Menu

安装

本文档将提供有关安装 ACP 的所有信息。

安装概述

安装概述

安装准备

前提条件

下载

节点预处理

安装

安装

Global Cluster 灾难恢复

Global Cluster 灾难恢复

安装概述

本文档面向具备 Linux 基础的系统管理员，详细说明 ACP `global` 集群的安装全流程。

在文档中，以下术语将被频繁使用，请注意区分：

- `global` 集群：指的是负责平台核心管理任务的集群，如集群管理、租户管理等功能。完成 `global` 集群的安装即完成 ACP 安装。安装完成后，便可按需接入或新建业务集群。
- 平台：通常指容器平台本身，是一个提供应用运行环境，集容器调度、管理、网络、存储等功能于一体的完整系统。平台是一个整体的概念，涵盖了 `global` 集群以及业务集群在内的所有组件。

> 提示
>
> 在安装前，请确保已完成容量规划、环境预处理及先决条件检查，确保各节点的硬件、网络和操作系统符合要求。 以下内容涵盖平台架构设计、安装方法和关键术语解释，帮助您在实际安装过程中把握核心要点。

目录

安装方法

`global` 集群的安装流程主要分为三个阶段：

1. 准备阶段

- 先决条件检查：确保所有节点硬件、网络及操作系统均符合要求，如内核版本、CPU 架构、网络配置等。
- 安装包下载与校验：登录 Customer Portal 获取最新安装包及签名文件，使用 GPG 工具验证包的完整性与正确性。
- 节点预处理：完成所有节点的预处理工作。

2. 执行阶段

- 安装包上传与解压：将安装包上传至目标控制节点（建议目录：`/root/cpaas-install`），并解压安装资源。
- 启动安装器：在控制节点上执行安装脚本（如 `bash setup.sh`），根据实际环境选择网络插件（Kube-OVN 或 Calico）、IP 协议模式（IPv4/IPv6/双栈）及 VIP 配置。
- 参数配置：访问安装器提供的 Web UI ，依次设置 Kubernetes 版本、集群网络、节点名称、访问地址及其他关键参数，完成 `global` 集群的安装。

3. 验证阶段

- 系统状态检查：安装完成后，登录平台 Web UI 检查集群状态及各组件运行情况。
- **CLI** 验证：通过命令行工具检查集群资源状态，确保所有服务正常运行，无异常或失败项。

术语解释

- **VIP**（**Virtual IP**） 虚拟 IP 地址，用于负载均衡 `global` 集群的 API Server、Web UI 等访问入口，确保高可用性。

后续章节将针对各安装阶段的详细操作、配置参数及验证方法做进一步说明，请在正式安装前仔细阅读并完成相应前置准备工作。

安装准备

前提条件

在安装 `global` 集群之前，您需要准备符合要求的硬件、网络和操作系统环境。

INFO

1. 平台当前不支持在已有 Kubernetes 环境中直接安装 `global` 集群。如果您的环境中已有 Kubernetes 集群，请先备份数据并清理环境后再进行安装。
2. 如果计划使用 global Cluster 灾备功能，请先阅读 global Cluster 灾备。

目录

容量规划

安装前，您必须根据目标和实际需求选择合适的安装场景。不同场景在基础设施资源配置和架构设计要求上存在显著差异。以下是三种典型场景的规划建议：

单节点

适用范围

适用于平台功能验证、演示或技术可行性测试。该场景仅用于验证平台核心功能，不承载生产级应用流量。资源配置处于最低水平。

资源配置要求

维度	规格要求
节点数量	1（物理机或虚拟机）
CPU	≥16 核
内存	≥32GB

架构说明

- 一体化部署：集群仅有一个节点，所有控制平面组件和应用均运行于该节点。
- 轻量负载：仅能承载不超过 10 个 Pod 的 Demo 应用。
- 非生产用途：不支持水平扩展，无法满足应用连续性和高可用性需求。

单集群

适用范围

面向 ISV（独立软件供应商）的标准化交付需求，`global` 集群同时负责平台管理和 ISV 应用的直接运行，无需创建额外的业务集群。

资源配置要求

部署方案	节点类型	数量	最低规格	推荐规格	备注
最低配置	控制平面节点	3	8 核 16GB + ISV 需求	12 核 24GB + ISV 需求	建议预留约 30% 冗余资源
	工作节点	0	—	—	ISV 应用与控制平面节点共享运行

