阿里云
容器服务Kubernetes版

常见问题

文档版本：20190725

法律声明

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

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

通用约定

格式	说明	样例
	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。	 **禁止:** 重置操作将丢失用户配置数据。
	该类警示信息可能导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **警告:** 重启操作将导致业务中断，恢复业务所需时间约10分钟。
	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 **说明:** 您也可以通过按Ctrl + A选中全部文件。
>	多级菜单递进。	设置 > 网络 > 设置网络类型
粗体	表示按键、菜单、页面名称等UI元素。	单击 **确定**。
`courier` 字体	命令。	执行 `cd /d C:/windows` 命令，进入Windows系统文件夹。
*## *	表示参数、变量。	`bae log list --instanceid` *`Instance_ID`*
[]或者[a|b]	表示可选项，至多选择一个。	`ipconfig` *`[-all|-t]`*
{}或者{a|b}	表示必选项，至多选择一个。	`swich` *`{stand | slave}`*

目录

1 漏洞修复公告

1.1 修复runc漏洞CVE-2019-5736的公告

阿里云容器服务已修复runc漏洞CVE-2019-5736。本文介绍该漏洞的影响范围及解决方法。

背景信息

Docker、containerd或者其他基于runc的容器在运行时存在安全漏洞，攻击者可以通过特定的容器镜像或者exec操作获取到宿主机runc执行时的文件句柄并修改掉runc的二进制文件，从而获取到宿主机的root执行权限。

漏洞CVE-2019-5736的详细信息，请参见CVE-2019-5736。

影响范围

- 对于阿里云容器服务而言，影响范围如下：

  Docker版本 < 18.09.2 的所有Docker Swarm集群和Kubernetes集群（不包含Serverless Kubernetes集群）
- 对于用户自建的Docker/Kubernetes环境而言，影响范围如下：

  Docker版本 < 18.09.2 或者使用 runc版本 <= 1.0-rc6的环境。

解决方法

阿里云容器服务已经修复该漏洞，新创建的1.11或1.12版本的Kubernetes集群中的Docker版本已修复该漏洞。您可以通过以下方法修复已有集群中的漏洞：

- 升级Docker。升级已有集群的Docker到18.09.2或以上版本。该方案会导致容器和业务中断。

- 仅升级runc（针对Docker版本17.06）。为避免升级Docker引擎造成的业务中断，可以按照以下步骤，逐一升级集群节点上的runc二进制。

  1. 执行以下命令定位docker-runc。docker-runc通常位于*/usr/bin/docker-runc*路径下。

     ```
     which docker-runc
     ```

  2. 执行以下命令备份原有的runc：

     ```
     mv /usr/bin/docker-runc /usr/bin/docker-runc.orig.$(date -Iseconds
     )
     ```

  3. 执行以下命令下载修复的runc：

     ```
     curl -o /usr/bin/docker-runc -sSL https://acs-public-mirror.oss-cn
     -hangzhou.aliyuncs.com/runc/docker-runc-17.06-amd64
     ```

  4. 执行以下命令设置docker-runc的可执行权限：

     ```
     chmod +x /usr/bin/docker-runc
     ```

  5. 执行以下命令测试runc是否可以正常工作：

     ```
     docker-runc -v
     # runc version 1.0.0-rc3
     # commit: fc48a25bde6fb041aae0977111ad8141ff396438
     # spec: 1.0.0-rc5
     ```

```
docker run -it --rm ubuntu echo OK
```

6. 如果是Kubernetes集群中的GPU节点，还需要完成以下步骤继续安装下nvidia-runtime。

   a. 执行以下命令定位nvidia-container-runtime。nvidia-container-runtime通常位于*/usr/bin/nvidia-container-runtime*路径下。

   ```
   which nvidia-container-runtime
   ```

   b. 执行以下命令备份原有的nvidia-container-runtime：

   ```
   mv /usr/bin/nvidia-container-runtime /usr/bin/nvidia-container-
   runtime.orig.$(date -Iseconds)
   ```

   c. 执行以下命令下载修复的nvidia-container-runtime：

   ```
   curl -o /usr/bin/nvidia-container-runtime -sSL https://acs
   -public-mirror.oss-cn-hangzhou.aliyuncs.com/runc/nvidia-
   container-runtime-17.06-amd64
   ```

   d. 执行以下命令设置nvidia-container-runtime的可执行权限：

   ```
   chmod +x /usr/bin/nvidia-container-runtime
   ```

   e. 执行以下命令测试nvidia-container-runtime是否可以正常工作：

   ```
   nvidia-container-runtime -v
   #  runc version 1.0.0-rc3
   #  commit: fc48a25bde6fb041aae0977111ad8141ff396438-dirty
   #  spec: 1.0.0-rc5

   docker run -it --rm -e NVIDIA_VISIBLE_DEVICES=all ubuntu nvidia
   -smi -L
   #  GPU 0: Tesla P100-PCIE-16GB (UUID: GPU-122e199c-9aa6-5063-
   0fd2-da009017e6dc)
   ```

   说明：

   本测试运行在GPU P100机型中，不同GPU型号测试方法会有区别。

1.2 修复Kubernetes Dashboard漏洞*CVE-2018-18264*的公告

阿里云容器服务Kubernetes 已修复Dashboard漏洞*CVE-2018-18264*，本文介绍该漏洞的影响版本及解决方法。阿里云容器服务Kubernetes内建的Kubernetes Dashboard是托管形态且已进行过安全增强，不受此漏洞影响。

背景信息

Kubernetes社区发现Kubernetes Dashboard安全漏洞*CVE-2018-18264*：使用Kubernetes Dashboard v1.10及以前的版本有跳过用户身份认证，及使用Dashboard登录账号读取集群秘钥信息的风险 。

阿里云容器服务Kubernetes内建的Kubernetes Dashboard是托管形态且已进行过安全增强，不受此漏洞影响。

安全漏洞*CVE-2018-18264*的详细信息，请参考：

- https://github.com/kubernetes/dashboard/pull/3289
- https://github.com/kubernetes/dashboard/pull/3400
- https://github.com/kubernetes/dashboard/releases/tag/v1.10.1

影响版本

如果您的Kubernetes集群中独立部署了Kubernetes Dashboard v1.10及之前版本（v1.7.0-v1.10.0），同时支持登录功能且使用了自定义证书。

解决方法

- 如果您不需要独立部署的Dashboard，请执行以下命令，将Kubernetes Dashboard从集群中删除。

```
kubectl --namespace kube-system delete deployment kubernetes-dashboard
```

- 如果您需要独立部署的Dashboard，请将Dashboard升级到v1.10.1版本，请参考https://github.com/kubernetes/dashboard/releases/tag/v1.10.1。
- 如果您使用阿里云容器服务Kubernetes版本托管的Dashboard，由于阿里云容器服务Kubernetes版本对此进行过安全增强，不受此漏洞影响，您仍可以直接在容器服务管理控制台上使用Dashboard。

1.3 修复Kubernetes漏洞*CVE-2018-1002105*公告

阿里云容器服务Kubernetes 已修复漏洞*CVE-2018-1002105*，本文介绍该漏洞的影响及解决方法。

背景信息

Kubernetes社区发现安全漏洞：*CVE-2018-1002105*。Kubernetes用户可通过伪造请求，在已建立的API Server连接上提升权限访问后端服务，目前阿里云容器服务Kubernetes 已第一时间修复此漏洞，请登录容器服务管理控制台升级您的Kubernetes版本。

漏洞*CVE-2018-1002105*详细信息，请参考：https://github.com/kubernetes/kubernetes/issues/71411。

影响版本

- Kubernetes v1.0.x-1.9.x

- Kubernetes v1.10.0-1.10.10 (fixed in v1.10.11)
- Kubernetes v1.11.0-1.11.4 (fixed in v1.11.5)
- Kubernetes v1.12.0-1.12.2 (fixed in v1.12.3)

影响配置

- 容器服务Kubernetes集群启用了扩展API Server，并且kube-apiserver与扩展API Server的网络直接连通。
- 容器服务Kubernetes集群开放了 pod exec/attach/portforward 接口，用户可以利用该漏洞获得所有的kubelet API访问权限。

