

# 阿里云 容器服务Kubernetes版

API参考（旧）

文档版本：20200702

# 法律声明

---

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

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。
2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。
3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。
5. 阿里云文档中所有内容，包括但不限于图片、架构设计、页面布局、文字描述，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

## 通用约定

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

# 目录

---

法律声明.....	I
通用约定.....	I
1 使用Kubernetes API.....	1
2 集群 API 调用方式.....	5
2.1 概述.....	5
2.2 公共参数.....	5
2.3 请求结构.....	7
2.4 返回参数.....	7
2.5 签名机制.....	7
3 集群 API 列表.....	12
3.1 创建Kubernetes集群.....	12
3.2 创建Kubernetes托管版集群.....	20
3.3 创建Serverless Kubernetes集群.....	29
3.4 创建Kubernetes边缘托管版集群.....	31
3.5 创建安全沙箱Kubernetes集群.....	36
3.6 扩容 Kubernetes 多可用区集群.....	42
3.7 扩容Kubernetes边缘托管版集群.....	45
3.8 扩容安全沙箱集群.....	48
3.9 获取集群 kubeconfig 接口.....	50
3.10 修改集群 tag 接口.....	52
3.11 查看资源 Tag 列表.....	53
3.12 添加已有ECS实例到Kubernetes集群.....	55
3.13 生成Kubernetes边缘托管集群的节点接入脚本.....	57
3.14 添加已有ENS节点至Kubernetes边缘托管集群.....	60
3.15 查询集群实例.....	62
3.16 查询所有集群实例.....	66
3.17 移除节点.....	68
3.18 删除集群.....	70

# 1 使用Kubernetes API

本文将基于cURL命令简单演示如何以REST的方式使用Kubernetes API，方便您使用开发语言原生的HTTPS方式操作Kubernetes集群。演示包括创建和删除Pod，创建和修改Deployment。

## 获取集群访问凭证kubeconfig

1. 登录[容器服务管理控制台](#)，在集群列表中，选择您要操作的集群，在右侧操作栏中单击**管理**。



2. 在集群管理信息页面您可以看到集群访问凭证（**kubeconfig**），将**kubeconfig**文件保存到本地。
3. 从kubeconfig文件中提取ca、key和apiserver信息，命令如下。

```
# cat ./kubeconfig |grep client-certificate-data | awk -F ' ' '{print $2}' |base64 -d > client-cert.pem
# cat ./kubeconfig |grep client-key-data | awk -F ' ' '{print $2}' |base64 -d > client-key.pem
# APISERVER=`cat ./kubeconfig |grep server | awk -F ' ' '{print $2}'`
```

## 使用cURL命令操作Kubernetes API

执行以下命令查看当前集群中所有namespaces。

```
# curl --cert client-cert.pem --key client-key.pem -k $APISERVER/api/v1/namespaces
```

- 常用的Pod相关操作。

执行以下命令查看default命名空间下的所有Pods。

```
# curl --cert client-cert.pem --key client-key.pem -k $APISERVER/api/v1/namespaces/default/pods
```

执行以下命令创建Pod（JSON格式）。

```
# cat nginx-pod.json
{
  "apiVersion": "v1",
  "kind": "Pod",
  "metadata": {
    "name": "nginx",
    "namespace": "default"
```

```
{
  "spec": {
    "containers": [
      {
        "name": "nginx",
        "image": "nginx:alpine",
        "ports": [
          {
            "containerPort": 80
          }
        ]
      }
    ]
  }
}
```

```
# curl --cert client-cert.pem --key client-key.pem -k $APISERVER/api/v1/namespaces/
default/pods -X POST --header 'content-type: application/json' -d@nginx-pod.json
```

执行以下命令创建Pod（YAML格式）。

```
# cat nginx-pod.yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  namespace: default
spec:
  containers:
  - name: nginx
    image: nginx:alpine
    ports:
    - containerPort: 80
```

```
# curl --cert client-cert.pem --key client-key.pem -k $APISERVER/api/v1/namespaces/default/pods -X POST --header 'content-type: application/yaml' --data-binary @nginx-pod.yaml
```

执行以下命令查询Pod状态。

```
# curl --cert client-cert.pem --key client-key.pem -k $APISERVER/api/v1/namespaces/default/pods/nginx
```

执行以下命令查询Pod logs。

```
# curl --cert client-cert.pem --key client-key.pem -k $APISERVER/api/v1/namespaces/default/pods/nginx/log
```

执行以下命令查询Pod的metrics数据（通过metric-server api）。

```
# curl --cert client-cert.pem --key client-key.pem -k $APISERVER/apis/metrics.k8s.io/v1beta1/namespaces/default/pods/nginx
```

执行以下命令删除Pod。

```
# curl --cert client-cert.pem --key client-key.pem -k $APISERVER/api/v1/namespaces/default/pods/nginx -X DELETE
```

- 常用的Deployment相关操作。

创建Deployment示例YAML文件如下。

```
# cat nginx-deploy.yaml
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: nginx-deploy
  labels:
    app: nginx
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:alpine
        ports:
        - containerPort: 80
      resources:
        requests:
          cpu: "2"
          memory: "4Gi"
```

```
# curl --cert client-cert.pem --key client-key.pem -k $APISERVER/apis/extensions/v1beta1/namespaces/default/deployments -X POST --header 'content-type: application/yaml' --data-binary @nginx-deploy.yaml
```

执行以下命令查看Deployment。

```
# curl --cert client-cert.pem --key client-key.pem -k $APISERVER/apis/extensions/v1beta1/namespaces/default/deployments
```

执行以下命令更新Deployment（修改replicas副本数量）。

```
# curl --cert client-cert.pem --key client-key.pem -k $APISERVER/apis/extensions/v1beta1/namespaces/default/deployments/nginx-deploy -X PATCH -H 'Content-Type: application/strategic-merge-patch+json' -d '{"spec": {"replicas": 4}}'
```

执行以下命令更新Deployments（修改容器镜像）。

```
# curl --cert client-cert.pem --key client-key.pem -k $APISERVER/apis/extensions/v1beta1/namespaces/default/deployments/nginx-deploy -X PATCH -H 'Content-Type: application/strategic-merge-patch+json' -d '{"spec": {"template": {"spec": {"containers": [{"name": "nginx", "image": "nginx:1.7.9"}]}}}}'
```

## 相关文档

[官方SDK](#)

[详细的Kubernetes API规范](#)

[其他Kubernetes集群访问方式](#)

推荐您使用Kubernetes官方维护的SDK（包括Go、Python、Java等语言）。



## 2 集群 API 调用方式

### 2.1 概述

对容器服务 API 接口的调用是通过向容器服务 API 的服务端地址发送 HTTP 请求，并按照接口说明在请求中加入相应请求参数来完成的。根据请求的处理情况，系统会返回处理结果。

1. [公共参数](#)
2. [请求结构](#)
3. [返回参数](#)
4. [签名机制](#)

### 2.2 公共参数

#### 公共请求头部

公共请求参数是指每个接口都需要使用到的请求参数。

参数名称	说明	选项
<b>Authorization</b>	用于验证请求合法性的认证信息，采用 AccessKeyId: Signature 的形式。	Required
<b>Content-Length</b>	RFC 2616 中定义的 HTTP 请求内容长度。	Required
<b>Content-Type</b>	RFC 2616 中定义的 HTTP 请求内容类型。	Required
<b>Content-MD5</b>	HTTP 协议消息体的 128-bit MD5 散列值转换成 BASE64 编码的结果。为了防止所有请求被篡改，建议所有请求都附加该信息。	Required
<b>Date</b>	请求的构造时间，目前只支持 GMT 格式。如果与 MNS 的服务器时间前后差异超过 15 分钟将返回本次请求非法。	Required
<b>Host</b>	访问 Host 值，例如：diku.aliyuncs.com。	Required

参数名称	说明	选项
<b>Accept</b>	客户端需要的返回值类型，支持 application/json 和 application/xml。	Required
<b>x-acis-version</b>	API 版本号。目前版本号为 2015-12-15。	Required
<b>x-acis-region-id</b>	地域（Region）指的是 ECS 实例所在的物理位置。	Required
<b>x-acis-signature-nonce</b>	唯一随机数，用于防止网络重放攻击。您在不同请求间要使用不同的随机数值。	Required
<b>x-acis-signature-method</b>	用户签名方式，目前只支持 HMAC-SHA1。	Required

### 示例

```
GET /clusters HTTP/1.1
Host: cs.aliyuncs.com
Accept: application/json
User-Agent: cs-sdk-python/0.0.1 (Darwin/15.2.0/x86_64;2.7.10)
x-acis-signature-nonce: f63659d4-10ac-483b-99da-ea8fde61eae3
Authorization: acs <yourAccessKeyId>:<yourSignature>
x-acis-signature-version: 1.0
Date: Wed, 16 Dec 2015 11:18:47 GMT
x-acis-signature-method: HMAC-SHA1
Content-Type: application/json;charset=utf-8
X-Acis-Region-Id: cn-beijing
Content-Length: 0
```

### 公共返回头部

您发送的每次接口调用请求，无论成功与否，系统都会返回一个唯一识别码 RequestId。

### 示例

XML 示例：

```
<?xml version="1.0" encoding="UTF-8"?>
<!--结果的根结点-->
<接口名称+Response>
| <!--返回请求标签-->
| <RequestId>4C467B38-3910-447D-87BC-AC049166F216</RequestId>
| <!--返回结果数据-->
</接口名称+Response>
```

JSON 示例：

```
{
  "RequestId": "4C467B38-3910-447D-87BC-AC049166F216"
  /* 返回结果数据 */
}
```

```
}
```

## 2.3 请求结构

本文为您介绍容器服务请求结构。

### 服务地址

阿里云容器服务的 OpenAPI 接入地址为 cs.aliyuncs.com。

### 通信协议

支持通过 HTTP 或 HTTPS 通道进行请求通信。为了获得更高的安全性，推荐您使用 HTTPS 通道发送请求。

### 请求方法

使用 HTTP 的 PUT、POST、GET、DELETE 等 HTTP Method 发送不同的请求。

### 请求参数

每个请求都需要包含公共请求参数和指定操作所特有的请求参数。

### 请求编码

请求及返回结果都使用 UTF-8 字符集进行编码。

## 2.4 返回参数

调用 API 服务后返回数据采用统一格式。返回的 HTTP 状态码为 2xx，代表调用成功；返回的 HTTP 状态码为 4xx 或 5xx，代表调用失败。调用成功返回的数据格式主要有 XML 和 JSON 两种，外部系统可以在请求时传入参数来制定返回的数据格式，默认为 XML 格式。

为了便于您查看，本文档中的返回示例做了格式化处理，实际返回结果是没有换行、缩进等处理的。

## 2.5 签名机制

本文主要介绍签名机制。

### 签名机制说明

Access Key ID 和 Access Key Secret 由阿里云官方颁发给访问者（可以通过阿里云官方网站申请和管理），其中 Access Key ID 用于标识访问者的身份；Access Key Secret 是用于加密签名字符串和服务器端验证签名字符串的密钥，必须严格保密，只有阿里云和用户知道。

容器服务会对每个访问的请求进行验证，每个向容器服务提交的请求，都需要在请求中包含签名（Signature）信息。容器服务通过使用 Access Key ID 和 Access Key Secret 进行对称加密的方法来

验证请求的发送者身份。如果计算出来的验证码和提供的一样即认为该请求是有效的；否则，容器服务将拒绝处理这次请求，并返回 HTTP 403 错误。

用户可以在 HTTP 请求中增加授权（Authorization）的 Head 来包含签名信息，表明这个消息已被授权。

容器服务要求将签名包含在 HTTP Header 中，格式为 `Authorization: acs [Access Key Id]:[Signature]`。

Signature 的计算方法如下：

```
Signature = base64(hmac-sha1(VERB + "\n"
+ ACCEPT + "\n"
+ Content-MD5 + "\n"
+ Content-Type + "\n"
+ Date + "\n"
+ CanonicalizedHeaders + "\n"
+ CanonicalizedResource))
```

- VERB 表示 HTTP 的 Method。例如示例中的 PUT。
- Accept 客户端需要的返回值类型，支持 application/json 和 application/xml。
- Content-MD5 表示请求内容数据的 MD5 值。
- Content-Type 表示请求内容的类型。
- Date 表示此次操作的时间，不能为空，目前只支持 GMT 格式。如果请求时间与 CAS 服务器时间相差超过 15 分钟，CAS 会判定此请求不合法，并返回 400 错误。错误信息及错误码详见本文档第 5 部分。例如示例中的 Thu, 17 Mar 2012 18:49:58 GMT。
- CanonicalizedHeaders 表示 HTTP 中以 x-acsc- 开始的字段组合。
- CanonicalizedResource 表示 HTTP 所请求资源的 URI（统一资源标识符）。例如示例中的 / clusters?name=my-clusters&resource=new。



#### 说明：

CanonicalizedHeaders（即以 x-acsc- 开头的 header）在签名验证前需要符合以下规范：

1. 将所有以 x-acsc- 为前缀的 HTTP 请求头的名字转换成小写字母。例如将 X-ACS-Meta-Name: TaoBao 转换为 x-acsc-meta-name: TaoBao。阿里云规范请求头的名字是大小写不敏感的，建议全部使用小写。
2. 如果一个公共请求头的值部分过长，则需要处理其中的 \t、\n、\r、\f 分隔符，将其替换为英文半角的空格。
3. 将上一步得到的所有 HTTP 阿里云规范头按照字典序进行升序排列。
4. 删除请求头和内容之间分隔符两端出现的任何空格。例如将 x-acsc-meta-name: TaoBao, Alipay 转换为 x-acsc-meta-name: TaoBao, Alipay。

