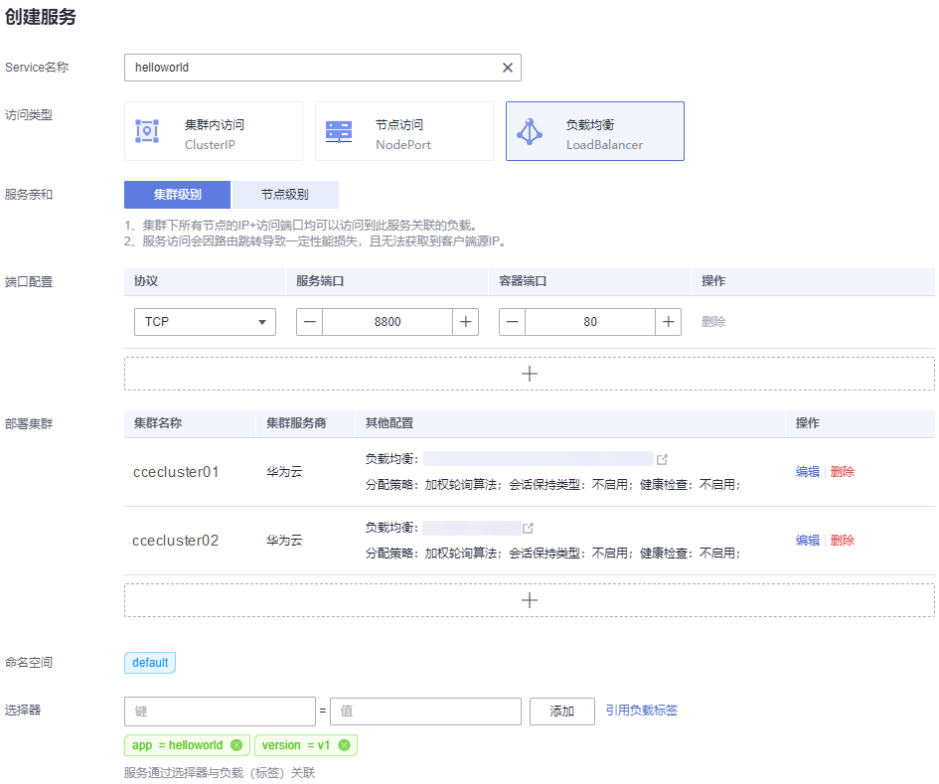
图 3-3 调度与差异化



步骤4 创建LoadBalancer访问。

1. 登录华为云UCS控制台，选择左侧导航栏中的“容器舰队”。
2. 单击已开通集群联邦的舰队名称，进入详情页面。
3. 在左侧导航栏选择“联邦管理 > 服务与路由”，单击右上角“创建服务”。
4. 完成参数填写，单击“确认”。
   - 访问类型：选择“负载均衡”。
   - 端口配置：选择TCP协议，填写服务端口、容器端口，如8800、80。
   - 部署集群：单击 ，依次添加ccecluster01和ccecluster02集群，负载均衡器选择共享型ELB实例，且必须和集群处于相同VPC中，如果列表中无可用ELB实例，单击“创建负载均衡器”前往ELB控制台进行创建。其他参数保持默认即可。
   - 选择器：服务通过选择器与负载标签关联，这里通过引用负载标签的方式来添加标签。

图 3-4 创建服务

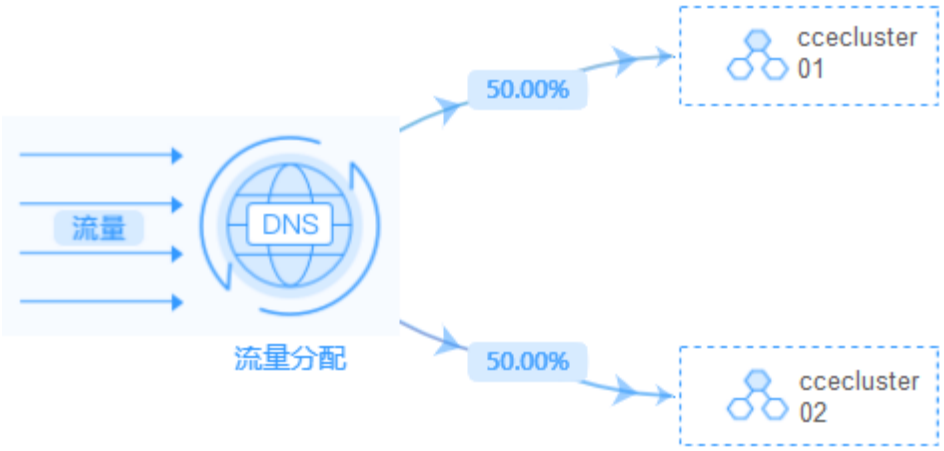


步骤5 创建域名访问。

1. 登录华为云UCS控制台，选择左侧导航栏中的“容器舰队”。
2. 单击已开通集群联邦的舰队名称，进入详情页面。
3. 在左侧导航栏选择“联邦管理 > 域名访问”，添加根域名。
4. 单击右上角“创建域名访问”，完成参数填写。
   - 目标服务：选择**步骤4**中创建的服务。
   - 流量配比模式：选择“自适应模式”，流量解析根据各集群后端实例数量自动分配权重。在本示例中，ccecluster01和ccecluster02集群的实例数均为1，那么正常情况下，两个集群将按照1:1的配比接收流量，如**图3-6**所示。

图 3-5 配置流量配比

图 3-6 流量配比拓扑图



----结束

多活容灾场景验证

按照上述集群应用部署操作，示例应用分别部署在集群“ccecluster01”和“ccecluster02”中，并以“负载均衡”类型的服务对外提供访问。**步骤5**中的域名访问创建成功后，系统自动为所选择的根域名添加解析记录，并且在UCS侧生成一个统一的对外访问路径（域名地址），因此，我们通过访问这个域名地址就可以验证流量的分配情况。

步骤1 获取域名访问地址。

1. 登录UCS控制台，选择左侧导航栏中的“容器舰队”。
2. 单击已开通集群联邦的舰队名称，进入详情页面。
3. 在左侧导航栏选择“联邦管理 > 域名访问”，列表中的“域名地址”即为域名访问地址。

图 3-7 域名地址

步骤2 在一台已连接公网的机器上执行如下命令，持续访问域名地址，查看集群应用处理状态。

- 正常情况下，两个集群上的应用均接收流量，并且各处理50%流量。

```
while true;do wget -q -O- helloworld.default.mcp-xxx.svc.xxx.co:8800; done
ccecluster01 is in Guangzhou.
ccecluster02 is in Shanghai.
ccecluster01 is in Guangzhou.
ccecluster02 is in Shanghai.
ccecluster01 is in Guangzhou.
ccecluster02 is in Shanghai.
...
```

- 当集群ccecluster01上的应用异常时（通过集群节点关机来模拟应用异常），系统将所有的流量路由到ccecluster02集群处理，用户感知不到异常。

```
while true;do wget -q -O- helloworld.default.mcp-xxx.svc.xxx.co:8800; done
ccecluster02 is in Shanghai.
ccecluster02 is in Shanghai.
ccecluster02 is in Shanghai.
```

```
ccecluster02 is in Shanghai.
ccecluster02 is in Shanghai.
ccecluster02 is in Shanghai.
...
```

返回UCS控制台，可以看到域名列表中的集群流量比例发生变化，由ccecluster02集群接管100%的流量，这与我们配置的流量配比模式以及观测到的现象均吻合。

图 3-8 域名列表

----结束

3.2 使用对等连接打通 CCE 集群网络

应用场景

在创建MCS对象前，需要保证集群间网络互通。其中，跨VPC的CCE集群间网络可以通过创建对等连接的方式打通。

本文将介绍如何通过创建对等连接的方式，为跨VPC的CCE集群打通节点间与容器间网络。

设置集群网络类型

将集群的网络类型设置为underlay以支持集群间Pod通信。支持underlay网络的CCE集群类型如下：

表 3-1 支持 underlay 网络的集群类型

CCE集群类型	网络类型	是否支持underlay网络
CCE集群	容器隧道网络	不支持
	VPC网络	支持
CCE Turbo集群	云原生网络2.0	支持

创建对等连接

步骤1 进入对等连接列表页面。

步骤2 在页面右上角区域，单击“创建对等连接”，并在弹出的对话框中，根据界面提示设置对等连接参数。参数详细说明请参见**表3-2**。

图 3-9 创建对等连接

表 3-2 创建对等连接参数说明

参数	是否必选	说明
对等连接名称	是	对等连接的名称。 由中文字符、英文字母、数字、中划线、下划线等构成，一般不超过64个字符。
本端VPC	是	本端集群的VPC，可以在下拉框中选择已有VPC。
本端VPC网段	是	本端VPC网段。
账户	是	可选择当前账户与其他账户，本例中选择当前账户。 • 当前账户：当对等连接中的对端VPC和本端VPC位于同一个账户下时，选择该项。 • 其他账户：当对等连接中的对端VPC和本端VPC位于不同账户下时，选择该项。
对端项目	是	当账户选择“当前账户”时，系统默认填充对应的项目，无需您额外操作。 比如VPC-A和VPC-B均为账户A下的资源，并且位于区域A，那么此处系统默认显示账户A下，区域A对应的项目。
对端VPC	是	对端集群的VPC，可以在下拉框中选择已有VPC。
对端VPC网段	是	对端VPC网段。 对端VPC网段不能和本端VPC网段相同或有重叠网段，否则对等连接路由可能会失效。

参数	是否必选	说明
描述	否	对该连接的描述信息。描述信息内容不能超过255个字符，且不能包含“<”和“>”。

步骤3 单击所创建的对等连接名称，进入对等连接详情页，单击“添加路由”，为对等连接添加目的地址为对端集群VPC网段的路由。

如图3-10所示，您需要填写的参数为路由中的两个“目的地址”，请参考表3-3进行配置。

图 3-10 添加路由

表 3-3 添加路由参数说明

参数	是否必选	说明
目的地址（对端）	是	对等连接另一端VPC内的地址，此处填写对端集群VPC网段。 集群VPC网段的查找方法如下： 1. 登录VPC控制台。 2. 左侧导航栏选择“虚拟私有云>我的VPC”，找到对应的对端虚拟私有云，复制其IPv4网段信息。 **图 3-11** 查找对端集群 VPC 网段 
目的地址（本端）	是	对等连接另一端VPC内的地址，此处填写本端集群VPC网段。 **注意** 请仔细检查路由中配置的目的地址信息，防止出现网段冲突。
描述	否	路由的描述信息，非必填项。 描述信息内容不能超过255个字符，且不能包含“<”和“>”。

