集群

概览

概览

不可变基础设施

不可变基础设施

托管集群

托管集群

导入第三方集群

注册集群

公有云集群初始化

公有云集群初始化。

如何操作

创建本地集群

创建本地集群

如何操作

为内置镜像仓库添加外部地址　　更新公共仓库凭证　　使用 **KubeC**

概览

从这里开始，选择是创建平台托管集群、评估托管控制平面，还是接入现有的 Kubernetes 环境。关于集群责任、控制平面拓扑和接入边界之间的平台级关系，请参阅集群管理模型。

目录

选择集群路径

如果你需要	从以下内容开始	关键边界
让平台为受支持的提供方预配机器，并使用 Immutable OS 管理操	关于不可变基础设施	平台拥有的生命周期仅适用于受支持的提供方和工作流范围内。

如果你需要	从以下内容开始	关键边界
作系统。		
在你准备并维护的机器上创建集群。	创建本地集群	平台在节点准备完成后管理 Kubernetes；节点操作系统生命周期仍由用户负责。
评估托管控制平面（HCP）。	关于托管控制平面	请查看 HCP 文档，了解当前的成熟度、操作系统、连接性和生产使用边界。
接入现有的 Kubernetes 环境。	第三方集群接入	外部所有者、发行版或提供方通常拥有 Kubernetes、节点和基础设施的生命周期。
在直接平台访问和反向连接接入之间进行选择。	导入第三方集群 和 注册集群	导入和注册是接入方式，而不是独立的 day-2 能力模型。

平台托管的集群创建

对于 ACP 创建并进行生命周期管理的集群，请根据基础设施责任选择创建路径。

基础设施责任	适用场景	继续使用
安装程序预配基础设施（IPI）	平台应为受支持的提供方集成预配机器，并通过 Immutable OS 管理节点操作系统。	关于不可变基础设施
用户预配基础设施（UPI）	你准备物理机或虚拟机，然后平台在这些节点上安装并管理 Kubernetes。	创建本地集群

安装程序预配基础设施

在安装程序预配基础设施（IPI）中，平台通过 Immutable OS 预配机器并管理节点操作系统。此模型适用于平台拥有所选提供方集成的受支持集群预配、扩缩容和升级生命周期的场景。

目前，平台支持使用 Alauda OS 进行不可变节点管理。有关不可变基础设施概念和提供方特定工作流，请参阅关于不可变基础设施。

用户预配基础设施

在用户预配基础设施中，用户在创建集群之前准备物理机或虚拟机。平台在这些节点上安装并管理 Kubernetes，而节点操作系统管理（包括预配、补丁和替换）仍由用户控制。

当组织已经拥有成熟的基础设施或操作系统管理流程时，此模型非常适合。有关集群创建工作流，请参阅创建本地集群。

如需通过 API 创建和管理用户预配基础设施集群，请参阅 不可变基础设施 API 参考↗，并进入 **Provider APIs > User-Provisioned Infrastructure Provider APIs**。

托管控制平面

托管控制平面是一种控制平面拓扑。每个托管集群都有自己的控制平面，而多个托管控制平面作为工作负载运行在管理集群上。

在 ACP 中，HCP 通过 Kamaji（`TenantControlPlane`）实现。有关当前支持范围和评估详情，请参阅关于托管控制平面。

第三方集群接入

第三方集群是由 灵雀云 之外提供的现有 Kubernetes 环境。它们可以包括标准 Kubernetes 环境、外部 Kubernetes 发行版或公共云 Kubernetes 服务。ACP 可以在文档中列出的前提条件和注意事项下提供集中式治理和运维，但接入并不会使 ACP 成为外部集群的 Kubernetes 生命周期、节点生命周期或提供方基础设施生命周期的所有者。

第三方集群通过导入或注册工作流进行接入：

接入方式	连接模型	适用场景
导入集群	`global` 集群使用提供的地址、CA 和凭据连接到目标集群 API server。	平台能够访问目标集群 API server，且 operator 能够提供所需的集群信息。
注册集群	目标集群中的反向代理服务发起注册，并与平台建立隧道。	目标集群应主动发起连接，或者无法直接访问目标 API server。

证书、审计数据、控制平面指标、入口、存储、节点操作、连接性以及 Extension 兼容性可能存在提供方特定的注意事项。请使用导入和注册工作流页面查看详细前提条件。

有关接入工作流，请参阅第三方集群接入、导入第三方集群和注册集群。

版本兼容性

使用Kubernetes 支持矩阵选择平台托管集群创建目标、验证工作负载集群升级前提条件，并验证第三方集群接入版本。这些是针对不同决策的独立范围。

升级前提行为

- 对于 ACP 4.3 及更高版本，工作负载集群只需在 `global` 集群升级之前保持在兼容版本范围内。
- 第三方接入范围与用于该升级前提条件的兼容版本范围相互独立。

ACP 4.2 及更早版本

- 在升级 `global` 集群之前，请先将工作负载集群升级到最新记录的兼容 Kubernetes 版本。
- 请使用 Kubernetes 支持矩阵作为已文档化版本映射的数值依据。

关于不可变基础设施

不可变基础设施使用不可变操作系统来部署 Kubernetes 集群。与传统的基于 OS 的集群不同，所有节点配置都已预置到镜像中，并且在部署后保持不变。集群升级和配置更改通过使用新镜像替换节点来应用，从而在整个集群生命周期中确保一致性、可靠性并简化运维。

ACP 在以下 provider 上支持不可变基础设施：Huawei DCS、VMware vSphere 和 Huawei Cloud Stack。计划支持裸金属。

> **Note**
>
> 因为 Immutable Infrastructure 的发版周期与灵雀云容器平台不同，所以 Immutable Infrastructure 的文档现在作为独立的文档站点托管在 Immutable Infrastructure ↗。

目录

任务

以下场景涵盖了整个集群生命周期中的不可变基础设施。

场景	文档
安装 `global` 集群	在不可变基础设施上安装 global 集群 ↗
升级 `global` 集群	在不可变基础设施上升级 global 集群 ↗
为 `global` 集群规划灾难恢复	在不可变基础设施上进行 global 集群灾难恢复 ↗
安装 provider 插件	安装 ↗
配置基础设施资源	基础设施资源 ↗
创建工作负载集群	创建集群 ↗
升级工作负载集群	升级集群 ↗
管理集群节点	管理节点 ↗
配置机器（OS 或内核级别）	机器配置 ↗

API 参考

有关 Cluster API 和 provider CRD 参考，请参见 API 参考 ↗ 。

托管集群

托管集群

托管集群

导入第三方集群

导入第三方集群

导入标准 **Kubernetes** 集群

导入 **OpenS**

导入 **Amazon EKS** 集群

导入 **GKE** 集群

导入华为云

将现有 CCE 集

导入 **Azure AKS** 集群

导入现有 Azure AKS 集群作为第三方集群

导入 **Alibaba Cloud ACK** 集群

导入腾讯云

注册集群

注册集群

公有云集群初始化

网络初始化

公有云集群网络初始化。

存储初始化

公有云集群存储初始化。

如何操作

导入集群的网络配置

获取导入集群信息

信任不安全的

从自定义命名的网卡采集网络数据

从自定义命名的网卡采集网络数据

为导入的标准 **Kubernetes** 集群配置审计采集

在导入的标准 Kubernetes 集群上启用 Kubernetes 审计日志记

第三方集群接入

第三方集群是现有的 Kubernetes 环境，其 Kubernetes 发行版、节点生命周期或提供商基础设施由 灵雀云 之外的实体管理。在控制台中，这些环境显示在 托管集群 下。

当满足连接、凭据、组件前置条件、提供商要求以及 Extension 兼容性要求时，ACP 可以为第三方集群提供集中治理、资源可见性、Extension 安装、应用运维和可观测性集成。接入不会使 ACP 成为外部集群的 Kubernetes 生命周期、节点生命周期或提供商基础设施生命周期的所有者。

目录

选择接入方式

方式	连接模型	适用场景
导入集群	`global` 集群使用提供的地址、CA 和凭据连接到目标集群 API server。	平台可以访问目标 API server，且管理员可以提供所需的集群信息。
注册集群	目标集群中的反向代理服务发起注册，并与平台建立隧道。	目标集群应主动发起连接，或者 `global` 集群到目标 API server 的直接访问受到限制。

接入后，集群列表中会一致地处理高级 Day 2 运维操作。节点运维、审计数据、控制平面指标、证书、ingress、存储以及 Extension 兼容性仍可能受到提供商特定限制的影响。

有关导入流程，请参见 导入第三方集群。有关反向连接接入，请参见 注册集群。

导入第三方集群

导入第三方集群

导入标准 **Kubernetes** 集群

导入 **OpenS**

导入 **Amazon EKS** 集群

导入 **GKE** 集群

导入华为云

将现有 CCE 集

导入 **Azure AKS** 集群

导入现有 Azure AKS 集群作为第三方集群

导入 **Alibaba Cloud ACK** 集群

导入腾讯云

导入第三方集群

当 `global` 集群可以访问目标集群 API server，且管理员可以提供所需的集群地址、CA 和凭证时，请使用导入方式。

目标环境	从以下内容开始
现有标准 Kubernetes 环境	导入标准 Kubernetes 集群
外部 Kubernetes 发行版	导入 OpenShift 集群
公有云 Kubernetes 服务	导入 Amazon EKS 集群、导入 GKE 集群、导入 Azure AKS 集群、导入阿里云 ACK 集群、导入腾讯云 TKE 集群 或 导入华为云 CCE 集群

特定于提供商的操作步骤在需要时包括提供商控制台或 CLI 步骤。提供商名称用于标识操作步骤路径；它不是单独的 ACP 集群模型。

导入标准 Kubernetes 集群

使用此操作步骤可将现有的标准 Kubernetes 集群（例如使用 **kubeadm** 部署的集群）导入为第三方集群。

目录

术语

术语	描述
Provider-managed Kubernetes service	由提供方负责控制平面节点和控制平面组件的 Kubernetes 服务。用户通常无法登录控制平面节点，也无法直接对其进行维护。
Self-managed Kubernetes cluster	由集群所有者负责维护控制平面节点和集群组件的 Kubernetes 集群。

前提条件

- 集群中的 Kubernetes 及相关组件必须满足 版本和参数要求。
- 如果运行时为 Containerd，请在集成前 更新 Containerd 配置，以确保分布式存储可以成功部署。

注意事项

默认情况下，平台会监控匹配 `eth.*|en.*|wl.*|ww.*` 的 NIC 流量。如果您的 NIC 使用不同的命名规范，请在集成后按照 从自定义命名的网卡收集网络数据 更新配置。

获取 **Registry** 地址

- 要使用平台在 **global** 集群 安装期间部署的 Registry，请在 global 控制节点上运行以下命令：

```
if [ "$(kubectl get productbase -o jsonpath='{.items[].spec.registry.preferPlatformURL}')" = 'false' ]; then
    REGISTRY=$(kubectl get cm -n kube-public global-info -o jsonpath='{.data.registryAddress}')
else
    REGISTRY=$(kubectl get cm -n kube-public global-info -o jsonpath='{.data.platformURL}' | awk -F // '{print $NF}')
fi
echo "Registry address: $REGISTRY"
```

- 要使用 **external registry**，请手动设置 **REGISTRY**：

```
REGISTRY=<external-registry-address>  # e.g., registry.example.cn:60080 or 192.168.134.43
echo "Registry address: $REGISTRY"
```

检查是否需要额外的 **Registry** 配置

1. 运行以下命令，检查 Registry 是否支持使用受信任的 CA 证书进行 HTTPS 通信：

```
REGISTRY=<registry-address-from-previous-step>

if curl -s -o /dev/null --retry 3 --retry-delay 5 -- "https://${REGISTRY}/v2/"; then
    echo 'Pass: Registry uses a trusted CA certificate. No extra config needed.'
else
    echo 'Fail: Registry does not support HTTPS or uses an untrusted certificate. Follow "Trust Insecure Registry".'
fi
```

2. 如果检查失败，请参阅 如何信任不安全的 Registry？

获取集群信息

请参阅 如何获取集群信息？。

集成集群

1. 在左侧导航中，进入 **Cluster Management** > **Clusters**。
2. 单击 **Import Cluster**。
3. 按如下所示配置参数：

参数	描述
Registry	存储所需平台组件镜像的 Registry。可选项：**Platform Default**（在 global 初始化期间配置）、**Private Registry**（需要地址、端口、用户名、密码）、**Public Registry**（需要 更新云凭证）。
Cluster Info	可手动输入，或从 KubeConfig 文件中解析。必填字段：**Cluster Address**、**CA Certificate**（如果手动输入，则需进行 Base64 解码）以及 **Authentication**（具有 cluster-admin 权限的 token 或客户端证书）。

4. 单击 **Check Connectivity**。平台将验证网络访问并自动检测集群类型。
5. 如果成功，单击 **Import** 完成导入。
   *可通过 **execution progress** 对话框查看进度（status.conditions）。集成完成后，集群会在列表中显示为健康状态。*

网络配置

请确保 global 集群与导入的集群之间的连通性。

导入后配置

如果您需要平台从导入的标准 Kubernetes 集群中收集审计数据，请在导入后为该集群配置 Kubernetes API server 审计日志。请参阅 如何为导入的标准 Kubernetes 集群配置审计收集？。

常见问题

为什么 “Add Node” 按钮是禁用状态？

不支持通过平台 UI 为导入的标准 Kubernetes 集群添加节点。请直接在目标集群中添加节点，或通过集群提供方添加。

支持哪些证书？

1. **Kubernetes Certificates**：仅可查看 API Server 证书；不支持其他证书，且不会自动轮转。
2. **Platform Component Certificates**：可查看并自动轮转。

不支持哪些功能？

- **Provider-managed Kubernetes services**：不提供审计日志。
- **Provider-managed Kubernetes services**：不支持 ETCD、Scheduler 和 Controller Manager 监控，仅提供 API Server 指标。
- **All clusters**：不支持 API Server 以外的证书。

如何修复 **Containerd** 运行时导致的分布式存储部署失败？

使用 Containerd 时，除非在所有节点上调整 Containerd 设置，否则分布式存储部署将会失败：

1. 编辑 `/etc/systemd/system/containerd.service`，设置 `LimitNOFILE=1048576`。
2. 运行 `systemctl daemon-reload`。
3. 重启 Containerd：`systemctl restart containerd`。
4. 在控制节点上，重启分布式存储 Pod：

```
kubectl delete pod --all -n rook-ceph
```

导入 OpenShift 集群

使用此特定于提供商的操作步骤，将现有 OpenShift 集群作为第三方集群导入。除非文档中说明了提供商特定的能力可以负责相应生命周期，否则外部分发版所有者仍需负责 Kubernetes 分发版、节点生命周期以及提供商基础设施生命周期。

目录

前提条件

- 集群的 Kubernetes 版本和参数必须满足 标准 Kubernetes 集群要求。
- 在集成过程中，需要执行 `kubectl` 命令。请在可以访问集群的堡垒机上安装 CLI 工具。
- 若要启用对节点、工作负载（Deployment、StatefulSet、DaemonSet）、Pods 和容器等指标的实时监控，请确保目标集群中已部署 **Prometheus**。

获取 **Registry** 地址

- 若要使用平台在 **global** 集群安装期间部署的 Registry，请在 global 控制节点上运行以下命令：

```
if [ "$(kubectl get productbase -o jsonpath='{.items[].spec.registry.preferPlatformURL}')" = 'false' ]; then
    REGISTRY=$(kubectl get cm -n kube-public global-info -o jsonpath='{.data.registryAddress}')
else
    REGISTRY=$(kubectl get cm -n kube-public global-info -o jsonpath='{.data.platformURL}' | awk -F // '{print $NF}')
fi
echo "Registry address is: $REGISTRY"
```

- 若要使用 外部 **Registry**，请手动设置 **REGISTRY** 变量：

```
REGISTRY=<external-registry-address>  # e.g., registry.example.cn:60080 or 192.168.134.43
echo "Registry address is: $REGISTRY"
```

检查是否需要额外的 **Registry** 配置

1. 运行以下命令，检查 Registry 是否支持 HTTPS 且使用受信任的 CA 证书：

```
REGISTRY=<registry-address-from-previous-step>

if curl -s -o /dev/null --retry 3 --retry-delay 5 -- "https://${REGISTRY}/v2/"; then
    echo 'Pass: Registry uses a trusted CA certificate. No extra config needed.'
else
    echo 'Fail: Registry does not support HTTPS or uses an untrusted certificate. Follow "Trust Insecure Registry".'
fi
```

2. 如果检查失败，请按照以下步骤操作。

信任不安全的 **Registry**

1. 登录到所有 **OCP** 集群节点。
2. 在每个节点上，配置 Registry 设置：

```
sudo -i
sudo chattr -i /

sudo mkdir -p /etc/systemd/system/crio.service.d/
cat | sudo tee /etc/systemd/system/crio.service.d/99-registry-cpaas-system.conf << 'EOF'
[Service]
ExecStart=
ExecStart=/usr/bin/crio \
          --insecure-registry='<registry-address>' \  # e.g., registry.example.cn:60080 or 192.168.134.43
          $CRIO_CONFIG_OPTIONS \
          $CRIO_RUNTIME_OPTIONS \
          $CRIO_STORAGE_OPTIONS \
          $CRIO_NETWORK_OPTIONS \
          $CRIO_METRICS_OPTIONS
EOF
```

3. 重启 `crio` ：

```
sudo systemctl daemon-reload && sudo systemctl restart crio
```

为集群配置 DNS

修改 global 集群中的 CoreDNS `ConfigMap`，以配置 DNS。

1. 在堡垒机上，获取 OCP 集群的基础域名：

   ```
   oc get dns cluster -o jsonpath='{.spec.baseDomain}'
   ```

   示例输出：

   ```
   ocp.example.com
   ```

2. 登录平台管理控制台，切换到 **global** 集群，然后进入 **Cluster Management** > **Resource Management**。