5. 将所有的头和内容用 \n 分隔符分隔拼成最后的 CanonicalizedHeaders。



#### 说明:

CanonicalizedResource 的格式规范: CanonicalizedResource 表示客户想要访问资源的规范描述, 需要将子资源和 query 一同按照字典序, 从小到大排列并以 & 为分隔符生成子资源字符串 (? 后的所有参数)。

```
http://cs.aliyuncs.com/clusters?name=my-clusters&resource=new
```

CanonicalizedResource 应该为:

```
/clusters?name=my-clusters&resource=new
```

### 签名示例

您可以通过该示例, 了解加签的步骤。

示例使用的 accessKeyId 和 accessKeySecret 分别为 access\_key\_id 和 access\_key\_secret。推荐您使用自己的 OpenAPI 调用程序, 来计算下面这个示例的加签串, 您自己的加签结果和示例结果。

请求的示例如下:

```
POST http://cs.aliyuncs.com/clusters?param1=value1&param2=value2 HTTP/1.1
Accept-Encoding: identity
Content-Length: 210
Content-MD5: 6U4ALMkKSj0PYbeQSHqgmA==
x-accs-version: 2015-12-15
Accept: application/json
User-Agent: cs-sdk-python/0.0.1 (Darwin/15.2.0/x86_64;2.7.10)
x-accs-signature-nonce: fbf6909a-93a5-45d3-8b1c-3e03a7916799
x-accs-signature-version: 1.0
Date: Wed, 16 Dec 2015 12:20:18 GMT
x-accs-signature-method: HMAC-SHA1
Content-Type: application/json;charset=utf-8
X-Acs-Region-Id: cn-beijing
Authorization: acs <yourAccessKeyId>:<yourSignature>
{"password": "Just$****", "instance_type": "ecs.m2.medium", "name": "my-test-cluster-9708****", "size": 1, "network_mode": "vpc", "data_disk_category": "cloud", "data_disk_size": 10, "ecs_image_id": "m-253l****l"}
```

请求构造过程

计算 Content-Length 和 Content-MD5

Content-Length



#### 说明:

示例 body 首位没有空格或换行符。

```
body: {"password": "Just$****", "instance_type": "ecs.m2.medium", "name": "my-test-
cluster-9708****", "size": 1, "network_mode": "vpc", "data_disk_category": "cloud",
data_disk_size": 10, "ecs_image_id": "m-253ll****"}
Content-Length: 210
```

## Content-MD5

```
body: {"password": "Just$****","instance_type": "ecs.m2.medium","name": "my-test-cluster-9708****","size": 1,"network_mode": "vpc","data_disk_category": "cloud","data_disk_size": 10,"ecs_image_id": "m-253ll****"}
# 计算 body 的 md5 值
md5(body): e94e002cc90a4a3d0f61b790487aa098
# 将 md5 值转化成字节数组。将 md5 中的每两个十六进制位合并，转化为一个字节。
# 例如：e9 -> 111111111111111111111111111111111101001 -> -23
bytes(md5(body)): [[-23], [78], [0], [44], [-55], [10], [74], [61], [15], [97], [-73], [-112], [72], [122], [-96], [-104]]
# 将得到的字节数组做一个 base64 转换
base64(bytes(md5(body))): 6U4ALMkKSj0PYbeQSHqgmA==
Content-MD5: 6U4ALMkKSj0PYbeQSHqgmA==
```

## 处理 CanonicalizedHeaders

```
# 将所有以'x-acsc-'开头的头部列出来
x-acsc-version: 2015-12-15
x-acsc-signature-nonce: ca480402-7689-43ba-acc4-4d2013d9d8d4
x-acsc-signature-version: 1.0
x-acsc-signature-method: HMAC-SHA1
X-Acs-Region-Id: cn-beijing
# 将请求名字变成小写，去掉每一行首尾的空格，并按照字典序进行排序。删除请求头和内容之间
分隔符两端出现的任何空格。
# 注意：最后一行没有换行符。
x-acsc-region-id:cn-beijing
x-acsc-signature-method:HMAC-SHA1
x-acsc-signature-nonce:fbf6909a-93a5-45d3-8b1c-3e03a7916799
x-acsc-signature-version:1.0
x-acsc-version:2015-12-15
```

## 计算 CanonicalizedResource

示例得到的 CanonicalizedResource，长度应该为 27。



**说明:**

第一行行尾有一个 `\n` 的换行符。

```
/clusters?param1=value1&param2=value2
```

## 计算 Signature

组装 `SignatureString`。示例中的加签字符串的长度为 307。除最后一行外，每一行行尾均有一个 `\n` 的换行符。

POST  
application/json

```
6U4ALMkKSj0PYbeQSHqgmA==
application/json;charset=utf-8
Wed, 16 Dec 2015 12:20:18 GMT
x-accs-region-id:cn-beijing
x-accs-signature-method:HMAC-SHA1
x-accs-signature-nonce:fbf6909a-93a5-45d3-8b1c-3e03a7916799
x-accs-signature-version:1.0
x-accs-version:2015-12-15
/clusters?param1=value1&param2=value2
```

### 计算 Signature

```
# 使用 accessKeySecret 来对加签字符串进行加密，其中示例使用的 accessKeySecret 是
access_key_secret。
hmac-sha1(SignatureString): fee03d405e421ebaf514adec881038c4b313584d
# 类似于 Content-MD5 的计算方式，将得到的加密串转化成字节数组。
# 将得到的字符数组做一个 base64 转换。得到最后的签名串。
base64(bytes(hmac-sha1(SignatureString))): ZmVlMDNkNDA1ZTQyMWVlYWY1MTRhZG
VjODgxMDM4YzRlMzEzNTg0ZA==
Signature: ZmVlMDNkNDA1ZTQyMWVlYWY1MTRhZGVjODgxMDM4YzRlMzEzNTg0ZA==
```

经过以上的处理，添加一些其他头部信息，最终构成的 HTTP 请求如下所示。

```
POST http://cs.aliyuncs.com/clusters?param1=value1&param2=value2 HTTP/1.1
Accept-Encoding: identity
Content-Length: 210
Content-MD5: 6U4ALMkKSj0PYbeQSHqgmA==
x-accs-version: 2015-12-15
Accept: application/json
User-Agent: cs-sdk-python/0.0.1 (Darwin/15.2.0/x86_64;2.7.10)
x-accs-signature-nonce: fbf6909a-93a5-45d3-8b1c-3e03a7916799
x-accs-signature-version: 1.0
Date: Wed, 16 Dec 2015 12:20:18 GMT
x-accs-signature-method: HMAC-SHA1
Content-Type: application/json;charset=utf-8
X-Acs-Region-Id: cn-beijing
Authorization: acs <yourAccessKeyId>:<yourSignature>
{"password": "Just$****", "instance_type": "ecs.m2.medium", "name": "my-test-cluster-
9708****", "size": 1, "network_mode": "vpc", "data_disk_category": "cloud", "data_disk_size
": 10, "ecs_image_id": "m-253ll****"}
```

## 3 集群 API 列表

### 3.1 创建Kubernetes集群

调用**CreateCluster**创建一个新的Kubernetes集群实例，并新建指定数量的节点。

#### 请求信息

请求行RequestLine

POST /clusters HTTP/1.1

特有请求头RequestHead

无，请参见[公共请求头部](#)。

请求体RequestBody



#### 说明：

创建集群时，请求参数需要正确组合，否则会导致集群创建失败。在容器服务控制台创建集群界面，提供了**生成集群创建OpenAPI参数**功能，可以为您提供准确的参数组合。请参见[#unique\\_12](#)。

```
{
  "disable_rollback": "失败是否回滚",
  "name": "集群名称",
  "timeout_mins": "集群创建超时时间",
  "cluster_type": "集群类型, Kubernetes",
  "region_id": "地域",
  "vpcid": "VPC ID",
  "master_vswitch_ids": "master节点交换机ID。填写3个vswitchid, 尽量是分别在3个不同az, 保证高可用",
  "master_instance_types": "master节点实例类型。填写3个实例规格",
  "master_count": "master实例个数, 支持3|5",
  "container_cidr": "容器POD CIDR",
  "service_cidr": "服务CIDR",
  "ssh_flags": "是否开放公网SSH登录",
  "cloud_monitor_flags": "是否安装云监控插件",
  "login_password": "节点SSH登录密码, 和key_pair二选一",
  "key_pair": "keypair名称, 和login_password二选一",
  "master_instance_charge_type": "Master实例付费类型, PostPaid|PrePaid",
  "master_period_unit": "包年包月单位, Month,Year, 只有在PrePaid下生效",
  "master_period": "包年包月时长, 只有在PrePaid下生效",
  "master_auto_renew": "Master节点是否自动续费",
  "master_auto_renew_period": "Master节点续费周期",
  "master_system_disk_category": "Master系统盘类型",
  "master_system_disk_size": "Master节点系统盘大小",
  "master_data_disk (已弃用)": "Master节点是否挂载数据盘",
  "master_data_disks": "Master节点数据盘配置",
  "master_system_disk_snapshot_policy_id": "Master节点系统盘快照策略ID",
  "worker_instance_charge_type": "Worker节点付费类型PrePaid|PostPaid",
```



```

"worker_period_unit": "包年包月单位, Month,Year, 只有在PrePaid下生效",
"worker_period": "包年包月时长, 只有在PrePaid下生效",
"worker_auto_renew": "Worker节点自动续费true/false",
"worker_auto_renew_period": "Worker节点续费周期",
"worker_instance_types": "Worker实例规格多实例规格参数",
"worker_vswitch_ids": "一台或多台虚拟交换机 ID, N 的取值范围为 [1, 5]",
"worker_system_disk_category": "Worker系统盘类型",
"worker_system_disk_size": "Worker节点系统盘大小",
"worker_data_disk (已弃用)": "Worker节点是否挂载数据盘",
"worker_data_disks": "Worker节点数据盘配置",
"worker_system_disk_snapshot_policy_id": "Worker节点系统盘快照策略ID",
"num_of_nodes": "Worker节点数",
"snat_entry": "是否配置SNATEntry",
"endpoint_public_access": "是否公网暴露集群endpoint",
"cpu_policy": "static|none",
"node_port_range": "节点端口范围, 默认30000-65535",
"proxy_mode": "网络模式, 可选值iptables|ipvs",
"addons": "选装addon, 数组格式对象",
"tags": "给集群打tag标签, 数组格式对象",
"security_group_id": "安全组ID",
"auto_snapshot_policy_id": "自动快照策略ID",
"taints": "给节点添加taint, 数组格式对象",
"deletion_protection": "是否开启集群删除保护, 防止通过控制台或api误删除集群",
"os_type": "运行pod的主机的操作系统类型, 例如: linux, Windows等",
"platform": "运行pod的主机的平台架构",
"runtime": "容器运行时, 默认为docker",
}

```

表 3-1: 请求体解释

名称	类型	必须	描述
<b>cluster_type</b>	string	是	集群类型。
<b>key_pair</b>	string	是	keypair名称。和login_password二选一。
<b>login_password</b>	string	是	SSH登录密码。密码规则为8 - 30 个字符, 且至少同时包含三项（大小写字母、数字和特殊符号）。和key_pair 二选一。
<b>master_instance_charge_type</b>	string	否	Master节点付费类型, 可选值为: <ul style="list-style-type: none"> <li>PrePaid: 预付费</li> <li>PostPaid: 按量付费</li> </ul> 默认为按量付费。
<b>master_instance_types</b>	list	是	Master节点ECS规格类型代码。更多详细信息, 参见 <a href="#">#unique_13</a> 。
<b>master_system_disk_category</b>	string	是	Master节点系统盘类型, 取值范围是: <ul style="list-style-type: none"> <li>cloud_efficiency: 高效云盘。</li> <li>cloud_ssd: SSD云盘。</li> </ul>
<b>master_system_disk_size</b>	int	是	Master节点系统盘大小, 单位为GiB。

名称	类型	必须	描述
<b>master_vswitch_ids</b>	list	是	Master节点交换机ID列表，交换机个数取值范围为1~3。为确保集群的高可用性，推荐您选择3个交换机，且分布在不同的可用区。
<b>num_of_nodes</b>	int	是	Worker节点数。范围是[0, 100]。
<b>name</b>	string	是	集群名称，集群名称可以使用大小写英文字母、中文、数字、中划线。
<b>region_id</b>	string	是	集群所在地域ID。
<b>snat_entry</b>	bool	是	<p>是否为网络配置SNAT。</p> <ul style="list-style-type: none"> <li>当已有VPC能访问公网环境时，设置为 false。</li> <li>当已有VPC不能访问公网环境时： <ul style="list-style-type: none"> <li>设置为true，表示配置SNAT，此时可以访问公网环境。</li> <li>设置为false，表示不配置SNAT，此时不能访问公网环境。</li> </ul> </li> </ul>
<b>worker_instance_types</b>	list	是	Worker节点ECS规格类型代码。更多详细信息，参见 <a href="#">#unique_13</a> 。
<b>worker_system_disk_category</b>	string	是	Worker节点系统盘类型。
<b>worker_system_disk_size</b>	int	是	Worker节点系统盘大小，单位为GiB。
<b>worker_vswitch_ids</b>	list	是	Worker节点的虚拟交换机ID。