推荐配置	控制平面节点	3	8 核 16GB	12 核 24GB	运行 Kubernetes 组件和平台控制平面
	工作节点	≥2	根据 ISV 应用负载需求	—	独立运行 ISV 应用，建议实现高可用部署

架构说明

- 生产就绪：`global` 集群同时运行控制平面组件和 ISV 应用。
- 平台与应用隔离：建议 ISV 应用运行在专用工作节点，避免与控制平面资源争用。
- 扩展策略：每新增 50 个应用 Pod，按 ISV 应用资源需求增加 1 个工作节点。

多集群

适用范围

适用于需要跨云跨地域统一管理多个业务集群的大型企业数据中心环境。随着业务集群数量增加，`global` 集群需动态扩展工作节点和资源配置，确保高可用和高性能。

核心特性

- 混合控制：支持跨云跨地域业务集群的统一管理。
- 弹性扩展：`global` 集群资源随接入业务集群数量线性扩展。
- 物理隔离：建议控制平面与计算平面物理隔离，保障系统稳定性。

基础资源配置

节点类型	数量	最低规格	推荐规格	备注
控制平面节点	3 或 5	8 核 16GB	16 核 32GB	启用反亲和策略
工作节点	≥2	12 核 24GB	24 核 48GB	建议预留 30% 资源冗余

动态扩展规则

2.1. 纵向扩展：单节点可按梯度升级（如 12 核 24GB → 24 核 48GB → 48 核 96GB）。

2.2. 横向扩展：每接入 10 个业务集群，建议计算资源扩容 20%（增加节点数或升级节点规格）。

2.3. 监控触发：CPU/内存利用率持续超过 70% 时，启动扩容措施。

架构说明

- 控制平面由 3 个节点组成 ETCD 集群，建议启用 TLS 加密和定期快照。
- 平台关键组件建议部署在独立工作节点，避免资源争用。
- 灾备建议：跨多个可用区部署，确保工作节点分布在至少 2 个物理故障域。

提示

1. 资源冗余：生产环境建议预留至少 30% 资源余量，以应对突发负载。
2. 网络规划：`global` 集群应部署在独立 VPC 或 VLAN 中，确保带宽 ≥1Gbps。
3. 存储隔离：ETCD 存储建议使用 NVMe SSD，并与应用存储物理隔离。

机器

INFO

本节描述构建高可用 `global` 集群的最低硬件要求。如果您已完成容量规划，请根据 容量规划 准备相应资源，或安装后根据需要进行扩容。

基本要求

集群至少需提供 **3** 台物理机或虚拟机作为控制平面节点。每个节点的最低配置如下：

类别	最低要求
CPU	≥ 8 核，主频 ≥ 2.5GHz 禁止超额配置；关闭节能模式
内存	≥ 16GB 禁止超额配置；建议使用至少六通道 DDR4

类别	最低要求
硬盘	单设备 IOPS ≥ 2000 吞吐量 ≥ 200MB/s 必须使用 SSD

ARM 架构要求

对于 ARM 架构（如 Kunpeng 920），建议配置提升至 x86 最低配置的 **2** 倍，但不少于 **1.5** 倍。

例如：x86 需 8 核 16GB，则 ARM 至少应达到 12 核 24GB，推荐配置为 16 核 32GB。

支持的操作系统及内核

INFO

1. 内核版本要求：
   以下内核版本均已由平台测试正式发布和验证。实际部署时，需严格遵循 **A.B.C** 主版本号，后续小版本可有差异。
2. 不支持环境：
   若操作系统、内核版本或 CPU 架构不符合要求，请联系技术支持。

Red Hat Enterprise Linux (RHEL)

- RHEL 7.8: `3.10.0-1127.el7.x86_64`
- RHEL 8.0 至 8.6: `4.18.0-80.el8.x86_64` 至 `4.18.0-372.9.1.el8.x86_64`