3. 编辑 `kube-system` 命名空间中的 `cpaas-coredns` `ConfigMap`。
   使用 OCP 基础域名和 DNS 服务器地址（来自集群节点上的 `/etc/resolv.conf`）添加一个新块。
   示例：

   ```
   Corefile: |
   ocp.example.com:1053 {
       log
       forward . 192.168.31.220
   }
   .:1053 {
       log
       forward . 192.168.31.220
   }
   ```

获取集群信息

请从以下方法中选择一种：

方法 1（推荐）：获取 KubeConfig 文件

1. 在堡垒机上搜索 `kubeconfig` 文件，并确认其中包含 admin 上下文。
2. 将 kubeconfig 文件从堡垒机复制到本地计算机：

```
scp root@<bastion-ip>:</path/to/kubeconfig> <local-path>
```

方法 2：使用令牌、API Server 地址和 CA 证书

请参见 如何获取集群信息？。

导入集群

1. 在左侧导航中，进入 **Cluster Management** > **Clusters**。
2. 单击 **Import Cluster**。
3. 配置参数：

参数	说明
Registry	存储平台组件镜像的 Registry。**Platform Default**：global 安装期间配置的 Registry。**Private Registry**：需要 Registry 地址、端口、用户名和密码。**Public Registry**：需要 更新云凭据。
Cluster Info	可以上传 KubeConfig 文件，也可以手动输入。**Cluster Address**：API Server 地址。**CA Certificate**：解码后的 Base64 CA 证书。**Authentication**：具有 cluster-admin 权限的令牌或客户端证书。

4. 单击 **Check Connectivity**。
5. 如果成功，单击 **Import**。进度可以在执行日志中查看。导入完成后，集群会在列表中显示为健康状态。

网络配置

确保 global 集群与已导入集群之间的网络连通性。请参见 已导入集群的网络配置。

部署附加组件

成功集成后，前往 **Marketplace** 部署所需的附加组件，例如监控、日志采集和日志存储。

在部署日志采集之前，请确保 `/var/cpaas/` *具有超过 50GB 的可用空间：*

```
df -h /var/cpaas
```

更新审计策略

您可以修改集群的审计策略（`spec.audit.profile`）：

- **Default**：记录读/写请求的元数据（OAuth access token 创建会记录正文）。
- **WriteRequestBodies**：记录所有请求的元数据以及写请求正文。
- **AllRequestBodies**：记录所有请求的元数据和正文。

敏感资源（例如 Secrets、Routes、OAuthClient）仅记录元数据。

使用以下命令更新：

```
oc edit apiserver cluster
```

常见问题

为什么 “Add Node” 按钮被禁用？

不支持通过平台 UI 添加节点。请使用该发行版提供商的方法。

支持哪些证书？

1. **Kubernetes Certificates**：仅可见 API Server 证书，不支持自动轮换。
2. **Platform Component Certificates**：可见并可自动轮换。

此特定于提供商的导入路径有哪些限制？

- 审计数据采集。
- ETCD、Scheduler、Controller Manager 监控（仅提供 API Server 指标）。
- 除 API Server 之外的证书。

导入 Amazon EKS 集群

使用此特定于 provider 的操作步骤，将现有的 Amazon EKS 集群导入为第三方集群。除非有已文档化的特定于 provider 的功能提供该生命周期，否则外部 provider 仍负责 provider 基础设施，以及由 provider 管理的 Kubernetes 和节点生命周期。

目录

前提条件

- 集群的 Kubernetes 版本和设置满足 导入标准 Kubernetes 集群的版本兼容性中的要求。
- 镜像仓库必须支持 HTTPS，并提供由公共 CA 签发的有效 TLS 证书。

准备环境

为符合 AWS EKS 安全最佳实践，请在 AWS CloudShell 中执行以下步骤。

1. 确保可连接到 AWS Management Console。
2. 搜索 `cloudshell`，然后打开 CloudShell↗。
3. 验证所选地域与目标集群的地域一致；如有需要，请切换。
4. CloudShell 就绪后，清空终端并运行：

```
# List clusters in the current region and verify your permissions
aws eks list-clusters

# <region-code> is the region of the cluster, e.g., us-west-1
# <my-cluster> is the cluster name from the previous output
aws eks update-kubeconfig --region <region-code> --name <my-cluster>

# The kubeconfig file is saved to "${HOME}/.kube/config"
# Save its content to a file, then upload it to the platform for parsin
g
cat "${HOME}/.kube/config"
```

5. 现在环境已准备就绪。对于后续步骤，例如 获取集群信息 和 导入集群，请在 CloudShell 中针对目标集群运行任何命令。

获取集群信息

获取导入令牌

来自公有云集群的 KubeConfig 不能直接用于导入。

要获取集群导入令牌，请参阅 如何获取集群信息？。

导入集群

1. 在左侧导航中，进入 集群管理 > 集群。
2. 单击 导入集群。
3. 按如下方式配置参数。

参数	描述
镜像仓库	存储集群所需平台组件镜像的仓库。- 平台默认：global 集群部署时配置的仓库。- 私有仓库：预先提供的、托管所需镜像的仓库。请提供 私有仓库地址、端口、用户名 和 密码。- 公有仓库：公网仓库。使用前，请按照 更新公有仓库云凭据 中的说明获取凭据。
集群信息	提示：上传 kubeconfig 文件，让平台自动解析。集群端点：目标集群暴露的外部 API server 地址。**CA** 证书：集群的 CA 证书。认证：使用上一步创建的、具有 **cluster administrator** 权限的 令牌。

4. 单击 检查连通性 以验证网络连通性并自动检测集群类型。检测到的类型会以徽标形式显示在表单右上角。
5. 连通性检查通过后，单击 导入，然后确认。
   提示：

   - 对于处于 **Importing** 状态的集群，单击详细信息图标，在 执行进度 对话框（`status.conditions`）中查看进度。
   - 导入成功后，集群列表会显示关键信息。集群状态为 Normal，并且可执行集群操作。

网络配置

确保 global 集群与已导入集群之间具有网络连通性。请参阅 已导入集群的网络配置。

后续步骤

初始化 **Ingress** 和存储

如果您需要 Ingress 和存储能力，请参阅 为 AWS EKS 初始化 Ingress 和 为 AWS EKS 初始化存储。

FAQ

导入后 **Add Node** 按钮被禁用。如何添加节点？

不支持从平台 UI 添加节点。请通过您的集群 provider 添加节点。

证书管理支持哪些已导入集群的证书？

1. **Kubernetes** 证书：仅可查看 API server 证书。其他 Kubernetes 证书不可见，也不会自动轮换。
2. 平台组件证书：在平台中可见，并支持自动轮换。

此特定于 **provider** 的导入路径有哪些限制？

- 不提供审计数据。
- 不支持 ETCD、Scheduler 和 Controller Manager 指标；仅提供部分 API server 图表。
- 除 Kubernetes API server 证书外，其他证书详情不可用。

导入 GKE 集群

使用此特定于提供商的操作步骤，将现有的 Google GKE 集群导入为第三方集群。除非文档中提供了特定于提供商的能力来负责该生命周期，否则外部提供商仍然负责提供商基础设施以及由提供商管理的 Kubernetes 和节点生命周期。

目录

前提条件

- 集群上的 Kubernetes 版本和组件满足导入公有云集群的版本要求。
- 确保集群类型为标准集群，且账户具有维护控制平面的权限。当前不支持 Autopilot clusters。
- 镜像仓库必须支持 HTTPS 访问，并提供由公共证书颁发机构认证的有效 TLS 证书。

准备操作环境

为符合 GKE 安全标准，以下步骤必须使用 Cloud Shell 执行。

1. 确保与 Google 的网络连通性。
2. 进入 Kubernetes Engine 功能中的 **Clusters** 页面↗，找到要导入的集群，点击集群详情，然后选择 **Connect** 按钮。
3. 在弹出的对话框中，复制用于配置 kubectl 命令行访问权限的命令，然后点击 **Run in Cloud Shell** 按钮。
4. 等待 Cloud Shell 就绪，清空命令行，粘贴上一步复制的内容，并执行。
5. 此时环境已准备就绪。后续在导入集群环境中执行的所有命令，例如 **Obtaining Cluster Information** 和 **Importing Cluster** 步骤中的命令，都应在 Cloud Shell 中执行。

获取集群信息

获取目标集群的 **API Server** 地址和 **CA** 证书

1. 进入 Kubernetes Engine 功能中的 **Clusters** 页面↗，并点击进入目标集群详情页。
2. API Server 地址可在 **External endpoints** 部分找到。
3. 要获取 CA 证书，请在 Cloud Shell 中使用以下任一方法：
   方法 **A**：从 kubeconfig 中获取 CA 证书：

   ```
   gcloud container clusters get-credentials <cluster-name> --zone <zone>
   kubectl config view --raw -o jsonpath='{.clusters[0].cluster.certificat
   e-authority-data}' | base64 -d
   ```

   方法 **B**：直接从集群中获取 CA 证书：

```
gcloud container clusters describe <cluster-name> --zone <zone> --forma
t='get(masterAuth.clusterCaCertificate)' | base64 -d
```

注意：在粘贴到导入表单之前，必须先对证书进行 Base64 解码。

获取目标集群令牌

公有云集群的 KubeConfig 文件不能直接用于导入集群。

请参见如何获取集群信息？以获取目标集群令牌。

导入集群

1. 在左侧导航栏中，点击 **Clusters** > **Clusters**。
2. 点击 **Manage Cluster** > **Import Cluster**。
3. 根据以下说明配置相关参数。

参数	描述
镜像仓库	用于存储集群所需平台组件镜像的仓库。**- Platform Default**：全局部署时配置的镜像仓库。**- Private Repository**：存放平台所需组件的预构建仓库。请输入访问镜像仓库所需的 私有镜像仓库地址、端口、用户名 和 密码。**- Public Repository**：使用互联网上的公共镜像仓库服务。使用前，请更新公共仓库云凭据以获取仓库认证权限。
集群信息	集群信息：包含目标集群令牌以及目标集群的 API Server 地址和 CA 证书。集群地址：目标集群暴露 API Server 供平台访问集群 API Server 的访问地址。**CA** 证书：目标集群的 CA 证书。注意：手动输入时，需要输入 Base64 解码后的证书。认证方式：目标集群的认证方式，需要使用上一步创建的具有集群管理权限的 **token**（令牌）进行认证。

4. 点击 **Check Connectivity** 以验证与目标集群的网络连通性，并自动识别集群类型，识别结果会以徽标的形式显示在表单右上角。
5. 连通性检查通过后，点击 **Import** 并确认。

TIP

- 在 **Importing** 状态的集群右侧点击 **Details** 图标，可在弹出的 **Execution Progress** 对话框中查看集群执行进度（status.conditions）。
- 集群导入成功后，您可以在集群列表中查看关键集群信息，集群状态将显示为正常，并且可以执行与集群相关的操作。

网络配置

确保 global 集群与导入集群之间的网络连通性。请参见导入集群的网络配置。

导入后操作

Ingress 和存储初始化

导入集群后，如果您需要使用 Ingress 和存储相关功能，请参见Google GKE Ingress Controller 配置和Google GKE 存储配置。

常见问题

导入集群后，“Add Node” 按钮变灰时，如何添加节点？

不支持通过平台界面添加节点。请联系集群提供商添加节点。

导入集群时，证书管理功能支持哪些证书？

1. **Kubernetes Certificates**：所有导入集群仅支持在平台证书管理界面查看 APIServer 证书信息。不支持查看其他 Kubernetes 证书，也不支持自动轮换。
2. **Platform Component Certificates**：所有导入集群都可以在平台证书管理界面查看平台组件证书信息，并支持自动轮换。

导入华为云 CCE 集群（公有云）

使用此特定于提供商的操作步骤，将现有的华为云 CCE 集群导入为第三方集群。除非某个已文档化的特定于提供商的功能提供了相应生命周期管理，否则外部提供商仍负责提供商基础设施以及提供商管理的 Kubernetes 和节点生命周期。

目录

前提条件

- 集群上的 Kubernetes 版本和参数满足标准 Kubernetes 集群组件版本和参数要求。
- 确保集群类型为华为云 CCE 集群，且账号具有维护控制平面的权限。当前不支持 Turbo 集群。
- 华为云 CCE 集群创建后默认不具备访问外部网络资源的能力。在导入集群之前，请确保待导入集群可以访问平台接入地址。

获取 **Image Registry** 地址

- 若要使用全局集群部署中平台部署的 image registry，请在全局集群的控制节点上执行以下命令以获取地址：

```
if [ "$(kubectl get productbase -o jsonpath='{.items[].spec.registry.pr
eferPlatformURL}')" = 'false' ]; then
    REGISTRY=$(kubectl get cm -n kube-public global-info -o jsonpath
='{.data.registryAddress}')
else
    REGISTRY=$(kubectl get cm -n kube-public global-info -o jsonpath
='{.data.platformURL}' | awk -F \// '{print $NF}')
fi
echo "Image registry address is: $REGISTRY"
```

- 若要使用外部 **image registry**，请手动设置 **REGISTRY** 变量。

```
REGISTRY=<external image registry address>  # Valid examples: registry.
example.cn:60080 or 192.168.134.43
echo "Image registry address is: $REGISTRY"
```

判断 **Image Registry** 是否需要额外配置

1. 执行以下命令，判断指定的 image registry 是否支持 HTTPS 访问，并且是否使用受信任 CA 机构签发的证书：

```
REGISTRY=<image registry address obtained from the "Obtain Image Regist
ry Address" section>

if curl -s -o /dev/null --retry 3 --retry-delay 5 -- "https://${REGISTR
Y}/v2/"; then
    echo 'Test passed: The image registry uses certificates issued by t
rusted CA authorities. You do not need to execute the content in the "T
rust Insecure Image Registry" section.'
else
    echo 'Test failed: The image registry does not support HTTPS or the
certificate is not trusted. See the "Trust Insecure Image Registry" sec
tion for configuration.'
fi
```

2. 如果测试失败，请参见如何信任不安全的 image registry？。

获取集群信息

1. 确保与华为云控制台网络连通。
2. 访问 `Cloud Container Engine CCE` 功能的集群管理页面↗；找到待导入集群并单击集群名称进入详情页面。
3. 如下图所示，按照导航找到下载 KubeConfig 文件按钮：`集群信息 - 连接信息 - kubectl - 配置`，并下载 KubeConfig 文件。

获取导入集群 **token**

公有云集群的 KubeConfig 文件不能直接用于集群导入。

请参见如何获取集群信息？获取导入集群 token。

导入集群

1. 在左侧导航栏中，单击集群管理 > 集群。
2. 单击导入集群。

3. 根据以下说明配置 `Image Registry` 相关参数。

参数	说明
Image Registry	用于存储集群所需平台组件镜像的仓库。 - 平台默认：全局集群部署时配置的 image registry。 - 私有仓库：预置的、用于存储平台所需组件的仓库。请输入访问该 image registry 所需的私有 **image registry** 地址、端口、用户名和密码。 - 公共仓库：使用位于公网的 image registry 服务。使用前，请更新公共 image registry 云凭证，以获取仓库认证权限。
集群信息	提示：上传 KubeConfig 文件，由平台自动解析并填充。 集群地址：导入集群暴露的 API Server 访问地址，用于平台访问导入集群的 API Server。 **CA** 证书：导入集群的 CA 证书。 认证方式：导入集群的认证方式，需要使用上一步创建的、具有集群管理权限的 **token** 进行认证。

4. 单击 `解析 KubeConfig 文件` 按钮，并提交上一步下载的 KubeConfig 文件。平台将自动解析并填充 `集群信息` 相关参数。
5. 单击检查连通性，检查与导入集群的网络连通性，并自动识别导入集群的类型。集群类型将以徽标形式显示在表单右上角。
6. 连通性检查通过后，单击导入并确认。
   提示：

   - 单击导入中状态的集群右侧的
     
     图标，可在弹出的执行进度对话框中查看集群的执行进度（status.conditions）。
   - 集群导入成功后，您可以在集群列表中查看集群的关键信息。集群状态显示为正常，并且可以执行与集群相关的操作。

网络配置

为确保全局集群与导入集群之间的网络连通性，请参见导入集群网络配置。

后续操作

Ingress（入站规则）和存储初始化

导入集群后，如果需要使用 Ingress（入站规则）和存储相关功能，请参见华为云 CCE 集群 Ingress 初始化配置和华为云 CCE 集群存储初始化配置。

常见问题

导入集群后，添加节点按钮是灰色的。如何添加节点？

不支持通过平台界面添加节点。请联系集群提供商添加节点。

导入集群支持哪些证书管理功能？

1. **Kubernetes** 证书：所有导入集群仅支持在平台证书管理界面查看 APIServer 证书信息。不支持查看其他 Kubernetes 证书以及自动轮转。
2. 平台组件证书：所有导入集群均可在平台证书管理界面查看平台组件证书信息，并支持自动轮转。

此特定于提供商的导入路径有哪些限制

- 不支持获取审计数据。
- 不支持 ETCD、Scheduler 和 Controller Manager 相关监控信息。支持 APIServer 部分监控图表。
- 除 Kubernetes APIServer 证书外，无法获取其他集群证书相关信息。

导入 Azure AKS 集群

使用此特定于提供商的操作步骤，将现有的 Azure AKS 集群导入为第三方集群。除非文档中说明了特定于提供商的能力，否则外部提供商仍负责提供商基础设施以及提供商托管的 Kubernetes 和节点生命周期。

目录

前提条件

- 集群上的 Kubernetes 版本和参数必须满足标准 Kubernetes 集群组件版本和参数要求。

> **TIP**
>
> - 如果 AKS 节点无法访问 global 集群，请参见如何配置 AKS 节点外部 IP 安全组规则。

- 镜像仓库必须支持 HTTPS 访问，并提供由公共证书颁发机构认证的有效 TLS 证书。

准备运行环境

为符合 Azure AKS 安全标准，必须使用 Cloud Shell 执行以下步骤。

1. 确保与 Azure Console 的网络连通性。
2. 打开Kubernetes 服务页面↗，找到要导入的集群，然后单击进入集群概览页面。
3. 单击 `Connect` 按钮，将打开一个标题为 `Connect to <导入集群名称>` 的浮动窗口。按照说明打开 Cloud Shell 并配置运行环境。