名称	类型	必须	描述
<b>addons</b>	list	否	<p>Kubernetes集群的addon插件的组合。</p> <ul style="list-style-type: none"> <li>addons的参数： <ul style="list-style-type: none"> <li>name：必填，addon插件的名称。</li> <li>version：可选，取值为空时默认取最新版本。</li> <li>config：可选，取值为空时表示无需配置。</li> </ul> </li> <li>网络插件：包含Flannel和Terway网络插件，二选一。</li> <li>日志服务：可选，如果不开启日志服务时，将无法使用集群审计功能。</li> <li>Ingress：默认开启安装Ingress组件。</li> </ul>
<b>container_cidr</b>	string	否	容器网段，不能和VPC网段冲突。当选择系统自动创建VPC时，默认使用172.16.0.0/16网段。
<b>cloud_monitor_flags</b>	bool	否	<p>是否安装云监控插件：</p> <ul style="list-style-type: none"> <li>true：表示安装。</li> <li>false：表示不安装。</li> </ul>
<b>cpu_policy</b>	string	否	CPU策略。集群版本为1.12.6及以上版本支持static 和 none两种策略。默认为none。
<b>disable_rollback</b>	bool	否	<p>失败是否回滚：</p> <ul style="list-style-type: none"> <li>true：表示失败不回滚。</li> <li>false：表示失败回滚。</li> </ul> <p>默认为true，如果选择失败回滚，则会释放创建过程中所生产的资源，不推荐使用false。</p>
<b>master_auto_renew</b>	bool	否	<p>Master节点是否自动续费，当<b>master_instance_charge_type</b>取值为PrePaid时才生效，可选值为：</p> <ul style="list-style-type: none"> <li>true：自动续费。</li> <li>false：不自动续费。</li> </ul>
<b>master_auto_renew_period</b>	int	否	自动续费周期，当选择预付费和自动续费时才生效，且为必选值。 <b>PeriodUnit=Month</b> 时，取值为{ "1" , "2" , "3" , "6" , "12" }。
<b>master_count</b>	int	否	Master实例个数，可选值3或者5。默认值为3。

名称	类型	必须	描述
<b>master_data_disk (已弃用)</b>	bool	否	Master节点是否挂载数据盘，可选择为： <ul style="list-style-type: none"> <li>• true: 挂载数据盘</li> <li>• false: 不挂载数据盘，默认为false</li> </ul>
<b>master_data_disks</b>	list	否	Master数据盘类型、大小等配置的组合。该参数只有在挂载Master节点数据盘时有效，包含以下参数： <ul style="list-style-type: none"> <li>• category: 数据盘类型。取值范围： <ul style="list-style-type: none"> <li>- cloud: 普通云盘</li> <li>- cloud_efficiency: 高效云盘</li> <li>- cloud_ssd: SSD云盘</li> </ul> </li> <li>• size: 数据盘大小，单位为GiB。</li> <li>• encrypted:</li> </ul>
<b>master_period</b>	int	否	包年包月时长，当 <b>master_instance_charge_type</b> 取值为 <b>PrePaid</b> 时才生效且为必选值，取值范围：PeriodUnit=Month时，Period取值：{ "1" , "2" , "3" , "6" , "12" }。
<b>master_period_unit</b>	string	否	当指定为PrePaid的时候需要指定周期。Month：以月为计时单位。
<b>node_port_range</b>	string	否	节点服务端口。取值范围为[30000, 65535]。
<b>proxy_mode</b>	string	否	kube-proxy 代理模式，支持 iptables 和 IPVS 两种模式。默认为 iptables 。
<b>endpoint_public_access</b>	bool	否	是否开启公网API Server： <ul style="list-style-type: none"> <li>• true: 默认为True，表示开放公网API Server。</li> <li>• false: 若设置为false，则不会创建公网的API Server，仅创建私网的API Server。</li> </ul>
<b>security_group_id</b>	string	否	指定集群ECS实例所属于的安全组ID。
<b>service_cidr</b>	string	否	服务网段，不能和VPC网段以及容器网段冲突。当选择系统自动创建VPC时，默认使用172.19.0.0/20网段。

名称	类型	必须	描述
<b>ssh_flags</b>	bool	否	是否开放公网SSH登录： <ul style="list-style-type: none"> <li>• true：表示开放。</li> <li>• false：表示不开放。</li> </ul>
<b>tags</b>	list	否	给集群打tag标签： <ul style="list-style-type: none"> <li>• key：标签名称。</li> <li>• value：标签值。</li> </ul>
<b>taints</b>	list	否	用于给节点做污点标记，通常用于 Pods 的调度策略。与之相对应的概念为：容忍（tolerance），若 Pods 上有相对应的 tolerance 标记，则可以容忍节点上的污点，并调度到该节点。
<b>kubernetes_version</b>	string	否	Kubernetes集群版本，默认最新版。ACK支持的KubernetesK8s版本：v1.16.6, v1.14.6和v1.12.6。
<b>timeout_mins</b>	int	否	集群资源栈创建超时时间，以分钟为单位，默认值 60。
<b>vpcid</b>	string	否	VPC ID，可空。如果不设置，系统会自动创建VPC，系统创建的VPC网段为192.168.0.0/16。 <div>  <b>说明：</b>  VpcId 和 vswitchid 只能同时为空或者同时都设置对应的值。 </div>
<b>worker_auto_renew</b>	bool	否	是否开启Worker节点自动续费，可选值为： <ul style="list-style-type: none"> <li>• true：自动续费。</li> <li>• false：不自动续费。</li> </ul>
<b>worker_auto_renew_period</b>	int	否	自动续费周期，当选择预付费和自动续费时才生效，且为必选值：PeriodUnit=Month时，取值{ "1" , "2" , "3" , "6" , "12" }。
<b>worker_data_disk (已弃用)</b>	string	否	是否挂载数据盘，可选择为： <ul style="list-style-type: none"> <li>• true：表示worker节点挂载数据盘。</li> <li>• false：表示worker节点不挂载数据盘。</li> </ul>

名称	类型	必须	描述
<b>worker_data_disks</b>	list	否	Worker数据盘类型、大小等配置的组合。该参数只有在挂载Worker节点数据盘时有效，包含以下参数： <ul style="list-style-type: none"> <li>category: 数据盘类型。取值范围： <ul style="list-style-type: none"> <li>cloud: 普通云盘。</li> <li>cloud_efficiency: 高效云盘。</li> <li>cloud_ssd: SSD云盘。</li> </ul> </li> <li>size: 数据盘大小，单位为GiB。</li> <li>encrypted: 是否对数据盘加密，true/false。</li> </ul>
<b>worker_instance_charge_type</b>	string	否	Worker节点付费类型，可选值为： <ul style="list-style-type: none"> <li>PrePaid: 预付费。</li> <li>PostPaid: 按量付费。</li> </ul> 默认为按量付费。
<b>worker_period</b>	int	否	包年包月时长，当 <b>worker_instance_charge_type</b> 取值为 <b>PrePaid</b> 时才生效且为必选值，取值范围：PeriodUnit=Month时，Period取值：{ "1" , "2" , "3" , "6" , "12" }。
<b>worker_period_unit</b>	string	否	当指定为PrePaid的时候需要指定周期。Month：以月为计时单位。
<b>deletion_protection</b>	bool	否	是否开启集群删除保护，防止通过控制台或api误删除集群。
<b>os_type</b>	string	否	运行pod的主机的操作系统类型，例如：linux, Windows等。
<b>platform</b>	string	否	运行pod的主机的平台架构。
<b>runtime</b>	json	否	容器运行时，一般为docker，包括2个信息：name和version。
<b>node_cidr_mask</b>	string	否	节点网络的网络前缀。node_cidr_mask通过cidr限制一个节点上能运行容器网络的Pod数量。例如node_cidr_mask是24的话，对应的容器数量是256，25对应的就是128。计算方式： $2^{(node\_cidr\_mask\text{的值}-24)-1}$ 。

## 返回信息

返回行 ResponseLine

HTTP/1.1 202 Accepted

特有返回头 ResponseHead

无，请参见[公共返回头部](#)。

表 3-2: 返回体 ResponseBody

名称	类型	描述
cluster_id	string	集群实例ID。
request_id	string	请求ID。
task_id	string	任务ID。系统自动分配，用户查询任务状态。

## 示例

请求示例

```
POST /clusters HTTP/1.1
<公共请求头>
{
  "cluster_type": "Kubernetes",
  "name": "my-test-Kubernetes-cluster",
  "region_id": "cn-beijing",
  "disable_rollback": true,
  "timeout_mins": 60,
  "kubernetes_version": "1.12.6-aliyun.1",
  "snat_entry": true,
  "endpoint_public_access": false,
  "cloud_monitor_flags": false,
  "node_cidr_mask": "25",
  "proxy_mode": "iptables",
  "tags": [],
  "addons": [{"name": "flannel"}, {"name": "aliyun-log-controller", "config": "{\"sls_project_name\": \"k8s-log-c64f6eab6a1764d3dbec3dc2b9e41****\"}"}, {"name": "nginx-ingress-controller", "config": "{\"IngressSlbNetworkType\": \"internet\"}"}, {"name": "nginx-ingress-controller", "config": "{\"IngressSlbNetworkType\": \"internet\"}"}, {"name": "nginx-ingress-controller", "config": "{\"IngressSlbNetworkType\": \"internet\"}"}],
  "node_port_range": "30000-32767",
  "login_password": "test****",
  "cpu_policy": "none",
  "master_count": 3,
  "master_vswitch_ids": ["vsw-2ze48rkq464rsdts****", "vsw-2ze48rkq464rsdts1****", "vsw-2ze48rkq464rsdts1****"],
  "master_instance_types": ["ecs.sn1.medium", "ecs.sn1.medium", "ecs.sn1.medium"],
  "master_system_disk_category": "cloud_efficiency",
  "master_system_disk_size": 40,
  "worker_instance_types": ["ecs.sn2.3xlarge"],
  "num_of_nodes": 3,
  "worker_system_disk_category": "cloud_efficiency",
  "worker_system_disk_size": 120,
  "vpcid": "vpc-2zegvl5etah5requ0****",
  "worker_vswitch_ids": ["vsw-2ze48rkq464rsdts****"],
```

```
"container_cidr": "172.20.XX.XX/16",
"service_cidr": "172.21.XX.XX/20",
"worker_data_disks": [{"category": "cloud_ssd", "size": 500}],
"master_data_disks": [{"category": "cloud_ssd", "size": 500}],
"taints": [{"key": "special", "value": "true", "effect": "NoSchedule"}]
}
```

返回示例

```
HTTP/1.1 202 Accepted
<公共响应头>
{
  "cluster_id": "cb95aa626a47740afbf6aa099b650****",
  "request_id": "687C5BAA-D103-4993-884B-C35E4314A1E1",
  "task_id": "T-5a54309c80282e39ea00002f"
}
```

## 3.2 创建Kubernetes托管版集群

调用**CreateCluster**创建一个Kubernetes托管版集群实例，并创建指定数量的节点。

### 请求信息

请求行RequestLine

```
POST /clusters HTTP/1.1
```

特有请求头RequestHead

无，请参见[公共请求头部](#)。

请求体RequestBody



#### 说明：

创建集群时，请求参数需要正确组合，否则会导致集群创建失败。在容器服务控制台创建集群界面，提供了**生成集群创建OpenAPI参数**功能，可以为您提供准确的参数组合。请参见[#unique\\_12](#)。

```
{
  "disable_rollback": "失败是否回滚。",
  "name": "集群名称。",
  "timeout_mins": "集群创建超时时间。",
  "cluster_type": "集群类型，Managed Kubernetes。",
  "region_id": "地域。",
  "vpcid": "VPC ID",
  "vswitch_ids": "一台或多台虚拟交换机ID，N的取值范围为[1,3]。",
  "container_cidr": "容器POD CIDR。",
  "service_cidr": "服务CIDR。",
  "cloud_monitor_flags": "是否安装云监控插件。",
  "login_password": "节点SSH登录密码，和key_pair二选一。",
  "key_pair": "keypair名称，和login_password二选一。",
  "worker_instance_charge_type": "Worker节点付费类型PrePaid|PostPaid。",
  "worker_period_unit": "包年包月单位，Month,Year，只有在PrePaid下生效。",
  "worker_period": "包年包月时长，只有在PrePaid下生效。",
}
```



```

"worker_auto_renew": "Worker节点自动续费true|false。",
"worker_auto_renew_period": "Worker节点续费周期。",
"worker_instance_types": "Worker实例规格多实例规格参数。",
"worker_system_disk_category": "Worker系统盘类型。",
"worker_system_disk_size": "Worker节点系统盘大小。",
"worker_data_disk": "是否挂载数据盘true|false。",
"worker_data_disks": "Worker节点数据盘配置。",
"num_of_nodes": "Worker节点数。",
"snat_entry": "是否配置SNATEntry",
"endpoint_public_access": "是否公网暴露集群endpoint。",
"proxy_mode": "网络模式, 可选值iptables|ipvs。",
"addons": "选装addon, 数组格式对象",
"tags": "给集群打tag标签, 数组格式对象。",
"security_group_id": "安全组ID。",
"taints": "给节点添加taint, 数组格式对象。",
"cpu_policy": "cpu的策略, static|none。",
"runtime": "容器的运行时环境, 一般为Docker。",
"platform": "运行pod的主机的平台架构。",
"os_type": "运行pod的主机的操作系统类型, 例如: linux, Windows等。",
"node_cidr_mask": "节点网络的网络ID的位数。",
"kubernetes_version": "kubernetes集群的版本, 默认最新版本。",
"deletion_protection": "是否开启集群删除保护, 防止通过控制台或API误删除集群。"
}

```


表 3-3: 请求体解释

名称	类型	必须	描述
<b>cluster_type</b>	string	是	集群类型。
<b>key_pair</b>	string	是	keypair名称。与login_password二选一。
<b>login_password</b>	string	是	SSH登录密码。密码规则为8 - 30个字符, 且至少同时包含三项（大小写字母、数字和特殊符号）。和key_pair二选一。
<b>name</b>	string	是	集群名称, 集群名称可以使用大小写英文字母、中文、数字、中划线。
<b>num_of_nodes</b>	int	是	Worker节点数。范围是[0, 100]。
<b>region_id</b>	string	是	集群所在地域ID。