警告

RHEL 7.8 不支持 **Calico Vxlan IPv6**。

CentOS

- CentOS 7.6 至 7.9: `3.10.0-1127` 至 `3.11`

> 警告
>
> CentOS 不支持 **Calico Vxlan IPv6**。

Ubuntu

- Ubuntu 20.04 LTS: `5.4.0-124-generic`
- Ubuntu 22.04 LTS: `5.15.0-56-generic`

> 警告
>
> 不支持 Ubuntu HWE（硬件启用）版本。

麒麟高级服务器版

- 麒麟 V10 SP3: `4.19.90-52.22.v2207.ky10.x86_64`

> 警告
>
> - 麒麟 V10、V10-SP1 和 V10-SP2 存在已知内核问题，可能导致 **NodePort** 网络访问失败，建议升级至 麒麟 **V10-SP3**。
> - ARM 架构仅支持 `Kunpeng 920`，其他型号请联系技术支持。

网络

安装前，必须预先配置以下网络资源。如果无法提供硬件 LoadBalancer，安装程序支持配置 **haproxy + keepalived** 作为软件负载均衡，但需注意：

- 性能较差：软件负载均衡性能低于硬件 LoadBalancer。
- 复杂度较高：若不熟悉 keepalived，可能导致 `global` 集群不可用，排查问题耗时且严重影响平台可靠性。

网络资源

资源	是否必需	数量	说明
`global` VIP	必需	1	用于集群内节点访问 kube-apiserver，配置于负载均衡设备以确保高可用。 该 IP 也可作为平台 Web UI 的访问地址。 与 `global` 集群处于同一网络的业务集群也可通过此 IP 访问 `global` 集群。
外部 IP	可选	按需	当存在与 `global` 集群不在同一网络的业务集群（如混合云场景）时，必须提供。其他网络的业务集群通过此 IP 访问 `global` 集群。 该 IP 需配置于负载均衡设备以确保高可用。 该 IP 也可作为平台 Web UI 的访问地址。
域名	可选	按需	若需通过域名访问 `global` 集群或平台 Web UI，请提前提供并确保域名解析正确。
证书	可选	按需	建议使用受信任证书以避免浏览器安全警告；若未提供，安装程序将生成自签名证书，但使用 HTTPS 时可能存在安全风险。

信息

以下情况必须提供域名：

1. `global` 集群需要支持 IPv6 访问。
2. 计划为 `global` 集群实施灾备方案。

注意

若平台需配置多个访问地址（如内外网地址），请根据上述表格提前准备相应 IP 或域名。安装时可在参数中配置，或安装后根据产品文档添加。

网络配置

类型	需求说明
网络速率	`global` 集群与业务集群同网速率 ≥1Gbps（推荐 10Gbps）；跨网速率 ≥100Mbps（推荐 1Gbps）。 速率不足将显著降低数据查询性能。
网络延迟	同网延迟 ≤2ms；跨网延迟 ≤100ms（推荐 ≤30ms）。
网络策略	请参考 LoadBalancer 转发规则 确保必要端口开放；使用 Calico CNI 时，确保启用 **IP-in-IP** 协议。
IP 地址段	`global` 集群节点应避免使用 172.16-32 网段。如已使用，请调整 Docker 配置（添加 bip 参数）以避免冲突。

LoadBalancer 转发规则

该规则确保 `global` 集群能正常接收来自 LoadBalancer 的流量。请根据下表检查网络策略，确保相关端口开放。

源 **IP**	协议	目标 **IP**	目标端口	说明
`global` VIP，外部 IP	TCP	所有控制平面节点 IP	443	通过 HTTPS 协议为平台 Web UI、镜像仓库和 Kubernetes API Server 提供访问服务。默认端口为 `443`。如需使用自定义 HTTPS 端口，请执行以下操作： • 将端口转发规则中的目标端口替换为自定义端口号。

源 **IP**	协议	目标 **IP**	目标端口	说明
				• 后续在 Web UI 安装参数中填写自定义端口号。
`global` VIP，外部 IP	TCP	所有控制平面节点 IP	6443	该端口为集群内节点访问 Kubernetes API Server 提供服务。
`global` VIP，外部 IP	TCP	所有控制平面节点 IP	11443	该端口为集群内节点访问镜像仓库提供服务。 注意：若计划使用外部镜像仓库替代 `global` 集群默认镜像仓库，无需配置此端口。