获取集群信息

获取导入令牌

公有云集群的 KubeConfig 文件不能直接用于集群导入。

请参见如何获取集群信息？以获取导入集群令牌。

导入集群

1. 在左侧导航栏中，单击 集群管理 > 集群。
2. 单击 导入集群。
3. 根据以下说明配置相关参数。

参数	说明
镜像仓库	存储集群所需平台组件镜像的仓库。- 平台默认值：部署 global 集群时配置的镜像仓库。- 私有仓库：预先构建的、存储平台所需组件镜像的仓库。输入访问镜像仓库所需的 私有镜像仓库地址、端口、用户名 和 密码。- 公共仓库：使用互联网上的公共镜像仓库服务。使用前，请更新公共镜像仓库云凭据以获取仓库认证权限。
集群信息	提示：上传 KubeConfig 文件后，平台将自动解析并填写信息。集群地址：导入集群对外暴露的 API Server 访问地址，平台通过该地址访问导入集群的 API Server。**CA** 证书：导入集群的 CA 证书。认证方式：导入集群的认证方式，需使用上一步创建的、具有集群管理权限的令牌进行认证。

4. 单击 检查连通性，验证与导入集群的网络连通性，并自动识别导入集群类型。集群类型将显示为表单右上角的徽标。
5. 连通性检查通过后，单击 导入 并确认。

TIP

- 单击处于 导入中 状态的集群右侧的 详情 图标，可在弹出的 执行进度 对话框中查看集群的执行进度（status.conditions）。
- 集群成功导入后，您可以在集群列表中查看集群的关键信息。集群状态将显示为正常，并且可以执行与集群相关的操作。

网络配置

确保 global 集群与导入的集群之间具备网络连通性。请参见导入集群的网络配置。

导入后的操作

Ingress（入站规则）和存储初始化

导入集群后，如果需要使用 Ingress（入站规则）和存储相关功能，请参见Azure AKS 集群 Ingress 初始化配置和Azure AKS 集群存储初始化配置。

常见问题

如何配置 AKS 节点外部 IP 安全组规则

默认情况下，节点仅具有内部 IP。外部 IP 配置在前端负载均衡器（LB）上，默认用于出站流量。此 LB 由 AKS principal 控制。直接手动修改此配置可能会导致问题。您可以通过 **Kubernetes** > 属性 > 基础设施资源组 > 网络安全组 > 添加全部出站/入站规则 允许流量通过。

如何访问 AKS 节点

要查看 Kubelet、CNI 和 kernel 等系统组件的日志，您需要先 SSH 到节点。建议使用 `kubectl-node-shell` 插件，而不是为每个节点分配公网 IP 地址。

选项 **1**：使用 **kubectl node-shell**

请参见 kubectl-node-shell 项目↗。

选项 **2**：使用 **debug**

请参见关于连接到 AKS 节点↗的提供商指导。

> **NOTE**
>
> 此示例需要 kubectl 1.25 或更高版本，其中包含 GA 版 `kubectl debug` 命令。

```
kubectl debug node/aks-newadd-41368356-vmss000002 -it --image=mcr.microsoft.com/dotnet/runtime-deps:6.0
chroot /host
```

使用内部负载均衡器的 Azure ALB

有关内部负载均衡器行为，请参见提供商指导↗。

```
apiVersion: v1
kind: Service
metadata:
  name: internal-app
  namespace: cpaas-system
  annotations:
    service.beta.kubernetes.io/azure-load-balancer-internal: "true"
spec:
  type: LoadBalancer
  ports:
  - name: http-port
    port: 80
    protocol: TCP
  - name: https-port
    port: 443
    protocol: TCP
  selector:
    service.cpaas.io/name: deployment-aks-alb
    service_name: alb2-aks-alb
```

使用外部负载均衡器的 Azure ALB

部署一个高可用 ALB，并将访问地址配置为外部 LB。

```
apiVersion: v1
kind: Service
metadata:
  name: azure-alb
  namespace: cpaas-system
spec:
  type: LoadBalancer
  ports:
  - name: http-port
    port: 80
    protocol: TCP
  - name: https-port
    port: 443
    protocol: TCP
  - name: prom-port
    port: 11780
    protocol: TCP
  - name: prom2-port
    port: 11781
    protocol: TCP
  - name: prom3-port
    port: 15012
    protocol: TCP
  selector:
    service_name: alb2-cpaas-system
```

如果已提前部署，可以使用以下命令对其进行修改。

```
kubectl edit helmrequest -n cpaas-system uat-cluster-aks-alb
```

导入集群后“添加节点”按钮变灰。如何添加节点？

不支持通过平台界面添加节点。请联系集群提供商添加节点。

导入集群的证书管理功能支持哪些证书？

1. **Kubernetes** 证书：所有导入的集群仅支持在平台证书管理界面中查看 APIServer 证书信息。其他 Kubernetes 证书无法查看，也不支持自动轮换。

2. 平台组件证书：所有导入的集群都可以在平台证书管理界面中查看平台组件证书信息，并支持自动轮换。

此特定于提供商的导入路径有哪些限制？

- 不支持审计数据检索。
- 不支持 ETCD、Scheduler 和 Controller Manager 相关监控信息。支持 APIServer 部分监控图表。
- 无法检索除 Kubernetes APIServer 证书之外的集群证书相关信息。

导入 Alibaba Cloud ACK 集群

使用此特定于提供方的操作步骤，将现有的 Alibaba Cloud ACK 托管集群或专有集群导入为第三方集群。除非文档中提供了特定于提供方的能力来实现相应生命周期管理，否则外部提供方仍负责提供方基础设施以及提供方托管的 Kubernetes 和节点生命周期。

> **TIP**
>
> 有关提供方集群类型的详细信息，请参阅提供方指南↗。

目录

前提条件

- 集群上的 Kubernetes 版本和参数满足导入标准 Kubernetes 集群的组件版本和参数要求。

获取镜像仓库地址

- 如需使用全局集群部署中的平台部署镜像仓库，请在全局集群的控制节点上执行以下命令以获取地址：

```
if [ "$(kubectl get productbase -o jsonpath='{.items[].spec.registry.preferPlatformURL}')" = 'false' ]; then
    REGISTRY=$(kubectl get cm -n kube-public global-info -o jsonpath='{.data.registryAddress}')
else
    REGISTRY=$(kubectl get cm -n kube-public global-info -o jsonpath='{.data.platformURL}' | awk -F \// '{print $NF}')
fi
echo "Image registry address is: $REGISTRY"
```

- 如需使用外部镜像仓库，请手动设置 **REGISTRY** 变量。

```
REGISTRY=<external image registry address>  # Valid examples: registry.example.cn:60080 or 192.168.134.43
echo "Image registry address is: $REGISTRY"
```

判断镜像仓库是否需要额外配置

1. 执行以下命令，判断指定镜像仓库是否支持 HTTPS 访问，并且是否使用由受信任 CA 机构签发的证书：

```
REGISTRY=<image registry address obtained from the "Get Image Registry Address" section>

if curl -s -o /dev/null --retry 3 --retry-delay 5 -- "https://${REGISTRY}/v2/"; then
    echo 'Test passed: The image registry uses certificates issued by trusted CA authorities. You do not need to execute the content in the "Trust Insecure Image Registry" section.'
else
    echo 'Test failed: The image registry does not support HTTPS or the certificate is not trusted. See the "Trust Insecure Image Registry" section for configuration.'
fi
```

2. 如果测试失败，请参阅如何信任不安全的镜像仓库？。

获取 KubeConfig

1. 登录 Alibaba Cloud 容器服务管理平台。
2. 在控制台左侧导航栏中，单击 **Clusters**。
3. 在 **Cluster List** 页面中，单击目标集群名称，或单击目标集群右侧 **Actions** 列下的 **Details**。
4. 在 **Cluster Information** 页面中，单击 **Connection Information** 选项卡，然后单击 **Generate Temporary KubeConfig**。
5. 在 **Temporary KubeConfig** 对话框中，设置临时凭证的有效期以及访问集群的方式（包括公网访问和内网访问）。
6. 单击 **Generate Temporary KubeConfig**，然后单击 **Copy** 复制内容，并将其保存到本地计算机上的 **KubeConfig** 文件中。
7. 集群成功导入后，您可以吊销临时凭证。

导入集群

1. 在左侧导航栏中，单击 **Cluster Management** > **Clusters**。
2. 单击 **Import Cluster**。

3. 根据以下说明配置相关参数。

参数	描述
Image Registry	存储集群所需平台组件镜像的仓库。**- Platform Default**：全局集群部署期间配置的镜像仓库。**- Private Registry**：预置的、用于存储平台所需组件镜像的仓库。您需要输入访问镜像仓库所需的私有镜像仓库地址、端口、用户名和密码。**- Public Registry**：使用互联网上的公共镜像仓库服务。使用前，请更新公共仓库云凭证，以获取仓库认证权限。
Cluster Information	提示：可以手动填写，也可以上传 KubeConfig 文件，由平台自动解析并填充。**Parse KubeConfig File**：上传获取到的 KubeConfig 文件后，平台将自动解析并填充 **Cluster Information**。您可以修改自动填充的信息。**Cluster Address**：集群对外暴露的 API Server 访问地址，用于平台访问集群的 API Server。**CA Certificate**：集群的 CA 证书。注意：手动输入时，需要输入 Base64 解码后的证书。**Authentication Method**：访问集群的认证方式。您需要使用具有集群管理权限的 **token** 或**证书认证（client certificate 和 key）**进行认证。

4. 单击 **Check Connectivity**，检查与待导入集群之间的网络连通性，并自动识别待导入集群的类型。集群类型将显示为表单右上角的徽标。
5. 连通性检查通过后，单击 **Import** 并确认。

> **TIP**
>
> - 单击 **Importing** 状态集群右侧的 **Details** 图标，可在弹出的 **Execution Progress** 对话框中查看集群的执行进度（status.conditions）。
> - 集群成功导入后，您可以在集群列表中查看集群的关键信息。集群状态将显示为正常，并且可以执行与集群相关的操作。

网络配置

确保全局集群与待导入集群之间的网络连通性。请参阅导入集群的网络配置。

FAQ

如何处理 **Alibaba Cloud** 监控与平台监控组件之间的端口冲突？

当 Alibaba Cloud 内置监控与平台监控组件同时存在时，会发生端口冲突。建议卸载 Alibaba Cloud 监控，仅保留平台监控。

如何为 **Alibaba Cloud** 集群使用公网访问？

如果 Alibaba Cloud 集群使用公网访问，您可以在 Alibaba Cloud 上绑定一个公网 IP。

导入集群后，添加节点按钮变为灰色。如何添加节点？

Alibaba Cloud ACK 托管集群和 **ACK** 专有集群都不支持通过平台界面添加节点。请在提供方后台添加节点，或联系集群提供方。

导入集群的证书管理功能支持哪些证书？

1. **Kubernetes** 证书：所有导入集群仅支持在平台证书管理界面查看 APIServer 证书信息。不支持查看其他 Kubernetes 证书，也不支持自动轮换。
2. 平台组件证书：所有导入集群都可以在平台证书管理界面查看平台组件证书信息，并支持自动轮换。

此特定于提供方的导入路径有哪些限制？

- **Alibaba Cloud ACK** 托管集群不支持获取审计数据。
- **Alibaba Cloud ACK** 托管集群不支持 ETCD、Scheduler、Controller Manager 相关监控信息，但支持部分 APIServer 监控图表。
- **Alibaba Cloud ACK** 托管集群和 **ACK** 专有集群都不支持获取除 Kubernetes APIServer 证书之外的集群证书相关信息。

导入腾讯云 TKE 集群

使用此特定于提供商的操作步骤，将现有的腾讯云 TKE 专用集群或托管集群导入为第三方集群。除非文档中说明了特定于提供商的能力会负责该生命周期，否则外部提供商仍然负责提供商基础设施以及提供商托管的 Kubernetes 和 node 生命周期。

> **TIP**
>
> 有关提供商集群类型的详细信息，请参阅 provider guide↗。

目录

前提条件

- 集群上的 Kubernetes 版本和参数满足 导入标准 Kubernetes 集群的组件版本和参数要求。
- 图像仓库必须支持 HTTPS 访问，并提供由公共证书颁发机构签发的有效 TLS 证书。

获取图像仓库地址

- 如需使用在 global 集群部署期间配置的 平台部署 图像仓库，请在 **global** 集群的控制节点上执行以下命令以获取地址：

```
if [ "$(kubectl get productbase -o jsonpath='{.items[].spec.registry.preferPlatformURL}')" = 'false' ]; then
    REGISTRY=$(kubectl get cm -n kube-public global-info -o jsonpath='{.data.registryAddress}')
else
    REGISTRY=$(kubectl get cm -n kube-public global-info -o jsonpath='{.data.platformURL}' | awk -F \// '{print $NF}')
fi
echo "Image registry address is: $REGISTRY"
```

- 如需使用 外部图像仓库，请手动设置 **REGISTRY** 变量。

```
REGISTRY=<external image registry address>  # Valid examples: registry.example.cn:60080 or 192.168.134.43
echo "Image registry address is: $REGISTRY"
```

判断图像仓库是否需要额外配置

1. 执行以下命令，判断指定图像仓库是否支持 HTTPS 访问，并且是否使用受信任 CA 签发的证书：

```
REGISTRY=<image registry address obtained from the "Obtain Image Regist
ry Address" section>

if curl -s -o /dev/null --retry 3 --retry-delay 5 -- "https://${REGISTR
Y}/v2/"; then
    echo 'Verification passed: The image registry uses a certificate is
sued by a trusted CA. It is not necessary to execute the content in the
"Trust Unsafe Image Registry" section.'
else
    echo 'Verification failed: The image registry does not support HTTP
S or the certificate is not trusted. See the "Trust Unsafe Image Regist
ry" section for configuration.'
fi
```

2. 如果验证失败，请参阅 如何信任不安全的图像仓库？。

获取 **KubeConfig**

1. 登录腾讯云容器服务管理平台。
2. 在 集群详情 > 基本信息 中，查看 **Cluster APIServer** 信息。
3. 根据实际客户网络选择 互联网访问 或 内网访问，然后下载 **Kubeconfig** 并保存到本地计算机。

导入集群

1. 在左侧导航栏中，单击 集群管理 > 集群。
2. 单击 导入集群。
3. 根据以下说明配置相关参数。

参数	说明
图像仓库	用于存储集群所需平台组件镜像的仓库。- 平台默认：global 部署期间配置的图像仓库。- 私有仓库：预先构建的仓库，用于存储平台所需的组件镜像。输入用于访问图像仓库的私有图像仓库地址、端口、用户名和密码。- 公共仓

参数	说明
	库：使用位于公网的图像仓库服务。使用前，请更新 公共仓库云凭据，以获取仓库认证权限。
集群信息	提示：可以手动填写，也可以通过 KubeConfig 文件上传，由平台自动解析并填充。解析 **KubeConfig** 文件：上传获取到的 KubeConfig 文件后，平台将自动解析并填充 集群信息，并且可以修改自动填充的信息。集群地址：集群对外暴露的 API Server 访问地址，用于平台访问集群的 API Server。**CA** 证书：集群的 CA 证书。注意：手动输入时，需要输入 Base64 解码后的证书。认证方式：访问集群的认证方式，需要使用具有 集群管理权限 的 **token**（Token）或 证书认证（客户端证书和密钥）。

4. 单击 检查连通性，验证与待导入集群之间的网络连通性，并自动识别待导入的集群类型。集群类型会以徽标的形式显示在表单右上角。
5. 连通性检查通过后，单击 导入 并确认。
   提示：

   - 在 导入中 状态的集群右侧，单击 详情 图标，可以在弹出的 执行进度 对话框中查看集群的执行进度（status.conditions）。
   - 集群成功导入后，可以在集群列表中查看集群的关键信息。集群状态显示为正常，并且可以执行与集群相关的操作。

网络配置

确保 global 集群与待导入集群之间的网络连通性。请参阅 导入集群的网络配置。

常见问题

导入集群后，“添加节点”按钮变为灰色。如何添加节点？

TKE 专用集群 和 **TKE** 托管集群 都不支持通过平台界面添加节点。请在提供商后端添加节点，或联系集群提供商。

导入集群的证书管理功能支持哪些证书？

1. **Kubernetes** 证书：所有已导入集群仅支持在平台证书管理界面查看 APIServer 证书信息。不支持查看其他 Kubernetes 证书，也不支持自动轮换。
2. 平台组件证书：所有已导入集群都可以在平台证书管理界面查看平台组件证书信息，并支持自动轮换。

此特定于提供商的导入路径有哪些限制

- **TKE** 托管集群 不支持获取审计数据。
- **TKE** 托管集群 不支持 ETCD、Scheduler、Controller Manager 相关的监控信息，但支持部分 APIServer 监控图表。
- **TKE** 托管集群 和 **TKE** 专用集群 都不支持获取除 Kubernetes APIServer 证书之外的集群证书相关信息。

注册集群

当目标第三方集群需要通过反向代理隧道主动向平台发起连接时，请使用注册方式。

目录

前提条件

- 根据目标环境的不同，目标集群中的 Kubernetes 版本和组件参数必须满足版本和参数要求。
- 镜像仓库必须支持 HTTPS 访问，并提供由公共认证机构认证的有效 TLS 证书。如果无法满足此要求，请参见如何信任不安全的镜像仓库？
  注意：平台在公网提供的 **Public Registry** 已满足 HTTPS 访问要求。您只需确认 **Platform Default** 和 **Private Registry** 是否支持 HTTPS 访问。
- 如果要连接的集群运行时组件是 Containerd，则需要在连接集群之前修改 Containerd 配置，以确保分布式存储成功部署。