名称	类型	必须	描述
<b>snat_entry</b>	bool	是	是否为网络配置SNAT。如果是自动创建VPC必须设置为 <b>true</b> 。如果使用已有VPC则根据是否具备出网能力来设置。
<b>vswitch_ids</b>	list	是	交换机ID。List长度范围为 [1, 3]。
<b>worker_system_disk_category</b>	string	是	Worker节点系统盘类型。
<b>worker_system_disk_size</b>	int	是	Worker节点系统盘大小。
<b>addons</b>	list	否	<p>Kubernetes集群的addon插件的组合。</p> <ul style="list-style-type: none"> <li>addons的参数： <ul style="list-style-type: none"> <li>name：必填，addon插件的名称。</li> <li>version：可选，取值为空时默认取最新版本。</li> <li>config：可选，取值为空时表示无需配置。</li> </ul> </li> <li>网络插件：包含Flannel和Terway网络插件，二选一。</li> <li>日志服务：可选，如果不开启日志服务时，将无法使用集群审计功能。</li> <li>Ingress：默认开启安装Ingress组件。</li> </ul>
<b>container_cidr</b>	string	否	容器网段，不能和VPC网段冲突。当选择系统自动创建VPC时，默认使用172.16.0.0/16网段。

名称	类型	必须	描述
<b>cloud_monitor_flags</b>	bool	否	是否安装云监控插件。
<b>disable_rollback</b>	bool	否	<p>失败是否回滚：</p> <ul style="list-style-type: none"> <li>• true：表示失败不回滚。</li> <li>• false：表示失败回滚。</li> </ul> <p>如果选择失败回滚，则会释放创建过程中所生产的资源，不推荐使用false。</p>
<b>public_slb</b>	bool	否	<p>是否开启公网API Server：</p> <ul style="list-style-type: none"> <li>• true：默认为True，表示开放公网API Server。</li> <li>• false：若设置为false，则不会创建公网的API Server，仅创建私网的API Server。</li> </ul> <div>  <b>说明：</b>            过期参数，替代参数请参见<b>endpoint_public_access</b>。         </div>
<b>proxy_mode</b>	string	否	kube-proxy代理模式，支持iptables和IPVS两种模式。默认为iptables。
<b>endpoint_public_access</b>	bool	否	<p>是否开启公网API Server：</p> <ul style="list-style-type: none"> <li>• true：默认为True，表示开放公网API Server。</li> <li>• false：若设置为false，则不会创建公网的API Server，仅创建私网的API Server。</li> </ul>

名称	类型	必须	描述
<b>security_group_id</b>	string	否	指定集群ECS实例所属于的安全组ID。
<b>service_cidr</b>	string	否	服务网段，不能和VPC网段以及容器网段冲突。当选择系统自动创建VPC时，默认使用172.19.0.0/20网段。
<b>tags</b>	list	否	给集群打tag标签： <ul style="list-style-type: none"> <li>key：标签名称。</li> <li>value：标签值。</li> </ul>
<b>taints</b>	list	否	用于给节点做污点标记，通常用于Pods的调度策略。与之相对应的概念为：容忍（tolerance），若Pods上有相对应的tolerance标记，则可以容忍节点上的污点，并调度到该节点。
<b>timeout_mins</b>	int	否	集群资源栈创建超时时间，以分钟为单位，默认值60分钟。
<b>vpcid</b>	string	否	VPC ID，可空。如果不设置，系统会自动创建VPC，系统创建的VPC网段为192.168.0.0/16。Vpcid和vswitchid只能同时为空或者同时都设置相应的值。
<b>worker_auto_renew</b>	bool	否	是否开启Worker节点自动续费，可选值为： <ul style="list-style-type: none"> <li>true：自动续费。</li> <li>false：不自动续费。</li> </ul>

名称	类型	必须	描述
<b>worker_auto_renew_period</b>	int	否	自动续费周期，当 <b>worker_instance_charge_type</b> 取值为 <b>PrePaid</b> 时才生效且为必选值。PeriodUnit=Month 时，取值为 { "1" , "2" , "3" , "6" ,
<b>worker_data_disk</b>	string	否	是否挂载数据盘，可选择： <ul style="list-style-type: none"> <li>true：表示worker节点挂载数据盘。</li> <li>false：表示worker节点不挂载数据盘。</li> </ul>
<b>worker_data_disks</b>	list	否	Worker数据盘类型、大小等配置的组合。该参数只有在挂载Worker节点数据盘时有效，包含以下参数： <ul style="list-style-type: none"> <li>category：数据盘类型。取值范围： <ul style="list-style-type: none"> <li>cloud：普通云盘。</li> <li>cloud_efficiency：高效云盘。</li> <li>cloud_ssd：SSD云盘。</li> </ul> </li> <li>size：数据盘大小，单位为GiB。</li> </ul>
<b>worker_data_disk_category</b>	int	否	数据盘类型 <div>  <b>说明：</b>            过期参数，替代参数请参见 <b>worker_data_disks</b> 参数中的 category 取值。         </div>

名称	类型	必须	描述
<b>worker_data_disk_size</b>	string	否	数据盘大小   <b>说明:</b> 过期参数，替代参数请参见 <b>worker_data_disks</b> 参数中的size取值。
<b>worker_instance_charge_type</b>	string	否	Worker节点付费类型，可选值为： <ul style="list-style-type: none"><li>PrePaid：预付费。</li><li>PostPaid：按量付费。</li></ul>
<b>worker_period</b>	int	否	包年包月时长，当 <b>worker_instance_charge_type</b> 取值为 <b>PrePaid</b> 时才生效且为必选值。 PeriodUnit=Month时，Period取值：{ “1” , “2” , “3” , “6” , “12” 。
<b>worker_period_unit</b>	string	否	当指定为PrePaid的时候需要指定周期。   <b>说明:</b> Month：以月为计时单位。
<b>worker_instance_types</b>	list	是	Worker节点ECS规格类型代码。更多详细信息，参见 <a href="#">#unique_13</a> 。
<b>cpu_policy</b>	string	否	CPU策略。集群版本为1.12.6及以上版本支持static和none两种策略。默认为none。

名称	类型	必须	描述
runtime	json	否	容器运行时，一般为docker，包括2个信息。 <ul style="list-style-type: none"><li>name：容器运行时名称。</li><li>version：容器运行时版本。</li></ul>
platform	string	否	运行Pod的主机的平台架构。
os_type	string	否	运行Pod的主机的操作系统类型，例如Linux，Windows。
node_cidr_mask	int	否	节点网络的网络前缀。node_cidr_mask通过cidr限制一个节点上能运行容器网络的Pod数量。例如node_cidr_mask是24的话，对应的容器数量是256，25对应的就是128。计算方式： $2^{(\text{node\_cidr\_mask}-24)-1}$ 。
kubernetes_version	string	否	Kubernetes集群的版本，默认最新版本。
deletion_protection	bool	否	是否开启集群删除保护，防止通过控制台或API误删除集群。

## 返回信息

返回行ResponseLine

```
HTTP/1.1 202 Accepted
```

特有返回头ResponseHead

无，请参见[公共返回头部](#)。

返回体ResponseBody

```
{
  "cluster_id": "string",
  "request_id": "string",
}
```

```
"task_id": "string"
}
```

## 示例

### 请求示例

```
POST /clusters HTTP/1.1
<公共请求头>
{
  "name": "test",
  "cluster_type": "my-test-Kubernetes-cluster",
  "disable_rollback": true,
  "timeout_mins": 60,
  "kubernetes_version": "1.12.6-aliyun.1",
  "region_id": "cn-beijing",
  "snat_entry": true,
  "cloud_monitor_flags": false,
  "endpoint_public_access": false,
  "node_cidr_mask": "25",
  "proxy_mode": "ipvs",
  "tags": [],
  "addons": [{"name": "flannel"}, {"name": "aliyun-log-controller", "config": "{\"sls_project_name\": \"k8s-log-c64f6eab6a1764d3dbee3dc2b9e41****\"}"}, {"name": "nginx-ingress-controller", "config": "{\"IngressSlbNetworkType\": \"internet\"}"}],
  "worker_instance_types": ["ecs.hfc5.xlarge"],
  "num_of_nodes": 3,
  "worker_system_disk_category": "cloud_efficiency",
  "worker_system_disk_size": 120,
  "worker_instance_charge_type": "PostPaid",
  "vpcid": "vpc-2zegvl5etah5requ09nec",
  "container_cidr": "172.20.0.0/16",
  "service_cidr": "172.21.0.0/20",
  "vswitch_ids": ["vsw-2ze48rkq464rsdts1****"],
  "worker_data_disks": [{"category": "cloud_ssd", "size": 500}],
  "login_password": "test@19****",
  "taint": [{"key": "special", "value": "true", "effect": "NoSchedule"}]
}
```

### 返回示例

```
HTTP/1.1 202 Accepted
<公共响应头>
{
  "cluster_id": "cb95aa626a47740afbf6aa099b65****",
  "request_id": "687C5BAA-D103-4993-884B-C35E4314A1E1",
  "task_id": "T-5a54309c80282e39ea00002f"
```



```
}
```

### 3.3 创建Serverless Kubernetes集群

调用**CreateCluster**创建一个新的Serverless Kubernetes集群实例。

**请求信息**

请求行RequestLine

POST /clusters HTTP/1.1

特有请求头RequestHead

无。请参见[公共请求头部](#)。

请求体RequestBody

```
{
  "cluster_type": "ASK",
  "name": "集群名称",
  "region_id": "地域",
  "zoneid": "可用区",
  "nat_gateway": "true", //是否创建NAT网关。
  "private_zone": "true", //是否开启privateZone用于服务发现。
  "vpc_id": "VPC ID", //不填则自动创建专有网络VPC。
  "tags": "给集群打tag标签, 数组格式对象。",
  "vswitch_id": "交换机ID。"
}
```

表 3-4: 请求体解释

名称	类型	必须	描述
<b>cluster_type</b>	string	是	集群类型。
<b>name</b>	string	是	集群名称， 集群名称可以使用大小写英文字母、中文、数字、中划线。
<b>private_zone</b>	bool	是	是否开启PrivateZone用于服务发现， 取值为true或者false。请参见 <a href="#">#unique_16</a> 。
<b>region_id</b>	string	是	集群所在地域ID。
<b>zoneid</b>	string	是	所属地域的可用区。
<b>vswitch_id</b>	string	否	交换机ID， 可空。若不设置，系统会自动创建交换机，系统自定创建的交换机网段为192.168.0.0/16。

名称	类型	必须	描述
<b>tags</b>	list	否	给集群打tag标签： <ul style="list-style-type: none"> <li>key：标签名称。</li> <li>value：标签值。</li> </ul>
<b>nat_gateway</b>	bool	否	是否创建NAT网关。取值为true或者false。如果不设置，系统默认为false。
<b>vpc_id</b>	string	否	VPC ID，可空。如果不设置，系统会自动创建VPC，系统创建的VPC网段为192.168.0.0/16。 <div>  <b>说明：</b> vpc_id和vswitch_id只能同时为空或者同时都设置对应的值。 </div>

## 返回信息

返回行ResponseLine

HTTP/1.1 202 Accepted

特有返回头ResponseHead

无。请参见[公共返回头部](#)。

**表 3-5: 返回体ResponseBody**

名称	类型	描述
<b>cluster_id</b>	string	集群实例ID。
<b>request_id</b>	string	请求ID。
<b>task_id</b>	string	任务ID。系统自动分配，用户查询任务状态。

## 示例

请求示例

```
POST /clusters HTTP/1.1
<公共请求头>
{
  "cluster_type": "ASK",
  "name": "my-test-Kubernetes-cluster",
  "region_id": "cn-beijing",
  "tags": [],
```

```
"vpcid": "vpc-2zegvl5etah5requ0****",
"vswitch_ids": ["vsw-2ze48rkq464rsdts1****"]
}
```

返回示例

```
HTTP/1.1 202 Accepted
<公共响应头>
{
  "cluster_id": "cb95aa626a47740afbf6aa099b650****",
  "request_id": "687C5BAA-D103-4993-884B-C35E4314A1E1",
  "task_id": "T-5a54309c80282e39ea00002f",
  "instanceId": "cb95aa626a47740afbf6aa099b650****"
}
```

## 3.4 创建Kubernetes边缘托管版集群

调用**CreateCluster**创建一个新的Kubernetes边缘托管版集群实例。

### 请求信息

请求行RequestLine

```
POST /clusters HTTP/1.1
```

特有请求头RequestHead

无，请参见[公共请求头部](#)。

请求体RequestBody

```
{
  "disable_rollback": "失败是否回滚。",
  "name": "集群名称。",
  "timeout_mins": "集群创建超时时间。",
  "cluster_type": "集群类型，ManagedKubernetes。",
  "profile": "边缘集群标志，Edge。",
  "region_id": "地域",
  "vpcid": "VPC ID",
  "vswitch_ids": "一台或多台虚拟交换机ID，N的取值范围为[1, 5]。",
  "container_cidr": "容器POD CIDR。",
  "service_cidr": "服务CIDR",
  "cloud_monitor_flags": "是否安装云监控插件。",
  "login_password": "节点SSH登录密码，和key_pair二选一。",
  "key_pair": "keypair名称，和login_password二选一。",
  "worker_instance_types": "Worker实例规格多实例规格参数。",
  "worker_system_disk_category": "Worker系统盘类型。",
  "worker_system_disk_size": "Worker节点系统盘大小。",
  "worker_data_disk": "是否挂载数据盘true/false",
  "worker_data_disk_category": "数据盘类型。",
  "worker_data_disk_size": "数据盘大小",
  "num_of_nodes": "Worker节点数",
  "snat_entry": "是否配置SNATEntry。",
  "endpoint_public_access": "是否公网暴露集群endpoint。",
  "tags": "给集群打tag标签，数组格式对象。"
```