步骤4 在对等连接详情页单击“添加路由”，为对等连接添加目的地址为对端集群容器网段的路由。

如图3-12所示，您需要填写的参数为路由中的两个“目的地址”，请参考表3-4进行配置。

图 3-12 添加路由

表 3-4 添加路由参数说明

参数	是否必选	说明
目的地址（对端）	是	对等连接另一端VPC内的地址，此处填写对端集群容器网段。 集群容器网段的查找方法如下： 1. 登录CCE控制台。 2. 单击目标集群名称，进入集群详情页，复制“网络信息>默认容器子网”中的IPv4网段信息。 **注意** 若存在多个容器网段，应为每个网段创建路由，以保证容器间的网络通信。 **图 3-13** 查找对端集群容器网段 
目的地址（本端）	是	对等连接另一端VPC内的地址，此处填写本端集群容器网段。 **注意** 请仔细检查路由中配置的目的地址信息，防止出现网段冲突。

参数	是否必选	说明
描述	否	路由的描述信息，非必填项。 描述信息内容不能超过255个字符，且不能包含“<”和“>”。

----结束

修改安全组

修改本端集群节点的安全组，在入方向规则中允许对端集群节点访问本端集群容器端口。

如图3-14所示，“协议端口”填写本端集群容器端口，“源地址”填写对端集群节点IP地址或网段。修改安全组的具体操作请参见更改集群节点的默认安全组。

图 3-14 修改安全组

验证集群间网络互通

步骤1 登录本端集群节点，执行以下命令，验证本端集群节点与对端集群节点的通信情况。

ping *对端集群节点的IP地址*

ping通则表示本端集群节点与对端集群节点间可以通信。

步骤2 进入本端集群容器，执行以下命令，验证本端集群容器与对端集群容器的通信情况。

curl *对端集群Pod的IP地址*

curl通则表示本端集群容器与对端集群容器间可以通信。

----结束

3.3 使用多集群负载伸缩扩缩工作负载

应用场景

在一些复杂的业务场景下，可能有固定时间段高峰业务，又有日常突发高峰业务，若只使用标准的FederatedHPA功能，需要足够的时间来扩展工作负载，在预期的负载峰

值可能会导致服务不可用。此种情况下，用户既期望能定时弹性伸缩应对固定时间段高峰业务，又期望能基于指标弹性伸缩应对日常突发高峰业务。联动FederatedHPA策略与CronFederatedHPA策略可实现复杂场景下的工作负载扩缩能力。

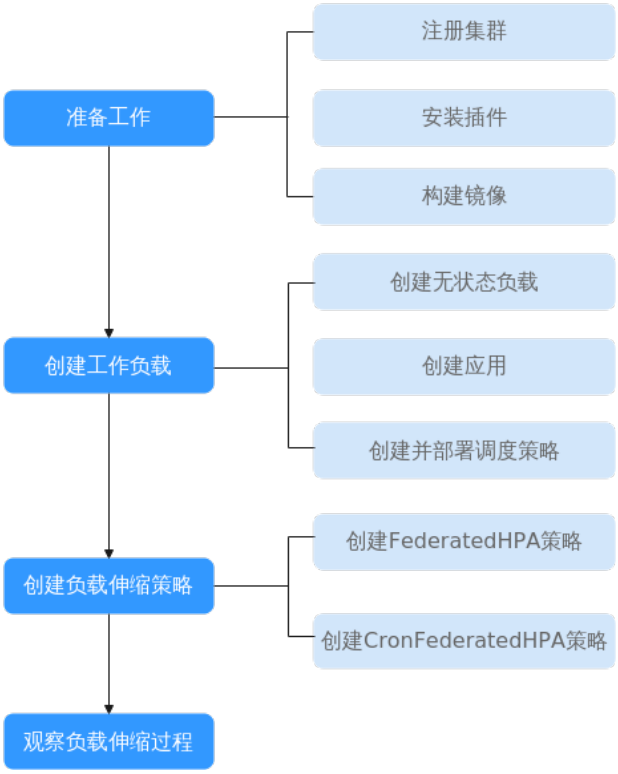
本小节将以hpa-example应用为例，指导您搭配使用FederatedHPA策略与CronFederatedHPA策略，实现复杂业务场景下的工作负载扩缩。

方案流程

使用负载伸缩策略的流程如图3-15，具体流程如下：

1. 准备工作。在创建负载伸缩策略前，您需要准备两个已注册至UCS的华为云集群，并为其安装Kubernetes Metrics Server插件，并构建一个名称为hpa-example的镜像。
2. 创建工作负载。基于准备工作中的镜像创建无状态工作负载，并配置服务，并为其创建与部署调度策略。
3. 创建负载伸缩策略。使用命令行工具创建FederatedHPA策略与CronFederatedHPA策略。
4. 观察负载伸缩过程。查看工作负载中的Pod的数量变动，观察所创建的负载伸缩策略效果。

图 3-15 复杂场景下负载伸缩策略使用流程



准备工作

- 注册两个华为云集群cluster01和cluster02。若您还未注册华为云集群，请参考**华为云集群**进行注册。
- 为集群安装Kubernetes Metrics Server插件。若未安装，请参考**Kubernetes Metrics Server**进行安装。

- 登录集群节点，准备一个算力密集型的应用。当用户请求时，需要先计算出结果后才返回给用户结果，如下所示。

  a. 创建一个名为index.php的PHP文件，文件内容是在用户请求时先循环开方1000000次，然后再返回“OK!”。

  vi index.php

  index.php文件的内容如下：

  ```
  <?php
    $x = 0.0001;
    for ($i = 0; $i <= 1000000; $i++) {
      $x += sqrt($x);
    }
    echo "OK!";
  ?>
  ```

  b. 使用如下命令编写Dockerfile制作镜像。

  vi Dockerfile

  Dockerfile的内容如下：

  ```
  FROM php:5-apache
  COPY index.php /var/www/html/index.php
  RUN chmod a+rx index.php
  ```

  c. 执行如下命令构建镜像，镜像名称为hpa-example，版本为latest。

  docker build -t hpa-example:latest .

  d. （可选）登录SWR管理控制台，在左侧导航栏选择“组织管理”，单击页面右上角的“创建组织”，创建一个组织。如已有组织可跳过此步骤。

  e. 在左侧导航栏选择“我的镜像”，单击右侧“客户端上传”，在弹出的页面中单击“生成临时登录指令”，单击 复制登录指令。

  f. 在集群节点上执行上一步中复制的登录指令，登录成功会显示“Login Succeeded”。

  g. 使用如下命令，为hpa-example镜像添加标签。

  docker tag [镜像名称1:版本名称1] [镜像仓库地址]/[组织名称]/[镜像名称2:版本名称2]

  表 3-5 标签参数说明

参数	参数说明
[镜像名称1:版本名称1]	请替换为您本地所要上传的实际镜像的名称和版本名称。
[镜像仓库地址]	请替换为5中登录指令末尾的域名。
[组织名称]	请替换为4中创建的组织名称。
[镜像名称2:版本名称2]	请替换为SWR镜像仓库中需要显示的镜像名称和镜像版本。

  命令示例如下：

  docker tag hpa-example:latest swr.cn-east-3.myhuaweicloud.com/cloud-develop/hpa-example:latest

h. 使用上传镜像至镜像仓库。

docker push [镜像仓库地址]/[组织名称]/[镜像名称2:版本名称2]

命令示例如下：

docker push swr.cn-east-3.myhuaweicloud.com/cloud-develop/hpa-example:latest

终端显示如下信息，表明上传镜像成功。

```
6d6b9812c8ae: Pushed
...
fe4c16cbf7a4: Pushed
latest: digest: sha256:eb7e3bbd*** size: **
```

i. 返回容器镜像服务控制台，在“我的镜像”页面，执行刷新操作后可查看到对应的镜像信息。

创建工作负载

步骤1 使用构建的hpa-example镜像创建无状态工作负载，Pod数为1。镜像地址与上传到的SWR仓库有关，需要替换为实际取值。

```
kind: Deployment
apiVersion: apps/v1
metadata:
  name: hpa-example
spec:
  replicas: 1
  selector:
    matchLabels:
      app: hpa-example
  template:
    metadata:
      labels:
        app: hpa-example
    spec:
      containers:
      - name: container-1
        image: 'hpa-example:latest'   # 替换为您上传到SWR的镜像地址
        resources:
          limits:                     # limits与requests建议取值保持一致，避免扩缩容过程中出现震荡
            cpu: 500m
            memory: 200Mi
          requests:
            cpu: 500m
            memory: 200Mi
      imagePullSecrets:
      - name: default-secret
```

步骤2 创建一个端口号为80的服务。

```
kind: Service
apiVersion: v1
metadata:
  name: hpa-example
spec:
  ports:
    - name: cce-service-0
      protocol: TCP
      port: 80
      targetPort: 80
      nodePort: 31144
  selector:
    app: hpa-example
  type: NodePort
```

步骤3 为工作负载和服务创建一个调度策略，并将其部署到cluster01和cluster02两个集群，使用权重拆分的方式部署，每个集群的权重为1，以保证两个集群的相同优先级。


```
apiVersion: policy.karmada.io/v1alpha1
kind: PropagationPolicy
metadata:
  name: hpa-example-pp
  namespace: default
spec:
  placement:
    clusterAffinity:
      clusterNames:
      - cluster01
      - cluster02
    replicaScheduling:
      replicaDivisionPreference: Weighted
      replicaSchedulingType: Divided
      weightPreference:
        staticWeightList:
        - targetCluster:
            clusterNames:
            - cluster01
          weight: 1
        - targetCluster:
            clusterNames:
            - cluster02
          weight: 1
  preemption: Never
  propagateDeps: true
  resourceSelectors:
  - apiVersion: apps/v1
    kind: Deployment
    name: hpa-example
    namespace: default
  - apiVersion: v1
    kind: Service
    name: hpa-example
    namespace: default
```

----结束

创建负载伸缩策略

步骤1 为工作负载创建FederatedHPA策略。

vi hpa-example-hpa.yaml

YAML文件内容如下。该策略作用于名称为hpa-example的负载，稳定窗口时长为扩容0秒、缩容100秒，最大Pod数为100、最小Pod数为2，包含一条系统指标规则，期望的CPU利用率为50%。

```
apiVersion: autoscaling.karmada.io/v1alpha1
kind: FederatedHPA
metadata:
  name: hpa-example-hpa                       # FederatedHPA策略名称
  namespace: default                        # 工作负载所在命名空间名称
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: hpa-example                       # 工作负载名称
  behavior:
    scaleDown:
      stabilizationWindowSeconds: 100           # 缩容的稳定窗口时长为100秒
    scaleUp:
      stabilizationWindowSeconds: 0             # 扩容的稳定窗口时长为0秒
  minReplicas: 2                            # 最小Pod数为2
  maxReplicas: 100                          # 最大Pod数为100
  metrics:
    - type: Resource
      resource:
```



```
      name: cpu                          # 扩缩指标基于CPU数据
      target:
        type: Utilization                # 指标类型为利用率
        averageUtilization: 50             # 期望的平均利用率
```