注意事项

网络接口流量监控默认识别名称匹配 `eth\.\|en\.\|wl\.*\|ww\.*` 的网络接口。如果目标集群使用其他命名规范，请参见从自定义命名网卡收集网络数据，并在注册后更新相应资源。

注册集群

1. 在左侧导航栏中，点击 **Clusters** > **Clusters**。
2. 点击 **Managed Clusters** > **Register Cluster**。
3. 根据以下说明，配置用于存储注册集群所需平台组件镜像的镜像仓库参数。

参数	描述
Platform Default	部署全局组件时使用的镜像仓库。
Private Registry	预先自行搭建的外部镜像仓库。您需要输入用于访问镜像仓库的 **Private Image Registry Address**、**Port**、**Username** 和 **Password**。
Public Registry	通过平台提供的公共镜像仓库拉取所需镜像。请确保目标集群可以访问公网。使用前，请参见更新公网镜像仓库云凭据以获取镜像仓库认证权限。

4. 单击 **Create**，在 **Registration Command** 页面获取注册命令，并在待注册集群中执行该命令。
   注意：注册命令的有效期为 24 小时。过期后请重新获取新的命令。

查看注册命令

在集群列表中找到待注册的集群，单击 **View Registration Command**。请在命令过期前执行注册命令。

常见问题

如何解决接入集群的运行时组件为 **Containerd** 时分布式存储部署失败的问题？

当接入集群的运行时组件为 Containerd 时，分布式存储部署会失败。要解决此问题，需要手动修改集群所有节点上的 Containerd 配置信息，并重启 Containerd。

注意：如果您在部署分布式存储之前按照以下步骤修改了 Containerd 配置，则无需执行步骤四。

1. 登录集群节点并编辑 `/etc/systemd/system/containerd.service` 文件，将 `LimitNOFILE` 参数值修改为 `1048576`。
2. 执行 `systemctl daemon-reload` 命令重新加载配置。
3. 执行 `systemctl restart containerd` 命令重启 Containerd。
4. 在集群控制节点执行 `kubectl delete pod --all -n rook-ceph` 命令，重启 rook-ceph 命名空间中的所有 Pod，使配置生效。

公有云集群初始化

网络初始化

AWS EKS 集群网络初始化配置

目录

支持概览

功能	支持状态	要求
LoadBalancer Service	支持	可选地 部署 AWS Load Balancer Controller。如果不部署此 Controller，LoadBalancer 功能将受到限制。

功能	支持状态	要求
Ingress	支持	可选地 部署 AWS Load Balancer Controller。可选启用 **Ingress Class** 功能（启用后，在通过表单界面创建 ingress 时，可以手动选择 ingress class）。

前提条件

- 准备两个带有 **kubernetes.io/role/elb** 标签的子网。对于共享子网，请添加 **kubernetes.io/cluster/<cluster-name>: shared** 标签。参见 为子网添加标签。
- 如果你已经创建了 EKS 集群，请 导入 Amazon EKS 集群。
- 在部署 AWS Load Balancer Controller 之前，请确保 kubectl、Helm、AWS CLI 和 eksctl 工具可用。
  注意：安装这些工具后，请使用创建集群的用户通过 AWS CLI 配置登录信息，并 测试 AWS CLI 和 eksctl 工具是否正确安装。
- 请提前获取 ACCOUNT_ID、**REGION** 和 **CLUSTER_NAME**，并将 `<ACCOUNT_ID>`、`<REGION>` 和 `<CLUSTER_NAME>` 替换为实际值。
  注意：**ACCOUNT_ID** 是创建集群的用户的 Account ID，**REGION** 是集群地域，**CLUSTER_NAME** 是集群名称。
- 更新并验证 Kubeconfig 配置文件。

配置步骤

部署 **AWS Load Balancer Controller**

注意：有关部署 AWS Load Balancer Controller 的详细信息，请参见 provider guide ↗。

配置 **OIDC Provider**

Kubernetes 集群使用 OpenID Connect (OIDC) 进行身份管理，并关联一个 OIDC issuer URL。要在集群中启用 AWS Identity 并允许 Service Accounts 使用 IAM 角色，请创建一个与集群 OIDC issuer URL 关联的 IAM OIDC Provider。

在 eksctl 中执行以下命令以配置 OIDC Provider：

```
eksctl utils associate-iam-oidc-provider --region=<REGION> --cluster=<CLU
STER_NAME> --approve
```

配置 **Service Account**

执行以下命令以创建一个 IAM policy，并创建一个名为 `aws-load-balancer-controller` 的 Service Account，将其与一个 IAM role 关联：

```
curl -o aws-load-balancer-controller-iam-policy.json https://raw.githubus
ercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.4.7/docs/in
stall/iam_policy.json
aws iam create-policy \
    --policy-name <CLUSTER_NAME>-AWSLoadBalancerControllerIAMPolicy \
    --policy-document file://aws-load-balancer-controller-iam-policy.json

eksctl create iamserviceaccount \
  --cluster=<CLUSTER_NAME> \
  --namespace=kube-system \
  --name=aws-load-balancer-controller \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --attach-policy-arn=arn:aws:iam::<ACCOUNT_ID>:policy/<CLUSTER_NAME>-AWS
LoadBalancerControllerIAMPolicy \
  --approve
```

将 **AWS Load Balancer Controller** 部署到集群

在 eksctl 中执行以下命令以部署 AWS Load Balancer Controller：

1. 添加 eks-charts 仓库：

   ```
   helm repo add eks https://aws.github.io/eks-charts
   ```

2. 更新本地仓库：

   ```
   helm repo update eks
   ```

3. 将 AWS Load Balancer Controller Helm Chart 部署到集群：

注意：`aws-load-balancer-controller` 是在 配置 Service Account 中创建的 **Service Account**。

```
helm install aws-load-balancer-controller eks/aws-load-balancer-control
ler \
  -n kube-system \
  --version=v2.4.7 \
  --set ingressClassConfig.default=true \
  --set clusterName=<CLUSTER_NAME> \
  --set serviceAccount.create=false \
  --set serviceAccount.name=aws-load-balancer-controller
```

创建 Ingress 和 LoadBalancer 服务

你可以同时创建 ingress 和 LoadBalancer 服务，也可以根据需要选择其中一种。

创建 **Ingress**

1. 在 **Container Platform** 中，点击左侧导航栏的 **Network** > **Ingress**。
2. 点击 **Create Ingress**，并在 **Ingress Class** 中选择 **EKS Ingress Class**。
3. 选择 **Protocol**。默认值为 **HTTP**。对于 **HTTPS**，请先 创建证书 并选择它。
4. 切换到 **YAML**，并添加以下 annotations。有关详细信息，请参见 annotation reference ↗：

```
alb.ingress.kubernetes.io/scheme: internet-facing ## Specify public acc
ess
alb.ingress.kubernetes.io/target-type: ip ## Route traffic directly to
pods
```

5. 点击 **Create**。

创建 **LoadBalancer Service**

1. 在 **Container Platform** 中，点击左侧导航栏的 **Network** > **Services**。
2. 点击 **Create Service**，并在 **External Access** 中选择 **LoadBalancer**。
3. 展开 **annotations**，并根据需要填写 LoadBalancer service annotations。
4. 点击 **Create**。

相关操作

测试 AWS CLI 和 eksctl 安装

- 执行以下命令。如果返回集群列表，则说明 AWS CLI 安装正确：

```
aws eks list-clusters
```

- 执行以下命令。如果返回集群列表，则说明 eksctl 安装正确：

```
eksctl get clusters
```

获取 ACCOUNT_ID

执行 `aws sts get-caller-identity` 以获取 **ACCOUNT_ID**。返回结果中的 `651168850570` 即为 **ACCOUNT_ID**：

```
{
  "ARN": "arn:aws:iam::651168850570:user/jwshi"
}
```

Kubeconfig 配置文件

1. 执行以下命令，更新指定地域的 Kubeconfig 文件：

```
aws eks --region <REGION>  update-kubeconfig --name <CLUSTER_NAME>
```

2. 执行以下命令验证 Kubeconfig 文件。如果正常返回信息，则表示配置正确：

```
kubectl get svc -n cpaas-system
```

为子网添加标签

1. 执行以下命令获取集群子网：

```
eksctl get cluster --name <CLUSTER_NAME>
```

2. 执行以下命令获取子网详情：

```
aws ec2 describe-subnets
```

3. 执行以下命令为子网添加标签。请将 `<subnet-id>` 替换为实际值。参见 Subnet auto-discovery↗ ：

   - 为子网添加 `kubernetes.io/role/elb` 标签：

   ```
   aws ec2 create-tags --resources <subnet-id> --tags Key=kubernetes.io/role/elb,Value="1"
   ```

   - 为共享子网添加 `kubernetes.io/cluster/<CLUSTER_NAME>: shared` 标签：

   ```
   aws ec2 create-tags --resources <subnet-id> --tags Key=kubernetes.io/cluster/<CLUSTER_NAME>,Value="shared"
   ```

创建证书

使用 HTTPS 协议时，请提前将 HTTPS 证书凭证保存为 Secret（TLS 类型）。

1. 在 **Container Platform** 中，点击左侧导航栏的 **Configuration** > **Secrets**。
2. 点击 **Create Secret**。
3. 选择 **TLS** 类型，并根据需要导入或填写 **Certificate** 和 **Private Key**。
4. 点击 **Create**。

AWS EKS 补充信息

目录

术语

缩写	全称	说明
eks-clb	经典负载均衡器	AWS 默认负载均衡器。在某些情况下存在问题，不推荐使用。
eks-nlb	网络负载均衡器	AWS 第 4 层负载均衡器，在 TCP/UDP 层进行负载均衡，适用于需要更高层网络控制的场景。
eks-alb	应用负载均衡器	AWS 第 7 层负载均衡器。与 eks-nlb 相比，eks-alb 可以解析 HTTP/HTTPS 协议，并更智能地分发请求，适用于 Web 应用。

缩写	全称	说明
aws-lb	AWS Load Balancer	安装在 Kubernetes 上的负载均衡器，可根据 Kubernetes 中的 LoadBalancer Services 和 Ingress 自动创建 eks-nlb 和 eks-alb，以满足应用负载均衡需求。
Platform Load Balancer	-	平台自有的第 7 层负载均衡器。
Service Annotations	-	以键值对形式附加到对象上的元数据。这些附加信息可以被识别和利用，从而增强并简化对 Kubernetes 资源各方面的管理。Annotations 可以是没有特定功能的说明性文本，也可以指定云提供商的配置或行为，或者指定配置参数和工具。功能非常强大。

重要说明

在创建负载均衡器时，建议手动配置 service annotations，以确保平台负载均衡器正确使用 aws-lb。如果未正确配置相应的 service annotations，平台将默认使用 eks-clb，而它存在与 UDP 相关的问题，可能会导致意外情况。

EKS 使用 aws-lb 为容器网络负载均衡器提供外部访问

服务注解配置说明

1. 在对应的集群中，使用 kubectl 执行以下命令，查找 kube-system 命名空间中名称包含 "aws-load" 的所有 Pod：

   ```
   kubectl get pod -n kube-system |grep aws-load
   ```

2. 创建负载均衡器；有关详细创建步骤和参数，请参见 AWS EKS 服务注解说明 中的负载均衡器创建章节。

- 如果上述命令没有返回相关 Pod，说明该集群未安装 AWS Load Balancer Controller，无需配置 service annotations，直接创建负载均衡器即可。
- 如果上述命令返回了相关 Pod，说明该集群已安装 AWS Load Balancer Controller。在对应集群中创建负载均衡器时，请添加以下 service annotations。注解详情请参见 AWS EKS 服务注解说明：
  - `service.beta.kubernetes.io/aws-load-balancer-type: external //Required`
  - `service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: ip //Required`
  - `service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing //Optional. Add this annotation if public network support is needed.`

访问地址获取方式

- 创建容器网络类型负载均衡器时，填写的 service annotations 会设置到平台负载均衡器对应的 LoadBalancer Service 上。
- 在公有云中，带有合适 service annotations 的 LoadBalancer Service 会被公有云识别并分配地址。平台负载均衡器会读取该地址，并将其设置为自身的访问地址。

华为云 CCE 集群网络初始化配置

目录

Support Overview

Feature	Support Status	Requirements
LoadBalancer Service	Default Support	无需额外部署。
Ingress	Default Support	可选启用 **Ingress Class** 功能（启用后，可通过表单界面创建 ingress 时手动选择 ingress 类）。无需额外部署。

Prerequisites

如果您已创建 CCE 集群，请导入 CCE 集群（公有云）。

Configuration Steps

您可以同时创建 ingress 和 LoadBalancer 服务，或根据需求选择其中一种。

Create Ingress

创建 ingress 有两种方式。推荐使用方式一：手动选择 **Ingress Class**。

注意：避免创建两个具有相同 **path** 的 ingress 资源。

（推荐）方式一：手动选择 **Ingress Class**

1. 在 **Container Platform** 中，点击左侧导航的 **Network** > **Ingress**。
2. 点击 **Create Ingress**，并在 **Ingress Class** 中选择 **CCE Ingress Class**。
3. 选择 **Protocol**，默认是 **HTTP**。若选择 **HTTPS**，请先创建证书并选择该证书。
4. 切换到 **YAML**，根据您的默认 Ingress Controller 类型添加以下注解。注解详情请参见使用注解配置负载均衡器↗：
   注意：请将下表注解中的 **values** 替换为实际环境值。

默认 **Ingress Controller** 类型	注解
Shared (自动创建)	`kubernetes.io/elb.autocreate: '{"type":"public","bandwidth_name":"{random}","bandwidth_chargemode":"traffic","bandwidth_size":5,"bandw` `kubernetes.io/elb.class: union`

Shared (复用)	`kubernetes.io/elb.class: union`
	`kubernetes.io/elb.id: <Load Balancer Instance ID>`
	`kubernetes.io/elb.port: '80'`

默认 Ingress Controller 类型	注解

Dedicated (自动创建)	`kubernetes.io/elb.autocreate: '{"type":"public","bandwidth_name":"<E`
	`Name>","bandwidth_chargemode":"traffic","bandwidth_size":5,"bandwidtl`
	`["<AZ A>","<AZ B>","<AZ C>"],"elb_virsubnet_ids":["<ELB Virtual Subne`
	`ID>"],"l7_flavor_name":"L7_flavor.elb.s1.small","l4_flavor_name":"L4_`
	`kubernetes.io/elb.class: performance`
	`kubernetes.io/elb.port: "80"`

Dedicated (复用)	`kubernetes.io/elb.class: performance`
	`kubernetes.io/elb.id: <Load Balancer Instance ID>`
	`kubernetes.io/elb.port: "80"`

5. 点击 **Create**。创建完成后，即可通过 ELB 访问集群服务。

方式二：使用默认 **Ingress Class**

1. 创建一个 IngressClass YAML 文件，内容如下。详情请参见默认 Ingress Class↗：

```yaml
apiVersion: networking.k8s.io/v1
kind: IngressClass
metadata:
  annotations:
    ingressclass.kubernetes.io/is-default-class: "true"
  name: cce
spec:
  controller: alauda/cce
```

2. 保存文件并应用到导入的集群，替换 `<filename.yaml>` 为实际 YAML 文件名：

```
kubectl apply -f <filename.yaml>
```

3. 在 **Container Platform** 中，点击左侧导航的 **Network** > **Ingress**。

4. 选择 **Protocol**，默认是 **HTTP**。若选择 **HTTPS**，请先创建证书并选择该证书。

5. 点击 **Create**。创建完成后，即可通过 ELB 访问集群服务。

Create LoadBalancer Service

1. 在 **Container Platform** 中，点击左侧导航的 **Network** > **Services**。
2. 点击 **Create Service**，并在 **External Access** 中选择 **LoadBalancer**。
3. 展开 **annotations**，根据需要填写 LoadBalancer 服务注解。
4. 点击 **Create**。

Related Operations

Create Certificate

使用 HTTPS 协议时，请提前将 HTTPS 证书凭据以 Secret（TLS 类型）形式保存。

1. 在 **Container Platform** 中，点击左侧导航的 **Configuration** > **Secrets**。
2. 点击 **Create Secret**。
3. 选择 **TLS** 类型，按需导入或填写 **Certificate** 和 **Private Key**。
4. 点击 **Create**。

Azure AKS 集群网络初始化配置

目录

支持概览

功能	支持状态	需求
LoadBalancer Service	默认支持	无需额外部署。
Ingress	支持	可选部署 Ingress Controller。可选启用 **Ingress Class** 功能（启用后，创建 ingress 时可通过表单界面手动选择 ingress 类）。

前提条件

如果您已创建 AKS 集群，请导入 Azure AKS 集群。

配置步骤

部署 Ingress Controller

AKS 使用 容器网络模式，并利用 **Nginx Ingress Controller** 管理负载均衡器，同时通过 **LoadBalancer** 类型的 **Services** 为容器内部网络中的虚拟 IP 地址（VIP）提供外部访问地址。

1. 登录 Microsoft Azure 并进入您已创建的 AKS 集群。
2. 在左侧导航中，点击 **Kubernetes Resources** > **Services and Ingresses**。
3. 点击 **Create**，从下拉菜单选择 **Ingress (Preview)**，系统将提示并自动创建 Ingress Controller。
4. 点击 **Enable** 并等待完成。

创建 Ingress 和 LoadBalancer 服务

您可以同时创建 ingress 和 LoadBalancer 服务，也可以根据需求选择其中之一。

创建 **Ingress**

1. 在 **Container Platform** 中，左侧导航点击 **Network** > **Ingress**。
2. 点击 **Create Ingress**，并为 **Ingress Class** 选择 **webapprouting.kubernetes.azure.com**。
3. 选择 **Protocol**，默认是 **HTTP**。若选择 **HTTPS**，请先创建证书并选择该证书。
4. 点击 **Create**。