}

表 3-6: 请求体解释

名称	类型	必须	描述
<b>cluster_type</b>	string	是	集群类型
<b>key_pair</b>	string	是	keypair名称，与login_password二选一。
<b>login_password</b>	string	是	SSH登录密码。密码规则为8 - 30个字符，且同时包含三项（大、小写字母，数字和特殊符号），和key_pair二选一。
<b>name</b>	string	是	集群名称，集群名称可以使用大小写英文字母、中文、数字、中划线。
<b>num_of_nodes</b>	int	是	Worker节点数。范围是[0, 300]。
<b>profile</b>	string	是	边缘集群标识，默认取值为Edge。
<b>region_id</b>	string	是	集群所在地域ID。
<b>snat_entry</b>	bool	是	是否为网络配置SNAT。如果是自动创建VPC必须设置为true。如果使用已有VPC则根据是否具备出网能力来设置。
<b>vswitch_ids</b>	list	是	交换机ID。List长度范围为 [1, 3]。
<b>worker_system_disk_category</b>	string	是	Worker节点系统盘类型。
<b>worker_system_disk_size</b>	int	是	Worker节点系统盘大小。

名称	类型	必须	描述
<b>container_cidr</b>	string	否	容器网段，不能和VPC网段冲突。当选择系统自动创建VPC时，默认使用172.16.0.0/16网段。
<b>cloud_monitor_flags</b>	bool	否	是否安装云监控插件。  <div>  <b>说明：</b>            如果您选择了使用日志或者监控服务，会自动购买一个云端ECS节点用于部署相应管控服务，那么需要配置云端worker的登录信息和资源规格信息。         </div>
<b>disable_rollback</b>	bool	否	失败是否回滚： <ul style="list-style-type: none"> <li>• true：表示失败不回滚。</li> <li>• false：表示失败回滚。</li> </ul> 如果选择失败回滚，则会释放创建过程中所生产的资源，不推荐使用false。
<b>proxy_mode</b>	string	否	kube-proxy代理模式，支持iptables和IPVS两种模式。默认为iptables。

名称	类型	必须	描述
<b>endpoint_public_access</b>	bool	否	<p>是否开启公网API Server:</p> <ul style="list-style-type: none"> <li>• true: 默认为True, 表示开放公网API Server。</li> <li>• false: 若设置为false, 则不会创建公网的API Server, 仅创建私网的API Server。</li> </ul> <div>  <b>说明:</b>            在边缘集群场景, 边缘节点通过公网和云端管控交互; 因此, 边缘集群需要开启公网访问。         </div>
<b>service_cidr</b>	string	否	服务网段, 不能和VPC网段以及容器网段冲突。当选择系统自动创建VPC时, 默认使用172.19.0.0/20网段。
<b>tags</b>	list	否	<p>给集群打tag标签:</p> <ul style="list-style-type: none"> <li>• key: 标签名称</li> <li>• value: 标签值</li> </ul>
<b>timeout_mins</b>	int	否	集群资源栈创建超时时间, 以分钟为单位, 默认值60分钟。
<b>vpcid</b>	string	否	VPC ID, 可空。如果不设置, 系统会自动创建VPC, 系统创建的VPC网段为192.168.0.0/16。Vpcid和vswitchid只能同时为空或者同时都设置相应的值。

名称	类型	必须	描述
<b>worker_data_disk</b>	string	否	是否挂载数据盘，可选择： <ul style="list-style-type: none"><li>true：表示worker节点挂载数据盘。</li><li>false：表示worker节点不挂载数据盘。</li></ul>
<b>worker_data_disk_category</b>	int	否	数据盘类型
<b>worker_data_disk_size</b>	string	否	数据盘大小

## 返回信息

返回行ResponseLine

```
HTTP/1.1 202 Accepted
```

特有返回头ResponseHead

无，请参见[公共返回头部](#)。

返回体ResponseBody

```
{
  "cluster_id": "string",
  "request_id": "string",
  "task_id": "string"
}
```

## 示例

请求示例

```
POST /clusters HTTP/1.1
<公共请求头>
{
  "name": "test",
  "cluster_type": "ManagedKubernetes",
  "profile": "Edge",
  "disable_rollback": true,
  "timeout_mins": 60,
  "region_id": "cn-beijing",
  "snat_entry": true,
  "cloud_monitor_flags": true,
  "endpoint_public_access": true,
  "tags": [],
  "worker_instance_types": ["ecs.hfc5.xlarge"],
  "num_of_nodes": 1,
  "worker_system_disk_category": "cloud_efficiency",
  "vpcid": "vpc-2zegvl5eta*****",
}
```

```
"container_cidr": "172.20.0.0/16",
"service_cidr": "172.21.0.0/20",
"vswitch_ids": ["vsw-2ze48rkq464rsdts1****"],
"login_password": "test@19****"
}
```

返回示例

```
HTTP/1.1 202 Accepted
<公共响应头>
{
  "cluster_id": "cb95aa626a47740afbf6aa099b65****",
  "request_id": "687C5BAA-D103-4993-884B-C35E4314****",
  "task_id": "T-5a54309c80282e39ea00****"
}
```

## 3.5 创建安全沙箱Kubernetes集群

调用**CreateCluster**创建一个新的安全沙箱容器Kubernetes集群。

### 请求信息

请求行RequestLine

```
POST /clusters HTTP/1.1
```

特有请求头RequestHead

无，请参见[公共请求头部](#)。

请求体RequestBody

```
{
  "name": "集群名称",
  "cluster_type": "集群类型: ManagedKubernetes",
  "disable_rollback": "true",
  "timeout_mins": "60",
  "region_id": "地域",
  "snat_entry": "是否配置SNATEntry。",
  "cloud_monitor_flags": "是否安装云监控插件。",
  "public_slb": "是否公网暴露集群endpoint。",
  "worker_instance_type": "worker实例规格",
  "num_of_nodes": "worker数量",
  "vpcid": "vpc id",
  "vswitch_ids": "worker节点虚拟交换机ID",
  "service_cidr": "服务CIDR",
  "login_password": "节点SSH登录密码，和key_pair二选一。",
  "key_pair": "keypair名称，和login_password二选一。",
  "worker_system_disk_category": "Worker系统盘类型",
  "worker_system_disk_size": "Worker节点系统盘大小",
  "worker_data_disk_category": "Worker节点数据盘类型",
  "worker_data_disk_size": "Worker节点数据盘大小",
  "worker_data_disk": "是否挂载数据盘true/false。",
  "worker_instance_charge_type": "Worker节点付费类型PrePaid|PostPaid",
  "worker_period_unit": "包年包月单位，Month,Year，只有在PrePaid下生效。",
  "worker_period": "包年包月时长，只有在PrePaid下生效。",
  "worker_auto_renew": "Worker节点自动续费true/false",
  "worker_auto_renew_period": "Worker节点续费周期",
}
```



```
"kubernetes_version": "kubernetes版本",
"addons": [{"name": "terway-eniip"}],
"runtime": {"name": "Sandboxed-Container.runv", "version": "1.0.0"},
"pod_vswitch_ids": "Pod虚拟交换机id数组，只支持1个虚拟交换机。"
}
```

表 3-7: 请求体解释

名称	类型	必须	描述
<b>addons</b>	list	是	<p>Kubernetes集群的addon插件的组合。</p> <ul style="list-style-type: none"> <li>addons的参数： <ul style="list-style-type: none"> <li>name：必填，addon插件的名称。</li> <li>version：可选，取值为空时默认取最新版本。</li> <li>config：可选，取值为空时表示无需配置。</li> </ul> </li> <li>网络插件：包含Flannel和Terway网络插件，二选一。</li> <li>日志服务：可选。 <ul style="list-style-type: none"> <li>如果不开启日志服务时，将无法使用集群审计功能。</li> <li>如果开启日志服务时，在addon数组中添加{"name": "aliyun-log-controller"}。</li> </ul> </li> </ul>
<b>cluster_type</b>	string	是	<p>集群类型</p> <p>安全沙箱容器当前支持的集群类型为ManagedKubernetes。</p>
<b>key_pair</b>	string	是	keypair名称，和login_password二选一。
<b>kubernetes_version</b>	string	是	安全沙箱容器仅支持1.14.6-aliyun.1及以上版本。
<b>login_password</b>	string	是	SSH登录密码。密码规则为8 ~ 30个字符，且同时包含三项（大小写字母、数字和特殊符号），和 <b>key_pair</b> 二选一。
<b>name</b>	string	是	集群名称，集群名称可以使用大小写英文字母、中文、数字、中划线。

名称	类型	必须	描述
<b>num_of_nodes</b>	int	是	Worker节点数。范围是[0, 100]。
<b>pod_vswitch_ids</b>	list	是	Pod使用的虚拟交换机的ID。
<b>region_id</b>	string	是	集群所在地域ID。
<b>runtime</b>	json	是	<ul style="list-style-type: none"> <li>name: 安全容器运行时的名称, 目前支持Sandboxed-Container.runv。</li> <li>version: 当前版本1.0.0。</li> </ul>
<b>snat_entry</b>	bool	是	<p>是否为网络配置SNAT。</p> <ul style="list-style-type: none"> <li>当已有VPC能访问公网环境时, 设置为false。</li> <li>当已有VPC不能访问公网环境时: <ul style="list-style-type: none"> <li>设置为true, 表示配置SNAT, 此时可以访问公网环境。</li> <li>设置为false, 表示不配置SNAT, 此时不能访问公网环境。</li> </ul> </li> </ul>
<b>vpcid</b>	string	是	<p>VPC ID。如果不设置, 系统会自动创建VPC, 系统创建的VPC网段为192.168.0.0/16。</p> <div>  <b>说明:</b>  <b>vpcid</b>和<b>vswitch_ids</b>只能同时为空或者同时都设置对应的值。 </div>
<b>vswitch_ids</b>	list	是	交换机ID。List长度范围为[1, 3]。
<b>worker_data_disk</b>	bool	是	是否挂载数据盘, 在安全沙箱容器场景下, 必须选择true。
<b>worker_data_disk_size</b>	string	是	数据盘大小, 单位为GiB。安全沙箱容器场景下, 必须大于200。
<b>worker_instance_type</b>	string	是	<p>安全沙箱容器实例规格。</p> <p>当前支持的实例规格为ecs.ebmg5s.24xlarge和ecs.ebmc5s.24xlarge。</p>
<b>worker_system_disk_category</b>	string	是	Worker节点系统盘类型。

名称	类型	必须	描述
<b>worker_system_disk_size</b>	int	是	Worker节点系统盘大小，单位为GiB。
<b>cloud_monitor_flags</b>	bool	否	是否安装云监控插件： <ul style="list-style-type: none"><li>• true：表示安装。</li><li>• false：表示不安装。</li></ul>
<b>disable_rollback</b>	bool	否	失败是否回滚： <ul style="list-style-type: none"><li>• true：表示失败不回滚。</li><li>• false：表示失败回滚。</li></ul> 默认为true，如果选择失败回滚，则会释放创建过程中所生产的资源，不推荐使用false。
<b>endpoint_public_access</b>	bool	否	是否开启公网API Server： <ul style="list-style-type: none"><li>• true：默认为True，表示开放公网API Server。</li><li>• false：若设置为false，则不会创建公网的API Server，仅创建私网的API Server。</li></ul>
<b>service_cidr</b>	string	否	服务网段，不能和VPC网段以及容器网段冲突。当选择系统自动创建VPC时，默认使用172.19.0.0/20网段。
<b>tags</b>	list	否	给集群打tag标签： <ul style="list-style-type: none"><li>• key：标签名称。</li><li>• value：标签值。</li></ul>
<b>timeout_mins</b>	int	否	集群资源栈创建超时时间，以分钟为单位，默认值60。
<b>worker_auto_renew</b>	bool	否	是否开启Worker节点自动续费，可选值为： <ul style="list-style-type: none"><li>• true：自动续费。</li><li>• false：不自动续费。</li></ul>

名称	类型	必须	描述
<b>worker_auto_renew_period</b>	int	否	<p>自动续费周期，当选择预付费和自动续费时才生效，且为必选值：</p> <ul style="list-style-type: none"> <li>PeriodUnit=Week时，取值：{ "1" , "2" , "3" }。</li> <li>PeriodUnit=Month时，取值{ "1" , "2" , "3" , "6" , "12" }。</li> </ul>
<b>worker_data_disk_category</b>	string	否	<p>数据盘类型，只在挂载Worker节点数据盘时有效，取值范围：</p> <ul style="list-style-type: none"> <li>cloud：普通云盘。</li> <li>cloud_efficiency：高效云盘。</li> <li>cloud_ssd：SSD云盘。</li> </ul>
<b>worker_instance_charge_type</b>	string	否	<p>Worker节点付费类型，可选值为：</p> <ul style="list-style-type: none"> <li>PrePaid：预付费。</li> <li>PostPaid：按量付费。</li> </ul> <p>默认为按量付费。</p>
<b>worker_period</b>	int	否	<p>包年包月时长，当<b>worker_instance_charge_type</b>取值为<b>PrePaid</b>时才生效且为必选值，取值范围：</p> <ul style="list-style-type: none"> <li>PeriodUnit=Week时，Period取值：{ "1" , "2" , "3" , "4" }。</li> <li>PeriodUnit=Month时，Period取值：{ "1" , "2" , "3" , "4" , "5" , "6" , "7" , "8" , "9" , "12" , "24" , "36" , "48" , "60" }。</li> </ul>
<b>worker_period_unit</b>	string	否	<p>当指定为PrePaid的时候需要指定周期。可选择为：</p> <ul style="list-style-type: none"> <li>Week：以周为计时单位。</li> <li>Month：以月为计时单位。</li> </ul>

## 返回信息

返回行ResponseLine

HTTP/1.1 202 Accepted

特有返回头ResponseHead

无，请参见[公共返回头部](#)。

**表 3-8: 返回体ResponseBody**

名称	类型	描述
cluster_id	string	集群实例ID。
request_id	string	请求ID。
task_id	string	任务ID。系统自动分配，用户查询任务状态。