步骤2 创建CronFederatedHPA策略。

vi cron-federated-hpa.yaml

YAML文件内容如下。该策略作用于名称为hpa-example-hpa的FederatedHPA策略，期望每天8:30扩容工作负载至10个Pod，每天10:00缩容工作负载至2个Pod。

```
apiVersion: autoscaling.karmada.io/v1alpha1
kind: CronFederatedHPA
metadata:
  name: cron-federated-hpa                  # CronFederatedHPA策略名称
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: FederatedHPA                      # 作用于FederatedHPA策略
    name: hpa-example-hpa                     # FederatedHPA的名称
  rules:
  - name: "Scale-Up"                        # 规则名称
    schedule: 30 08 * * *                   # 触发时间
    targetReplicas: 10                      # 目标Pod数，非负整数
    timeZone: Asia/Shanghai                   # 时区
  - name: "Scale-Down"                      # 规则名称
    schedule: 0 10 * * *                    # 触发时间
    targetReplicas: 2                       # 目标Pod数，非负整数
    timeZone: Asia/Shanghai                   # 时区
```

----结束

验证负载伸缩结果

步骤1 查看FederatedHPA策略，结果显示工作负载的CPU使用率为0%。

kubectl get FederatedHPA hpa-example-hpa

```
NAME              REFERENCE                TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
hpa-example-hpa   Deployment/hpa-example   0%/50%    1         100       1          6m
```

步骤2 通过如下命令访问工作负载，其中{ip:port}为负载的访问地址，可以在工作负载的详情页中查询。

while true;do wget -q -O- http://*{ip:port}*; done

步骤3 观察工作负载自动扩容过程。

kubectl get federatedhpa hpa-example-hpa --watch

查看FederatedHPA策略，可以看到6m23s时负载的CPU使用率为200%，超过了目标值，此时触发了FederatedHPA策略，将工作负载扩容为4个Pod，随后的几分钟内，CPU使用并未下降，直到到8m16s时CPU使用率才开始下降，这是因为新创建的Pod并不一定创建成功，可能是因为资源不足Pod处于Pending状态，这段时间内在扩容节点。

到8m16s时CPU使用率开始下降，说明Pod创建成功，开始分担请求流量，到8分钟时下降到81%，还是高于目标值，在容忍度范围外，说明还会再次扩容，到9m31s时再次扩容到7个Pod，这时CPU使用率降为51%，在容忍度范围内，不会再次扩缩，因此此后Pod数量一直稳定在7个。

```
NAME              REFERENCE                TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
hpa-example-hpa   Deployment/hpa-example   0%/50%     1         100       1          6m
hpa-example-hpa   Deployment/hpa-example   200%/50%   1         100       1          6m23s
```



```
hpa-example-hpa   Deployment/hpa-example   200%/50%   1   100   4   6m31s
hpa-example-hpa   Deployment/hpa-example   210%/50%   1   100   4   7m16s
hpa-example-hpa   Deployment/hpa-example   210%/50%   1   100   4   7m16s
hpa-example-hpa   Deployment/hpa-example   90%/50%    1   100   4   8m16s
hpa-example-hpa   Deployment/hpa-example   85%/50%    1   100   4   9m16s
hpa-example-hpa   Deployment/hpa-example   51%/50%    1   100   7   9m31s
hpa-example-hpa   Deployment/hpa-example   51%/50%    1   100   7   10m16s
hpa-example-hpa   Deployment/hpa-example   51%/50%    1   100   7   11m
```

查看FederatedHPA策略事件，可以看到策略的生效时间。

kubectl describe federatedhpa hpa-example-hpa

步骤4 停止访问负载，观察工作负载自动缩容过程。

查看FederatedHPA策略，可以看到从13m开始CPU使用率为21%，18m时Pod数量缩为3个，到23m时Pod数量缩为1个。

kubectl get federatedhpa hpa-example-hpa --watch

```
NAME              REFERENCE                TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
hpa-example-hpa   Deployment/hpa-example   50%/50%    1         100       7          12m
hpa-example-hpa   Deployment/hpa-example   21%/50%    1         100       7          13m
hpa-example-hpa   Deployment/hpa-example   0%/50%     1         100       7          14m
hpa-example-hpa   Deployment/hpa-example   0%/50%     1         100       7          18m
hpa-example-hpa   Deployment/hpa-example   0%/50%     1         100       3          18m
hpa-example-hpa   Deployment/hpa-example   0%/50%     1         100       3          19m
hpa-example-hpa   Deployment/hpa-example   0%/50%     1         100       3          19m
hpa-example-hpa   Deployment/hpa-example   0%/50%     1         100       3          19m
hpa-example-hpa   Deployment/hpa-example   0%/50%     1         100       3          19m
hpa-example-hpa   Deployment/hpa-example   0%/50%     1         100       3          23m
hpa-example-hpa   Deployment/hpa-example   0%/50%     1         100       3          23m
hpa-example-hpa   Deployment/hpa-example   0%/50%     1         100       1          23m
```

查看FederatedHPA策略事件，可以看到策略的生效时间。

kubectl describe federatedhpa hpa-example-hpa

步骤5 达到CronFederatedHPA策略的触发时间后，观察工作负载的自动扩缩容过程。

可以看到118m时Pod数量扩为4个，到123m时Pod数量扩为10个。

kubectl get cronfederatedhpa cron-federated-hpa --watch

```
NAME                REFERENCE                TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
cron-federated-hpa  Deployment/hpa-example   50%/50%    1         100       1          112m
cron-federated-hpa  Deployment/hpa-example   21%/50%    1         100       1          113m
cron-federated-hpa  Deployment/hpa-example   0%/50%     1         100       4          114m
cron-federated-hpa  Deployment/hpa-example   0%/50%     1         100       4          118m
cron-federated-hpa  Deployment/hpa-example   0%/50%     1         100       4          118m
cron-federated-hpa  Deployment/hpa-example   0%/50%     1         100       4          119m
cron-federated-hpa  Deployment/hpa-example   0%/50%     1         100       7          119m
cron-federated-hpa  Deployment/hpa-example   0%/50%     1         100       7          119m
cron-federated-hpa  Deployment/hpa-example   0%/50%     1         100       7          119m
cron-federated-hpa  Deployment/hpa-example   0%/50%     1         100       7          123m
cron-federated-hpa  Deployment/hpa-example   0%/50%     1         100       10         123m
cron-federated-hpa  Deployment/hpa-example   0%/50%     1         100       10         123m
```

查看CronFederatedHPA策略事件，可以看到策略的生效时间。

kubectl describe cronfederatedhpa cron-federated-hpa

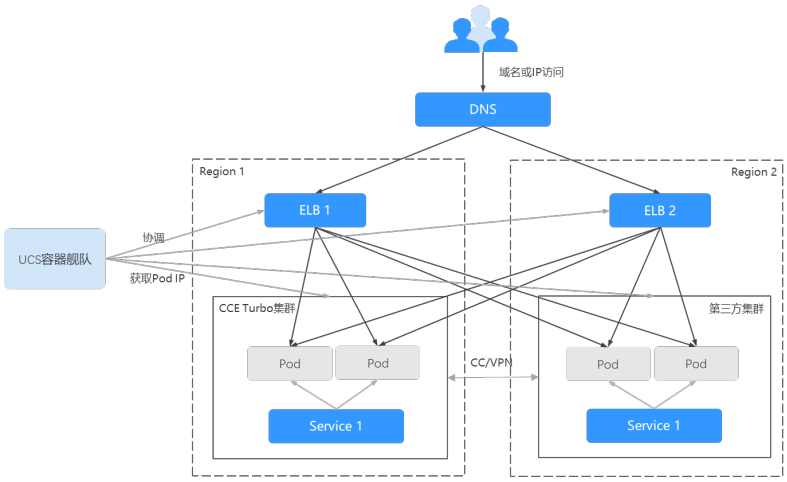
----结束

3.4 通过 MCI 实现跨集群业务流量分发

应用场景

在分布式集群场景下，为了提供低延迟的服务，企业的应用可能部署在不同区域、不同厂商的云端上，在某个地区集群发生故障时，该地区业务也随之会受到影响。使用MCI，可进行跨地域集群的流量分发，实现跨地域的应用故障迁移。

图 3-16 MCI 实现跨集群流量分发架构图



准备工作

- 准备两个部署于不同Region的CCE Turbo 1.21及以上版本集群，或者网络模型为underlay的Kubernetes集群。
- 规划应用部署的地域，并购买相应地域的ELB实例服务，为保证跨Region容灾能力，请保证两个ELB实例，跨Region部署。该ELB实例需要为独享型、支持应用型（HTTP/HTTPS）、支持私网（有私有IP地址），并且开启了跨VPC后端开关，具体创建步骤请参见**创建独享型负载均衡器**。
- 打通ELB的VPC与Kubernetes集群间的网络，确保ELB实例与容器Pod IP网络可达，并保证成员集群间网络网段不冲突。
- 准备联邦内可用的工作负载（Deployment）和服务（Service），若无请参考**无状态负载**和**集群内访问（ClusterIP）**进行创建。

通过 MCI 实现跨地域应用故障迁移

本小节以部署于两个区域的CCE Turbo集群“cce-cluster01”、“cce-cluster02”为例，通过创建绑定至多地域ELB实例的MCI对象，结合华为云提供的DNS域名解析能力，部署支持跨Region容灾的服务公网访问入口，验证应用的高可用容灾能力。

步骤1 将集群注册到UCS、接入网络并加入容器舰队，具体操作请参见**注册集群**。

步骤2 为集群所在舰队开通集群联邦，并确保集群已成功接入集群联邦。具体操作请参见**集群联邦**。

步骤3 创建联邦工作负载，并配置对应的服务。

以nginx镜像为例，将在cce-cluster01与cce-cluster-02集群上部署nginx的工作负载，并配置相应的服务。

步骤4 分别至对应的Region环境创建ELB实例。

网络配置中，开启IP类型后端（跨VPC后端）开关，VPC选择cce-cluster01所在的VPC，并新创建弹性公网IP。分别记录ELB实例1、ELB实例2的ID。