创建 **LoadBalancer** 服务

1. 在 **Container Platform** 中，左侧导航点击 **Network** > **Services**。
2. 点击 **Create Service**，并为 **External Access** 选择 **LoadBalancer**。

3. 展开 **annotations**，根据需要填写 LoadBalancer 服务注解。
4. 点击 **Create**。

相关操作

创建证书

使用 HTTPS 协议时，需提前将 HTTPS 证书凭据以 Secret（TLS 类型）形式保存。

1. 在 **Container Platform** 中，左侧导航点击 **Configuration** > **Secrets**。
2. 点击 **Create Secret**。
3. 选择 **TLS** 类型，导入或填写 **Certificate** 和 **Private Key**。
4. 点击 **Create**。

Google GKE 集群网络初始化配置

目录

支持概览

功能	支持状态	需求
LoadBalancer Service	默认支持	无需额外部署。
Ingress	默认支持	可选启用 **Ingress Class** 功能（启用后，可通过表单界面创建 ingress 时手动选择 ingress 类）。无需额外部署。

前提条件

如果您已创建 GKE 集群，请导入 GKE 集群。

配置步骤

部署 **Ingress Controller**

无需手动部署。GKE 提供了一个托管的内置 Ingress controller，称为 GKE Ingress。该 controller 将 Ingress 资源映射到 Google Cloud 负载均衡器，用于处理 GKE 中的 HTTP(S) 工作负载，使配置更简单且更自动化。

创建 **Ingress** 和 **LoadBalancer Service**

您可以同时创建 ingress 和 LoadBalancer 服务，或根据需求选择其中之一。

创建 **Ingress**

1. 在 **Container Platform** 中，点击左侧导航的 **Network** > **Ingress**。
2. 点击 **Create Ingress**，并在 **Ingress Class** 中选择 **GKE Ingress Class**。
3. 选择 **Protocol**，默认是 **HTTP**。若选择 **HTTPS**，请先创建证书并选择该证书。
4. 点击 **Create**。等待约 5 分钟，GKE 平台会自动为 ingress 分配公网 IP 地址。
   注意：不同的 ingress 资源会分配不同的公网 IP 地址。

创建 **LoadBalancer Service**

1. 在 **Container Platform** 中，点击左侧导航的 **Network** > **Services**。
2. 点击 **Create Service**，并在 **External Access** 中选择 **LoadBalancer**。
3. 展开 **annotations**，根据需要填写 LoadBalancer 服务注解。
4. 点击 **Create**。

相关操作

在 **Google Cloud** 中查看 **Ingress** 资源

1. 进入 **Google Cloud > Kubernetes Engine**，点击左侧导航的 **Services and Ingress**。
2. 点击 **INGRESS**。
3. 在列表中查看对应 Ingress 资源的信息。

创建证书

使用 HTTPS 协议时，请提前将 HTTPS 证书凭据保存为 Secret（TLS 类型）。

1. 在 **Container Platform** 中，点击左侧导航的 **Configuration** > **Secrets**。
2. 点击 **Create Secret**。
3. 选择 **TLS** 类型，按需导入或填写 **Certificate** 和 **Private Key**。
4. 点击 **Create**。

存储初始化

公共云存储初始化概述

在导入受支持的公共云 Kubernetes 服务后，以及在工作负载中依赖持久化存储功能之前，请使用存储初始化流程。外部提供商仍负责提供商基础设施和提供商托管的存储后端；ACP 负责配置并使用导入集群中暴露的存储类。

下表汇总了针对每条特定于提供商的初始化路径所记录的默认存储类和操作。

目录

存储类支持

AWS EKS 集群

存储类型	默认存储类	使用 **RWO** 访问模式创建 **PVC**	使用 **RWX** 访问模式创建 **PVC**	**PVC** 扩容	**P**
文件存储	efs-sc	支持	支持	不支持	不
块存储	ebs-sc	支持	不支持	支持	不

Huawei Cloud CCE 集群

存储类型	默认存储类	使用 **RWO** 访问模式创建 **PVC**	使用 **RWX** 访问模式创建 **PVC**	**PVC** 扩容	**P**
文件存储	csi-nas	不支持	支持	支持	不
块存储	csi-disk	支持	不支持	支持	不

Azure AKS 集群

存储类型	默认存储类	使用 **RWO** 访问模式创建 **PVC**	使用 **RWX** 访问模式创建 **PVC**	**PVC** 扩容	**P**
文件存储	azurefile	支持	支持	支持	不
块存储	default	支持	不支持	支持	不

Google GKE 集群

存储类型	默认存储类	使用 **RWO** 访问模式创建 **PVC**	使用 **RWX** 访问模式创建 **PVC**	**PVC** 扩容	**P**
文件存储	standard-rwx	支持	支持	支持	不

存储类型	默认存储类	使用 **RWO** 访问模式创建 **PVC**	使用 **RWX** 访问模式创建 **PVC**	**PVC** 扩容	**P**
块存储	standard-rwo	支持	不支持	支持	不

AWS EKS 集群存储初始化配置

平台与 AWS EKS 的集成及存储初始化配置。

目录

约束与限制

- 默认的 efs-sc 文件存储类在挂载后可能不支持权限修改，可能导致 PostgreSQL、Jenkins 等部分应用无法正常运行。
- AL2023 AMI 不支持 A1 系列实例，导致 EBS 块存储插件（Amazon EBS CSI Driver）无法正常部署。EBS CSI 驱动已实现多架构/ARM 的 GA 支持，因此限制在于 AMI/实例支持，而非驱动本身。如果需要使用 EBS 块存储类，请避免使用以下实例类型，建议使用 Graviton2/3 替代方案：
  - a1.medium

- a1.large
- a1.xlarge
- a1.2xlarge
- a1.4xlarge

推荐替代方案：使用 m6g、c6g、r6g、t4g 等 Graviton2/3 实例系列，提供更优性能和完整的 EBS CSI 驱动支持。

前提条件

- 确保已安装 kubectl↗ 和 AWS CLI 工具。
- 若已创建 EKS 集群，请导入 Amazon EKS 集群；否则，创建 AWS EKS 集群。
- 在 EKS 集群中部署 EFS 文件存储插件 **Amazon EFS CSI Driver** 和 EBS 块存储插件 **Amazon EBS CSI Driver**。
  注意：使用 EFS 文件存储时，请在 EKS 所在地域创建文件存储，并记录 文件系统 **ID**。

配置步骤

创建存储类

1. 进入 平台管理，点击左侧导航中的 存储管理 > 存储类。
2. 点击 创建存储类 旁的下拉菜单 > 从 **YAML** 创建。
3. 在 YAML 文件中添加以下内容，根据需要创建默认存储类。默认文件存储类名称为 **efs-sc**，默认块存储类名称为 **ebs-sc**。

   - EFS 文件存储
     注意：将 `<File System ID>` 替换为实际的 文件系统 **ID**，例如 `fileSystemId: fs-05aef9e1edd309f2b`。

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: efs-sc
provisioner: efs.csi.aws.com
parameters:
  provisioningMode: efs-ap
  fileSystemId: <File System ID>
  directoryPerms: "755"
```

- EBS 块存储

```
allowVolumeExpansion: true
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ebs-sc
provisioner: ebs.csi.aws.com
reclaimPolicy: Delete
volumeBindingMode: WaitForFirstConsumer
```

4. 点击 创建。
   注意：若默认存储类不满足需求，可按上述步骤创建新的存储类并根据需要修改参数。详见可用存储类参数配置。

修改存储类项目分配

1. 在左侧导航中点击 存储管理 > 存储类。
2. 点击存储类名称为 **efs-sc** 或 **ebs-sc** 旁的三点按钮 > 更新项目。
3. 根据需要选择 项目分配 方式，点击 更新 将存储类分配给项目。

相关操作

配置可用存储类参数

- EFS 文件存储可用参数

参数	可选值	默认值	可选	描述
az		""	是	用于跨账户 与 az 关联的 户挂载；未 载目标进行
basePath			是	动态创建访 指定时，访 统根目录下
directoryPerms			否	创建 访问点 限。
uid			是	创建 访问点 用户 ID。
gid			是	创建 访问点 组 ID。
gidRangeStart		50000	是	创建 访问点 POSIX 组 II 置了 uid/gid 数。
gidRangeEnd		7000000	是	POSIX 组 II 置了 uid/gid 数。

参数	可选值	默认值	可选	描述
subPathPattern			是	构造动态创 在子路径的 符串和有限 nfs-subdir-e chart 中的 " 量。可选参 .PVC.name 和 .PV.nam
ensureUniqueDirectory		true	是	动态创建时 时，在 subl 模式后追加 不会意外指 意：仅在确 设置为 false
provisioningMode	efs-ap		否	EFS 卷类型 点。
fileSystemId			否	创建访问点

- EBS 块存储可用参数
  注意：不同卷类型的性能参数详见 Amazon EBS 卷类型↗。

参数	可选值	默认值	描述
"allowAutoIOPSPerGBIncrease"	true, false	false	设置为 "true" 时，当 iopsPerGB * <卷大小> 过低，无法满足 AWS 支持的 IOPS 范围时，CSI 驱动会自动增加卷的 IOPS。确保动态创建始终成功，即使用户指定

参数	可选值	默认值	描述
			的 PVC 容量或 iopsPerGB 过小，但可能产生额外费用，因为此类卷的 IOPS 高于 iopsPerGB 要求。
"blockExpress"	true, false	false	通过将 io2 卷的 IOPS 限制提升至 256000，创建 io2 Block Express 卷，但创建的 IOPS 超过 64000 的卷无法挂载到不支持 io2 Block Express 的实例。
"blockSize"			格式化底层文件系统时使用的块大小。仅适用于 Linux 节点且文件系统类型为 ext2、ext3、ext4 或 xfs。
"bytesPerINode"			格式化底层文件系统时每个 inode 占用的字节数。仅适用于 Linux 节点且文件系统类型为 ext2、ext3 或 ext4。

参数	可选值	默认值	描述
"csi.storage.k8s.io/fstype"	xfs, ext2, ext3, ext4	ext4	创建卷时格式化的文件系统类型，区分大小写。
"encrypted"	true, false	false	是否需要加密卷。
"inodeSize"			格式化底层文件系统时使用的 inode 大小。仅适用于 Linux 节点且文件系统类型为 ext2、ext3、ext4 或 xfs。inode 是文件系统中存储文件和目录元数据的数据结构。
"iops"			每秒 I/O 操作次数，适用于 IO1、IO2 和 GP3 卷。
"iopsPerGB"			每 GiB 每秒的 I/O 操作次数，适用于 IO1、IO2 和 GP3 卷。
"kmsKeyId"			用于加密卷的密钥完整 ARN。未指定时，AWS 使用卷所在区域的默认 KMS 密钥，并自动生成名为 /aws/ebs 的密钥。

参数	可选值	默认值	描述
"numberOfINodes"			格式化底层文件系统时指定的 inode 数量。仅适用于 Linux 节点且文件系统类型为 ext2、ext3 或 ext4。
"throughput"		125	吞吐量，单位 MiB/s。仅在指定 gp3 卷类型时有效。为空时默认为 125 MiB/s。详见 Amazon EBS 卷类型↗。
"type"	io1, io2, gp2, gp3, sc1, st1, standard, sbp1, sbg1	gp3	EBS 卷类型。

华为云 CCE 集群存储初始化配置

平台与华为云 CCE 集成及存储初始化配置。

目录

约束与限制

集群 PVC 数量有限制，账户存储容量有配额。您可以通过支持工单申请提升配额。

前提条件

- 如果您已创建 CCE 集群，请导入 CCE 集群（公有云）。

配置步骤

1. 进入 平台管理，然后在左侧导航中单击 存储管理 > 存储类。
2. 单击名为 **csi-nas** 或 **csi-disk** 的存储类旁边的三个点 > 更新项目。
   注意：导入 CCE 集群后，会生成默认存储类。块存储建议使用 **csi-disk**，文件存储建议使用 **csi-nas**。请参见 默认存储类说明。
3. 根据需要选择 项目分配 方式，然后单击 更新，将 **csi-nas** 或 **csi-disk** 存储类分配给项目。

默认存储类说明

存储类名称	存储类类型	说明
（推荐）**csi-disk**	块存储	推荐使用。
（推荐）**csi-nas**	文件存储	推荐使用。仅适用于支持 csi-nas 服务的区域。
csi-disk-topology	块存储	当节点跨可用区时，自动进行云硬盘拓扑分配。
csi-local	本地存储	
csi-local-topology	本地存储	
csi-obs	对象存储	
csi-sfsturbo	超高性能文件存储	超高性能文件存储无法动态创建持久卷。

常见问题

PVC 创建失败

由于 PVC 数量已达到上限，会出现以下错误。您可以通过支持工单申请增加配额：

```
message: "ShareLimitExceeded: Maximum number of shares allowed (10) excee
ded."
```

账户欠费

由于欠费，PVC 创建失败并出现以下错误。请先解决付款问题，然后重试：

```
message: "Your account is suspended and resources cannot be used"
```

Azure AKS 集群存储初始化配置

平台与 Azure AKS 的集成及存储初始化配置。

目录

约束与限制

默认的 azurefile 文件存储类在挂载后可能不支持权限修改，这可能导致某些应用如 PostgreSQL 和 Jenkins 无法正常运行。

前提条件

- 如果您已创建 AKS 集群，请导入 Azure AKS 集群。

配置步骤

1. 进入 平台管理，点击左侧导航中的 存储管理 > 存储类。
2. 点击名为 **default** 或 **azurefile** 的存储类旁的三点 > 更新项目。
   注意：导入 AKS 集群后，会生成以下默认存储类。推荐使用 **default** 作为块存储，**azurefile** 作为文件存储。详见默认存储类说明。
3. 根据需要选择 项目分配 方式，点击 更新，将 **default** 或 **azurefile** 存储类分配给项目。
   注意：如果默认存储类不满足需求，可按照上述步骤创建新的存储类并根据需要修改参数。详见可用存储类参数。

相关信息

默认存储类说明

存储类名称	存储类类型	说明
（推荐）**azurefile**	文件存储	使用 Azure 标准存储创建 Azure 文件共享。
（推荐）**default**	块存储	使用 Azure StandardSSD 存储创建托管磁盘。
azurefile-csi	文件存储	使用 Azure 标准存储创建 Azure 文件共享。
azurefile-csi-nfs	文件存储	使用 Azure 标准存储创建 Azure 文件共享，支持 NFS 协议。
azurefile-csi-premium	文件存储	使用 Azure 高级存储创建 Azure 文件共享。
azurefile-premium	文件存储	使用 Azure 高级存储创建 Azure 文件共享。
managed	块存储	使用 Azure StandardSSD 存储创建托管磁盘。
managed-csi	块存储	使用 Azure StandardSSD 本地冗余存储（LRS）创建托管磁盘。

存储类名称	存储类类型	说明
managed-csi-premium	块存储	使用 Azure 高级本地冗余存储（LRS）创建托管磁盘。
managed-premium	块存储	使用 Azure 高级存储创建托管磁盘。

可用存储类参数

- 有关块存储可选参数及含义，请参见在 Azure Kubernetes Service (AKS) 中使用 Azure 磁盘创建和使用卷↗。
- 有关文件存储可选参数及含义，请参见在 Azure Kubernetes Service (AKS) 中使用 Azure 文件创建和使用卷↗。

Google GKE 集群存储初始化配置

Google GKE 与存储初始化配置的平台集成。

目录

限制与约束

- 默认文件存储类型 PVC 的最小容量为 1T。如果在创建时将容量设置为小于 1T，则会自动扩容到 1T。
- 默认文件存储存在容量限制。您可以通过支持工单申请扩容。

- 默认文件存储的创建和删除操作都需要较长时间。如果资源长时间处于 `Creating` 状态，请继续监控该操作。

前提条件

- 创建集群时，在 Google Cloud Platform **Cluster > Features** 页面下的 **Other** 部分，勾选 **Enable Compute Engine Persistent Disk CSI Driver** 和 **Enable Filestore CSI Driver** 选项。
- 在 Google Cloud Platform 上启用 **Cloud Filestore API** 和 **Google Kubernetes Engine API**。请参见 使用 Filestore CSI 驱动程序访问 Filestore 实例↗。
- 调整 Google Cloud Platform 上的区域文件存储配额。请参见 资源配额和限制↗。
- 如果您已创建 GKE 集群，请导入 GKE 集群。

配置步骤

1. 进入 **Platform Management**，然后在左侧导航中点击 **Storage Management** > **Storage Classes**。
2. 点击名为 **standard-rwx** 或 **standard-rwo** 的存储类旁边的三个点 > **Update Project**。
   注意：导入 GKE 集群后，会生成默认存储类。文件存储推荐使用 **standard-rwx**，块存储推荐使用 **standard-rwo**。请参见 默认存储类说明。
3. 按需选择 **Project Assignment** 方式，然后单击 **Update**，将 **standard-rwx** 或 **standard-rwo** 存储类分配给项目。
   注意：如果默认存储类不满足需求，请按照上述步骤创建新的存储类，并根据需要修改参数。请参见 可用的存储类参数。

相关信息

默认存储类说明

存储类名称	存储类类型	描述
（推荐）**standard-rwx**	文件存储	使用 Basic HDD Filestore service tier ↗ 。
（推荐）**standard-rwo**	块存储	使用平衡型持久磁盘。
premium-rwx	文件存储	使用 Basic SSD Filestore service tier ↗ 。
premium-rwo	块存储	SSD 持久磁盘。
enterprise-rwx	文件存储	使用 Enterprise Filestore tier ↗ 。
enterprise-multishare-rwx	文件存储	使用 Enterprise Filestore tier ↗ 。请参见 Filestore multishares for Google Kubernetes Engine ↗ 。

可用的存储类参数

- 有关块存储可选参数及其含义，请参见 Storage options ↗ 。
- 有关文件存储可选参数及其含义，请参见 Service tiers ↗ 。

常见问题

文件存储类型存储类 PVC 创建失败

- 出现以下错误是因为项目中未启用 Cloud Filestore API，或者缺少相应权限。请参见 前提条件 进行解决：