## 示例

请求示例

```
POST /clusters HTTP/1.1
<公共请求头>
{
  "name": "test-sandbox",
  "cluster_type": "ManagedKubernetes",
  "disable_rollback": true,
  "timeout_mins": 60,
  "region_id": "cn-hangzhou",
  "snat_entry": true,
  "cloud_monitor_flags": false,
  "endpoint_public_access": true,
  "worker_instance_type": "ecs.ebmg5s.24xlarge",
  "num_of_nodes": 2,
  "vpcid": "vpc-bp1iybm49v9jgb50xxxxx",
  "vswitch_ids": ["vsw-bp1ue9z93i9zpcblxxxxx"],
  "service_cidr": "192.168.0.0/16",
  "login_password": "xxxxxxxx",
  "worker_system_disk_category": "cloud_efficiency",
  "worker_system_disk_size": 200,
  "worker_data_disk_category": "cloud_efficiency",
  "worker_data_disk_size": 200,
  "worker_data_disk": true,
  "worker_instance_charge_type": "PostPaid",
  "kubernetes_version": "1.14.6-aliyun.1",
  "addons": [{"name": "terway-eniip"}],
  "runtime": {"name": "Sandboxed-Container.runv", "version": "1.0.0"},
  "pod_vswitch_ids": ["vsw-bp18nj6xxfoc2ci2xxxxx"]
}
```

返回示例

HTTP/1.1 202 Accepted

```
<公共响应头>
{
  "cluster_id": "cb95aa626a47740afbf6aa099b65****",
  "request_id": "687C5BAA-D103-4993-884B-C35E4314A1E1",
  "task_id": "T-5a54309c80282e39ea00002f"
}
```

## 3.6 扩容 Kubernetes 多可用区集群

调用 **ScaleOutCluster** 增加集群中 Worker 节点的数量（支持多可用区实例）。

### 请求信息

请求行 RequestLine

```
POST /api/v2/clusters/{cluster_id} HTTP/1.1
```

表 3-9: 请求行参数 URI Param

名称	类型	是否必须	描述
cluster_id	string	是	集群 ID。

特有请求头 RequestHead。

无，请参见[公共请求头部](#)。

请求体 RequestBody。

```
{
  "key_pair": "keypair名称, 和login_password 二选一",
  "vswitch_ids": "一台或多台虚拟交换机 ID, N 的取值范围为 [1,3]",
  "worker_instance_types": "Worker实例规格多实例规格参数",
  "worker_system_disk_category": "Worker系统盘类型",
  "worker_system_disk_size": "Worker节点系统盘大小",
  "worker_data_disk": "是否挂载数据盘 true/false",
  "tags": "给集群打tag标签, 数组格式对象",
  "count": "扩容的数量",
  "worker_data_disks": "Worker节点数据盘配置"
}
```

表 3-10: 请求体解释

名称	类型	必须	描述
count	int	是	扩容数量。
vswitch_ids	list	否	Worker 节点的交换机 ID。

名称	类型	必须	描述
<b>worker_data_disk</b>	bool	是	是否挂载数据盘，可选择为： <ul style="list-style-type: none"><li>• true：表示 worker 节点挂载数据盘。</li><li>• false：表示 worker 节点不挂载数据盘。</li></ul>
<b>worker_instance_types</b>	list	是	Worker 节点 ECS 规格类型代码。更多详细信息，参见 <a href="#">#unique_13</a> 。
<b>key_pair</b>	string	是	keypair 名称。与 login_password 二选一。
<b>login_password</b>	string	是	扩容的 worker 节点密码。密码规则为8 - 30 个字符，且同时包含三项（大小写字母、数字和特殊符号），和 key_pair 二选一。
<b>worker_data_disks</b>	list	否	Worker 数据盘类型、大小等配置的组合。该参数只有在挂载 Worker 节点数据盘时有效，包含以下参数： <ul style="list-style-type: none"><li>• category：数据盘类型。取值范围如下。<ul style="list-style-type: none"><li>- cloud：普通云盘。</li><li>- cloud_efficiency：高效云盘。</li><li>- cloud_ssd：SSD 云盘。</li></ul></li><li>• size：数据盘大小，单位为GiB。</li></ul>
<b>worker_system_disk_category</b>	string	否	Worker 节点系统盘类型。

名称	类型	必须	描述
<b>worker_system_disk_size</b>	int	否	Worker节点系统盘大小，单位为 GiB。

## 返回信息

返回行 ResponseLine

```
HTTP/1.1 202 Accepted
```

特有返回头 ResponseHead

无，请参见[公共返回头部](#)。

返回体 ResponseBody

```
{
  "cluster_id": "string",
  "request_id": "string",
  "task_id": "string",
  "instanceId": "string"
}
```

## 示例

请求示例

```
POST /api/v2/clusters/Cccfd68c474454665ace07efce924**** HTTP/1.1
<公共请求头>
{
  "key_pair": "common", // KeyPair
  "vswitch_ids": ["vsw-uf684tfrpwup8gcs****"],
  "worker_instance_types": ["ecs.c5.xlarge"],
  "worker_system_disk_category": "cloud_efficiency",
  "worker_system_disk_size": 120,
  "worker_data_disk": false,
  "worker_data_disks": [{"category": "cloud_ssd", "size": 500}],
  "tags": [],
  "count": 1
}
```

返回示例

```
HTTP/1.1 202 Accepted
<公共响应头>
{
  "cluster_id": "Cccfd68c474454665ace07efce924****",
  "request_id": "687C5BAA-D103-4993-884B-C35E4314A1E1",
  "task_id": "T-5a54309c80282e39ea00002f",
  "instanceId": "Cccfd68c474454665ace07efce924****"
}
```



```
}

```

3.7 扩容Kubernetes边缘托管版集群

调用**ScaleOutCluster**增加集群中Worker节点数量。目前只能扩容边缘节点服务(Edge Node Service, ENS)实例。

请求信息

请求行RequestLine

```
POST /api/v2/clusters/{cluster_id} HTTP/1.1

```

表 3-11: 请求行参数URI Param

名称	类型	是否必须	描述
cluster_id	string	是	集群ID。

特有请求头RequestHead

无，请参见[公共请求头部](#)。

请求体RequestBody

```
{
  "timeout_mins": 超时时间,
  "worker_instance_type": ens型号,
  "ens_region_id": ens region id,
  "worker_image_id": 实例使用的操作系统镜像,
  "ens_internet_charge_type": 计费类型,
  "worker_period": 购买时长,
  "worker_auto_renew": 是否自动续费,
  "worker_auto_renew_period": 续费时长,
  "login_password": 扩容实例密码,
  "count": 扩容数量,
  "worker_system_disk_size": 系统盘大小,
  "worker_data_disk_size": 数据盘大小,
  "is_edge_worker": 是否为边缘节点
}
```

表 3-12: 请求体解释

名称	类型	必须	描述
count	int	是	扩容数量。
worker_instance_type	string	是	ENS型号。
ens_region_id	string	是	ENS的区域ID。

名称	类型	必须	描述
worker_image_id	string	是	实例使用的OS镜像。
ens_internet_charge_type	string	是	第一次新购时必须填写。如果用户已有资源，则按已存在有效的计费方式。 <ul style="list-style-type: none"> <li>BandwidthByDay：日峰值带宽。</li> <li>95BandwidthByMonth：月95峰值带宽。</li> </ul>
worker_period	int	是	购买资源的时长，单位为：月。取值范围：1~9, 12。
login_password	string	是	密码。
worker_system_disk_size	int	是	系统盘大小，单位为GiB。取值：20~100之间10的倍数，并且要大于镜像大小。
is_edge_worker	bool	是	取值必须为true。
worker_data_disk_size	int	否	数据盘大小。0表示不需要数据盘，20~200之间的10的倍数。
worker_auto_renew	bool	否	是否要自动续费。取值范围： <ul style="list-style-type: none"> <li>True：自动续费。</li> <li>False：不自动续费。</li> </ul> 默认值：False。
worker_auto_renew_period	int	否	每次自动续费的时长，当参数AutoRenew取值True时为必填。取值范围：1~12。
timeout_mins	int	否	超时时间。

## 返回信息

返回行ResponseLine

```
HTTP/1.1 202 Accepted
```

特有返回头ResponseHead

无，请参见[公共返回头部](#)。

返回体

```
{
  "cluster_id": "string",
  "request_id": "string",
  "task_id": "string"
}
```

## 示例

请求示例

```
POST /api/v2/clusters/Cccfd68c474454665ace07efce924**** HTTP/1.1
<公共请求头>
{
  "timeout_mins": 60,
  "worker_instance_type": "ens.sn1.tiny",
  "ens_region_id": "cn-beijing-telecom",
  "worker_image_id": "m-2QVLO2T8NYgm8CNQVg15gF",
  "ens_internet_charge_type": "BandwidthByDay",
  "worker_period": 1,
  "worker_auto_renew": true,
  "worker_auto_renew_period": 1,
  "login_password": "Hello1234!",
  "count": 1,
  "worker_system_disk_size": 20,
  "worker_data_disk_size": 0,
  "is_edge_worker": true
}
```

返回示例

```
HTTP/1.1 202 Accepted
<公共响应头>
{
  "cluster_id": "Cccfd68c474454665ace07efce924****",
  "request_id": "687C5BAA-D103-4993-884B-C35E4314A1E1",
  "task_id": "T-5a54309c80282e39ea00002f"
```

```
}
```

## 3.8 扩容安全沙箱集群

调用**ScaleOutCluster**增加安全沙箱容器集群中Worker节点的数量（支持多可用区实例）。

### 请求信息

请求行RequestLine

```
POST /api/v2/clusters/{cluster_id} HTTP/1.1
```

表 3-13: 请求行参数 URI Param

名称	类型	是否必须	描述
<b>cluster_id</b>	string	是	集群ID。

特有请求头 RequestHead

无，请参见[公共请求头部](#)。

请求体 RequestBody

```
{
  "login_password": "节点的登录密码，和keypair参数二选一",
  "worker_instance_charge_type": "PostPaid",
  "vswitch_ids": "一台或多台虚拟交换机 ID，N 的取值范围为 [1, 3]",
  "worker_instance_types": "Worker实例规格多实例规格参数",
  "worker_system_disk_category": "Worker系统盘类型",
  "worker_system_disk_size": "worker系统盘大小",
  "worker_data_disk": "是否挂载数据盘 true|false",
  "worker_data_disk_category": "Worker数据盘类型",
  "worker_data_disk_size": "worker数据盘大小",
  "tags": "给集群打tag标签，数组格式对象",
  "count": "扩容的数量"
}
```

表 3-14: 请求体解释

名称	类型	必须	描述
<b>count</b>	int	是	扩容数量。
<b>login_password</b>	string	是	扩容的worker节点密码。密码规则为8 - 30 个字符，且同时包含三项（大、小写字母，数字和特殊符号）。和 key_pair 二选一。

名称	类型	必须	描述
<b>key_pair</b>	string	是	keypair名称。与 login_password二选一。
<b>vswitch_ids</b>	list	是	Worker节点的交换机ID。
<b>worker_data_disk</b>	bool	是	是否挂载数据盘，可选择为： <ul style="list-style-type: none"><li>• true：表示worker节点挂载数据盘。</li><li>• false：表示worker节点不挂载数据盘。</li></ul> 在安全沙箱容器场景下，必须选择true。
<b>worker_data_disk_size</b>	string	是	数据盘大小，单位为GiB，安全沙箱容器场景下，必须大于200。
<b>worker_instance_types</b>	list	是	安全沙箱容器实例规格。目前支持： <ul style="list-style-type: none"><li>• ecs.ebmg5s.24xlarge</li><li>• ecs.ebmc5s.24xlarge</li><li>• ecs.ebmgn6i.24xlarge</li></ul>
<b>worker_data_disk_category</b>	string	否	数据盘类型。
<b>worker_system_disk_category</b>	string	否	Worker节点系统盘类型。
<b>worker_system_disk_size</b>	int	否	Worker节点系统盘大小，单位为GiB。

## 返回信息

返回行 ResponseLine

HTTP/1.1 202 Accepted

特有返回头 ResponseHead

无，请参见[公共返回头部](#)。

返回体 ResponseBody

```
{
  "cluster_id": "string",
  "request_id": "string",
  "task_id": "string"
}
```

## 示例

请求示例

```
POST /api/v2/clusters/Cccfd68c474454665ace07efce924**** HTTP/1.1
<公共请求头>
{
  "login_password": "xxxxxxx",
  "worker_instance_charge_type": "PostPaid",
  "vswitch_ids": ["vsw-2zes3rfz7bmk0nxxxxxxx"],
  "worker_instance_types": ["ecs.ebmg5s.24xlarge"],
  "worker_system_disk_category": "cloud_efficiency",
  "worker_system_disk_size": 120,
  "worker_data_disk": true,
  "worker_data_disk_category": "cloud_efficiency",
  "worker_data_disk_size": 200,
  "tags": [],
  "count": 1,
  "disable_rollback": false
}
```

返回示例

```
HTTP/1.1 202 Accepted
<公共响应头>
{
  "cluster_id": "Cccfd68c474454665ace07efce924****",
  "request_id": "687C5BAA-D103-4993-884B-C35E4314A1E1",
  "task_id": "T-5a54309c80282e39ea00002f"
}
```

## 3.9 获取集群 kubeconfig 接口

调用 **DescribeClusterUserKubeconfig** 返回包含当前登录用户身份信息的 Kubernetes 集群访问 kubeconfig。

请求信息

请求行 RequestLine

```
GET /k8s/{clusterid}/user_config
```

特有请求头 RequestHead

无，请参见[公共请求头部](#)。

## 返回信息

返回行 ResponseLine

```
HTTP/1.1 200 OK
```

特有返回头 ResponseHead

无，请参见[公共返回头部](#)。

返回体 ResponseBody

```
{
  "config": "string"
}
```

表 3-15: 返回体解释

名称	类型	描述
<b>clusterid</b>	String	集群 ID。
<b>config</b>	String	当前用户对应的集群访问 kubeconfig。