提示

- 建议在 LoadBalancer 上配置健康检查以监控端口状态。
- 若计划为 `global` 集群实施灾备方案，需为所有控制平面节点开放端口 `2379`，用于主备集群间 ETCD 数据同步。
- 平台默认仅支持 HTTPS，如需支持 HTTP，需为所有控制平面节点开放 HTTP 端口。

下载安装包

在安装之前，您需要下载并解压安装包。如果需要 Extensions Package，请先解压 Core Package，然后将 Extensions Package 解压到同一目录中，之后再开始安装。

下载安装包

登录 **Customer Portal** 下载安装包。如果您尚未注册账号，请联系技术支持。

提供两种类型的安装包：

- **Core Package**：所有安装必需
- **Extensions Package**：可选，仅在需要额外功能时下载

Core Package 文档

下载核心包 installer-core-v4.x.y-zzzz.tar。这里的“zzzz”可能是 x86、arm 或 hybrid，取决于用户选择的架构（以下同理）。 使用 tar 命令解压核心包：

```
tar xvf installer-core-v4.x.y-zzzz.tar
```

Extensions Package 文档

下载扩展包 installer-extensions-v4.x.y-zzzz.tar。这里的“zzzz”必须与核心包匹配。扩展包包含核心包中未包含的插件。 如果下载了 Extensions Package，需要在安装前将扩展包解压到与核心包相同的目录：

```
tar xvf installer-extensions-v4.x.y-zzzz.tar -C <the core package extraction
directory>
```

示例： 如果核心包解压到了 /root，解压后 /root 下会有一个 installer 目录。 您应在 installer-extensions-v4.x.y-zzzz.tar 所在目录执行以下命令：`tar xvf installer-extensions-v4.x.y-zzzz.tar -C /root/`

安装指南将详细介绍 Core Package 的安装流程。Extensions Package 文档将提供使用和配置扩展的具体说明。

节点预处理

在安装 `global` 集群之前，所有节点（控制平面节点和工作节点）必须完成预处理。

目录

执行快速配置脚本

ACP 安装包提供了一个用于快速配置节点的脚本。

解压安装包后，在 `res` 目录下可以获得 `init.sh` 脚本文件。将该脚本文件复制到节点上，并确保拥有 `root` 权限。

执行脚本：

```
bash init.sh
```

WARNING

`init.sh` 无法保证以下所有检查均被正确处理，仍需继续执行下面的步骤。

节点检查

以下列出了节点必须完成的所有检查。根据节点的角色，所需检查会有所不同。例如，有些检查仅适用于控制平面节点。

检查分为两类：

- ✅ 表示必须通过的检查。
- ⚠️ 表示在特定场景下必须满足的检查，请根据说明判断是否满足对应条件，若满足则必须解决。

以下是检查列表：

- 操作系统与内核
  - ✅ 机器的 grub 启动配置必须包含参数 `transparent_hugepage=never`。
  - ✅ CentOS 7.x 系统机器的 grub 启动配置必须包含参数 `cgroup.memory=nokmem`。
  - ✅ 检查内核模块 `ip_vs`、`ip_vs_rr`、`ip_vs_wrr` 和 `ip_vs_sh` 是否已启用。
  - ⚠️ 当内核版本低于 4.19.0（或 RHEL 低于 4.18.0）时，检查内核模块 `nf_conntrack_ipv4` 和（IPv6 时）`nf_conntrack_ipv6` 是否已启用。
  - ⚠️ 若 `global` 集群计划使用 `Kube-OVN` CNI，内核模块 `geneve` 和 `openvswitch` 必须启用。
  - ✅ 禁用 apparmor/selinux 和防火墙。
  - ✅ 禁用 `swap`。
- 用户与权限
  - ✅ 节点的 SSH 用户具有 `root` 权限，且可无密码使用 `sudo`。
  - ✅ `/etc/ssh/sshd_config` 中的 `UseDNS` 和 `UsePAM` 参数必须设置为 `no`。
  - ✅ 执行 `systemctl show --property=DefaultTasksMax` 返回 `infinity` 或非常大的值，否则需调整 `/etc/systemd/system.conf`。
- 节点网络
  - ✅ `hostname` 必须符合以下规则：