```
failed to provision volume with StorageClass "standard-rwx": rpc error:
code = PermissionDenied desc = googlespi: Error 403: Cloud Filestore AP
I has not been used in project alauda-proj-1234 before or it is disable
d.
...
resion: SERVICE_DISABLED
```

- 出现以下错误是因为超出了存储配额。请参见 前提条件 进行解决：

```
failed to provision volume with StorageClass "standard-rwx": rpc error:
code = ResourceExhausted desc = googlespi: Error 429: Quora limit 'Stan
dardStorageGbPerRegion' has been exceeded. Limit 2048 in region asia-ea
st1.
 rateLimitExceeded
```

块存储类型存储类 **PVC** 无法正常绑定

出现以下错误是因为节点的 CSINode 未为 pd.csi.storage.gke.io 驱动程序配置内容。您可以通过重启 **Compute Engine Persistent Disk CSI Driver** 来解决。

注意：更新此插件会使集群不可用。更新过程大约需要 5-10 分钟。

```
Warning ProvisioningFailed 18m (x14 over 39m) pd.csi.storage.gke.io_gke-5
cb9bddae4d1430eb8ad-01f4-2084-vm_4b4e70bd-e2db-4779-9102-fee83a657ced fai
led to provision volume with StorageClass "standard": error generating ac
cessibility requirements: no available topology found
Normal ExternalProvisioning 4m35s (x143 over 39m) persistentvolume-contro
ller waiting for a volume to be created, either by external provisioner
"pd.csi.storage.gke.io" or manually created by system administrator
Normal Provisioning 3m19s (x18 over 39m) pd.csi.storage.gke.io_gke-5cb9bd
dae4d1430eb8ad-01f4-2084-vm_4b4e70bd-e2db-4779-9102-fee83a657ced External
provisioner is provisioning volume for claim "acp-gke-test/standard"
```

如何操作

导入集群的网络配置

获取导入集群信息

信任不安全的

从自定义命名的网卡采集网络数据

从自定义命名的网卡采集网络数据

为导入的标准 **Kubernetes** 集群配置审计采集

在导入的标准 Kubernetes 集群上启用 Kubernetes 审计日志记

导入集群的网络配置

在导入前，所需的连通路径可使 `global` 集群访问目标集群。导入后，导入集群中的平台组件也可能需要访问 `global` 集群。对于告警和日志收集等需要双向连通性的组件，请添加 platform URL 注解。

目录

前提条件

准备域名、域名指向的**IP** 地址以及导入集群可以访问的对应有效证书。

注意：

- 该域名不能与平台访问地址相同。
- 确保域名的端口（HTTPS 端口，与平台访问地址使用相同端口）可以将流量转发到 global 集群的所有控制节点。

添加平台 **URL** 注解

将该注解添加到导入集群资源中。

1. 在左侧导航栏中，点击集群管理 > 集群。
2. 点击 **global**。
3. 点击操作 > **CLI** 工具，并使用以下命令替换相关参数：

```
kubectl annotate --overwrite -n cpaas-system clusters.cluster.x-k8s.io
<imported cluster name> \
        cpaas.io/platform-url=<prepared domain address, e.g.: https://w
ww.domain.cn>
```

代码示例：

```
kubectl annotate --overwrite -n cpaas-system clusters.cluster.x-k8s.io
cluster-imported \
        cpaas.io/platform-url=https://www.domain.cn
```

如何获取导入集群信息？

目录

问题描述

获取连接导入集群所需的配置，以便授权平台访问并管理该集群。

前提条件

- 可正常使用的 `kubectl` 环境。对于公有云集群，强烈建议使用云厂商提供的 **CloudShell**。 如果没有 CloudShell，建议使用 Linux 或 macOS。
- 已获取导入集群的 KubeConfig 文件，并将其复制到安装了 `kubectl` 的环境中。为避免误操作到错误环境，强烈建议使用以下一种非破坏性方式：
  - 备份方式：先将现有 kubeconfig 复制到安全位置：`cp "${HOME}/.kube/config" "${HOME}/.kube/config.backup"`

- 隔离方式：将 `KUBECONFIG` 环境变量指向导入的 kubeconfig：`export KUBECONFIG="/path/to/imported/kubeconfig"`
- 合并方式：使用 kubectl 的 merge/flatten，且不丢失现有 context：
  1. `export KUBECONFIG="/path/to/imported/kubeconfig:${HOME}/.kube/config"`
  2. `kubectl config view --flatten > "${HOME}/.kube/merged.kubeconfig"`
  3. `export KUBECONFIG="${HOME}/.kube/merged.kubeconfig"`

获取集群信息

获取集群令牌

1. 运行以下命令：

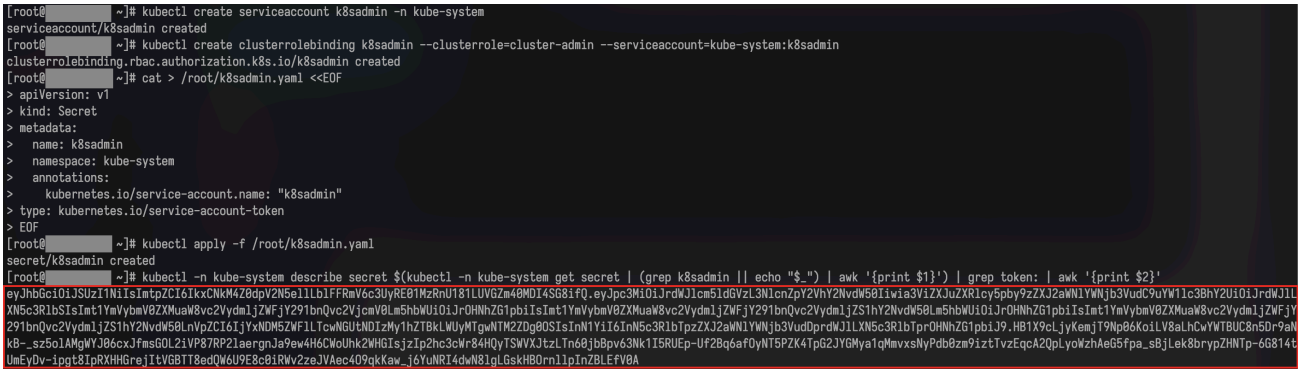
```
# [Important] The following operations support bash only

# Manually create a secret, bind a service account, and generate a non-expiring token
kubectl get ns cpaas-system > /dev/null 2>&1 || kubectl create namespace cpaas-system
kubectl create serviceaccount k8sadmin -n cpaas-system
kubectl create clusterrolebinding k8sadmin --clusterrole=cluster-admin --serviceaccount=cpaas-system:k8sadmin

cat | kubectl apply -f - <<EOF
apiVersion: v1
kind: Secret
metadata:
  name: k8sadmin
  namespace: cpaas-system
  annotations:
    kubernetes.io/service-account.name: "k8sadmin"
type: kubernetes.io/service-account-token
EOF

kubectl -n cpaas-system describe secret \
    $(kubectl -n cpaas-system get secret | (grep k8sadmin || echo "$_") | awk '{print $1}') \
    | grep -F 'token:' | awk '{print $2}'
```

WARNING

此操作步骤会创建一个 cluster-admin 凭证，且该凭证设计为不过期。

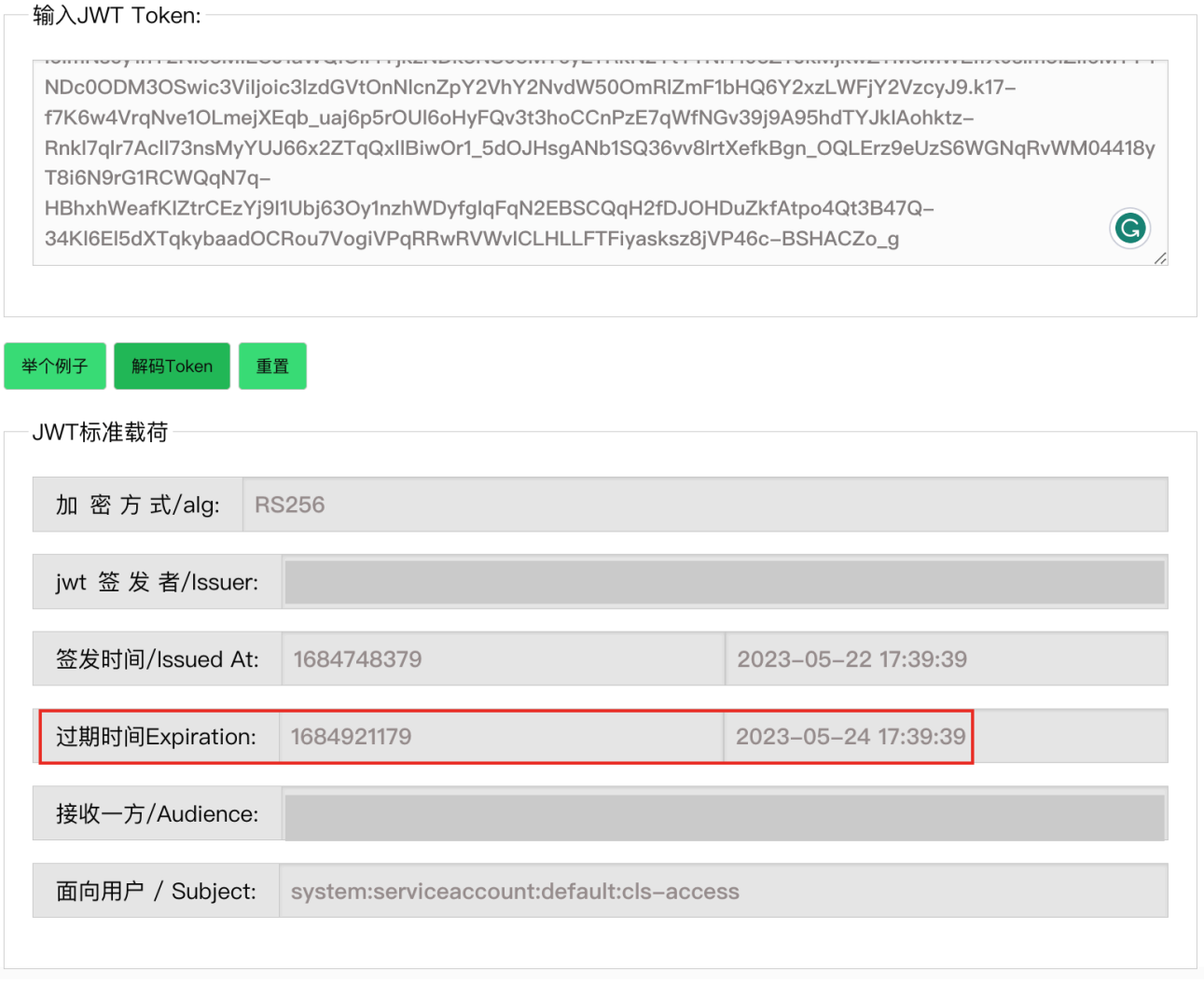
- 如有可能，优先采用仅授予所需资源最小权限的 RBAC。
- 请妥善保管令牌；在不再需要时及时轮换/撤销。
- 限制可读取 `cpaas-system` 中 `Secret` 对象的人员范围。

2. 下图为上一步获取到的令牌示例。



3. 验证令牌的过期时间。

   使用任何支持解析 JWT 令牌的工具来分析该令牌，并确认其过期时间。 如果在解析结果中能找到 expiration 字段（包含 “exp” 的键，如下所示），则平台在该时间之后将无法管理导入集群。 在这种情况下，请停止操作并联系技术支持。



> **TIP**
>
> 过期时间记录在原始 JWT payload 中的 `"exp": 1684486916,`。该值为 UNIX 时间戳，可转换为 UTC 时间。

完成后进行清理（撤销访问权限）：

```
kubectl delete clusterrolebinding k8sadmin
kubectl -n cpaas-system delete secret k8sadmin
kubectl -n cpaas-system delete serviceaccount k8sadmin
```

获取导入集群 **API server** 地址和 **CA** 证书

> **TIP**
>
> 如果你已经通过导入集群页面上的 `Parse KubeConfig File` 功能获取了 API server 地址和 CA 证书，可跳过此步骤。

1. 运行以下命令：

```
# View the import cluster API server addresses. There may be multiple a
ddresses; choose the one that fits your environment.
kubectl --kubeconfig "${KUBECONFIG:-${HOME}/.kube/config}" config view
--show-managed-fields=false --flatten --raw -ojsonpath='{$.clusters..cl
uster.server}'
addr_apiserver='<Selected API server address>'

# Get the CA certificate for the API server specified above
kubectl --kubeconfig "${KUBECONFIG:-${HOME}/.kube/config}" config view
--show-managed-fields=false --flatten --raw \
    -ojsonpath="{$.clusters[?(@.cluster.server == '${addr_apiserve
r}')].cluster.certificate-authority-data}" \
    | { base64 -d 2>/dev/null || base64 -D; }
```

如何信任不安全的镜像仓库？

目录

问题描述

托管组件镜像的镜像仓库平台可能不提供 HTTPS 服务，或者没有由公共证书颁发机构签发的有效 TLS 证书。如果您信任该仓库，请按照以下步骤配置您的容器运行时。

配置对不安全镜像仓库的信任

注意：

- 所有需要使用镜像的节点，包括新加入的节点，都必须进行配置并重启 Containerd。
- Containerd v1.4/v1.5 与 v1.6 的配置略有不同，请根据您的版本选择相应步骤。

1. 在导入集群的每个节点上运行以下命令：

    - 备份配置文件

```
mkdir -p '/var/backup-containerd-confs/'
if ! [ -f /etc/containerd/config.toml ]; then
    echo '未找到 Containerd 配置。请检查 containerd 是否正确安装。如仍无法解决，请联系技术支持。'
    exit 1
else
    cp /etc/containerd/config.toml /var/backup-containerd-confs/config.toml_$(date +%F_%T)
fi
```

- 获取 Containerd 运行时版本

```
# 获取 containerd 版本
# 与 v1.6 版本对比，选择相应步骤
ctr --version | grep -Eo 'v[0-9]+\.[0-9]+\.[0-9]+'
```

Containerd v1.4 v1.5 不安全仓库配置

2. 在导入集群的每个节点上运行以下命令：

- 编辑 `/etc/containerd/config.toml`

```
# 添加到配置文件的示例内容
# 方括号内为节名称，若文件已有同名节，则合并内容
[plugins."io.containerd.grpc.v1.cri".registry]
    [plugins."io.containerd.grpc.v1.cri".registry.mirrors]
        [plugins."io.containerd.grpc.v1.cri".registry.mirrors."<registry-address>"]
            endpoint = ["https://<registry-address>", "http://<registry-address>"]
        [plugins."io.containerd.grpc.v1.cri".registry.mirrors."192.168.134.43"]
            endpoint = ["https://192.168.134.43", "http://192.168.134.43"]
    [plugins."io.containerd.grpc.v1.cri".registry.configs]
        [plugins."io.containerd.grpc.v1.cri".registry.configs."<registry-address>".tls]
            insecure_skip_verify = true
        [plugins."io.containerd.grpc.v1.cri".registry.configs."192.168.134.43".tls]
            insecure_skip_verify = true
```

- 重启 Containerd。

```
systemctl daemon-reload && systemctl restart containerd
```

Containerd v1.6 不安全仓库配置

2. 在导入集群的每个节点上运行以下命令：

- 检查配置中是否存在 `config_path`。

```
if ! grep -qF 'config_path' /etc/containerd/config.toml; then
    if grep -qE '\[plugins."io.containerd.grpc.v1.cri".registry.(mirr
ors|configs)(\.|\])' /etc/containerd/config.toml; then
        echo '请按照“Containerd v1.4 v1.5 不安全仓库配置”中的步骤操作。'
    else
        cat >> /etc/containerd/config.toml << 'EOF'
[plugins."io.containerd.grpc.v1.cri".registry]
    config_path = "/etc/containerd/certs.d/"
EOF
    fi
fi

config_path_var=$(grep -F '/etc/containerd/certs.d' /etc/containerd/c
onfig.toml)
if [ -z "$config_path_var" ]; then
   echo '文件中 config_path 的值异常，请检查！'
   exit 1
fi
```

- 创建 `hosts.toml` 文件。
  如果前面的命令输出了 `请按照“Containerd v1.4 v1.5 不安全仓库配置”中的步骤操作。`，请参见 *Containerd v1.4 v1.5 不安全仓库配置*。

```
REGISTRY='<registry address obtained in the "Get the registry addres
s" section>'

mkdir -p "/etc/containerd/certs.d/$REGISTRY/"
cat > "/etc/containerd/certs.d/$REGISTRY/hosts.toml" << EOF
server = "$REGISTRY"
[host."http://$REGISTRY"]
  capabilities = ["pull", "resolve", "push"]
  skip_verify = true
[host."https://$REGISTRY"]
  capabilities = ["pull", "resolve", "push"]
  skip_verify = true
EOF
```

- 重启 Containerd。

```
systemctl daemon-reload && systemctl restart containerd
```

从自定义命名的网卡采集网络数据

目录

场景描述

创建业务集群后，平台监控默认只能识别匹配 `eth.*|en.*|wl.*|ww.*` 等模式的网卡名称。对于用户自定义的网卡名称，监控页面无法查看网络流量数据。为此，平台支持修改相关资源参数，手动采集网卡流量数据。

操作步骤

1. 登录 global 集群的控制节点，使用 kubectl 执行以下命令。
2. 首先，在 global 集群中查找对应业务集群的 moduleinfo 资源名称：