阿里云容器服务Kubernetes集群配置

- 阿里云容器服务Kubernetes集群的API Server默认开启了RBAC，通过主账号授权管理默认禁止了匿名用户访问。同时Kubelet 启动参数为`anonymous-auth=false`，提供了安全访问控制，防止外部入侵。
- 对于使用子账号的多租户容器服务Kubernetes集群用户，子账号可能通过pod exec/attach/portforward 接口越权访问。如果集群只有管理员用户，则无需过度担心。
- 子账号在不经过主账号自定义授权的情况下默认不具有聚合API资源的访问权限。

解决方法

请登录容器服务管理控制台，升级您的集群，升级的注意事项及具体的操作步骤，请参考#unique_7。

- 如果您的集群版本为1.11.2请升级到1.11.5。
- 如果您的集群版本为1.10.4请升级到1.10.11或1.11.5版本。
- 如果您的集群版本为1.9及以下版本，请升级到1.10.11或1.11.5版本。在1.9版本升级1.10或1.11版本时，如集群使用了云盘数据卷，需在控制台先升级flexvolume插件。

> 说明:
> 在容器服务管理控制台上，选择目标集群，单击导航栏更多 > 系统组件升级，在系统组件升级页面，选择flexvolume组件，单击升级。

由于Serverless Kubernetes在此漏洞发生前已进行加固，用户不受影响。

1.4 修复Kubectl cp漏洞*CVE-2019-11246*的公告

Kubernetes 最近公布了另一个kubectl cp 相关漏洞 CVE-2019-11246 ，此漏洞可能允许攻击者利用kubectl cp命令，采用路径遍历（Path Traversal）的方式将容器tar包中的恶意文件写入所在主机上的任何路径，该过程仅受本地用户的系统权限限制。 详细信息请参见Kubernetes披露。

背景信息

该漏洞与不久前的CVE-2019-1002101漏洞影响相似，由于之前的相关漏洞修复。

kubectl cp命令用于用户容器和主机之间的文件拷贝，当从容器中拷贝文件时，Kubernetes 会首先在容器中执行tar命令创建相应的归档文件，然后发送给客户端，kubectl会在用户主机上进行相应解压操作。

如果容器tar包中包含恶意文件，当攻击者具有kubectl cp命令的执行权限时，可以利用路径遍历(Path Traversal。

官方修复pr请参见 https://github.com/kubernetes/kubernetes/pull/76788。

影响范围

- kubectl v1.11.x 及以前版本
- kubectl v1.12.1-v1.12.8（fixed in v1.12.9）
- kubectl v1.13.1-v1.13.5（fixed in v1.13.6）
- kubectl v1.14.1（fixed in v1.14.2）

> 说明:
> 您可以通过运行`kubectl version --client`命令，来查看 kubectl 版本。

解决方案

通过升级kubectl 的版本来修复该漏洞。请参考安装 Kubectl，升级kubectl客户端，安装成功后请再次确认客户端版本号。

- 如果您的 kubectl 版本为1.12.x 请升级到1.12.9。
- 如果您的 kubectl 版本为1.13.x 请升级到1.13.6。
- 如果您的 kubectl 版本为1.14.x，请升级到1.14.2。
- 如果您的 kubectl 版本为1.11及以下版本，请升级到1.12.9、1.13.6或1.14.2版本。

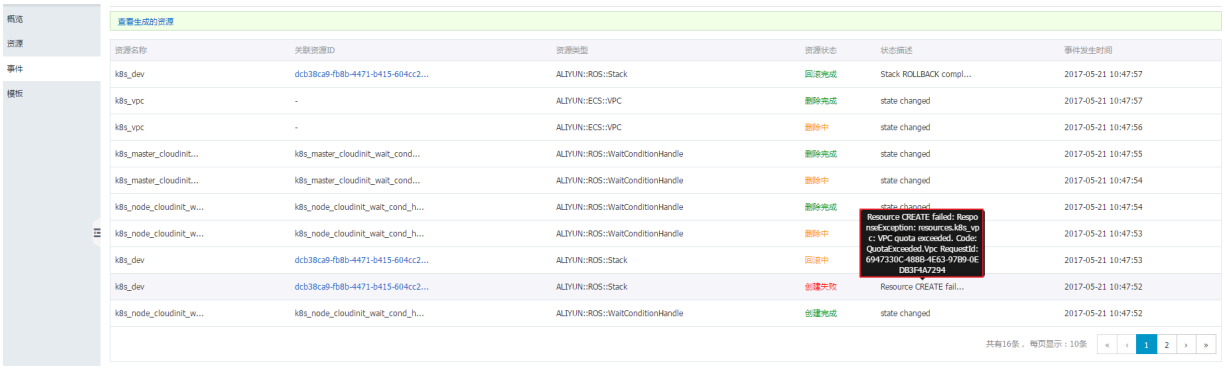
2 集群创建失败

查看失败原因

您可以通过查看集群的创建事件来查看集群创建失败的原因。

登录 ROS 管理控制台

选择集群所在的地域，选择所需的集群并单击右侧的管理，单击左侧导航栏中的事件，将鼠标移动到失败事件上查看具体的失败报错信息。



上图中的报错信息显示由于 VPC 达到配额导致集群创建失败。

失败异常码及解决方法

- Resource CREATE failed: ResponseException: resources.k8s_SNat_Eip: Elastic IP address quota exceeded Code: QuotaExceeded.Eip

  解决方法：释放多余的 EIP 或者提交 VPC 工单提高 EIP 限额。

- Resource CREATE failed: ResponseException: resources.k8s_master_slb_internet: The maximum number of SLB instances is exceeded. Code: ORDER.QUANTITY_INVALID

  解决方法：释放多余的 SLB 实例或者提交 SLB 工单提高 SLB 限额。

- Resource CREATE failed: ResponseException: resources.k8s_vpc: VPC quota exceeded. Code: QuotaExceeded.Vpc

  解决方法：释放多余的 VPC 或者提交 VPC 工单提高 VPC 限额。

- Status Code: 403 Code: InvalidResourceType.NotSupported Message: This resource type is not supported;

  解决方法：ECS 没有库存或者类型不支持，请选择其他 ECS 规格重试。

- Resource CREATE failed: ResponseException: resources.k8s_master_1: The specified image does not support cloud-init. Code: ImageNotSupportCloudInit

  解决方法：使用自定义镜像创建集群，自定义镜像必须是基于最新的 CentOS 镜像。
- Resource CREATE failed: ResponseException: resources.k8s_nodes: The resource is out of stock in the specified zone. Please try other types, or choose other regions and zones. Code: OperationDenied.NoStock

  解决方法：当前所选实例规格已售罄，请选择其他可用区或实例规格。
- Resource CREATE failed: ResponseException: resources.k8s_NAT_Gateway: A route entry already exists, which CIDR is '0.0.0.0/0' Code: RouterEntryConflict.Duplicated

  解决方法：当前 VPC 路由表内已存在默认路由，请移除默认路由，或取消勾选为专有网络创建SNAT后重试。
- Resource CREATE failed: ResponseException: resources.KubernetesWorkerRole: The number of role is limited to 200. Code: LimitExceeded.Role

  解决方法：RAM 角色数量已达到配额限制，请清理部分角色或提交 RAM 工单提高配额。
- Resource CREATE failed: ResponseException: resources.k8s_NAT_Gateway: The Account failed to create order. Code: OrderFailed

  解决方法：下单失败，请提交工单咨询。
- Resource CREATE failed: ResponseException: resources.k8s_master_1: This operation is forbidden by Aliyun RiskControl system. Code: Forbidden.RiskControl

  解决方法：您的账户出现异常，详情请联系客服
- Resource CREATE failed: ResponseException: resources.k8s_master_slb_internet: Your account does not have enough balance. Code: PAY.INSUFFICIENT_BALANCE

  解决方法：创建按量付费实例需要您账户余额大于100元，请先充值。
- Resource CREATE failed: ResponseException: resources.k8s_nodes: Your account does not have enough balance. Code: InvalidAccountStatus.NotEnoughBalance

  解决方法：创建按量付费实例需要您账户余额大于100元，请先充值。
- Resource CREATE failed: WaitConditionFailure: resources.k8s_node_cloudinit_wait_cond: See output value for more details.