步骤5 分别获取租户的两个区域的项目ID1、项目ID2。

在华为云console控制台，单击右上角的账户名-我的凭证，查询对应区域的项目ID。

步骤6 使用kubectl连接集群联邦，具体操作请参见**使用kubectl连接集群**。

步骤7 分别创建并编辑对应两个Region的mci.yaml 文件。

创建MCI资源，文件内容定义如下所示，详细的参数定义请参见**使用MCI**。

kubectl apply -f mci.yaml

```
apiVersion: networking.karmada.io/v1alpha1
kind: MultiClusterIngress
metadata:
  name: nginx-ingress-region1
  namespace: default
  annotations:
    karmada.io/elb.id: xxxxxxx # Region1的ELB实例ID
    karmada.io/elb.port: "80" # Region1的ELB实例监听器端口
    karmada.io/elb.projectid: xxxxxxx # Region1的租户项目ID
    karmada.io/elb.health-check-flag: "on" #开启健康检查，实现故障切流
```

```
spec:
  ingressClassName: public-elb
  rules:
  - host: demo.localdev.me
    http:
      paths:
      - backend:
          service:
            name: nginx
            port:
              number: 8080
        path: /
        pathType: Prefix
---
apiVersion: networking.karmada.io/v1alpha1
kind: MultiClusterIngress
metadata:
  name: nginx-ingress-region2
  namespace: default
  annotations:
    karmada.io/elb.id: xxxxxxx # Region2的ELB实例ID
    karmada.io/elb.port: "801" # Region2的ELB实例监听器端口
    karmada.io/elb.projectid: xxxxxxx # Region2的租户项目ID
    karmada.io/elb.health-check-flag: "on" #开启健康检查，实现故障切流
spec:
  ingressClassName: public-elb
  rules:
  - host: demo.localdev.me
    http:
      paths:
      - backend:
          service:
            name: nginx
            port:
              number: 8080
        path: /
        pathType: Prefix
```

步骤8 检查ELB监听器后端是否正常挂载后端服务器组、后端实例是否运行正常，健康检查是否正常。

注意

请提前放开容器的安全组。以CCE Turbo集群为例，请在集群总览页面>网络信息>默认容器子网安全组中放开其他地域的ELB实例的网段。

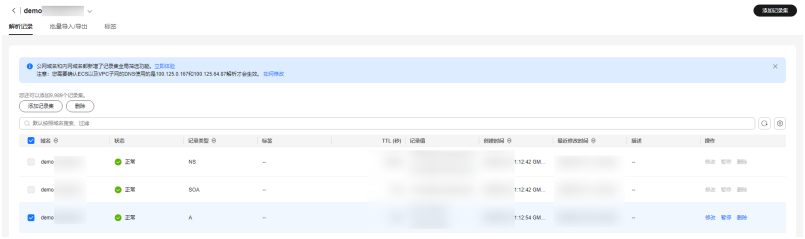
----结束

配置 DNS 访问

本文以华为云的内网DNS为例，您也可自行配置DNS。

步骤1 创建内网DNS，在ECS上通过公网的方式访问服务，ECS请先绑定EIP或者NAT配置公网出口。

- 创建与ECS相同VPC的内网域名，该域名为MCI中指定的域名。
- 将两个ELB实例的公网IP分别添加至集群的记录集。



步骤2 在ECS上通过域名curl demo.localdev.me访问对应的服务，查询返回，返回200为正常。

----结束

跨地域应用故障迁移验证

示例应用分别部署在集群“ccecluster-01”和“ccecluster-02”中，并以公网EIP的方式提供了服务的访问入口。

故障场景构造

构造单地域故障的场景，以Region1故障为例，执行以下操作，构造单地域故障：

步骤1 休眠Region1的cce-cluster01集群，并关机集群下的节点。

步骤2 解绑Region1的ELB实例的EIP1。

----结束

容灾能力验证

步骤1 在DNS的域名解析页面，在记录集中手动删除Region1的ELB实例绑定的ELB IP地址。

步骤2 检查ELB的实例后端是否显示存在健康检查结果异常的后端服务器。

步骤3 在ECS上访问对应的服务，检查服务是否访问正常，返回结果是否为200。

----结束

3.5 UCS 双集群高可用部署

应用场景

大企业场景提供多集群多活方案，做小故障域降低逻辑层面故障的风险，提供原有生态的兼容，最大限度降低在业务发布、运维等方面的适配工作量。

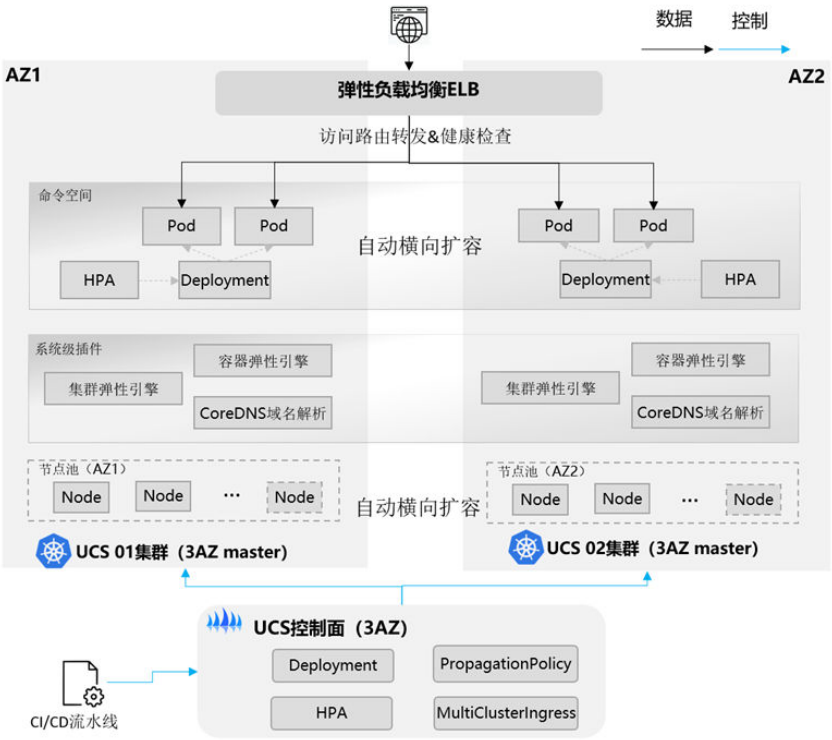
通过UCS提供双集群多活容灾，可以确保在任何一个可用区或集群发生故障时，不影响服务整体可用性。

约束限制

- 您需要拥有至少两个Kubernetes版本为1.21及以上的可用CCE turbo集群（如下文中ucs01与ucs02），并且集群分布在不同的AZ。

方案架构简介

- UCS控制面：
  - UCS控制面3AZ部署：UCS当前默认多AZ部署，使用集群联邦，详情可参见**开通集群联邦**。
  - UCS控制面负责管理ucs01与ucs02集群，集群加入容器舰队，详请可参见**管理容器舰队**。
- CCE集群：
  - 2个CCE turbo集群，集群控制面节点3AZ部署，创建集群详情可参见**购买Standard/Turbo集群**；
  - 集群计算节点分别创建AZ1、AZ2节点池，创建节电池详情可参见**创建节点池**。
  - 集群内分别安装**集群弹性引擎（Cluster AutoScaler）**，**容器弹性引擎（HPA Controller）**、**域名解析（CoreDNS）**等插件，详情可参见**插件概述**。
- 弹性负载均衡ELB：
  - ELB实例多AZ部署，详情可参见**ELB资源使用多AZ部署**。
  - ELB将访问流量根据路由策略分发到后端多个Pod实例，同时结合**健康检查**功能，流量只分发后端正常工作的Pod实例，详情可参见**配置流量分配策略分发流量**。
- 应用部署：通过CICD流水线对接UCS控制面，创建应用deployment、弹性伸缩策略HPA以及分发策略PropagationPolicy均衡部署到双集群，并通过**MultiClusterIngress**发布应用，详情可参见**使用kubectl命令实现UCS高可用部署操作步骤**。



容器级容错实施建议

容器级容错旨在通过配置健康检查和自动重启机制，确保容器应用的高可用性和可靠性。应用部署需要遵守以下规范：

项目	描述	说明
应用无状态化	应用必须做无状态和幂等处理	服务的无状态化是部署的多个服务模块(进程)，使其完全对等。也就是部署多个Pod实例，请求到任一实例的处理结果是一样的。这些Pod实例不存储业务的上下文信息，比如session、登录、业务上下文相关的信息。只会根据每次请求携带的数据进行相应的业务处理。 幂等指的是使用相同的参数多次调用相同的API,对后端产生的影响是一致的。
应用副本数	每个应用负载Deployment实例数满足业务容量规划和可用性要求。 • 必须：每个负载的实例数不小于2。 • 建议：4个实例，每个集群有2个实例。	配合多集群方案，满足生产中应用高可用性要求。为集群级别和可用区（AZ）级别故障域隔离创造条件。

项目	描述	说明
应用健康检查	每个应用必须配置： ● 启动探针（Startup Probe）：适用于启动时间较长的应用，可以避免在应用尚未完全启动时就进行就绪检查。 ● 存活探针（Liveness Probe）：用于检测容器是否还在运行，如果失败，Kubernetes会重启容器。 ● 就绪探针（Readiness Probe）：用于确定容器是否已经准备好接受流量，如果失败，该容器将不会被认为是“就绪状态”，从而不会接收到服务流量。 配置合理的检查间隔和超时： ● 设置合理的periodSeconds（检查间隔）和timeoutSeconds（超时时间），以平衡检查频率和系统负载。 ● 对于启动探针，可以设置较长的间隔和超时，以适应应用的启动时间。	容器出现故障或无法正常工作，系统可以自动重启该容器，从而提高应用的可用性和可靠性。
弹性伸缩	业务支持自动扩缩容的能力即配置指标弹性HPA，并且要求： ● 必须：HPA minReplicas不小于2，HPA maxReplicas大于等于minReplicas。 ● 建议：HPA minReplicas值为4	业务按需使用资源，最大程度地减少资源浪费。在业务流量突增以及集群级故障时，应用能够自动扩容，保障业务不受损
优雅停机	应用必须支持优雅停机	优雅停机是在对应用进程发送停止指令之后，能保证正在执行的业务操作不受影响。应用接收到停止指令之后的步骤应该是，停止接收访问请求，等待已经接收到的请求处理完成，并能成功返回，这时才真正停止应用。

项目	描述	说明
ELB健康检查	弹性负载均衡ELB流量分发正常工作的后端。 ● 必须：应用提供健康检查接口，并在ELB上配置健康检查	在个别实例异常、节点异常或者整个AZ、整个集群故障时，能快速地隔离故障实例，保证业务访问成功率。
容器镜像	容器镜像体积最大应不超过1G。容器镜像标签使用具体的版本号。 ● 必须：容器镜像标签禁止使用latest	体积小的镜像有利于分发、快速启动；镜像使用具体的版本号才能做版本控制。
资源配额	资源配额应为资源申请量的两倍数值	预防应用升级、应用扩容场景时，因资源配额不足导致失败的情况。