```
kubectl get moduleinfo | grep -E 'prometheus|victoriametrics'
```

示例输出：

```
global-6448ef7f7e5e3924c1629fad826372e7     global     prometheus
prometheus                              Running   v3.15.0-zz231204040711-9d
1fc12474c2   v3.15.0-zz231204040711-9d1fc12474c2   v3.15.0-zz2312040407
11-9d1fc12474c2
ovn-0954f21f0359720e8c115804376b3e7e        ovn        prometheus
prometheus                              Running   v3.15.0-zz231204040711-9d
1fc12474c2   v3.15.0-zz231204040711-9d1fc12474c2   v3.15.0-zz2312040407
11-9d1fc12474c2
```

3. 编辑业务集群对应的 moduleinfo 资源，将 `ovn-0954f21f0359720e8c115804376b3e7e` 替换为上一步查到的业务集群 moduleinfo 资源名称：

```
kubectl edit moduleinfo ovn-0954f21f0359720e8c115804376b3e7e
```

4. 添加 valuesOverride 字段，并根据注释信息修改字段和正则表达式：

```
spec:
  valuesOverride: # 如果该字段不存在，需要在 spec 下新增 valuesOverride 字段
及以下参数
    ait/chart-cpaas-monitor:
      ovn: # 替换为业务集群名称
        indicator:
          networkDevice: eth.*|em.*|en.*|wl.*|ww.*|[A-Z].*i|custom_inte
rface # 将 custom_interface 替换为自定义正则表达式，确保正确匹配网卡名称
```

5. 等待 10 分钟后，检查节点监控页面的网络相关图表，确认修改生效。

如何为导入的标准 Kubernetes 集群配置审计采集？

目录

场景说明

将标准 Kubernetes 集群导入平台后，必须先在该集群上启用 Kubernetes API server 审计日志记录，平台才能从该集群采集审计数据。

本文档中的操作步骤适用于控制平面节点由您自行管理的标准 Kubernetes 集群，例如基于 kubeadm 的集群。对于您无法登录或修改控制平面节点的云厂商托管 Kubernetes 服务，则不适用。

前提条件

- 标准 Kubernetes 集群已导入平台。
- 您可以登录集群中的每个控制平面节点。

- 集群使用标准的 kubeadm 风格 API server 静态 Pod manifest 路径：`/etc/kubernetes/manifests/kube-apiserver.yaml`。

操作步骤

1. 为审计策略创建本地 `policy.yaml` 文件。
   根据 Kubernetes 版本设置 `apiVersion`：

   - Kubernetes 1.24 之前：`audit.k8s.io/v1beta1`
   - Kubernetes 1.24 及更高版本：`audit.k8s.io/v1`

   使用以下内容：

```
apiVersion: audit.k8s.io/v1
kind: Policy
omitStages:
  - "RequestReceived"
rules:
  - level: None
    users:
      - system:kube-controller-manager
      - system:kube-scheduler
      - system:serviceaccount:kube-system:endpoint-controller
    verbs: ["get", "update"]
    namespaces: ["kube-system"]
    resources:
      - group: ""
        resources: ["endpoints"]
  - level: None
    nonResourceURLs:
      - /healthz*
      - /version
      - /swagger*
  - level: None
    resources:
      - group: ""
        resources: ["events"]
  - level: None
    resources:
      - group: "devops.alauda.io"
  - level: None
    verbs: ["get", "list", "watch"]
  - level: None
    namespaces:
      - kube-system
      - cpaas-system
      - alauda-system
      - istio-system
      - kube-node-lease
    resources:
      - group: "coordination.k8s.io"
        resources: ["leases"]
  - level: None
    resources:
      - group: "authorization.k8s.io"
        resources: ["subjectaccessreviews", "selfsubjectaccessreviews"]
```

```
    - group: "authentication.k8s.io"
      resources: ["tokenreviews"]
  - level: Metadata
    resources:
    - group: ""
      resources: ["secrets", "configmaps"]
  - level: RequestResponse
    resources:
    - group: ""
    - group: "aiops.alauda.io"
    - group: "apps"
    - group: "app.k8s.io"
    - group: "authentication.istio.io"
    - group: "auth.alauda.io"
    - group: "autoscaling"
    - group: "asm.alauda.io"
    - group: "clusterregistry.k8s.io"
    - group: "crd.alauda.io"
    - group: "infrastructure.alauda.io"
    - group: "monitoring.coreos.com"
    - group: "networking.istio.io"
    - group: "networking.k8s.io"
    - group: "portal.alauda.io"
    - group: "rbac.authorization.k8s.io"
    - group: "storage.k8s.io"
    - group: "tke.cloud.tencent.com"
    - group: "devopsx.alauda.io"
    - group: "core.katanomi.dev"
    - group: "deliveries.katanomi.dev"
    - group: "integrations.katanomi.dev"
    - group: "builds.katanomi.dev"
    - group: "operators.katanomi.dev"
    - group: "tekton.dev"
    - group: "operator.tekton.dev"
    - group: "eventing.knative.dev"
    - group: "flows.knative.dev"
    - group: "messaging.knative.dev"
    - group: "operator.knative.dev"
    - group: "sources.knative.dev"
    - group: "operator.devops.alauda.io"
  - level: Metadata
```

TIP

如果集群版本早于 1.24，仅将 `apiVersion` 字段更改为 `audit.k8s.io/v1beta1`。其余策略内容保持不变。

2. 将 `policy.yaml` 上传到每个控制平面节点上的 `/etc/kubernetes/audit/` 目录。

> **WARNING**
>
> - 如果集群有多个控制平面节点，请将该文件上传到每个节点。
> - 如果目录不存在，请手动创建：`/etc/kubernetes/audit/`

3. 更新每个控制平面节点上的 `/etc/kubernetes/manifests/kube-apiserver.yaml`。在 `spec.containers[].command` 中添加或更新以下与审计相关的标志：

标志	是否必需	说明
`--audit-policy-file`	是	必须设置为 `/etc/kubernetes/audit/policy.yaml`。
`--audit-log-format`	是	必须设置为 `json`。
`--audit-log-path`	是	必须设置为 `/etc/kubernetes/audit/audit.log`。
`--audit-log-mode`	否	推荐值：`batch`。
`--audit-log-maxsize`	否	审计日志文件的最大大小，单位为 MiB。推荐值：`200`。
`--audit-log-maxbackup`	否	保留的审计日志文件数量。推荐值：`2`。

示例：

```
- --audit-log-format=json
- --audit-log-maxbackup=2
- --audit-log-maxsize=200
- --audit-log-mode=batch
- --audit-log-path=/etc/kubernetes/audit/audit.log
- --audit-policy-file=/etc/kubernetes/audit/policy.yaml
```

4. 将审计目录挂载配置添加到同一个 `kube-apiserver.yaml` 文件中。
   在 `spec.containers[].volumeMounts` 下添加以下项：

```
- mountPath: /etc/kubernetes/audit
  name: k8s-audit
```

   在 `spec.volumes` 下添加以下项：

```
- hostPath:
    path: /etc/kubernetes/audit
    type: DirectoryOrCreate
  name: k8s-audit
```

> **WARNING**
>
> - 如果集群有多个控制平面节点，请在每个节点上更新 manifest。
> - `volumeMounts[].name` 的值必须与对应的 `volumes[].name` 值一致。
> - 不要更改挂载路径 `/etc/kubernetes/audit`。

5. 保存文件并验证配置是否生效。
   检查每个控制平面节点上是否生成了 `/etc/kubernetes/audit/audit.log`。如果该文件存在并且包含审计记录，则说明配置已生效。

```
ls -l /etc/kubernetes/audit/audit.log
tail -n 20 /etc/kubernetes/audit/audit.log
```

创建本地集群

目录

前提条件

节点要求

1. 如果您从下载安装包下载了单架构安装包，请确保节点机器的架构与安装包一致。否则，由于缺少对应架构的镜像，节点将无法启动。
2. 请确认节点操作系统和内核受支持。详情请参见支持的 OS 和内核。
3. 对节点机器执行可用性检查。具体检查项请参见节点预处理 > 节点检查。
4. 如果节点机器 IP 无法通过 SSH 直接访问，请为节点提供 SOCKS5 代理。`global` 集群将通过该代理服务访问节点。

负载均衡

对于生产环境，集群控制平面节点需要负载均衡器，以确保高可用。

如果您的环境中已有硬件负载均衡器，建议使用它。
如果没有，可以启用 `Self-built VIP`，它使用 keepalived 提供软件负载均衡能力。

建议：使用硬件负载均衡器时，建议将集群 Endpoint 配置为域名。如果集群安装后硬件负载均衡器的 VIP 发生变化，您可以更新 DNS 记录，而无需重新配置集群。

注意：`Self-built VIP` 不支持域名配置。

负载均衡器必须能够正确转发到集群中所有控制平面节点上的 `6443`、`11780` 和 `11781` 端口。

如果您的集群只有一个控制平面节点，并且使用该节点的 IP 作为 Cluster Endpoint 参数，那么后续无法从单节点扩展为高可用的多节点部署。因此，即使是单节点集群，我们也建议提供负载均衡器。

启用 `Self-built VIP` 时，您需要准备：

1. 一个可用的 VRID
2. 支持 VRRP 协议的主机网络
3. 所有控制平面节点和 VIP 必须位于同一子网中，且 VIP 不能与任何节点 IP 重复

连接 `global` 集群和业务集群

平台要求 `global` 集群与业务集群之间能够相互访问。如果它们不在同一网络中，您需要：

1. 为业务集群提供 `External Access`，以确保 `global` 集群可以访问它。网络要求必须确保 `global` 能够访问所有控制平面节点上的 `6443`、`11780` 和 `11781` 端口。
2. 为 `global` 添加一个业务集群可访问的额外地址。创建业务集群时，请将该地址添加到集群注解中，键为 `cpaas.io/platform-url`，值设置为 `global` 的公网访问地址。

镜像仓库

集群镜像支持 Platform Built-in、Private Repository 和 Public Repository 选项。这些选项用于配置业务集群使用的平台镜像源。

- **Platform Built-in**：使用为 `global` 集群配置的平台镜像源。该镜像源可以是随 `global` 部署的 Registry，也可以是安装过程中选择的外部平台镜像仓库。如果镜像源是内置 Registry 且业务集群无法访问 `global`，请参见为内置 Registry 添加外部地址。
- **Private Repository**：为该业务集群使用其他由客户管理的镜像仓库。在选择之前，请上传所选 Kubernetes 版本和架构所需的完整镜像内容，并满足准备外部平台镜像仓库中描述的能力和访问条件。用于生产时，还应遵循 HTTPS、DNS、可用性、监控、备份、恢复和保留方面的生产建议。
- **Public Repository**：使用平台的公共镜像仓库。使用前，请完成更新公共仓库凭据。

容器网络

如果您计划在集群中使用 Kube-OVN 的 Underlay，请准备 Kube-OVN Underlay 物理网络。

创建步骤

1. 进入管理员视图，然后在左侧导航栏中单击 **Clusters/Clusters**。
2. 单击 **Create Cluster**。
3. 按照以下说明配置各个部分：Basic Info、Container Network、Node Settings 和 Extended Parameters。

基本信息

参数	说明

Kubernetes Version	所有可选版本都经过严格测试，具有稳定性和兼容性。 建议：选择最新版本以获得最佳功能和支持。
Container Runtime	默认提供 Containerd 作为容器运行时。
Cluster Network Protocol	支持三种模式：IPv4 单栈、IPv6 单栈、IPv4/IPv6 双栈。 注意：如果选择双栈模式，请确保所有节点都已正确配置 IPv6 地址；设置后无法更改网络协议。
Cluster Endpoint	`IP Address / Domain`：如果没有域名，请输入预先准备好的域名或 VIP。 `Self-built VIP`：默认禁用。仅当您未提供 LoadBalancer 时才启用。启用后，安装程序会自动部署 `keepalived`，以支持软件负载均衡。 `External Access`：当集群与 `global` 集群不处于同一网络环境时，请输入为集群准备的外部可访问地址。

容器网络

Kube-OVN

Kube-OVN 是由 灵雀云 开发的云原生 Kubernetes 容器网络编排系统。它为跨云网络管理、传统网络架构集成、基础设施互联以及边缘集群部署场景提供 Kubernetes 网络能力。

参数	说明
Subnet	也称为 Cluster CIDR，表示默认子网段。集群创建后，可以添加额外的子网。
Transmit Mode	**Overlay**：在基础设施之上抽象出的虚拟网络，不占用物理网络资源。创建 Overlay 默认子网时，集群中的所有 Overlay 子网都使用相同的 cluster NIC 和 node NIC 配置。 **Underlay**：这种传输方式依赖物理网络设备。它可以直接为 Pod 分配物理网络地址，从而与物理网络实现更好的性能和连通性。Underlay 子网中的节点必须具有多个 NIC，并且用于桥接网络的 NIC 必须专用于 Underlay，不能承载 SSH 等其他流量。创建 Underlay 默认子网时，cluster NIC 实际上是桥接网络的默认 NIC，而 node NIC 是桥接网络中的 node NIC 配置。 • **Default Gateway**：物理网络网关地址，即 Cluster CIDR 段的网关地址（必须位于 Cluster CIDR 地址范围内）。 • **VLAN ID**：虚拟局域网标识符（VLAN 编号），例如 `0`。 • **Reserved IPs**：设置不会被自动分配的保留 IP，例如子网中已被其他设备使用的 IP。
Service CIDR	Kubernetes 中类型为 ClusterIP 的 Services 使用的 IP 地址范围。不能与默认子网范围重叠。
Join CIDR	在 Overlay 传输模式下，这是节点与 Pod 之间通信使用的 IP 地址范围。不能与默认子网或 Service CIDR 重叠。

Calico

Calico 是一种三层网络解决方案，可为容器提供安全的网络连接。

参数	说明
Default Subnet	也称为 Cluster CIDR，表示默认子网段。集群创建后，可以添加额外的子网。
Service CIDR	Kubernetes 中类型为 ClusterIP 的 Services 使用的 IP 地址范围。不能与默认子网范围重叠。

Flannel

Flannel 为集群中的所有容器提供扁平网络环境，使在不同节点主机上创建的容器在整个集群中拥有唯一的虚拟 IP 地址。Pod 子网会按照掩码平均分配给集群节点，每个节点上的 Pod 会从分配给该节点的网段中获取 IP 地址。这提高了容器之间的通信效率，而无需考虑 IP 转换问题。

参数	说明
Cluster CIDR	集群启动时创建的 Pod 使用的 IP 地址范围。支持设置当前容器网络下每个节点可分配给 Pod 的最大 IP 地址数量。 注意：平台会根据上述配置自动计算集群可容纳的最大节点数，并显示在输入框下方的提示中。 重要：集群创建后，集群网络无法更改。在创建集群之前请仔细规划网络。

Service CIDR	Kubernetes 中类型为 ClusterIP 的 Services 使用的 IP 地址范围。不能与容器子网范围重叠。

Custom

如果您需要安装其他网络插件，请选择 **Custom** 模式。集群成功创建后，您可以手动安装网络插件。

参数	说明
Cluster CIDR	集群启动时创建的 Pod 使用的 IP 地址范围。
Service CIDR	Kubernetes 中类型为 ClusterIP 的 Services 使用的 IP 地址范围。不能与容器子网范围重叠。

节点设置

参数	说明
Network Interface Card	集群网络插件使用的主机网络接口设备名称。 注意： • 当为 Kube-OVN 默认子网选择 Underlay 传输模式时，必须指定网络接口名称，该接口将作为桥接网络的默认 NIC。 - 平台默认对名称类似于 `eth.|en.|wl.|ww.` 的接口进行网络接口流量监控识别。如果您使用的接口命名规则不同，请在集群纳管后参见从自定义

	名称的网络接口收集网络数据修改相关资源，以确保平台能够正确监控网络接口流量。
Node Name	您可以选择使用节点 IP 或主机名作为平台上的节点名称。 注意：当选择使用主机名作为节点名称时，请确保添加到集群中的节点主机名是唯一的。
Nodes	添加节点到集群，或从草稿中恢复临时保存的节点信息。有关添加节点的详细参数说明，请参见下文。
Monitoring Type	支持 **Prometheus** 和 **VictoriaMetrics**。 当选择 **VictoriaMetrics** 作为监控组件时，必须配置 **Deploy Type**： **- Deploy VictoriaMetrics**：部署所有相关组件，包括 **VMStorage**、**VMAlert**、**VMAgent** 等。 **- Deploy VictoriaMetrics Agent**：仅部署日志采集组件 **VMAgent**。使用此部署方式时，您需要关联平台中另一个集群上已部署的 VictoriaMetrics 实例，为该集群提供监控服务。
Monitoring Nodes	选择用于部署集群监控组件的节点。支持选择允许部署应用的计算节点和控制平面节点。 为避免影响集群性能，建议优先选择计算节点。集群成功创建后，存储类型为 **Local Volume** 的监控组件将部署在所选节点上。

添加节点参数

参数	说明
Type	**Control Plane Node**：负责运行集群中的 kube-apiserver、kube-scheduler、kube-controller-manager、etcd、容器网络以及部分平台管理组件。当启用 **Application Deployable** 时，控制平面节点也可以用作计算节点。 **Worker Node**：负责承载集群中运行的业务 Pod。
IPv4 Address	节点的 IPv4 地址。对于以内网模式创建的集群，请输入节点的私有 **IP**。
IPv6 Address	当集群启用 IPv4/IPv6 双栈时有效。填写节点的 IPv6 地址。
Application Deployable	当 **Node Type** 为 **Control Plane Node** 时有效。是否允许在该控制平面节点上部署业务应用，将业务相关 Pod 调度到该节点。
Display Name	节点的显示名称。
SSH Connection IP	通过 SSH 服务访问节点时可连接的 IP 地址。 如果您可以使用 `ssh <username>@<node's IPv4 address>` 登录节点，则不需要填写此参数；否则，请输入节点的公网 IP 或 NAT 外网 IP，以确保 `global` 集群和代理能够通过该 IP 连接到节点。
Network Interface Card	输入节点使用的网络接口名称。网络接口配置生效的优先级如下（从左到右，优先级依次降低）：

	Kube-OVN Underlay：Node NIC > Cluster NIC **Kube-OVN Overlay**：Node NIC > Cluster NIC > 节点默认路由对应的 NIC **Calico**：Cluster NIC > 节点默认路由对应的 NIC **Flannel**：Cluster NIC > 节点默认路由对应的 NIC
Associated Bridge Network	注意：创建集群时不支持桥接网络配置；此选项仅在向已创建 Underlay 子网的集群中添加节点时可用。 请选择现有的添加桥接网络。如果您不想使用桥接网络的默认 NIC，可以单独配置节点 NIC。
SSH Port	SSH 服务端口号，例如 `22`。
SSH Username	SSH 用户名，需要是具有 root 权限的用户，例如 `root`。
Proxy	是否通过代理访问节点的 SSH 端口。当 `global` 集群无法通过 SSH 直接访问待添加节点时（例如 `global` 集群与业务集群不在同一子网；节点 IP 是 `global` 集群无法直接访问的内网 IP），需要开启此开关并配置代理相关参数。配置代理后，可通过代理实现节点访问和部署。

	注意：当前仅支持 SOCKS5 代理。 **Access URL**：代理服务器地址，例如 `192.168.1.1:1080`。 **Username**：访问代理服务器的用户名。 **Password**：访问代理服务器的密码。
SSH Authentication	登录到所添加节点时所使用的认证方式及相应认证信息。可选项包括： **Password**：需要具有 root 权限的用户名及对应的 **SSH password**。 **Key**：需要具有 root 权限的 **private key** 及 **private key password**。
Save Draft	将当前对话框中已配置的数据保存为草稿，并关闭 **Add Node** 对话框。 在不离开 **Create Cluster** 页面时，您可以选择 **Restore from draft** 打开 **Add Node** 对话框，并恢复保存为草稿的配置数据。 注意：从草稿恢复的数据是最近一次保存的草稿数据。

扩展参数

注意：

- 除必需配置外，不建议设置扩展参数，因为错误的设置可能会导致集群不可用，并且在集群创建后无法修改。
- 如果输入的 **Key** 与默认参数 **Key** 重复，则会覆盖默认配置。

操作步骤

1. 单击 **Extended Parameters** 以展开扩展参数配置区域。您可以为集群选择性设置以下扩展参数：

参数	说明
Kubelet Parameters	`kubeletExtraArgs`，Kubelet 的附加配置参数。 注意：当填写 **Container Network** 的 **Node IP Count** 参数时，系统会自动生成一个默认的 **Kubelet Parameter** 配置，其键为 `max-pods`，值为 **Node IP Count**。该配置用于设置集群中任意节点可运行的 Pod 最大数量。此配置不会在界面中显示。 在 **Kubelet Parameters** 区域新增 `max-pods: maximum number of runnable pods` 键值对会覆盖默认值。允许填写任意正整数，但建议使用默认值（Node IP Count）或填写不超过 `256` 的值。
Controller Manager Parameters	`controllerManagerExtraArgs`，Controller Manager 的附加配置参数。
Scheduler Parameters	`schedulerExtraArgs`，Scheduler 的附加配置参数。
APIServer Parameters	`apiServerExtraArgs`，APIServer 的附加配置参数。
APIServer URL	`publicAlternativeNames`，证书中签发的 APIServer 访问地址。仅可输入 IP 或域名，最长 253 个字符。

Cluster Annotations	集群注解信息，以键值对形式在元数据中标记集群特征，供平台组件或业务组件获取相关信息。

4. 单击 **Create**。您将返回集群列表页面，集群状态将显示为 **Creating**。

创建后步骤

查看创建进度

在集群列表页面，您可以查看已创建集群的列表。对于处于 **Creating** 状态的集群，您可以查看其执行进度。

操作步骤

1. 单击集群状态右侧的 **View Execution Progress** 小图标。
2. 在弹出的执行进度对话框中，您可以查看集群的执行进度（status.conditions）。
   提示：当某个类型处于进行中状态，或者在失败状态下带有原因时，将鼠标悬停在对应原因（蓝色文本）上，可查看该原因的详细信息（status.conditions.reason）。

关联项目

集群创建完成后，您可以在项目管理视图中将其添加到项目中。

如何操作

为内置镜像仓库添加外部地址

目录

概述

当 `global` 集群使用 `Platform Built-in` 镜像仓库时，业务集群通常也会使用该镜像仓库拉取镜像。该镜像仓库不仅为 `global` 集群内的组件提供服务，还必须可供业务集群节点访问。

在某些场景下，业务集群节点无法直接访问 `global` 集群的镜像仓库地址。例如，`global` 集群位于私有数据中心，而业务集群位于公有云或边缘环境中。

为平台的默认镜像仓库配置一个外部可访问的地址，以便业务集群拉取镜像。

前提条件

在开始之前，请准备以下内容：

- 业务集群节点可访问的域名
- 该域名所指向的 IP 地址
- 该域名的有效 SSL 证书

> 警告
>
> - 域名必须与平台访问地址不同
> - 确保该域名的 IP 地址能够将流量转发到 `global` 集群的所有控制平面节点

操作步骤

为平台镜像仓库配置证书和路由规则

1. 将该域名的有效证书复制到 `global` 集群的任一控制平面节点上
2. 创建一个包含域名证书的 TLS secret：

   ```
   kubectl create secret tls registry-address.tls --cert=<certificate-file
   name> --key=<key-filename> -n kube-system
   ```

   示例：

   ```
   kubectl create secret tls registry-address.tls --cert=custom.crt --key=
   custom.key -n kube-system
   ```

   注意：创建证书后，请监控 `global` 集群中 `kube-system` 命名空间下 **registry-address.tls** secret 的过期时间，并在过期前替换证书。
3. 在 `global` 集群的任一控制平面节点上创建 ingress 规则：