## 示例

请求示例

```
GET /k8s/c5b5e80b0b64a4bf6939d2d8fbbc5****/user_config HTTP/1.1
<公共请求头>
```

返回示例

```
HTTP/1.1 200 Ok
<公共响应头>
{
  "config": "xxxxxxx"
```

```
}

```

### 3.10 修改集群 tag 接口

调用 **ModifyClusterTags** 修改当前 Kubernetes 集群的 tag 接口。

请求信息

请求行 RequestLine

```
POST /clusters/{cluster_id}/tags HTTP/1.1

```

表 3-16: 请求行参数 URI Param

名称	类型	是否必须	描述
cluster_id	string	是	集群 ID。

特有请求头 RequestHead

无，请参见[公共请求头部](#)。

请求体 RequestBody

```
[
  {
    "key": "xxxxxxx",
    "value": "xxxxxxx"
  }
]

```

表 3-17: 请求体解释

名称	类型	必须	描述
key	string	否	标签名称。
value	string	否	标签值。

返回信息

返回行 ResponseLine

```
HTTP/1.1 200 OK

```

特有返回头 ResponseHead

无，请参见[公共返回头部](#)。



## 示例

### 请求示例

```
POST /clusters/c5b5e80b0b64a4bf6939d2d8fbbc5****/tags HTTP/1.1
<公共请求头>

[
  {
    "key": "xxxxxxxx",
    "value": "xxxxxxxx"
  }
]
```

### 返回示例

```
HTTP/1.1 200 Ok
<公共响应头>
```

## 3.11 查看资源 Tag 列表

调用 ListTagResources，查询可见的资源标签关系。

### 请求语法

```
GET /tags HTTPS|HTTP
```

### 请求参数

名称	类型	是否必选	示例值	描述
resource_type	String	是	CLUSTER	资源类型定义。
next_token	String	否	1d2db86sca 4384811e0b 5e8707e*****	下一个查询开始的Token。
resource_ids	String	否	["xxxxx","xxxxx"]	要查询的集群 ID 列表。
tags	String	否	[{"key":"env","value":"dev"}, {"key":"dev","value":"IT"}]	要查询的 tag 列表，采用 JSON 字符串的形式，最多包含 20 个子项限制。



#### 说明：

- 参数 resource\_ids 和 tags 二者需要选一个传入，否则提示异常信息。
- 用户查询结果仅为可见标签，不可见标签无法查询。

## 返回参数

名称	类型	示例值	描述
request_id	String	F99407AB-2FA9-489E-A259-40CF6D*****	当前请求的 Request ID。
tag_resources	Struct	-	标签资源组。
tag_resource	Array	-	标签资源。
resource_id	String	xxxxxx	集群 ID。
resource_type	String	ALIYUN::CS::CLUSTER	资源类型。
tag_key	String	env	标签键。
tag_value	String	dev	标签值。

## 示例

### 请求示例

```
GET /tags?
resource_ids=["xxx","yyy"]
&resource_type=CLUSTER
&tags=[{"key":"env","value":"dev"}, {"key":"dep","value":"IT"}]
```

### 正常返回示例

#### XML格式

```
<request_id>3D8795D9-8FF5-46B2-86E6-E3B4073169BE</request_id>
<tag_resources>
  <tag_resource>
    <resource_type>ALIYUN::CS::CLUSTER</resource_type>
    <resource_id>xxxxx****</resource_id>
    <tag_key>env</tag_key>
    <tag_value>dev</tag_value>
  </tag_resource>
</tag_resources>
```

#### JSON格式


```
{
  "request_id": "3D8795D9-8FF5-46B2-86E6-E3B4073169BE",
  "tag_resources": {
    "tag_resource": [
      {
        "resource_type": "ALIYUN::CS::CLUSTER",
        "resource_id": "xxxx****",
        "tag_key": "env",
        "tag_value": "dev"
      }
    ]
  }
}
```

```
}

```

### 3.12 添加已有ECS实例到Kubernetes集群

调用**AttachInstances**添加已有实例到集群。

 **说明：**

添加过程中会替换系统盘，需要提前做好数据备份。

请求信息

请求行RequestLine

```
POST /clusters/{cluster_id}/attach HTTP/1.1

```

表 3-18: 请求行参数URI Param

名称	类型	是否必须	描述
cluster_id	string	是	集群ID。

特有请求头RequestHead

无，请参见[公共请求头部](#)。

请求体RequestBody

```
{
  "password": "ECS实例root登录密码",
  "instances": "要添加的实例数组",
  "format_disk": "是否格式化数据盘",
  "key_pair": "密钥对",
  "tags": "给节点打tag标签, 数组格式对象"
}

```

表 3-19: 请求体解析

名称	类型	是否必须	描述
password	String	是	ECS实例密码。密码规则为8 - 30个字符，且至少同时包含三项（大小写字母、数字和特殊符号）。
instances	Array	是	已有实例的数组。
format_disk	bool	否	是否格式化数据盘。

名称	类型	是否必须	描述
<b>key_pair</b>	string	否	密钥对名称。 <ul style="list-style-type: none"><li>Windows实例，忽略该参数。默认为空。即使填写了该参数，仍旧只执行Password的内容。</li><li>Linux实例的密码登录方式会被初始化成禁止。</li></ul>
<b>tags</b>	list	否	给节点打tag标签： <ul style="list-style-type: none"><li>key：标签名称。</li><li>value：标签值。</li></ul>

## 返回信息

返回行ResponseLine

```
HTTP/1.1 202 OK
```

特有返回头ResponseHead

无，请参见[公共返回头部](#)。

返回体ResponseBody

```
{
  "list": [
    {
      "code": "200",
      "instanceId": "i-2zee3oiwcyoz7kwd****",
      "message": "successful"
    },
    {
      "code": "200",
      "instanceId": "i-2ze0lgm3y6iylcbt****",
      "message": "successful"
    }
  ],
  "task_id": "T-5a544aff80282e39ea000039"
```

```
}
```

## 示例

请求示例

```
POST /clusters/Cccfd68c474454665ace07efce924****/attach HTTP/1.1
<公共请求头>
{
  "password": "Hello1234",
  "tags": [],
  "instances": [
    "i-xxxx",
    "i-yyyy"
  ]
}
```

返回示例

```
HTTP/1.1 202 Accepted
<公共响应头>
{
  "list": [
    {
      "code": "200",
      "instanceId": "i-xxxx",
      "message": "successful"
    },
    {
      "code": "200",
      "instanceId": "i-yyyy",
      "message": "successful"
    }
  ],
  "task_id": "T-5a544aff80282e39ea000039"
}
```

## 3.13 生成Kubernetes边缘托管集群的节点接入脚本

调用**DescribeEdgeClusterAttachScripts**添加已有节点到边缘Kubernetes托管集群。该API返回唯一的可执行脚本，用户获取脚本后，在已有节点上执行即可完成该节点的接入。

### 请求信息

请求行RequestLine

```
POST /clusters/{cluster_id}/attachscript HTTP/1.1
```

表 3-20: 请求行参数URI Param

名称	类型	是否必须	描述
cluster_id	string	是	集群ID。

特有请求头RequestHead

无，请参见[公共请求头部](#)。

请求体RequestBody

```
{
  "options": "{
    \"flannelface\": \"flannel绑定网卡\",
    \"enableiptables\": \"是否启用iptables\",
    \"manageRuntime\": \"是否使用接入工具管理runtime\",
    \"nodeNameStrategy\": \"节点的命名机制\",
    \"nodeName\": \"节点名称\",
    \"nodeNamePrefix\": \"节点名称前缀\",
    \"enabledAddons\": [
      \"kube-proxy\",
      \"flannel\",
      \"coredns\"
    ]
  }"
}
```

表 3-21: 请求体解析

名称	类型	是否必须	描述
<b>options</b>	json	是	节点的接入配置。
<b>enabledAddons</b>	array	否	需要安装的组件列表，默认为空，不安装。普通节点需要配置为["kube-proxy", "flannel", "coredns"]。
<b>enableIptables</b>	bool	否	是否开启iptables，默认值true。
<b>flannelface</b>	String	否	flannel使用的网卡名。默认值是eth0。
<b>gpuVersion</b>	String	否	表示要接入的节点是否为GPU节点，默认为空，当前支持的GPU版本是Nvidia_Tesla_T4。
<b>manageRuntime</b>	bool	否	是否由edgeadm安装并检测Runtime。

名称	类型	是否必须	描述
<b>nodeName</b>	string	否	设置节点名。  只有在 <code>--nodename-strategy=customized</code> 时生效。
<b>nodeNamePrefix</b>	string	否	节点名称前缀。 <ul style="list-style-type: none"><li>如果前缀为空，则默认使用hostname。</li><li>如果不为空，则取值为prefix+UUID。</li></ul>
<b>nodeNameStrategy</b>	bool	否	节点生成策略。目前支持四种策略： <ul style="list-style-type: none"><li>hostname：主机名。</li><li>random：随机字符串。</li><li>randomWithPrefix：指定前缀+随机字符串。</li><li>customized：自定义。</li></ul> 默认值为hostname。

## 返回信息

返回行ResponseLine

```
HTTP/1.1 202 OK
```

特有返回头ResponseHead

无，请参见[公共返回头部](#)。

返回体ResponseBody

```
"wget http://aliacs-k8s-cn-hangzhou.oss-cn-hangzhou.aliyuncs.com/public/pkg/run/attach/1.12.6-aliyunedge.1/edgeadm -O edgeadm; chmod u+x edgeadm; ./edgeadm join --openapi-token=XXX --node-spec="{\"flannelface\":\"eth0\", \"enableiptables\":
```

```
true,\"assumeYes\":true,\"manageRuntime\":true,\"nodeNameStrategy\":\"hostname\\\",\\\"enabledAddons\":[\"kube-proxy\\\",\\\"flannel\\\",\\\"coredns\"]}\""
```

## 示例

### 请求示例

```
POST /clusters/Cccfd68c47445466XXXX/attachscript HTTP/1.1
<公共请求头>
{
  "options": "{
    \"flannelInterface\": \"eth0\",
    \"enableIptables\": true,
    \"manageRuntime\": true,
    \"nodeNameStrategy\": \"hostname\",
    \"enabledAddons\": [
      \"kube-proxy\",
      \"flannel\",
      \"coredns\"
    ]
  }"
}
```

### 返回示例

```
HTTP/1.1 202 Accepted
<公共响应头>
"wget http://aliacs-k8s-cn-hangzhou.oss-cn-hangzhou.aliyuncs.com/public/pkg/run/attach/1.12.6-aliyunedge.1/edgeadm -O edgeadm; chmod u+x edgeadm; ./edgeadm join --openapi-token=XXX --node-spec="{\"flannelInterface\": \"eth0\", \"enableIptables\": true, \"assumeYes\": true, \"manageRuntime\": true, \"nodeNameStrategy\": \"hostname\\\",\\\"enabledAddons\":[\"kube-proxy\\\",\\\"flannel\\\",\\\"coredns\"]}\""
```

## 3.14 添加已有ENS节点至Kubernetes边缘托管集群

调用**AttachInstances**接口添加已有ENS节点至边缘托管集群。

### 请求信息

```
POST /clusters/{cluster_id}/attach HTTP/1.1
```

表 3-22: 请求行参数URI Param

名称	类型	是否必须	描述
cluster_id	string	是	集群ID。

特有请求头RequestHead

无，请参见[公共请求头部](#)。

请求体RequestBody

```
{
```



```
"instances": "要添加的实例数组。",
"is_edge_worker": "是否为边缘节点。",
}
```

表 3-23: 请求体解析

名称	类型	是否必须	描述
instances	Array	是	已有实例的数组。
is_edge_worker	bool	是	是否为边缘节点。接入 ENS 节点时需要配置为 true。

返回信息

返回行ResponseLine

HTTP/1.1 202 OK

特有返回头ResponseHead

无，请参见[公共返回头部](#)。

返回体ResponseBody

```
{
  "list": [
    {
      "code": "200",
      "instanceId": "i-5j2zjis****",
      "message": "successful"
    },
    {
      "code": "200",
      "instanceId": "i-5j443uf****",
      "message": "successful"
    }
  ],
  "task_id": "T-5d6f733d9e408ec74f000002"
}
```

示例

请求示例

```
POST /clusters/Cccfd68c474454665ace07efce924****/attach HTTP/1.1
<公共请求头>
{
  "is_edge_worker": true
  "instances": [
    "i-xxxx",
    "i-yyyy"
  ]
}
```

```
}
```

返回示例

```
HTTP/1.1 202 Accepted
<公共响应头>
{
  "list": [
    {
      "code": "200",
      "instanceId": "i-xxxx",
      "message": "successful"
    },
    {
      "code": "200",
      "instanceId": "i-yyyy",
      "message": "successful"
    }
  ],
  "task_id": "T-5d6f733d9e408ec74f000002"
}
```