节点级容错配置建议

节点级容错是指当某个节点发生故障时，可以将Pod自动重新调度到其他健康节点上。

项目	描述	说明
节点故障自动驱逐	当节点出现异常，变为不可用状态时，容器将在该容忍时间后自动驱逐，默认为300s。默认对所有的容器生效，用户也可以为指定pod进行差异化容忍配置，此时将以Pod配置的容忍时长为准。	无特殊需求建议保持默认配置，容忍时间配置过小可能导致容器在网络抖动等一些短时故障场景下频繁迁移影响业务，容忍时间配置过大可能导致容器在节点故障时长时间无法迁移导致业务受损。
集群节点弹性	节点弹性伸缩，也就是资源层面的弹性伸缩。CA(Cluster AutoScaling)会检查所有Pending状态的Pod，根据用户配置的扩缩容策略，选择出一个最合适的节点池进行扩容。	当集群资源不够时需要CA扩容节点，使得集群有足够资源；而当HPA缩容后集群会有大量空余资源，这时需要CA缩容节点释放资源，才不至于造成浪费。自动扩容支撑单边资源量，CA的上限应设置为支撑所有流量所需资源量；

通过 kubectl 命令恢复集群级/AZ 级故障（可选）

集群关键系统组件出现故障或者集群升级策略不当、升级配置有误、操作人员执行有误等人为因素导致集群整体不可用或者出现AZ站点级别的故障时，UCS提供手动切流的能力。通过创建Remedy对象将MultiClusterIngress流量从故障集群上摘除。

通过Kubectl命令恢复故障步骤如下：

步骤1 使用kubectl连接集群联邦，详细操作请参见**通过kubectl连接集群**。

步骤2 集群故障后，在执行机上创建并编辑remedy.yaml文件，文件内容如下所示，参数定义请参见**表3-6**。

vi remedy.yaml

示例YAML定义了一个Remedy对象，触发条件为空，表示无条件触发，集群联邦控制器会立即将ucs01上的流量摘除。在集群故障恢复后，删除该Remedy对象，ucs01上的流量会自动恢复，由此保证单集群的故障不会影响服务的可用性。

```
apiVersion: remedy.karmada.io/v1alpha1
kind: Remedy
metadata:
  name: foo
spec:
  clusterAffinity:
    clusterNames:
      - ucs01
  actions:
  - TrafficControl
```

表 3-6 Remedy 参数说明

参数	描述
spec.clusterAffinity.clusterNames	策略关注的集群名列表。仅在该列表中的集群会执行指定动作，为空时不会执行任何动作。
spec.decisionMatches	触发条件列表。当上述集群列表中指定的集群满足任一触发条件时，即会执行指定动作。当列表为空时，表示无条件触发。
conditionType	触发条件的类型。当前仅支持ServiceDomainNameResolutionReady类型，即CPD上报的CoreDNS域名解析状态。
operator	判断逻辑，仅支持Equal和NotEqual两种值，即等于和不等于。
conditionStatus	触发条件的状态。
actions	策略要执行的动作，目前仅支持TrafficControl，即流量控制。

步骤3 集群故障恢复后，删除该Remedy对象。

```
kubectl delete remedy foo
```

步骤4 检查集群ucs01上的流量已自动恢复，手动切流成功。

----结束

使用 kubectl 命令实现 UCS 高可用部署操作步骤

前置条件：

- 使用kubectl连接集群联邦，详细操作请参见**通过kubectl连接集群**。
- 已创建可使用的独享性ELB实例，并绑定弹性公网，详情可参见购买**独享型负载均衡器**。

说明

以下均为示例yaml，请根据实际情况修改参数内容。

实践操作操作步骤：

使用UCS高可用部署，需要进行指定资源下发规则，示例如下yaml:

```
apiVersion: policy.karmada.io/v1alpha1
kind: ClusterPropagationPolicy
metadata:
  name: karmada-global-policy # 策略名
spec:
  resourceSelectors: # 分发策略关联的资源,支持同时分发多个资源对象
  - apiVersion: apps/v1 # group/version
    kind: Deployment # 资源类型kind
  - apiVersion: apps/v1
    kind: DaemonSet
  - apiVersion: v1
    kind: Service
  - apiVersion: v1
    kind: Secret
  - apiVersion: v1
    kind: ConfigMap
  - apiVersion: v1
    kind: ResourceQuota
  - apiVersion: autoscaling/v2
    kind: HorizontalPodAutoscaler
  - apiVersion: autoscaling/v2beta2
    kind: HorizontalPodAutoscaler
  - apiVersion: autoscaling.cce.io/v2alpha1
    kind: CronHorizontalPodAutoscaler
  priority: 0 # 数值越大，优先级越高
  conflictResolution: Overwrite # ConflictResolution声明当正在传播的资源已存在于目标集群中时， 默认为“Abort”，这意味着停止传播以避免意外覆盖。Overwrite则表示强制覆盖。
  placement: # 放置规则即把关联资源分发到哪些集群
    clusterAffinity: # 配置集群亲和性
      clusterNames: # 使用集群名选择集群
      - ucs01 # 集群名：需要修改为环境中实际集群名
      - ucs02 # 集群名：需要修改为环境中实际集群名
    replicaScheduling: # 实例调度策略
      replicaSchedulingType: Divided # 实例拆分
      replicaDivisionPreference: Weighted # 根据权重拆分
      weightPreference: # 权重选项
        staticWeightList: # 静态权重表： ucs01的权重为1（分配到大约1/2的实例数），ucs02权重为1 （分配到大约1/2的实例数）
        - targetCluster: # 目标集群
            clusterNames:
            - ucs01 # 集群名：需要修改为环境中实际集群名
          weight: 1 # 权重为1
        - targetCluster: # 目标集群
            clusterNames:
             - ucs02 # 集群名：需要修改为环境中实际集群名
          weight: 1 # 权重为1
    clusterTolerations: # 集群容忍，当集群master不健康或不可达时，应用不作驱逐处理
    - key: cluster.karmada.io/not-ready
      operator: Exists
      effect: NoExecute
    - key: cluster.karmada.io/unreachable
      operator: Exists
      effect: NoExecute
```

若创建了HPA，需要拆分hpa中的最小实例数，可使用如下示例yaml（可选）：

说明

以下yaml中clusterNum为集群数量，示例集群数量为2，请根据实际场景配置。

```
apiVersion: config.karmada.io/v1alpha1
kind: ResourceInterpreterCustomization
metadata:
  name: hpa-min-replica-split-ric
spec:
  customizations:
    replicaResource:
      luaScript: |
```



```
      function GetReplicas(obj)
        clusterNum = 2
        replica = obj.spec.minReplicas
        if ( obj.spec.minReplicas == 1 )
        then
          replica = clusterNum
        end
        return replica, nil
      end
  replicaRevision:
    luaScript: |
      function ReviseReplica(obj, desiredReplica)
        obj.spec.minReplicas = desiredReplica
        return obj
      end
target:
  apiVersion: autoscaling/v2
  kind: HorizontalPodAutoscaler
```

创建configmap实例，示例如下yaml:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: demo-configmap
  namespace: default       #命名空间，默认为default
data:
  foo: bar
```

创建deployment实例，示例如下yaml:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: demo
  namespace: default       #命名空间，默认为default
  labels:
    app: demo
spec:
  replicas: 2
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
  selector:
    matchLabels:
      app: demo
  template:
    metadata:
      labels:
        app: demo
    spec:
      affinity:
        podAntiAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
          - labelSelector:
              matchExpressions:
              - key: app
                operator: In
                values:
                - demo
            topologyKey: kubernetes.io/hostname
      containers:
      - env:
        - name: POD_NAME
          valueFrom:
            fieldRef:
              fieldPath: metadata.name
        - name: POD_NAMESPACE
```



```
        valueFrom:
          fieldRef:
            apiVersion: v1
            fieldPath: metadata.namespace
      - name: POD_IP
        valueFrom:
          fieldRef:
            apiVersion: v1
            fieldPath: status.podIP
      envFrom:
      - configMapRef:
          name: demo-configmap
      name: demo
      image: nginx    #若使用“开源镜像中心”的镜像，可直接填写镜像名称；若使用“我的镜像”中的镜像，请在SWR中获取具体镜像地址。
      command:
        - /bin/bash
      args:
        - '-c'
        - 'sed -i "s/nginx/podname: $POD_NAME podIP: $POD_IP/g" /usr/share/nginx/html/index.html;nginx "-g" "daemon off;"'
      imagePullPolicy: IfNotPresent
      resources:
        requests:
          cpu: 100m
          memory: 100Mi
        limits:
          cpu: 100m
          memory: 100Mi
```

创建hpa实例，示例如下yaml:

```
apiVersion: autoscaling/v2beta2
kind: HorizontalPodAutoscaler
metadata:
  name: demo-hpa
  namespace: default       #命名空间，默认为default
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: demo
  minReplicas: 2
  maxReplicas: 4
  metrics:
    - type: Resource
      resource:
        name: cpu
        target:
          type: Utilization
          averageUtilization: 30
  behavior:
    scaleDown:
      policies:
      - type: Pods
        value: 2
        periodSeconds: 100
      - type: Percent
        value: 10
        periodSeconds: 100
      selectPolicy: Min
      stabilizationWindowSeconds: 300
    scaleUp:
      policies:
      - type: Pods
        value: 2
        periodSeconds: 15
      - type: Percent
        value: 20
        periodSeconds: 15
```



```
    selectPolicy: Max
    stabilizationWindowSeconds: 0
```

创建service实例，示例如下yaml:

```
apiVersion: v1
kind: Service
metadata:
  name: demo-svc
  namespace: default    #命名空间，默认为default
spec:
  type: ClusterIP
  selector:
    app: demo
  sessionAffinity: None
  ports:
    - name: http
      protocol: TCP
      port: 8080
      targetPort: 8080
```

创建mci实例，示例如下yaml:

```
apiVersion: networking.karmada.io/v1alpha1
kind: MultiClusterIngress
metadata:
  name: demo-mci # MCI的名字
  namespace: default       #命名空间，默认为default
  annotations:
    karmada.io/elb.id: xxx # TODO: ELB实例ID
    karmada.io/elb.projectid: xxx #TODO: ELB实例的项目ID
    karmada.io/elb.port: "8080" #TODO: ELB监听端口
    karmada.io/elb.health-check-flag: "on"
    karmada.io/elb.health-check-option.demo-svc: '{"protocol":"TCP"}'
spec:
  ingressClassName: public-elb  # ELB类型，固定值
  rules:
  - host: demo.localdev.me # 对外暴露的域名 TODO：修改实际地址
    http:
      paths:
      - backend:
          service:
            name: demo-svc # 暴露的service名字
            port:
              number: 8080  # 暴露service端口
        path: /
        pathType: Prefix # 前缀匹配
```

验证双集群高可用业务

用户在执行机上执行如下命令，验证双集群高可用业务:

- 获取HOSTNAME与ELBIP
  ```
  kubectl get mci demo-mci -oyaml
  ```



```
[root@ecs-ab7b test]# kubectl --kubeconfig=/[redacted]et mci demo-mci -oyaml
apiVersion: networking.karmada.io/v1alpha1
kind: MultiClusterIngress
metadata:
  annotations:
    karmada.io/elb.health-check-flag: "on"
    karmada.io/elb.health-check-option.demo-svc: '{"protocol":"TCP"}'
    karmada.io/elb.id: [redacted]
    karmada.io/elb.port: "8080"
    karmada.io/elb.projectid: [redacted]f
    kubectl.kubernetes.io/last-applied-configuration: |
      {"apiVersion":"networking.karmada.io/v1alpha1","kind":"MultiClusterIngress","metadata":{"annotations":{"k
ol\":\"TCP\"}","karmada.io/elb.id":"[redacted]0","karmada.io/elb.port":"8080","karmada
},"spec":{"ingressClassName":"public-elb","rules":[{"host":"demo.localdev.me","http":{"paths":[{"backend":{"serv
  creationTimestamp: "[redacted]0:44Z"
  finalizers:
  - karmada.io/multi-cluster-ingress-controller
  generation: 1
  name: demo-mci
  namespace: default
  resourceVersion: [redacted]SJ9
  uid: [redacted]e1
spec:
  ingressClassName: public-elb
  rules:
  - host: demo.localdev.me
    http:
      paths:
      - backend:
          service:
            name: demo-svc
            port:
              number: 80
        path: /
        pathType: Prefix
status:
  loadBalancer:
    ingress:
    - hostname: demo.localdev.me
      ip: [redacted]97
    - ip: 16[redacted]4
  serviceLocations:
  - clusters:
    - turbo-123-1
    - turbo-123-2
    name: demo-svc
[root@ecs-ab7b test]#
```

- 多次访问业务，回显不同PODNAME和PODID，表示实现双集群访问成功

  curl -H "host:demo.localdev.me" http://[ELBIP]:8080/

```
[root@ecs-ab7b test]# curl -H "host:demo.localdev.me" http://[redacted]:8080/
<!DOCTYPE html>
<html>
<head>
<title>Welcome to podname: demo-[redacted]782 podIP: [redacted]214!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to podname: d[redacted]2 podIP: [redacted]214!</h1>
<p>If you see this page, the podname: [redacted]782 podIP: [redacted]214 web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://podname: [redacted]782 podIP: [redacted].214.org/">podname: [redacted]7782 podIP: [redacted]214.org</a>.<br/>
Commercial support is available at
<a href="http://podname: [redacted]782 podIP: [redacted]214.com/">podname: [redacted]7782 podIP: [redacted]214.com</a>.</p>

<p><em>Thank you for using podname: [redacted]7782 podIP: [redacted]214.</em></p>
</body>
</html>
[root@ecs-ab7b test]#
```

```
[root@ecs-ab7b test]# curl -H "host:demo.localdev.me" http://[redacted]:8080/
<!DOCTYPE html>
<html>
<head>
<title>Welcome to podname: [redacted]-r9jhc podIP: [redacted]178!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to podname: [redacted]9jhc podIP: [redacted].178!</h1>
<p>If you see this page, the podname: [redacted]9jhc podIP: [redacted]178 web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://podname: [redacted]9jhc podIP: [redacted].178.org/">podname: [redacted]r9jhc podIP: [redacted].178.org</a>.<br/>
Commercial support is available at
<a href="http://podname: [redacted]r9jhc podIP: [redacted]178.com/">podname: [redacted]9jhc podIP: [redacted]178.com</a>.</p>

<p><em>Thank you for using podname: [redacted]-r9jhc podIP: [redacted].178.</em></p>
</body>
</html>
[root@ecs-ab7b test]#
```

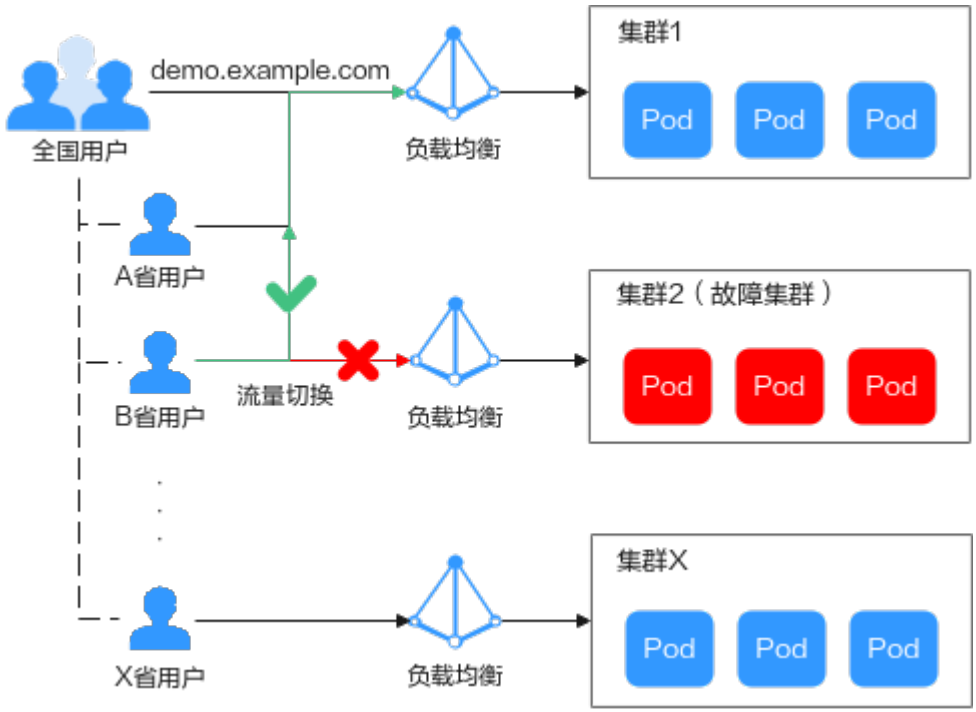
4 流量分发

4.1 使用流量分发实现应用故障倒换

应用场景

在分布式集群场景下，为了给用户提供低延迟的服务，应用可能部署在不同区域、不同厂商的云端上，在某个地区集群发生故障时，该地区的用户访问也随之会受到影响。利用UCS的流量管理和应用数据管理功能，可以实现多云多集群场景下的应用故障倒换、调度和迁移，故障倒换方案示意如图4-1所示。

图 4-1 多云集群应用故障倒换示意图



约束限制

- 您需要拥有两个Kubernetes版本为1.19及以上的可用集群，并且各个集群中至少拥有一个可用节点。
- 您需要已有一个公网域名，并添加至华为云云解析（DNS）服务，具体操作请参考**快速添加网站域名解析**。

环境搭建

步骤1 将集群注册到UCS并接入网络。具体操作请参见**注册集群**。

例如，将集群“ccecluster01”、“ccecluster02”添加至UCS，并查看集群是否处于正常运行状态。

步骤2 在添加至UCS的两个集群中分别创建一个工作负载。

说明

为展示流量切换的效果，本实践中两个集群的容器镜像版本不同。

- 集群“ccecluster01”：示例应用版本号为1.0.0。
- 集群“ccecluster02”：示例应用版本号为2.0.0。

图 4-2 创建工作负载

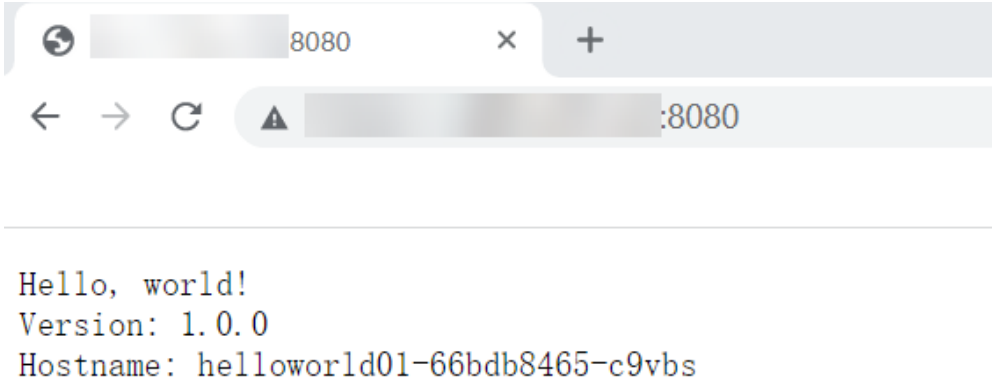
步骤3 分别为两个集群中的应用创建“负载均衡”类型的服务。

说明

仅支持访问类型为“负载均衡”的服务，其他类型的服务将被自动过滤。

步骤4 浏览器访问负载均衡IP地址，查看部署结果。

图 4-3 查看部署结果



----结束

功能验证

按照上述集群应用部署操作，示例应用分别部署在集群“ccecluster01”、“ccecluster02”中，并以“负载均衡”类型的服务对外提供访问。

下面将通过UCS的流量分发功能，实现多集群应用的故障倒换，验证应用的高可用容灾能力。

说明

实践中的应用仅作示例，在实际生产环境中可替换为用户自有应用，且对示例集群的提供商、地域、数量不作限制。

步骤1 登录UCS控制台，在左侧导航栏中单击“流量分发”。

步骤2 在流量管理控制台页面，单击右上角“创建流量策略”，填写域名地址解析，设置本例中的测试域名为“demo.example.com”。

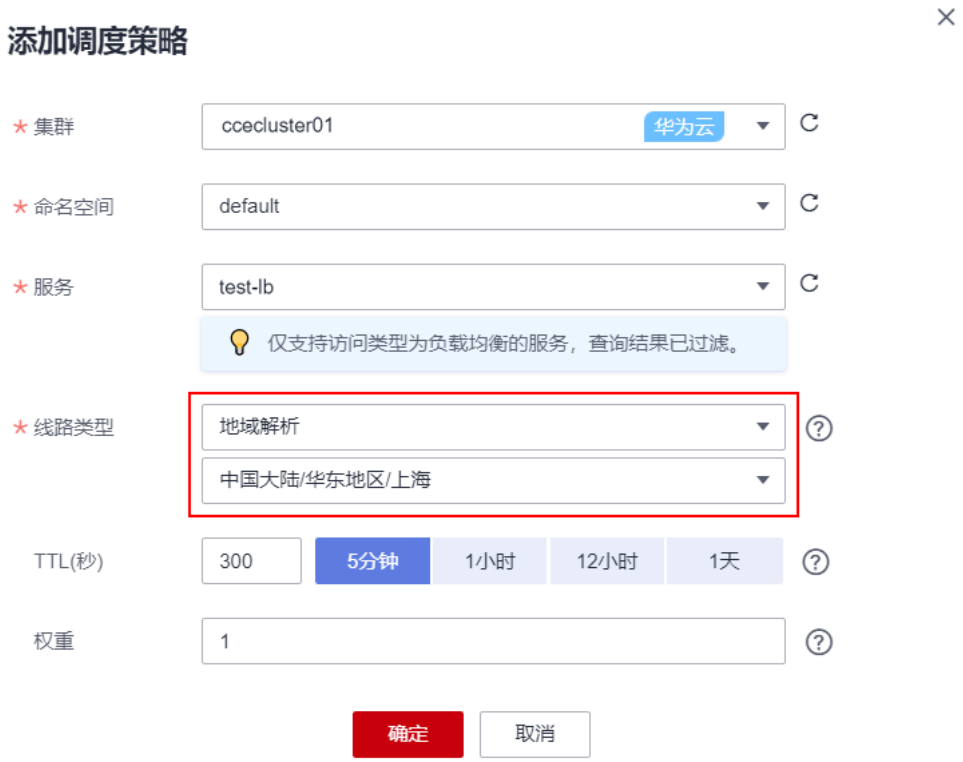
图 4-4 创建流量策略

步骤3 为两个集群服务分别添加调度策略，添加完成后单击“确定”。

本示例中，为模拟不同地域下的集群应用部署，添加三条调度策略：

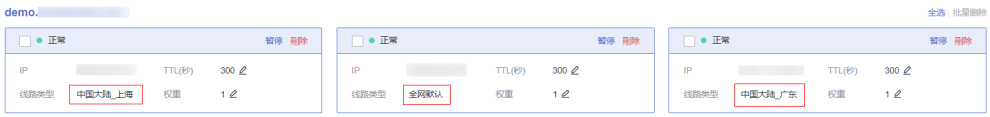
- 集群“ccecluster01”线路类型设置为“地域解析-中国大陆/华东地区/上海”。
- 集群“ccecluster02”线路类型设置为“地域解析-中国大陆/华南地区/广东”。
- 为域名添加默认线路解析记录，设置集群“ccecluster01”线路类型为“全网默认”。如不设置默认线路解析将会造成指定线路外的地区用户访问失败。