  解决方法：配置集群出错，请稍后重试或提交工单咨询。

- Resource CREATE failed: WaitConditionTimeout: resources.k8s_master1_cloudinit_wait_cond: 0 of 2 received:

  解决方法：配置集群出错，请稍后重试或提交工单咨询。

- Resource CREATE failed: ResponseException: resources.k8s_master_1: The request processing has failed due to some unknown error. Code: UnknownError

  解决方法：未知错误，请稍后重试或提交工单咨询。

- Resource CREATE failed: ResponseException: resources.k8s_nodes: The request processing has failed due to some unknown error. Code: UnknownError

  解决方法：未知错误，请稍后重试或提交工单咨询。

3 删除 Kubernetes 集群失败：ROS stack 无法删除

问题原因

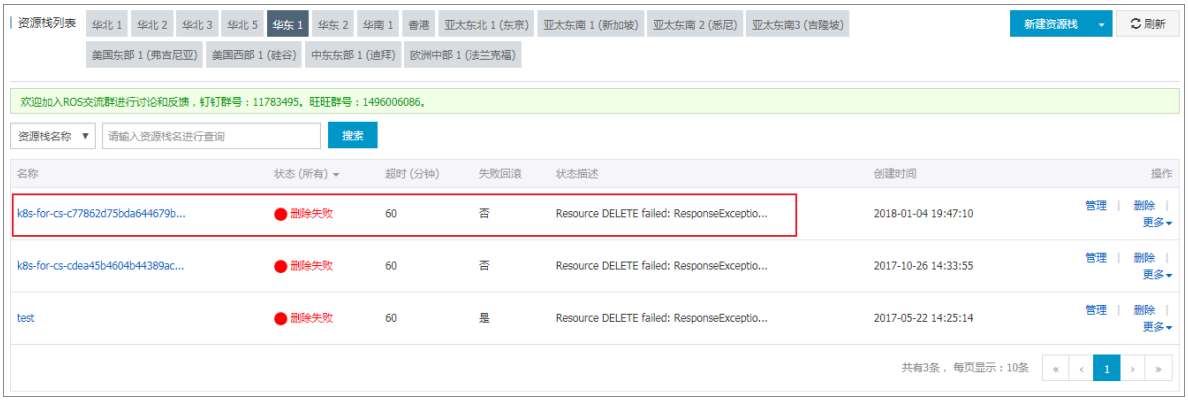
用户在 ROS 创建的资源下手动添加了一些资源（比如在 ROS 创建的 VPC 下手动添加了一个 VSwitch），ROS 是没有权限删除这些资源的。这就会导致 ROS 删除 Kubernetes 资源时无法处理该 VPC，最终导致删除失败。

> **说明:**
> 有关创建 Kubernetes 集群时 ROS 自动创建的资源，参见#unique_11。

解决办法

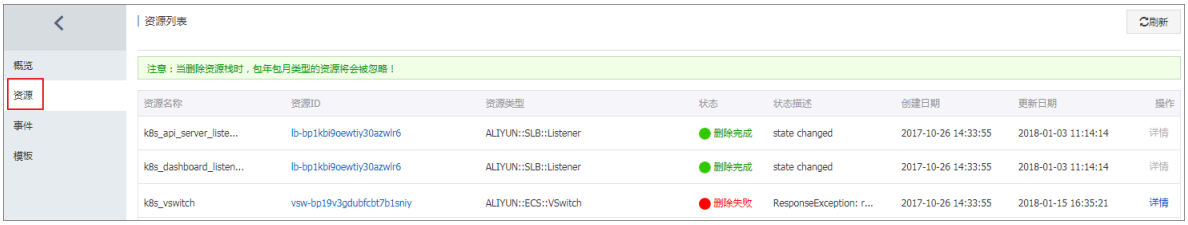
1. 集群删除失败时（集群的状态显示删除失败），跳转到 ROS 管理控制台。

2. 选择集群所在的地域，找到集群对应的资源栈 `k8s-for-cs-{cluster-id}`，可以看到其状态为删除失败。



3. 单击资源栈的名称进入资源栈详情页面，单击左侧导航栏中的资源。

   您可以看到哪些资源删除失败了。本示例中负载均衡下的 VSwitch 删除失败。



4. 进入删除失败的资源所在产品的控制台，并找到该资源。

   本示例中，登录 VPC 管理控制台，找到集群所在的 VPC，并在该 VPC 下找到删除失败的 VSwitch。

5. 单击 VSwitch 右侧的删除 尝试手动删除。

   本示例中，由于 VSwitch 下还有资源未释放，所以删除失败。

   手动释放该 VSwitch 下的资源，然后再次尝试删除该 VSwitch。

6. 使用类似的方法手动删除 Kubernetes 集群下所有删除失败的资源，然后再次尝试删除 Kubernetes 集群。

4 收集 Kubernetes 诊断信息

当Kubernetes集群出现问题，或者节点异常时，您需要收集Kubernetes诊断信息。

说明:

- 当集群异常时，需要在master节点完成收集。
- 当worker节点异常时，则需要在master节点和异常的worker节点上完成收集。

完成以下步骤在master/worker节点收集诊断信息。

1. 在 master/worker节点下载诊断脚本，并增加运行权限。

```
curl -o /usr/local/bin/diagnose_k8s.sh http://aliacs-k8s-cn-hangzhou
.oss-cn-hangzhou.aliyuncs.com/public/diagnose/diagnose_k8s.sh
chmod u+x /usr/local/bin/diagnose_k8s.sh
```

2. 执行诊断脚本。

```
diagnose_k8s.sh
 ......
 + echo 'please get diagnose_1514939155.tar.gz for diagnostics'
 ##每次执行诊断脚本，产生的日志文件的名称不同
 please get diagnose_1514939155.tar.gz for diagnostics
 + echo '请上传 diagnose_1514939155.tar.gz'
 请上传 diagnose_1514939155.tar.gz
```

3. 列出并上传产生的日志文件。

```
cd /usr/local/bin
ls -ltr|grep diagnose_1514939155.tar.gz            ##注意替换为生成的日
志文件名
```

5 Helm 手动升级

登录到Kubernetes集群master节点，参见通过 kubectl 连接 Kubernetes 集群。

执行以下命令。

```
helm init --tiller-image registry.cn-hangzhou.aliyuncs.com/acs/tiller:
v2.11.0 --upgrade
```

其中镜像地址可使用对应region的vpc域名，比如杭州region的机器就可以替换为registry-vpc.cn-hangzhou.aliyuncs.com/acs/tiller:v2.11.0。

然后等待`tiller`健康检查通过，就可以通过执行`helm version`查看版本升级情况。

说明:

这里只会升级Helm 服务端版本，客户端可以通过直接下载对应的client binary 使用。

Helm 2.11.0 client 下载地址: https://github.com/helm/helm/releases/tag/v2.11.0。目前阿里云支持的最新版本为2.11.0。

Helm 客户端和服务端版本都升级完毕后，执行`helm version`命令，可看到如下信息。

```
$ helm version
Client: &version.Version{SemVer:"v2.11.0", GitCommit:"2e55dbe1fd
b5fdb96b75ff144a339489417b146b", GitTreeState:"clean"}
Server: &version.Version{SemVer:"v2.11.0", GitCommit:"2e55dbe1fd
b5fdb96b75ff144a339489417b146b", GitTreeState:"clean"}
```

6 如何支持私有镜像

```
kubectl create secret docker-registry regsecret --docker-server=
registry-internal.cn-hangzhou.aliyuncs.com --docker-username=abc@
aliyun.com --docker-password=xxxxxx --docker-email=abc@aliyun.com
```

其中：

- regsecret：指定密钥的键名称，可自行定义。
- —docker-server：指定 Docker 仓库地址。
- —docker-username: 指定 Docker 仓库用户名。
- —docker-password：指定 Docker 仓库登录密码。
- —docker-email：指定邮件地址（选填）。

yml 文件加入密钥参数。

```
containers:
    - name: foo
      image: registry-internal.cn-hangzhou.aliyuncs.com/abc/test:1.0
imagePullSecrets:
    - name: regsecret
```

其中：

- `imagePullSecrets` 是声明拉取镜像时需要指定密钥。
- `regsecret` 必须和上面生成密钥的键名一致。
- `image` 中的 Docker 仓库名称必须和 `--docker-server` 中的 Docker 仓库名一致。