```
REGISTRY_DOMAIN_NAME=<www.registry.com> # Replace with your accessible
domain name
cat << EOF | kubectl create -f -
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  annotations:
    nginx.ingress.kubernetes.io/backend-protocol: HTTPS
  name: registry-address
  namespace: kube-system
  labels:
    service_name: registry
spec:
  rules:
    - host: $REGISTRY_DOMAIN_NAME
      http:
        paths:
          - backend:
              service:
                name: registry
                port:
                  number: 443
            path: /v2/
            pathType: ImplementationSpecific
          - backend:
              service:
                name: registry
                port:
                  number: 443
            path: /v2/_catalog
            pathType: ImplementationSpecific
          - backend:
              service:
                name: registry
                port:
                  number: 443
            path: /v2/.+/tags/list
            pathType: ImplementationSpecific
          - backend:
              service:
                name: registry
                port:
                  number: 443
```

```
            path: /v2/.+/manifests/[A-Za-z0-9_+.-:]+
            pathType: ImplementationSpecific
          - backend:
              service:
                name: registry
                port:
                  number: 443
            path: /v2/.+/blobls/[A-Za-z0-9-:]+
            pathType: ImplementationSpecific
          - backend:
              service:
                name: registry
                port:
                  number: 443
            path: /v2/.+/blobls/uploads/[A-Za-z0-9-:]+
            pathType: ImplementationSpecific
          - backend:
              service:
                name: registry
                port:
                  number: 443
            path: /auth/token
            pathType: ImplementationSpecific
  tls:
    - secretName: registry-address.tls
      hosts:
        - $REGISTRY_DOMAIN_NAME
EOF
```

如果返回类似 `... created` 的响应，则表示 ingress 创建成功。

4. 检查是否存在 Registry Service 资源：

```
kubectl -n kube-system get svc | grep registry
```

如果该 Service 不存在，请使用以下命令创建：

```
cat << EOF | kubectl create -f -
apiVersion: v1
kind: Service
metadata:
  labels:
    name: registry
    service_name: registry
  name: registry
  namespace: kube-system
spec:
  ports:
    - protocol: TCP
      port: 443
      targetPort: 60080
  selector:
    component: registry
  type: ClusterIP
EOF
```

5. 使用该域名从镜像仓库拉取镜像，以测试配置：

```
crictl pull <registry-domain-name>/automation/qaimages:helloworld
```

或者

```
podman pull <registry-domain-name>/automation/qaimages:helloworld
```

Updating Public Repository Credentials

目录

Overview

`Public Repository` 是一个由平台提供的公开互联网镜像仓库服务。当您希望集群使用 `Public Repository` 作为镜像仓库时，需要更新内置的 `public-registry-credential` Cloud Credentials，以确保平台有权限从公共仓库拉取镜像。

Procedure

1. 登录 灵雀云 **Customer Portal**，并从右上角 用户信息 下拉菜单中的 企业管理 部分下载贵组织的认证文件。
2. 在 管理员 控制台左侧导航栏中，进入 集群 > **Cloud Credential**。
3. 找到名为 `public-registry-credential` 的云凭证，点击右侧下拉菜单中的 更新。
4. 在 上传公共仓库地址 部分，上传您从 灵雀云 **Customer Portal** 下载的认证文件。

5. 点击 更新 以应用更改。

使用 KubeConfig 访问集群

目录

简介

当你需要通过 ACP CLI（`ac`）、`kubectl`、`helm`、`client-go` 或类似的 Kubernetes 客户端从本地访问集群时，请使用 KubeConfig。

KubeConfig 文件是一个 YAML 客户端配置文件，它不是 Kubernetes 资源。KubeConfig 文件通常定义目标集群端点、客户端凭证以及将二者组合起来的上下文。

ACP CLI（`ac`）使用标准 kubeconfig，并提供类似 kubectl 的体验。它与 `kubectl` 工作流兼容，同时还增加了平台特定的上下文、集群和会话操作。如果你已经使用 `ac login`，ACP CLI 可以自动配置 kubeconfig 并创建上下文。下面的访问路径包括从平台下载的 KubeConfig 文件，以及手动组装的基于令牌的 KubeConfig 文件。

适用场景

在以下场景中使用这些访问路径：

- 集群管理员需要一个可直接使用的 KubeConfig，用于集群管理、故障排查或检查。
- 开发人员或项目用户需要一个与其自身 ACP 身份和 RBAC 权限相匹配的 KubeConfig。
- 流水线或连接器需要一个与个人日常账号相独立的凭证。

选择访问路径

使用下表选择合适的访问路径：

访问路径	凭证来源	实际生效权限	生命周期控制	最适合的场景
平台下载的 KubeConfig	从平台下载的 KubeConfig 文件中嵌入的凭证	由下载文件中嵌入的凭证决定	由当前平台对该下载文件的行为控制	集群管理员和模板准备
基于令牌的 KubeConfig	为用户账号创建的 ACP API 令牌	仅限于令牌所有者的身份和 RBAC 范围	由令牌过期或吊销状态控制	开发人员、项目用户和自动化账号

> 当前行为
>
> 目前，从 ACP 下载的 KubeConfig 文件默认有效期为 10 年。ACP 当前未为此路径提供 CA 替换，也不支持为此路径使用用户自定义的集群 CA。

如果你需要更严格的身份隔离或更短的生命周期控制，请使用基于令牌的 KubeConfig，而不是长期重复使用下载的文件。

前提条件

开始之前，请确保满足以下条件：

- 你可以登录 ACP 并访问目标集群。
- 你已在本地机器上安装 ACP CLI（`ac`）或 `kubectl`，并将在该机器上使用 KubeConfig 文件。
- 如果你想使用基于令牌的 KubeConfig，可以从 **Profile** > **API Tokens** 创建一个 ACP API 令牌。有关详细信息，请参见 简介。
- 如果你想将 KubeConfig 用于流水线或连接器，请在创建其令牌之前准备一个专用账号，并只授予所需权限。

操作步骤

下载并使用平台提供的 KubeConfig

当你需要直接使用平台提供的 KubeConfig 文件访问集群时，请使用此路径。

1 下载 KubeConfig 文件

在 **Administrator** 视图中，进入 **Clusters** > **Clusters**，选择目标集群，打开集群详情页，单击 **Actions**，然后选择 **Download Kubeconfig**。

将下载的文件保存到本地路径，例如 `~/.kube/<cluster-name>.yaml`。

2 选择 KubeConfig 文件

将 ACP CLI（`ac`）或 `kubectl` 指向下载的文件：

```
export KUBECONFIG="$HOME/.kube/<cluster-name>.yaml"
kubectl config get-contexts
kubectl config current-context
ac config current-context
```

如果你管理多个集群，请在运行命令之前使用 `kubectl config use-context <context-name>` 或 `ac config use-context <context-name>` 切换到正确的上下文。如果你使用 ACP CLI 会话，也可以使用 `ac config use-cluster <cluster-name>` 在当前平台会话中切换集群。

3 验证集群访问

运行只读命令以确认 KubeConfig 文件按预期工作：

```
kubectl get nodes
kubectl get pods -A
ac get nodes
ac get pods -A
```

如果命令返回了预期的集群资源，则说明该 KubeConfig 文件可直接使用。

当你需要进行集群管理时，可以直接使用这个下载的文件。你也可以将其作为模板，构建一个同时适用于 `ac` 和 `kubectl` 的基于令牌的 KubeConfig。

使用自己的 ACP API 令牌构建基于令牌的 KubeConfig

当你希望 KubeConfig 权限与某个特定的 ACP 用户或自动化账号保持一致时，请使用此路径。

此路径不会提升权限。KubeConfig 文件仅包含连接和认证设置。实际访问边界仍然取决于令牌所有者及该身份的 RBAC 分配。

1 创建 ACP API 令牌

打开 **Profile** > **API Tokens**，创建一个新令牌，并设置与使用场景匹配的过期时间。

请安全保存该令牌。如果你计划将其用于自动化，请避免使用个人日常账号。

2 准备 KubeConfig 模板

使用以下任一方法：

- 下载平台提供的 KubeConfig 文件，并保存一份单独的副本用于基于令牌的访问。
- 手动创建一个标准 KubeConfig 文件。

如果你手动创建文件，请使用以下格式填写目标集群端点：

```
https://<platform-access-address>/kubernetes/<cluster-name>
```

如果你不想从头构建集群部分，请先下载平台提供的 KubeConfig，然后复用其中的集群和 CA 字段作为模板。

3 在 KubeConfig 文件中设置 ACP API 令牌

将 `users[].user.token` 的值替换为你为目标用户或自动化账号创建的 ACP API 令牌。

以下示例展示了一个基于令牌的 KubeConfig：

KubeConfig.yaml

```
apiVersion: v1
kind: Config
clusters:
  - name: <cluster-name>
    cluster:
      server: https://<platform-access-address>/kubernetes/<cluster-n
ame>
      certificate-authority-data: <base64-encoded-ca-data>
users:
  - name: <user-name>
    user:
      token: <platform-api-token>
contexts:
  - name: <context-name>
    context:
      cluster: <cluster-name>
      user: <user-name>
      namespace: <namespace>
current-context: <context-name>
```

`namespace` 是可选项。它会为命令设置默认命名空间，但不会扩大令牌的权限范围。

4 验证实际生效权限

使用新的 KubeConfig 文件并验证返回的权限是否与预期角色一致：

```
export KUBECONFIG="$HOME/.kube/<user-or-automation>.yaml"
kubectl auth can-i get pods -n <namespace>
kubectl get pods -n <namespace>
ac config current-context
ac get pods -n <namespace>
```

如果 `kubectl auth can-i` 返回 `no`，请授予所需角色或使用其他账号。不要将 KubeConfig 变更视为 RBAC 变更的替代方案。

为流水线或连接器访问使用专用账号

对于流水线或连接器访问，请使用专用账号，而不是个人用户账号：

1. 为流水线或连接器创建一个专用账号。
2. 仅向该账号授予所需的项目级或租户级权限。
3. 为该账号创建一个 ACP API 令牌。
4. 使用该令牌构建一个基于令牌的 KubeConfig。
5. 将令牌或 KubeConfig 文件存储在自动化平台的密钥管理系统中。
6. 通过创建新令牌、更新密钥并吊销旧令牌来轮换凭证。

这种模式可为自动化提供独立凭证，但在 ACP 中并不存在原生的项目级 KubeConfig 对象。

生命周期与吊销

生命周期行为取决于你使用的访问路径：

访问路径	控制有效性的因素	当前行为
平台下载的 KubeConfig	当前平台对该下载文件的行为	下载的 KubeConfig 目前默认有效期为 10 年。ACP 当前未为此路径提供 CA 替换或用户自定义的集群 CA。
基于令牌的 KubeConfig	`users[].user.token` 中的 ACP API 令牌	只有在令牌有效期间，KubeConfig 才可继续使用。当令牌过期或被吊销后，KubeConfig 也会停止工作。

如果你需要更短的有效期或更便捷的吊销方式，请优先使用基于令牌的路径。

验证

准备好 KubeConfig 文件后，请执行以下检查：

1. 运行 `kubectl config get-contexts`，确认所需上下文存在。
2. 运行只读命令，例如 `kubectl get pods -A` 或 `kubectl get pods -n <namespace>`，并确认响应与预期访问范围一致。
3. 对于基于令牌的 KubeConfig，运行 `kubectl auth can-i <verb> <resource> -n <namespace>`，并确认结果与令牌所有者的角色一致。

4. 如果你使用 ACP CLI，请运行 `ac config current-context` 或 `ac namespace`，并确认当前活动上下文与预期的集群和命名空间一致。
5. 如果你使用 ACP CLI，请运行 `ac get pods -n <namespace>` 或 `ac get nodes`，并确认响应与预期访问范围一致。

后续步骤

- 有关 ACP API 令牌管理，请参见 简介。
- 有关 ACP CLI 基础知识，请参见 CLI 入门。
- 有关 ACP CLI 上下文和会话管理，请参见 管理 CLI 配置文件。
- 有关 ACP CLI 与 kubectl 的兼容性，请参见 ac 和 kubectl 命令的使用。