图 4-5 添加调度策略



步骤4 此时已为测试域名“demo.example.com”添加了三条解析，用户流量将根据设置的线路类型和权重正常访问两个集群中的应用。

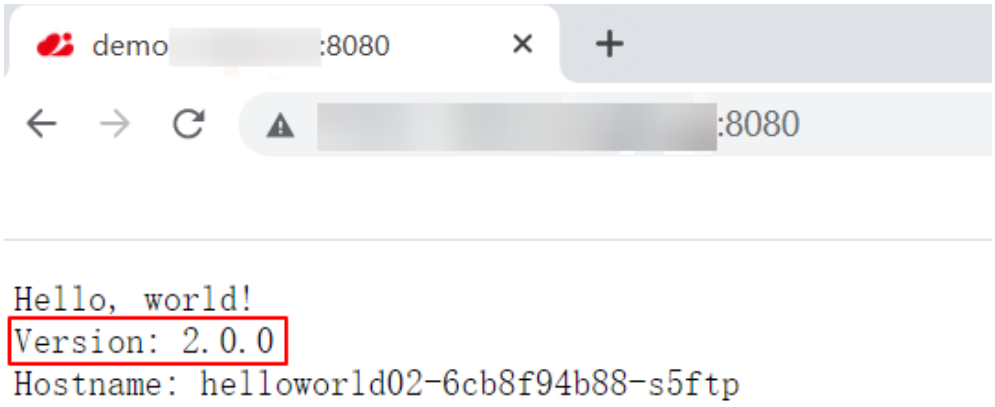
图 4-6 调度策略列表



- 上海地区用户：将访问集群“ccecluster01”中的应用，版本为1.0.0。
- 广东地区用户：将访问集群“ccecluster02”中的应用，版本为2.0.0。
- 其他用户：将默认访问集群“ccecluster01”中的应用，版本为1.0.0。

步骤5 广东地区用户通过域名“demo.example.com”访问应用，版本为2.0.0，说明访问的是集群“ccecluster02”中的应用。

图 4-7 查看访问结果



步骤6 此时手动停止集群“ccecluster02”中的应用，将实例个数调整为0，模拟环境故障。

图 4-8 调整实例个数

步骤7 广东地区用户访问应用时，依旧被解析至集群“ccecluster02”，返回错误。

此时需要在“流量分发”页面单击集群“ccecluster02”对应调度策略的“暂停”按钮，进行应用故障倒换。

图 4-9 暂停调度策略

广东地区用户访问域名“demo.example.com”时，不再解析至集群“ccecluster02”，只会将默认线路解析结果返回，用户访问到集群“ccecluster01”，访问正常。待运维人员完成故障集群修复后，可单击“启用”按钮重新使用该线路解析。

----结束

5 服务网格

5.1 第三方注册中心接入能力

ASM提供了服务网格对接Nacos注册中心功能，便于将Nacos上的微服务同步到网格中，实现流量治理等功能。

操作步骤

步骤1 通过**使用kubectl连接网格控制面**获取kubeconofig的证书内容。

步骤2 登录云容器引擎控制面，单击选择任意集群，进入详情页。

说明

建议选择网格对应的VPC下的集群。若连接其他VPC下的集群，则需要参考**UCS服务网格 集群连通方法**打通集群所在VPC和网格对应VPC。

步骤3 在左侧导航栏，单击“配置与密钥”，单击“密钥”页签，显示集群的密钥信息。

步骤4 选择“命名空间”为“asm-system”，单击右上角“创建密钥”。

步骤5 设置密钥参数，单击右下角“创建密钥”，完成密钥创建。

- 名称：自定义名称。例如：kubeconfig。

注意

创建的密钥的名称不要使用mesh-kubeconfig，因为apiserver也会自动创建这个名字的密钥，如果使用了这个名称可能会被覆盖。

- 描述：（可选）。
- 密钥类型：选择“其他”，填写密钥类型为“cfe/secure-opaque”。
- 密钥数据：添加密钥数据，键为“kubeconfig”，值为**步骤1**中获取到的kubeconfig证书内容。

步骤6 单击左侧导航栏“插件与模板”，在“可安装插件”页签中找到“asm-service-controller”插件，单击“安装”。

步骤7 填写配置参数，单击右下角“安装”，完成插件安装。

- meshKubeconfigSecret：为步骤**步骤5**中创建的密钥名称。
- source：目前包含“nacos”，代表对接的第三方注册中心nacos的相关信息，其中包含4个参数。
  - name：为该注册中心名称。
  - addr：为nacos的ip及端口。
  - allnamespaces：为是否需要同步nacos全部命名空间中的服务，当allnamespaces为false时，需要填充namespaces，表示需要同步的nacos的命名空间。
  - namespaces：表示需要同步的nacos的命名空间。

⚠ 注意

若插件运行的集群为CCE turbo类型集群，在安装插件完成后，需要参考**为Pod配置EIP**为asm-system命名空间下，名为asm-service-controller的pod绑定eip，才能正常使用该插件功能。

----结束

5.2 UCS 服务网格 集群连通方法

5.2.1 同 region 集群打通方法

以两个北京四集群为例，网格控制面也位于北京四，两个集群在不同的VPC中，需要使用VPC对等连接打通网络以使用网格功能。

网段约束

1. 各集群所在的VPC网段不能冲突。
2. 各集群所设置的容器网段不能冲突。
3. CCE网络插件实现会在路由表中添加路由，为了防止路由冲突造成网络无法联通，集群的VPC网段不能与其他集群的容器网段冲突。

操作步骤

步骤1 登录虚拟私有云控制台，单击“虚拟私有云>对等连接”，单击右上角“创建对等连接”。

步骤2 填写参数，选择需要打通的两个VPC，单击“确定”，创建对等连接。

步骤3 在弹出的提示信息中，单击“立即添加”，根据VPC对等连接提示在路由表中添加路由。

步骤4 单击“添加路由”，目的地址为对端VPC网段路由，下一跳类型为对等连接，下一跳为2创建的对等连接，单击“继续添加”，目的地址为对端集群网段路由，下一跳类型为对等连接，下一跳为2创建的对等连接，单击“确定”，完成路由表配置。需要在对端VPC路由表中做相同的操作。

步骤5 登录虚拟私有云控制台，单击“虚拟私有云>访问控制>安全组”，选择 {集群名}-cce-node-xxx 的安全组，单击安全组名称，查看安全组详情。

注意

标准版cce集群需放通node安全组，turbo集群要放通node安全组和eni安全组。

步骤6 单击“入方向规则”，单击“添加规则”，填写“协议端口”和“IP地址”信息，单击“确定”。放通来自另一个VPC网段以及对应集群的容器网段的请求。（在对端VPC安全组做同样的操作）

步骤7 查看添加的安全组规则。

----结束

5.2.2 跨 region 集群打通方法

以北京四、广州region为例，进行跨region集群引入网格，其中北京四为网格控制面所在region。

网段约束

1. 各集群所在的VPC网段不能冲突。
2. 各集群所设置的容器网段不能冲突。
3. CCE网络插件实现会在路由表中添加路由，为了防止路由冲突造成网络无法联通，集群的VPC网段不能与其他集群的容器网段冲突。

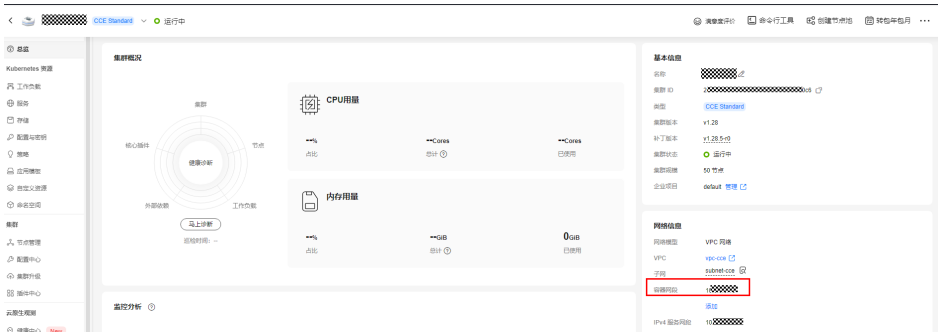
操作步骤

步骤1 登录云连接CC控制台，单击右侧“创建云连接”按钮。

步骤2 弹出创建云连接界面，填写参数信息，单击“确定”，完成创建。

步骤3 单击步骤2中创建的云连接，在弹出的页签中，单击“加载网络实例”进入页面，再单击“网络实例-加载网络实例”，选择对应region及VPC，并展开其他网段，填写对应region集群的容器网段。

步骤4 登录云容器引擎控制面，单击集群名称进入，在总览页面查看网络信息内容，获取到容器网段。



步骤5 所有需要通过云连接打通的集群，其VPC都需要接入到云连接中，查看VPC接入生效的方法如下：

登录虚拟私有云控制台，单击“虚拟私有云>路由表>对应VPC实例名称”，云连接会在VPC中添加两条路由。

步骤6 单击**步骤2**中创建的云连接，在弹出的页签中单击“带宽包>购买带宽包”，根据实际情况进行配置，注意将带宽包绑定之前创建的云连接实例。

步骤7 单击**步骤2**中创建的云连接，在弹出的页签中单击“域间带宽”，根据使用情况，在region之间分配域间带宽。

步骤8 登录虚拟私有云控制台，单击“虚拟私有云>访问控制>安全组”，选择 {集群名}-cce-node-xxx 的安全组，单击安全组名称，查看安全组详情。

注意

标准版cce集群需放通node安全组，turbo集群要放通node安全组和eni安全组。

步骤9 单击“入方向规则”，单击“添加规则”，填写“协议端口”和“IP地址”信息，单击“确定”。用于放通其他region连接控制面istiod与控制面kubeapiserver的请求。例如：在北京四VPC放通广州VPC与容器网段（相同操作**步骤8**-**步骤9**也需要在广州执行）。