详情信息参见官方文档 使用私有仓库。

实现无密钥编排

为了避免每次使用私有镜像部署时，都需要引用密钥，您可将secret添加到namespace的default service account中，参见Add ImagePullSecrets to a service account。

首先找到前面创建的拉取私有镜像的secret。

```
# kubectl get secret regsecret
NAME        TYPE                                DATA      AGE
regsecret   kubernetes.io/dockerconfigjson   1         13m
```

本例中采用手动配置的方式，修改命名空间的默认服务帐户default，从而将此secret作为imagePullSecret。

首先创建一个*sa.yaml*配置文件，将服务账号default的配置导入到该文件中。

```
kubectl get serviceaccounts default -o yaml > ./sa.yaml
```

```
cat  sa.yaml

apiVersion: v1
kind: ServiceAccount
metadata:
  creationTimestamp: 2015-08-07T22:02:39Z
  name: default
  namespace: default
  resourceVersion: "243024"              ##注意该项
  selfLink: /api/v1/namespaces/default/serviceaccounts/default
  uid: 052fb0f4-3d50-11e5-b066-42010af0d7b6
secrets:
- name: default-token-uudgeoken-uudge
```

执行`vim sa.yaml`命令，删除resourceVersion，并增加拉取镜像的密钥配置项imagePullSecrets。修改后的配置如下。

```
apiVersion: v1
kind: ServiceAccount
metadata:
  creationTimestamp: 2015-08-07T22:02:39Z
  name: default
  namespace: default
  selfLink: /api/v1/namespaces/default/serviceaccounts/default
  uid: 052fb0f4-3d50-11e5-b066-42010af0d7b6
secrets:
- name: default-token-uudge
imagePullSecrets:                    ##增加该项
- name: regsecret
```

随后用*sa.yaml*配置文件替换default的服务账号配置。

```
kubectl replace serviceaccount default -f ./sa.yaml
serviceaccount "default" replaced
```

以一个tomcat编排为例，执行`kubectl create -f` 命令创建。

```
apiVersion: apps/v1beta2 # for versions before 1.8.0 use apps/v1beta1
kind: Deployment
metadata:
  name: tomcat-deployment
  labels:
    app: tomcat
spec:
  replicas: 1
  selector:
    matchLabels:
      app: tomcat
  template:
    metadata:
      labels:
        app: tomcat
    spec:
      containers:
      - name: tomcat
        image: registry-internal.cn-hangzhou.aliyuncs.com/abc/test:1.0
              #替换为您自己的私有镜像地址
```

```
        ports:
        - containerPort: 8080
```

若配置正常，Pod会启动成功。然后执行`kubectl get pod tomcat-xxx -o yaml`**命令，你可看到以下配置项。**

```
spec:
  imagePullSecrets:
  - nameregsecretey
```

7 存储卷常见问题

存储卷挂载不上

检查Flexvolume是否安装

在master节点上执行下面命令。

```
# kubectl get pod -n kube-system | grep flexvolume

flexvolume-4wh8s             1/1       Running   0          8d
flexvolume-65z49             1/1       Running   0          8d
flexvolume-bpc6s             1/1       Running   0          8d
flexvolume-l8pml             1/1       Running   0          8d
flexvolume-mzkpv             1/1       Running   0          8d
flexvolume-wbfhv             1/1       Running   0          8d
flexvolume-xf5cs             1/1       Running   0          8d
```

查看flexvolume pod状态是否为Running，且运行的数量与节点数量相同；

如果没有安装，请参考安装插件。

如果运行状态不对，请参考插件运行日志分析。

检查动态存储插件是否安装

如果使用云盘的动态存储功能，需要确认是否安装动态存储插件，执行下面命令。

```
# kubectl get pod -n kube-system | grep alicloud-disk

alicloud-disk-controller-8679c9fc76-lq6zb     1/1 Running   0   7d
```

如果没有安装，请参考安装插件。

如果运行状态不对，请参考插件运行日志分析。

如何查看存储相关日志

Flexvolume日志（master1上执行）

执行get命令查看出错的pod：

```
# kubectl get pod -n kube-system | grep flexvolume
```

执行log命令，查看出错pod的日志：

```
# kubectl logs flexvolume-4wh8s -n kube-system
# kubectl describe pod flexvolume-4wh8s -n kube-system
```

```
#在pod描述最后若干行是pod运行状态的描述，可以根据描述分析错误；
```

云盘、NAS、OSS驱动日志查看：

```
# 查看host节点上持久化的日志；
# 如果某个Pod挂载失败，查看pod所在的节点地址：

# kubectl describe pod nginx-97dc96f7b-xbx8t | grep Node
Node: cn-hangzhou.i-bp19myla3uvnt6zihejb/192.168.247.85
Node-Selectors:  <none>

# 登陆节点，查看日志：

# ssh 192.168.247.85
# ls /var/log/alicloud/flexvolume*
flexvolume_disk.log  flexvolume_nas.log  flexvolume_o#ss.log

可以看到云盘、nas、oss挂载的日志；
```

provsioner插件日志（master1上执行）

执行get命令查看出错的pod：

```
# kubectl get pod -n kube-system | grep alicloud-disk
```

执行log命令，查看出错pod的日志：

```
# kubectl logs alicloud-disk-controller-8679c9fc76-lq6zb -n kube-
system
# kubectl describe pod alicloud-disk-controller-8679c9fc76-lq6zb -n
kube-system

#在pod描述最后若干行是pod运行状态的描述，可以根据描述分析错误；
```

Kubelet日志

```
# 如果某个Pod挂载失败，查看pod所在的节点地址：

# kubectl describe pod nginx-97dc96f7b-xbx8t | grep Node
Node: cn-hangzhou.i-bp19myla3uvnt6zihejb/192.168.247.85
Node-Selectors:  <none>

# 登陆节点，查看kubelet日志：

# ssh 192.168.247.85
# journalctl -u kubelet -r -n 1000 &> kubelet.log

# -n的值表示期望看到的日志行数；
```

上述为获取flexvolume、provsioner、Kubelet错误日志的方法，如果无法根据日志修复状态，可以附带日志信息联系阿里云技术支持；

云盘常见问题

云盘挂载失败，出现timeout错误

如果节点为手动添加，可能是由于sts权限的问题导致，需要手动配置RAM权限：#unique_18。

云盘挂载失败，出现Size错误

创建云盘对Size有如下要求。

说明:

- 普通云盘：最小5Gi
- 高效云盘：最小20Gi
- SSD云盘：最小20Gi

云盘挂载失败，出现zone错误；

ECS挂载云盘时，必须在同一个region下面的相同zone内，否则不能挂载成功。

升级系统后，云盘有时报错：input/output error

1. 升级flexvolume到v1.9.7-42e8198或以后版本。
2. 对于已经出问题的Pod，需要重建。

升级命令：

```
# kubectl set image daemonset/flexvolume acs-flexvolume=registry.cn-
hangzhou.aliyuncs.com/acs/flexvolume:v1.9.7-42e8198 -n kube-system
```

Flexvolume版本信息：可登录容器镜像服务控制台，单击左侧导航栏中的镜像搜索，搜索acs/flexvolume，获取Flexvolume最新版本信息。

NAS常见问题

NAS挂载时间太长

如果NAS卷包含的文件量很大，且在挂载模板中配置了chmod参数，可能导致挂载时间过长的问题；可以去掉chmod参数。

NAS挂载失败，出现timeouot错误

检查nas挂载点和集群是否在同一个vpc内，否则无法挂载。

OSS常见问题

oss挂载失败

- 检查使用的ak是否正确。
- 检查OSS挂载使用的URL是否网络可达。

集群升级后，容器内oss挂载目录不可用

升级集群、重启kubelet的时候，由于容器网络会重启，导致ossfs进程重启。

ossfs进程重启会导致主机与容器目录映射失效，这时需要重启容器，或重建Pod。

可以通过配置健康检查实现容器或Pod的自动重启，请参考使用阿里云 OSS。

8 如何选择Kubernetes集群网络插件：Terway和Flannel

本文详细介绍了容器服务在Kubernetes集群创建时提供的两种网络插件：Terway和Flannel，为您在创建集群选择网络插件时提供参考。

在创建Kubernetes集群时，阿里云容器服务提供两种网络插件：Terway和Flannel：

- Flannel：使用的是简单稳定的社区的Flannel CNI插件，配合阿里云的VPC的高速网络，能给集群高性能和稳定的容器网络体验，但功能偏简单，支持的特性少，例如：不支持基于Kubernetes标准的Network Policy。
- Terway：是阿里云容器服务自研的网络插件，功能上完全兼容Flannel，支持将阿里云的弹性网卡分配给容器，支持基于Kubernetes标准的NetworkPolicy来定义容器间的访问策略，支持对单个容器做带宽的限流。对于不需要使用Network Policy的用户，可以选择Flannel，其他情况建议选择Terway。了解更多Terway网络插件的相关内容，请参见如何使用Terway网络插件。