### 3.15 查询集群实例

调用 **DescribeClusterDetail** 根据集群 ID，查看集群的详细信息。

请求信息

请求行 RequestLine

```
GET /clusters/{cluster_id} HTTP/1.1
```

请求行参数 URI Param

名称	类型	是否必须	描述
cluster_id	string	是	集群 ID。

特有请求头 RequestHead

无，请参见[公共请求头部](#)。

请求体 RequestBody

无

返回信息

返回行 ResponseLine

```
HTTP/1.1 200 OK
```

特有返回头 ResponseHead

无, 请参见[公共返回头部](#)。

## 返回体 ResponseBody

```
{
  "name": "sian-devk8s",
  "cluster_id": "cdde1f21ae22e483ebcb068a6eb7f****",
  "size": 1,
  "region_id": "cn-bei****",
  "state": "running",
  "cluster_type": "ManagedKubernetes",
  "created": "2019-11-25T15:50:20+08:00",
  "updated": "2020-01-13T23:01:03+08:00",
  "init_version": "1.14.8-aliyun.1",
  "current_version": "1.14.8-aliyun.1",
  "meta_data": "{\n  \"Addons\": [\n    {\n      \"name\": \"flannel\",\n      \"version\": \"\",\n      \"disabled\": false,\n      \"required\": \"\",\n      \"config\": \"\"\n    },\n    {\n      \"name\": \"flexvolume\",\n      \"version\": \"\",\n      \"disabled\": false,\n      \"required\": \"\",\n      \"config\": \"\"\n    },\n    {\n      \"name\": \"alicloud-disk-controller\",\n      \"version\": \"\",\n      \"disabled\": false,\n      \"required\": \"\",\n      \"config\": \"\"\n    },\n    {\n      \"name\": \"logtail-ds\",\n      \"version\": \"\",\n      \"disabled\": false,\n      \"required\": \"\",\n      \"config\": \"\"\n    },\n    {\n      \"name\": \"IngressDashboardEnabled\",\n      \"version\": \"\",\n      \"disabled\": false,\n      \"required\": \"\",\n      \"config\": \"\"\n    },\n    {\n      \"name\": \"nginx-ingress-controller\",\n      \"version\": \"\",\n      \"disabled\": false,\n      \"required\": \"\",\n      \"config\": \"\"\n    },\n    {\n      \"name\": \"IngressSlsNetworkType\",\n      \"version\": \"\",\n      \"disabled\": false,\n      \"required\": \"\",\n      \"config\": \"\"\n    },\n    {\n      \"name\": \"kube-flannel-ds\",\n      \"version\": \"\",\n      \"disabled\": false,\n      \"required\": \"\",\n      \"config\": \"\"\n    }\n  ],\n  \"Capabilities\": {\n    \"AnyAZ\": true,\n    \"CSI\": true,\n    \"CpuPolicy\": true,\n    \"DeploymentSet\": true,\n    \"HpcCluster\": true,\n    \"Network\": \"Flannel\",\n    \"NodeCIDRMask\": \"25\",\n    \"NodeNameMode\": true,\n    \"ProxyMode\": \"ipvs\",\n    \"PublicSLB\": true,\n    \"SLSProjectName\": true,\n    \"SandboxRuntime\": false,\n    \"Taint\": true,\n    \"TerwayEniip\": true,\n    \"UserData\": true\n  },\n  \"ClusterDomain\": \"\",\n  \"DockerVersion\": \"18.09.2\",\n  \"EtcVersion\": \"v3.3.8\",\n  \"HasSandboxRuntime\": false,\n  \"KubernetesVersion\": \"1.14.8-aliyun.1\",\n  \"MultiAZ\": false,\n  \"NameMode\": \"\",\n  \"OSType\": \"Linux\",\n  \"Platform\": \"CentOS\",\n  \"PodVswitchId\": \"\",\n  \"Provider\": \"\",\n  \"ResourceGroupId\": \"rg-acfmyvw3wjmb3uq\",\n  \"SubClass\": \"default\",\n  \"SupportPlatforms\": null,\n  \"VersionSpec\": null,\n  \"VpcCidr\": \"192.168.0.0/16\"\n}",
  "resource_group_id": "rg-acfmyvw3wjmb3uq****",
  "instance_type": "",
  "vpc_id": "vpc-2zecuu62b9zw7a7qn****",
  "vswitch_id": "vsw-2zete8s4qocqg0mf****",
  "vswitch_cidr": "",
  "data_disk_size": 0,
  "data_disk_category": "cloud",
  "security_group_id": "sg-2zedf74ifulatvx0z2ag",
  "tags": [
    {
      "key": "ack.aliyun.com",
      "value": "cdde1f21ae22e483ebcb068a6eb7f358d"
    }
  ],
  "zone_id": "cn-beiji****",
  "-": "PayByTraffic",
  "network_mode": "vpc",
  "subnet_cidr": "172.20.0.0/16",
  "master_url": "{\n  \"api_server_endpoint\": \"https://47.93.19X.XXX:XXXXhttps://mirana.cs-cn-beijing.aliyuncx.xxx:xxxxhttps://192.168.X.XX:XXXX\"\n}",
  "external_loadbalancer_id": "lb-2ze3buguz3gx9920zwf24",
  "port": 0,
  "node_status": "",
  "cluster_healthy": "",
  "docker_version": "18.09.2",
  "swarm_mode": false,
  "gw_bridge": "",
  "upgrade_components": {
    "Kubernetes": {
      "component_name": "Kubernetes",

```

```

    "version": "1.14.8-aliyun.1",
    "next_version": "",
    "changed": "",
    "can_upgrade": false,
    "force": false,
    "policy": "",
    "ExtraVars": null,
    "ready_to_upgrade": "",
    "message": "",
    "exist": false,
    "required": false
  }
},
"private_zone": false,
"profile": "",
"deletion_protection": true,
"capabilities": null,
"enabled_migration": false,
"need_update_agent": false,
"outputs": [
  {
    "Description": "Log Info Output",
    "OutputKey": "LastKnownError",
    "OutputValue": null
  },
  {
    "Description": "Error msg of ess scaling instance",
    "OutputKey": "NodesScalingErrorInfo",
    "OutputValue": null
  },
  {
    "Description": "The mode we use in kube-proxy.",
    "OutputKey": "ProxyMode",
    "OutputValue": "ipvs"
  },
  {
    "Description": "ScalingGroup ID",
    "OutputKey": "ScalingGroupID",
    "OutputValue": "asg-2zebrr08gt5uluh7u5ar"
  },
  {
    "Description": "Ids of worker node",
    "OutputKey": "NodeInstanceIDs",
    "OutputValue": [
      "i-2ze4ymrjuocpbc0cftdj",
      "i-2ze4ymrjuocpbc0cftdk",
      "i-2ze4ymrjuocpbc0cftdl"
    ]
  },
  {
    "Description": "Count of ess scaling instance",
    "OutputKey": "NodesScalingAddedInstances",
    "OutputValue": 3
  }
]
}

```

### 返回体解释

Cluster 的格式

名称	类型	描述
<b>agent_version</b>	string	Agent 版本号。
<b>cluster_id</b>	String	集群 ID，集群的唯一标识。
<b>created</b>	string	集群的创建时间。
<b>external_loadbalancer_id</b>	String	集群负载均衡服务的 ID。
<b>master_url</b>	string	集群 Master 地址，您可以通过该地址连接您的集群进行相关操作。更多详细信息，参见 <a href="#">#unique_29</a> 。
<b>name</b>	string	集群名称，由您在创建集群时指定，在每个用户下唯一。
<b>network_mode</b>	String	集群网络模式（VPC 网络）。
<b>region_id</b>	String	集群所在地域 ID。
<b>security_group_id</b>	String	安全组 ID。
<b>size</b>	String	节点数。
<b>state</b>	String	集群状态，running stoped。
<b>updated</b>	string	最后更新时间。
<b>vpc_id</b>	string	VPC ID。
<b>vswitch_id</b>	string	VSwitch ID。

## 示例

### 请求示例

```
GET /clusters/C5b5e80b0b64a4bf6939d2d8fbbc5**** HTTP/1.1
<公共请求头>
```

### 返回示例

```
HTTP/1.1 200 Ok
<公共响应头>
{
  "agent_version": "0.5-e56dab3",
  "cluster_id": "c978ca3eaacd3409a9437db07598f****",
  "created": "2015-12-11T03:52:40Z",
  "external_loadbalancer_id": "1518f2b7e4c-cn-beijing-btc-a01",
  "master_url": "https://182.***.***.***:17589",
  "name": "my-python-cluster-039de960",
  "network_mode": "vpc",
  "region_id": "cn-bei****",
  "security_group_id": "sg-25yqj***",
  "size": 5,
  "state": "running",
  "updated": "2015-12-15T15:01:58Z",
```

```
"vpc_id": "",
"vswitch_id": ""
}
```

## 3.16 查询所有集群实例

调用 **DescribeClusters** 查看您在容器服务中创建的所有集群（包括 Swarm 和 Kubernetes 集群）。

### 请求信息

请求行 RequestLine

```
GET /clusters HTTP/1.1
```

特有请求头 RequestHead

无，请参见 [公共请求头部](#)。

请求体 RequestBody

无

### 返回信息

返回行 ResponseLine

```
HTTP/1.1 200 OK
```

特有返回头 ResponseHead

无，请参见 [公共返回头部](#)。

返回体 ResponseBody

```
[
  {
    "agent_version": "string",
    "cluster_id": "string",
    "created": "datetime",
    "external_loadbalancer_id": "string",
    "master_url": "string",
    "name": "string",
    "network_mode": "string",
    "region_id": "string",
    "security_group_id": "string",
    "size": "numbers",
    "state": "string",
    "updated": "datetime",
    "vpc_id": "string",
    "vswitch_id": "string"
  }
]
```

]

表 3-24: 返回体解释（Cluster 的格式）

名称	类型	描述
agent_version	string	Agent 版本号。
cluster_id	String	集群 ID，集群的唯一标识。
created	string	集群的创建时间。
external_loadbalancer_id	String	集群负载均衡服务的 ID。
master_url	string	集群 Master 地址，您可以通过该地址连接您的集群进行相关操作。更多详细信息，参见 <a href="#">#unique_31</a> 。
name	string	集群名称，由您在创建集群时指定，在每个用户下唯一。
network_mode	String	集群网络模式（VPC 网络：vpc）。
region_id	String	集群所在地域 ID。
security_group_id	String	安全组 ID。
size	String	节点数。
state	String	集群状态。
updated	string	最后更新时间。
vpc_id	string	VPC ID。
vswitch_id	string	VSwitch ID。

## 示例

### 请求示例

```
GET /clusters HTTP/1.1
<公共请求头>
```

### 返回示例

```
HTTP/1.1 200 OK
<公共响应头>
[
  {
    "agent_version": "0.5-e56dab3",
    "cluster_id": "c978ca3eaacd3409a9437db07598f****",
    "created": "2015-12-11T03:52:40Z",
    "external_loadbalancer_id": "1518f2b7e4c-cn-beijing-btc-a01",
```

```

    "master_url": "https://182.92.245.56:17589",
    "name": "my-python-cluster-039de960",
    "network_mode": "vpc",
    "region_id": "cn-beijing",
    "security_group_id": "sg-25yqjuxhz",
    "size": 5,
    "state": "running",
    "updated": "2015-12-15T15:01:58Z",
    "vpc_id": "",
    "vswitch_id": ""
  },
  {
    "agent_version": "0.5-e56dab3",
    "cluster_id": "c1eb19e0093204cbb86c3a80334d2****",
    "created": "2015-12-15T14:26:58Z",
    "external_loadbalancer_id": "151a6099de1-cn-beijing-btc-a01",
    "master_url": "https://182.92.245.56:11905",
    "name": "my-test-cluster-002b3f3d",
    "network_mode": "vpc",
    "region_id": "cn-beijing",
    "security_group_id": "sg-25rg2ws9f",
    "size": 1,
    "state": "running",
    "updated": "2015-12-15T14:43:55Z",
    "vpc_id": "",
    "vswitch_id": ""
  }
]

```

## 3.17 移除节点

调用 **DeleteClusterNodes** 从 Kubernetes 集群中移除多余的 Worker 节点。



### 说明:

- 移除节点会涉及 Pod 迁移，可能会影响业务，请在业务低峰期操作。
- 操作过程中可能存在非预期风险，请提前做好相关的数据备份。
- 操作过程中，后台会把当前节点设置为不可调度状态。
- 移除节点仅移除 Worker 节点，不会移除 Master 节点。

### 请求信息

#### 请求行 RequestLine

POST /clusters/{cluster\_id}/nodes HTTP/1.1

#### 请求行参数 URI Param

名称	类型	是否必须	描述
cluster_id	string	是	集群 ID。

#### 特有请求头 RequestHead



无，请参见[公共请求头部](#)。

### 请求体 RequestBody

```
{
  "nodes": "要移除的node_name数组",
  "release_node": "是否同时释放ecs节点"
}
```

### 请求体解析

名称	类型	是否必须	描述
<b>nodes</b>	Array	是	已有实例的数组。
<b>release_node</b>	bool	否	是否同时释放 ECS 节点，默认不释放。

### 返回信息

#### 返回行 ResponseLine

HTTP/1.1 202 OK

#### 特有返回头 ResponseHead

无，请参见[公共返回头部](#)。

### 示例

#### 请求示例

```
POST /clusters/Cccfd68c474454665ace07efce924****/attach HTTP/1.1
<公共请求头>
{
  "password": "Hello1234",
  "nodes": [
    "i-2ze74qyw9s3ge4k0****",
    "i-2zeipuf9l4tj659q****"
  ]
}
```

#### 返回示例

```
HTTP/1.1 202 Accepted
<公共响应头>
```

### 3.18 删除集群

调用 **DeleteCluster** 根据集群 ID，删除集群实例，并释放集群所有节点资源。

**请求信息**

请求行 RequestLine

```
DELETE /clusters/{cluster_id} HTTP/1.1
```

表 3-25: 请求行参数 URI Param

名称	类型	是否必须	描述
cluster_id	string	是	集群 ID。

特有请求头 RequestHead

无，请参见[公共请求头部](#)。

请求体 RequestBody

无

**返回信息**

返回行 ResponseLine

```
HTTP/1.1 202 Accepted
```

特有返回头 ResponseHead

无，请参见[公共返回头部](#)。

返回体 ResponseBody

无

**示例**

请求示例

```
DELETE /clusters/Cccfd68c474454665ace07efce924**** HTTP/1.1
<公共请求头>
```

返回示例

```
HTTP/1.1 202 Accepted
<公共响应头>
```