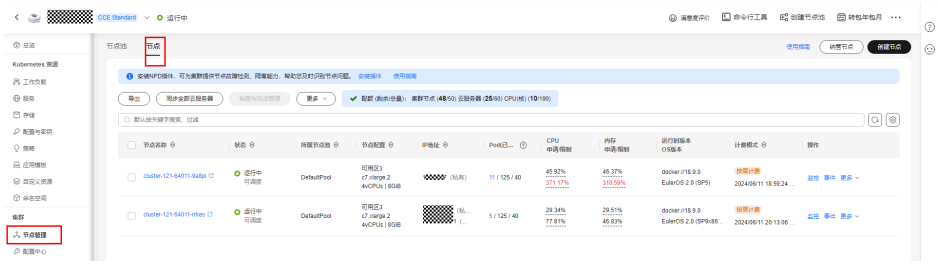
步骤10 查看添加的安全组规则。

----结束

5.2.3 如何确认集群连通

VPC 网段之间的网络连通

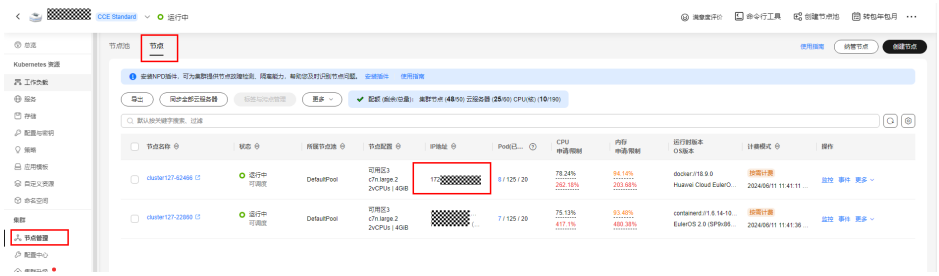
步骤1 登录云容器引擎控制台，选择本端集群，进入集群详情页，单击左侧导航栏“节点管理”，进入节点详情页。



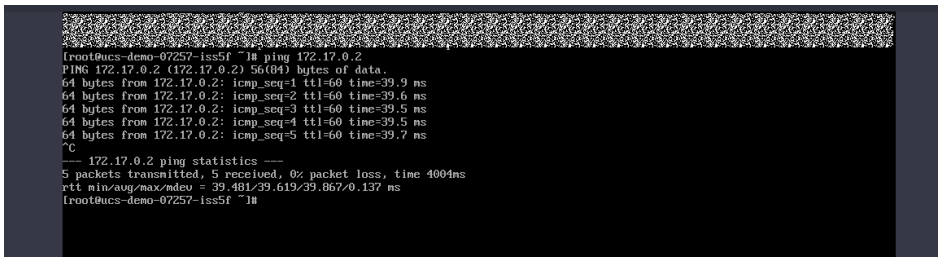
步骤2 单击“节点名称”下的节点，在弹出的页面中单击右上角“远程登录”，选择VNC方式登录。

步骤3 根据界面提示，输入账号和密码，成功进入Linux环境中。

步骤4 在云容器引擎控制台中，选择对端集群，进入集群详情页，单击左侧导航栏“节点管理”，进入节点详情页。



步骤5 在**步骤3**中，使用Linux命令查看网络是否连通。对端集群节点IP为**步骤4**中的节点IP。例如：ping 172.X.X.X。

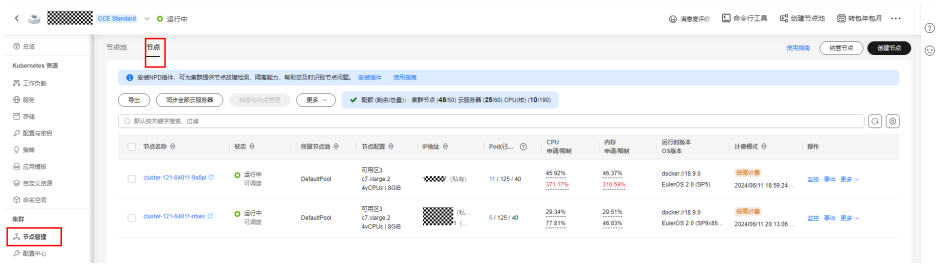


步骤6 在对端集群中执行相同的操作。

----结束

容器网段之间的网络连通

步骤1 登录云容器引擎控制台，选择本端集群，进入集群详情页，单击左侧导航栏“节点管理”，进入节点详情页。

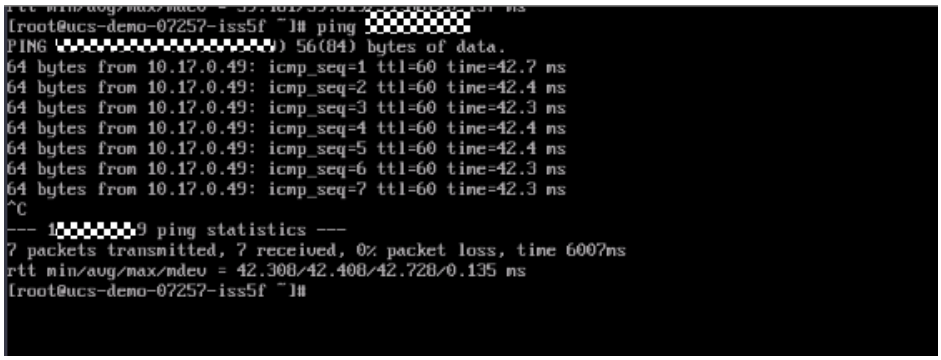


步骤2 单击“节点名称”下的节点，在弹出的页面中单击右上角“远程登录”，选择VNC方式登录。

步骤3 根据界面提示，输入账号和密码，成功进入Linux环境中。

步骤4 在云容器引擎控制台中，选择对端集群，进入集群详情页，单击左侧导航栏“工作负载>容器组”，进入容器Pod详情页。

步骤5 在**步骤3**中，使用Linux命令查看网络是否连通。对端集群PodIP为**步骤4**中的PodIP地址。。



步骤6 在对端集群中执行相同的操作。

----结束

5.3 为南北向服务网关的目标服务配置灰度发布

使用场景

服务网关是网格的流量入口，网格外部的客户端通过服务网关访问网格内的服务。目前默认是基于**Kubernetes Gateway API**模型实现网关能力，网格服务详情中的灰度发布策略只适用于东西向网格内部服务间；对于南北向入口网关的目标服务，如果需要配置灰度发布策略，可参考下文为入口网关的目标服务配置灰度发布策略。

说明

东西向网格内部服务间灰度发布，使用的是Istio的VirtualService/DestinationRule模型，依赖DestinationRule*subsets* 来定义服务的版本。

南北向入口网关的目标服务灰度发布，使用的是Kubernetes Gateway API的后端服务定义（backend service definitions），依赖定义多个service来定义服务的版本。

前提条件

- 已**启用网格**。
- 已**添加集群到网格**。
- 已**创建服务网关**。
- 已在集群创建v1、v2多个版本的工作负载

操作步骤

步骤1 创建nginx-v1服务

进入**CCE Console**页面，单击在网格已添加的CCE集群名称进入集群详情页，单击“服务-服务”，选择对应命名空间，单击“创建服务”按钮。

参数填写说明：

- Service名称：自定义服务名称，例如nginx-v1。
- 访问类型：选择集群内访问。
- 选择器：单击“引用负载标签”，选择对应的工作负载，例如nginx。
- 端口配置：容器端口填写业务容器进程监听端口，例如80。服务端口填写通过service访问的端口，例如5566。

步骤2 创建nginx-v2服务

参考**步骤1**创建**nginx-v2**服务。

步骤3 创建基于流量比例的路由

进入**华为云UCS控制台**，依次单击“服务网格-要配置的网格名称-服务网关-网关路由-HTTP路由-YAML创建”。

使用以下内容，创建nginx-canary网关路由。

```
apiVersion: gateway.networking.k8s.io/v1beta1
kind: HTTPRoute
metadata:
  name: nginx-canary # 网关路由名
  namespace: whtest # 网关路由所在的命名空间
spec:
  parentRefs:
    - group: gateway.networking.k8s.io
      kind: Gateway
      name: gwtest1 # 网关名
      namespace: whtest # 网关所在的命名空间
```

```
rules:
  - backendRefs:
      - group: ''
        kind: Service
        name: nginx-v1 # nginx-v1服务的服务名
        port: 5566 # nginx-v1服务的服务端口
        weight: 30 # nginx-v1服务的流量比例
      - group: ''
        kind: Service
        name: nginx-v2 # nginx-v2的服务名
        port: 5566 # nginx-v2服务的端口
        weight: 70 # nginx-v2服务的流量比例
    matches:
      - path:
          type: PathPrefix
          value: /
```

该配置表示路由规则引用whtest命名空间下名为gwtest1的Gateway资源。因为未指定监听器名称，此处会尝试引用该Gateway的所有监听器。对于路径前缀为/的请求，将30%流量路由到同命名空间下的nginx-v1服务的5566端口，将70%流量路由到同命名空间下的nginx-v2服务的5566端口。

步骤4 验证基于流量比例的路由生效

等待几秒钟待新规则配置下发成功，通过网关访问目标服务nginx应用，查看路由规则是否生效。

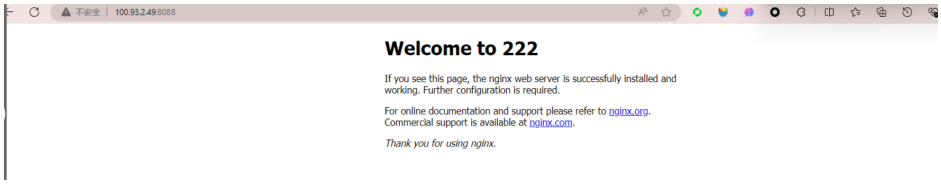
查看方法如下：

在浏览器中输入地址 http://$GATEWAY_ELB_IP:$GATEWAY_PORT/ ，其中，$GATEWAY_ELB_IP 是路由引用的whtest命名空间下名为gwtest1的网关的负载均衡公网地址；

$GATEWAY_PORT是gwtest1网关的监听器对外端口。

预期结果：

反复多次刷新浏览器，约有70%的时间可以看到v2版本的nginx服务内容。



----结束

相关文档

Istio Traffic Shifting Task

Gatewy API HTTP traffic splitting