9 如何手动安装alicloud-application-controller

在阿里云容器服务中，1.10.4及以上版本默认安装alicloud-application-controller，来提供一种基于 CRD 的分批发布的能力。

说明:

Kubernetes最新集群已经默认安装alicloud-application-controller，旧版本集群可以通过手动安装的方式部署alicloud-application-controller，要求Kubernetes集群版本最低为v1.9.3。

您可通过`kubectl create -f alicloud-application-controller.yml`命令部署alicloud-application-controller，在*alicloud-application-controller.yml*中输入如下的编排模板。

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: alicloud-application-controller
  labels:
    owner: aliyun
    app: alicloud-application-controller
  namespace: kube-system
spec:
  replicas: 1
  selector:
    matchLabels:
      owner: aliyun
      app: alicloud-application-controller
  template:
    metadata:
      labels:
        owner: aliyun
        app: alicloud-application-controller
      annotations:
        scheduler.alpha.kubernetes.io/critical-pod: ''
    spec:
      tolerations:
      - effect: NoSchedule
        operator: Exists
        key: node-role.kubernetes.io/master
      - effect: NoSchedule
        operator: Exists
        key: node.cloudprovider.kubernetes.io/uninitialized
      containers:
        - name: alicloud-application-controller
          image: registry.cn-hangzhou.aliyuncs.com/acs/aliyun-app-
lifecycle-manager:0.1-c8d5da8
          imagePullPolicy: IfNotPresent
      serviceAccount: admin
```

10 安全组常见问题

本文介绍容器服务Kubernetes集群由于安全组导致网络不通的问题原因及解决方法。

问题现象

容器之间网络不通。

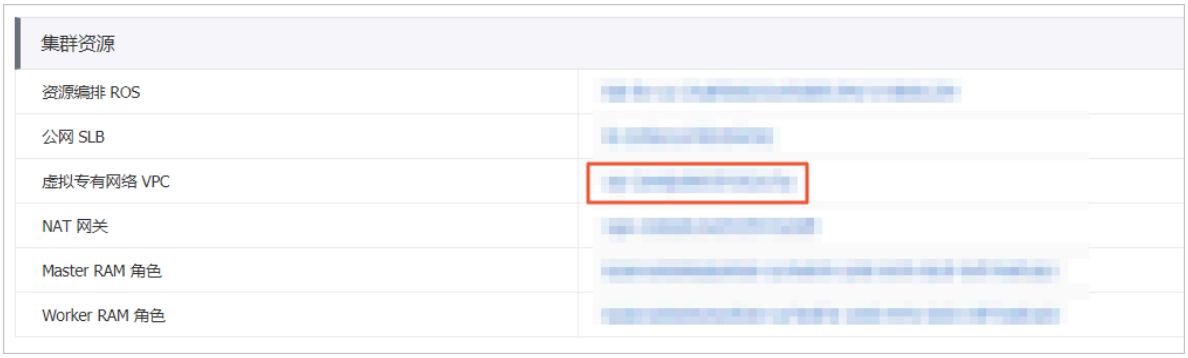
问题原因

- 入方向授权对象为Pod 网络 CIDR，且协议类型为全部的规则被删除。
- 新增ECS实例的安全组与集群所在的安全组不同。

解决方法

问题原因一：入方向授权对象为Pod 网络 CIDR，且协议类型为全部的规则被删除。

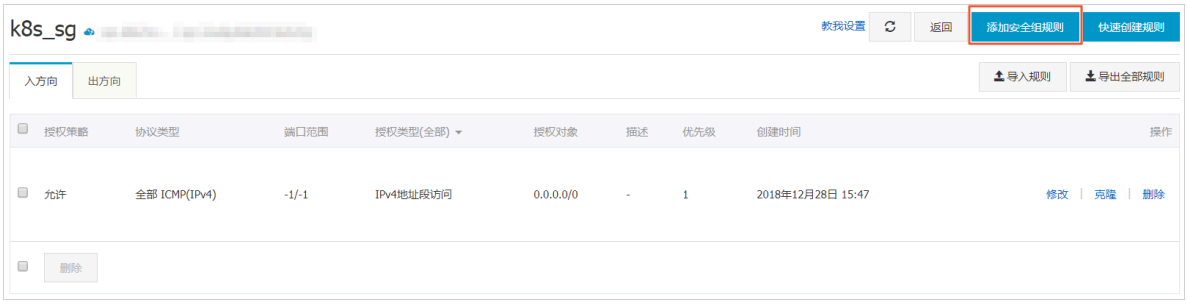
1. 登录容器服务管理控制台。
2. 在 Kubernetes 菜单下，单击左侧导航栏的集群，进入集群列表页面。
3. 单击目标集群名称，查看集群的详细信息。
4. 单击集群资源区域的虚拟专有网络 VPC，跳转到专有网络管理控制台的专有网络详情页面。



5. 单击网络资源区域的安全组右侧的数字，跳转到云服务器管理控制台的安全组列表页面。

6. 单击目标安全组操作列的配置规则，查看安全组的详细信息。

7. 在入方向页面，单击右上角添加安全组规则。



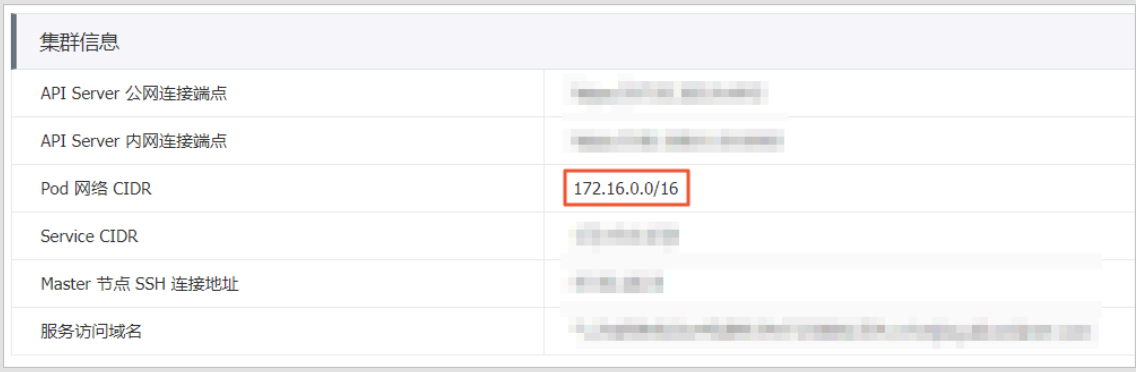
8. 填写协议类型及授权对象。

协议类型请选择全部。

授权对象填写为Pod 网络 CIDR。

说明:

- Pod 网络 CIDR可在容器服务管理控制台集群详细信息页面，集群信息区域查看。



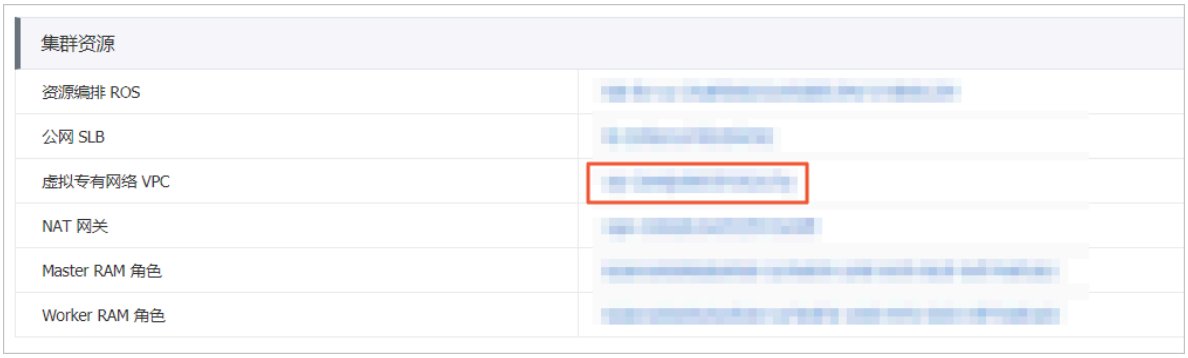
- 授权对象的设置，可参考#unique_24。

执行结果

入方向授权对象为Pod 网络 CIDR，且协议类型为全部的规则已添加。

问题原因二：新增的ECS实例的安全组与集群所在的安全组不同。

1. 登录容器服务管理控制台。
2. 单击左侧导航栏集群，在集群列表页面，选择目标集群。
3. 单击目标集群操作列的集群名称。
4. 单击集群资源区域的虚拟专有网络VPC，跳转到专有网络管理控制台查看网络资源信息。

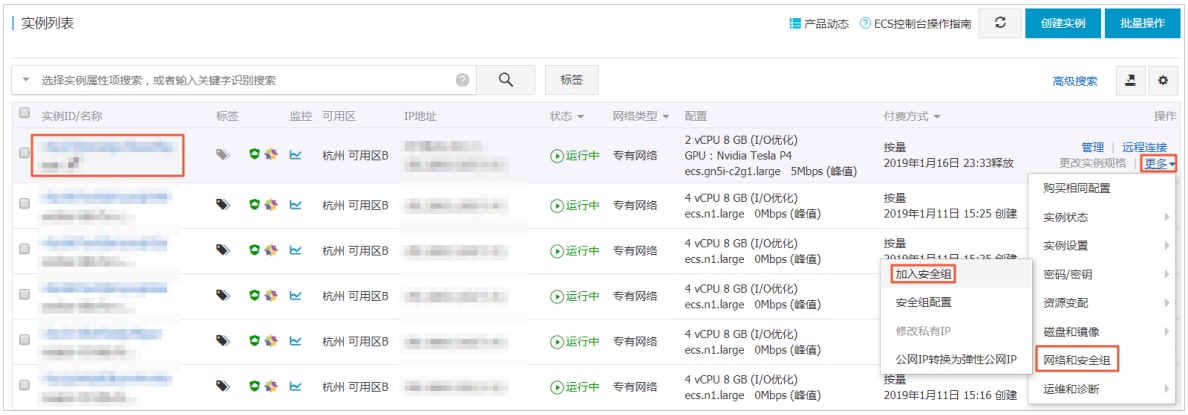


5. 在专有网络详情页面，单击网络资源区域安全组右侧的数字，跳转到云服务器ECS管理控制台的安全组列表页面查看安全组的详细信息。

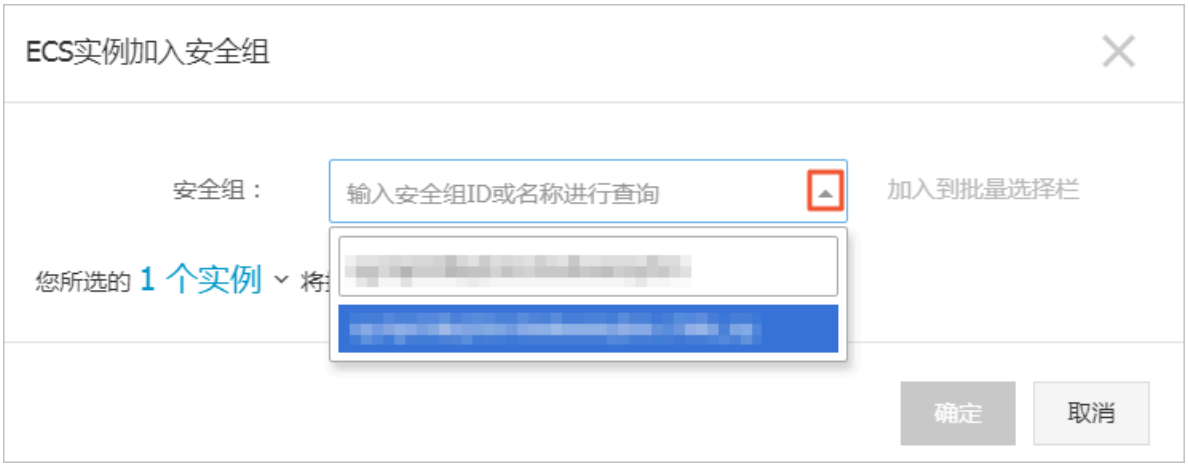
6. 在安全组列表页面，查看安全组的名称。

7. 在云服务器ECS管理控制台，单击左侧导航栏实例。

8. 在实例列表页面，单击目标实例的操作列的更多 > 网络和安全组 > 加入安全组，进入ECS实例加入安全组页面。



9. 单击安全组复选框右侧的下拉箭头，输入步骤7查询到的集群安全组名称。



10.单击确定。

执行结果

1. 单击云服务器管理控制台左侧导航栏的实例，在实例列表页面，单击目标实例的名称。
2. 单击左侧导航栏的本实例安全组。
3. 在安全组列表区域，可看到ECS实例已加入集群所在的安全组。

11 Istio 常见问题

您可通过本文了解 Istio 的常见问题及解决方法。

集群内无法访问集群外的 URL

问题现象

集群内无法访问集群外的 URL。

问题原因

缺省情况下，Istio 服务网格内的 Pod，由于其 iptables 将所有外发流量都透明的转发给了 Sidecar，所以这些集群内的服务无法访问集群之外的 URL，而只能处理集群内部的目标。

解决方法

- 通过定义 `ServiceEntry` 来调用外部服务。
- 配置 Istio 使其直接放行对特定IP地址范围的访问。

详细内容，可参考Control Egress Traffic

Tiller 版本较低

问题现象

安装过程中遇到如下提示：

```
Can't install release with errors: rpc error: code = Unknown desc =
Chart incompatible with Tiller v2.7.0
```

问题原因

Tiller 版本较低，需要升级。

解决方法

执行以下任意一条命令，升级 Tiller 版本。

> **说明:**
> 此处请升级到v2.10.0及以上版本。

升级到v2.11.0版本：

```
helm init --tiller-image registry.cn-hangzhou.aliyuncs.com/acs/tiller:
v2.11.0 --upgrade
```

升级到v2.10.0版本：

```
helm init --tiller-image registry.cn-hangzhou.aliyuncs.com/acs/tiller:
v2.10.0 --upgrade
```

说明：

Tiller版本升级后，建议将客户端也升级到相应版本，客户端下载地址，请参考https://github.com/helm/helm/releases。

CRD（Custom Resource Definitions）版本问题

问题现象

在第一次创建或升级 Istio 1.0 时遇到如下提示：

```
Can't install release with errors: rpc error: code = Unknown desc =
apiVersion "networking.istio.io/v1alpha3" in ack-istio/charts/pilot/
templates/gateway.yaml is not available
```

问题原因

CRD 不存在或版本较低，需要安装最新版本的 CRD。

说明：

仅Helm版本为2.10.0及之前版本，会遇到此问题。2.10.0之后的版本，系统自动升级 CRD。

解决方法

1. 下载新版本的 Istio，可参考Downloading the Release。
2. 执行以下命令，安装最新版本的 CRD。

```
kubectl apply -f install/kubernetes/helm/istio/templates/crds.yaml -
n istio-system
```

3. 如果启用了`certmanager`，需要执行以下命令安装相关的 CRD。

```
kubectl apply -f install/kubernetes/helm/istio/charts/certmanager/
templates/crds.yaml
```

子账号无法安装 Istio

问题现象

安装过程中遇到类似如下提示：

```
Error from server (Forbidden): error when retrieving current
configuration of:
Resource: "apiextensions.k8s.io/v1beta1, Resource=customresourcedefini
tions", GroupVersionKind: "apiextensions.k8s.io/v1beta1, Kind=
CustomResourceDefinition"
```

问题原因

当前使用账号不具有安装 Istio 的权限。

解决方法

- 切换为主账号登录。
- 对子账号赋予相应的权限，例如自定义角色中的 `cluster-admin`，可参考子账号RBAC权限配置指导。

卸载 Istio 后 CRD 残留

问题现象

卸载 Istio 后，CRD 残留。

问题原因

卸载 Istio，系统不会删除 CRD，需要执行命令删除。

解决方法

1. Helm为2.9.0及之前版本，需要执行以下命令，删除 Job 资源。

```
kubectl -n istio-system delete job --all
```

2. 执行以下命令，删除 CRD。

```
kubectl delete crd `kubectl get crd | grep -E 'istio.io|certmanager.
k8s.io' | awk '{print $1}'`
```

删除后custom resource对象 残留

问题现象

卸载 Istio 后，custom resource对象 残留。

问题原因

卸载 Istio，用户先删除了crd，但没有删除custom resource对象，需要执行命令删除。

解决方法

1. 执行`kubectl edit istio -n istio-system istio-config`命令。

2. 删除`finalizers`下的`- istio-operator.finializer.alibabacloud.com`。

12 更多问题和交流 请扫码加入钉钉交流群

关于Serverless Kubernetes集群更多问题和交流，请扫描二维码加入钉钉交流群：

13 如何给Kubernetes集群指定安全组

当前暂不支持给Kubernetes集群指定安全组。但是创建Kubernetes集群时，容器服务ACK会自动创建一个默认安全组，您可以通过修改默认安全组的规则，达到指定安全组的效果。

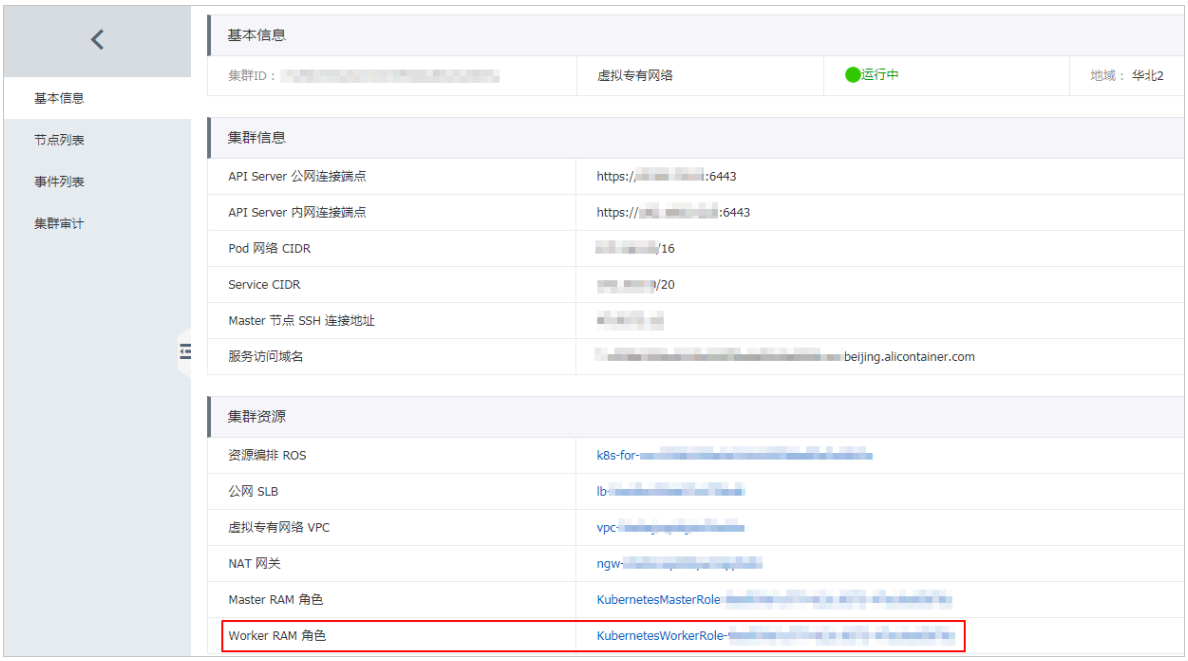
14 如何在Kubernetes集群指定RAM中的自定义角色

Kubernetes集群暂不支持指定RAM中的自定义角色，但是在Kubernetes集群中创建worker时会自动生成一个角色，您可以通过给该角色添加policy的方式，来实现角色授权。

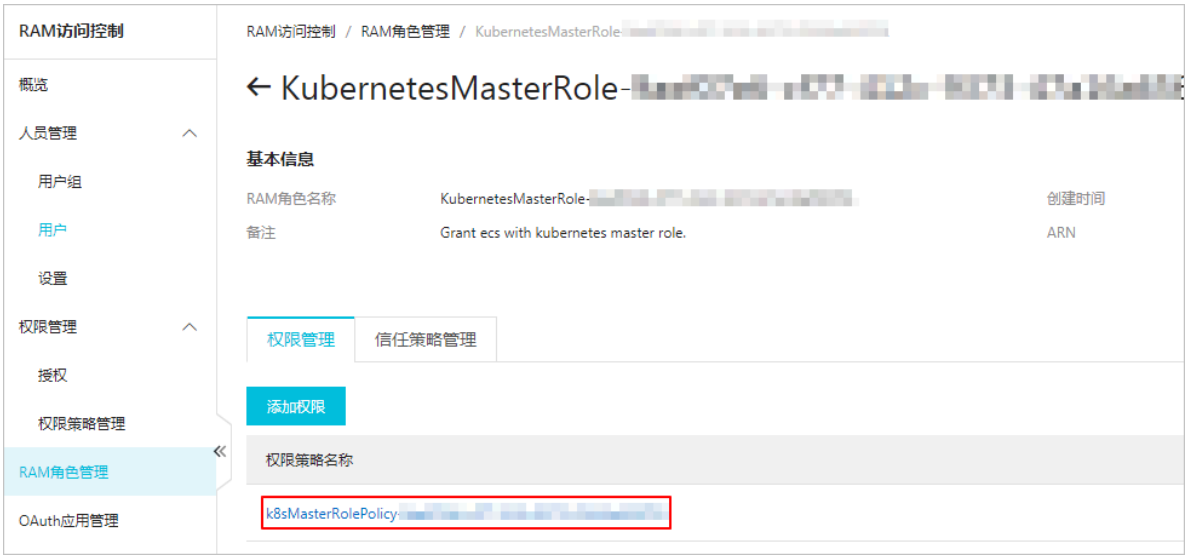
操作步骤

1. 登录 容器服务管理控制台。
2. 在 Kubernetes 菜单下，选择集群 > 集群，在集群列表页面，单击目标集群。

3. 进入基本信息页面，在集群资源区域，单击Worker RAM 角色。



4. 跳转到RAM控制台页面，在RAM 角色管理页面，单击权限策略名称。



5. 进入策略权限管理页面，单击修改策略内容，将如下内容填入后，单击确定。本示例中是给该角色授予伸缩和删除集群的权限，如果您想了解更多的权限，请参见表 1。

```
{
            "Action": [
              "cs:ScaleCluster",
              "cs:DeleteCluster"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
```

15 子账号如何给其他子账号进行RBAC授权

本文主要介绍子账号如何给其他子账号进行RBAC授权。

背景信息

默认情况下，某子账号不具备对其他子账号进行RBAC授权的能力，需要首先确保某子账号在目标集群或命名空间上被授予预置的管理员角色或自定义中的cluster-admin角色，同时给某子账号授予RAM相应权限，策略内容包括：

- 查看其他RAM子账号
- 授予RAM权限策略
- 查看RBAC权限配置
- RBAC授权能力

前提条件

某子账号已在目标集群或命名空间上被授予预置的管理员角色或自定义中的cluster-admin角色。

操作步骤

1. 授予某子账号RAM权限。

   登录RAM控制台，给某子账号授予RAM权限，请参见自定义RAM授权策略。

   授权策略内容示例参考：

```
{
    "Statement": [{
            "Action": [
                "ram:Get*",
                "ram:List*",
                "cs:GetUserPermissions",
                "cs:GetSubUsers",
                "cs:GrantPermission"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "ram:AttachPolicyToUser",
                "ram:AttachPolicy"
            ],
            "Effect": "Allow",
            "Resource":  [
                "acs:ram:*:*:policy/xxxxxx",
                "acs:*:*:*:user/*"
            ]
        }
    ],
    "Version": "1"
```

```
}
```

说明:

xxxxxx需要替换成您想要绑定的RAM策略名称。例如，您替换成*，表示某子账号拥有所有RAM策略的授权绑定能力。

2. **某子账号给其他子账号授权。**

   当某子账号完成了上述策略的绑定后，即拥有对其他子账号在指定策略范围内的RAM授权能力，以及在集群内的RBAC授权能力。您可以参考子账号RBAC权限配置指导，对其他子账号进行授权。

16 旧版本CCM如何支持SLB重命名

Cloud Controller Manager 组件 v1.9.3.10后续版本创建的SLB支持自动打TAG从而可以重命名，而 v1.9.3.10及之前的版本，您需要手动给该SLB打上一个特定的TAG从而支持SLB重命名。

前提条件

- 只有 Cloud Controller Manager 组件 v1.9.3.10及之前版本创建的SLB才需要手动打TAG的方式来支持重命名。
- service类型为Loadbalancer。

操作步骤

1. 登录到Kubernetes集群master节点，参见 通过 kubectl 连接 Kubernetes 集群。
2. 执行 `# kubectl get svc -n ${namespace} ${service}`命令，查看该service类型及IP。

```
# kubectl get svc -n ${namespace} ${service}
nginx-local   LoadBalancer   172.19.[redacted]   47.111.[redacted]   8900:31598/TCP   33d
```

> 说明:
> 您需要将${namespace}与${service}替换为所选集群的命名空间及服务名称。

3. 执行以下命令，生成该SLB所需要的TAG。

```
# kubectl get svc -n ${namespace} ${service} -o jsonpath="{.metadata.uid}"|awk -F "-" '{print "kubernetes.do.not.delete: "substr("a"$1$2$3$4$5,1,32)}'
```

```
# kubectl get svc -n ${namespace} ${service} -o jsonpath="{.metadata.uid}"|awk -F "-" '{print "ku
kubernetes.do.not.delete: a05ff996d0b3a11e999c600163f00d43
```

4. 登录负载均衡控制台负载均衡控制台根据步骤2中所获取的IP，在其所在的region搜索到该SLB。
5. 根据步骤3生成的KEY值和 VALUE值（分别对应上图的1和2），为该SLB打上一个TAG。详情请参见添加标签

17 集群证书更新说明

本文档主要为您介绍证书更新相关自动化操作说明。

当集群证书过期前两个月左右，在容器服务控制台集群列表页面会出现更新证书的红色按钮提示您更新证书，同时您会收到相应的站内信或短信通知。

您可以通过单击该红色按钮进行集群证书的自动更新，整个更新时长依据集群节点个数而定，一般5~10分钟即可完成更新。更新成功后相应证书有效期会延长五年。

注意事项

- 在证书更新过程中集群 Kubernetes 原生组件会有短暂的重启过程，建议您在非业务高峰期执行更新操作。
- 更新操作不会影响已有的线上服务。

备份

节点类型	备份内容
Master	· /etc/kubernetes/ · /var/lib/kubelet/pki · /etc/systemd/system/kubelet.service.d/10-kubeadm.conf · /etc/kubeadm/ · 需要备份的业务数据 说明: 其中，*/var/lib/kubelet/pki*和需要备份的业务数据的路径下，如果不存在数据，则无需备份。
Worker	· /etc/kubernetes/ · /etc/systemd/system/kubelet.service.d/10-kubeadm.conf · /var/lib/kubelet/pki/* · 需要备份的业务数据 说明: 其中，*/var/lib/kubelet/pki/**和需要备份的业务数据路径下，如果不存在数据，则无需备份。

证书更新

表 17-1: Master节点

证书或 conf 名称	路径
· apiserver.crt · apiserver.key	/etc/kubernetes/pki
· apiserver-kubelet-client.crt · apiserver-kubelet-client.key	/etc/kubernetes/pki
· front-proxy-client.crt · front-proxy-client.key	/etc/kubernetes/pki
· dashboard.crt · dashboard.key	/etc/kubernetes/pki/dashboard
· kubelet.crt · kubelet.key 说明: 如果不存在*kubelet.key*，则无需更新。	/var/lib/kubelet/pki 说明: 如果不存在数据，则无需更新。
admin.conf	/etc/kubernetes
kube.conf	/etc/kubernetes
controller-manager.conf	/etc/kubernetes
scheduler.conf	/etc/kubernetes
kubelet.conf	/etc/kubernetes
config	~/.kube/
· kubelet-client-current.pem 或 kubelet-client.crt\ · kubelet-client.key 说明: 如果不存在*kubelet-client.key*，则无需更新。	/var/lib/kubelet/pki 说明: 如果不存在数据，则无需更新。

表 17-2: Worker 节点

证书或 conf 名称	路径
· kubelet.crt · kubelet.key 说明: 如果不存在*kubelet.key*，则无需更新。	/var/lib/kubelet/pki 说明: 如果不存在数据，则无需更新。
· kubelet-client-current.pem 或 kubelet-client.crt · kubelet-client.key 说明: 如果不存在*kubelet-client.key*，则无需更新。	/var/lib/kubelet/pki 说明: 如果不存在数据，则无需更新。
kubelet.conf	/etc/kubernetes

18 修复GPU节点容器启动问题

问题描述

在某些特定 Kubernetes 版本中的GPU节点上，重启Kubelet和Docker时，发现没有容器被启动。

```
# service kubelet stop
Redirecting to /bin/systemctl stop kubelet.service
# service docker stop
Redirecting to /bin/systemctl stop docker.service
# service docker start
Redirecting to /bin/systemctl start docker.service
# service kubelet start
Redirecting to /bin/systemctl start kubelet.service

# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED
          STATUS              PORTS               NAMES
```

问题定位

执行如下命令，查看Docker 的 Cgroup Driver。

```
docker info | grep -i cgroup
Cgroup Driver: cgroupfs
```

此时发现的Cgroup Driver类型是cgroupfs。

解决方案

1. 备份*/etc/docker/daemon.json*，完成后，执行如下命令更新*/etc/docker/daemon.json*。

```
cat >/etc/docker/daemon.json <<-EOF
{
    "default-runtime": "nvidia",
    "runtimes": {
        "nvidia": {
            "path": "/usr/bin/nvidia-container-runtime",
            "runtimeArgs": []
        }
    },
    "exec-opts": ["native.cgroupdriver=systemd"],
    "log-driver": "json-file",
    "log-opts": {
        "max-size": "100m",
        "max-file": "10"
    },
    "oom-score-adjust": -1000,
    "storage-driver": "overlay2",
    "storage-opts":["overlay2.override_kernel_check=true"],
    "live-restore": true
}
```

```
EOF
```

2. 执行如下命令，重启 Docker 和 Kubelet。

```
# service kubelet stop
Redirecting to /bin/systemctl stop kubelet.service
# service docker restart
Redirecting to /bin/systemctl restart docker.service
# service kubelet start
Redirecting to /bin/systemctl start kubelet.service
```

3. 执行如下命令，确认Docker 的 Cgroup Driver 的类型为 systemd。

```
# docker info | grep -i cgroup
Cgroup Driver: systemd
```

19 多集群 config 合并和切换

本文为您提供了多集群 config 合并和切换的方法。

当前您可以在容器服务控制台集群详情页面，获取到当前登录用户对指定集群的 kubeconfig 访问凭证，或者通过调用 openapi 的方式获取指定 config内容，当您的阿里云账号具有多个集群的访问权限时，可以通过如下命令合并多个集群的 kubeconfig 内容合并到一个统一的凭证文件中。

```
KUBECONFIG=file1:file2:file3 kubectl config view --merge --flatten >
 ~/.kube/all-config
export KUBECONFIG=~/.kube/all-config
```

- 执行如下命令，查看所有的可使用的 context 上下文。

  ```
  kubectl config get-contexts
  ```

- 执行如下命令，查看 config 配置的帮助信息。

  ```
  kubectl config --help
  ```

- 执行如下命令，切换 context 上下文配置。

  ```
  kubectl config use-context {your-contexts}
  ```

更多 kubeconfig 相关使用介绍，请参见官方使用文档 。