    - 不超过 36 个字符。
    - 以字母或数字开头和结尾。
    - 仅包含小写字母、数字、`-` 和 `.`，但不能包含 `.-`、`..` 或 `-.`。
- ✅ `/etc/hosts` 中的 `localhost` 必须解析为 `127.0.0.1`。
- ✅ `/etc/resolv.conf` 文件必须存在且包含 `nameserver` 配置，但不能包含以 172 开头的地址（需禁用 systemd-resolved）。
- ⚠️ `/etc/resolv.conf` 文件不应配置 search 域（如必须配置，请参见 配置 Search Domain）。
- ✅ 机器的 IP 地址不能是回环、多播、链路本地、全 0 或广播地址。
- ✅ 执行 `ip route` 必须返回默认路由或指向 `0.0.0.0` 的路由。
- ✅ 节点不得占用以下端口：
    - 控制平面节点：`2379`、`2380`、`6443`、`10249` ~ `10256`
    - 安装程序所在节点：`8080`、`12080`、`12443`、`16443`、`2379`、`2380`、`6443`、`10249` ~ `10256`
    - 工作节点：`10249` ~ `10256`
- ✅ 若 `global` 集群使用 **Kube-OVN** 或 **Calico**，确保以下端口未被占用：
    - **Kube-OVN**：`6641`、`6642`
    - **Calico**：`179`
- ⚠️ 确保 Docker 所需的网络段 `172.16.x.x` ~ `172.32.x.x` 的 IP 地址未被占用。如该网络段 IP 被占用且无法更改，请联系技术支持。

- 软件与目录要求
    - ✅ 必须安装以下工具：`ip`、`ss`、`tar`、`swapoff`、`modprobe`、`sysctl`、`md5sum` 以及 `scp` 或 `sftp`。
    - ⚠️ 若计划使用本地存储 **TopoLVM** 或 **Rook**，需安装 `lvm2`。
    - ✅ 不允许存在 `/etc/systemd/system/kubelet.service` 文件。
    - ✅ `/tmp` 挂载参数中不得包含 `noexec`。
    - ✅ 删除与 `global` 集群组件冲突的软件包（详见 删除冲突软件包）。
    - ✅ 以下文件若存在，必须删除：

  - `/var/lib/docker`
  - `/var/lib/containerd`
  - `/var/log/pods`
  - `/var/lib/kubelet/pki`
- 跨节点检查
  - ✅ `global` 集群节点间不得存在网络防火墙限制。
  - ✅ 集群中每个节点的 `hostname` 必须唯一。
  - ✅ 所有节点时区必须统一，且时间同步误差 ≤ 10 秒。

附录

删除冲突软件包

安装前，节点上可能已在 docker/containerd 环境中运行应用，或安装了与 `global` 集群冲突的软件，因此需要检查并卸载冲突的软件包。

DANGER

- 为避免应用中断或数据丢失，请务必确认是否存在冲突软件包。发现冲突时，请制定应用切换方案并备份数据后再卸载。
- 卸载冲突软件包后，还需检查 `/usr/local/bin/` 等目录中是否存在其他潜在冲突的二进制文件（如与 docker、containerd、runc、podman、容器网络、容器运行时或 Kubernetes 相关的软件）。

以下命令供参考。

CentOS / RedHat

检查：

```
for x in \
    docker docker-client docker-common docker-latest \
    podman-docker podman \
    runc \
    containernetworking-plugins \
    apptainer \
    kubernetes kubernetes-master kubernetes-node kubernetes-client \
    ; do
    rpm -qa | grep -F "$x"
done
```

卸载：

```
for x in \
    docker docker-client docker-common docker-latest \
    podman-docker podman \
    runc \
    containernetworking-plugins \
    apptainer \
    kubernetes kubernetes-master kubernetes-node kubernetes-client \
    ; do
    yum remove "$x"
done
```

Ubuntu

检查：

```
for x in \
    docker.io \
    podman-docker \
    containerd \
    rootlesskit \
    rkt \
    containernetworking-plugins \
    kubernetes \
    ; do
    dpkg-query -l | grep -F "$x"
done

for x in \
    kubernetes-worker \
    kubectl kube-proxy kube-scheduler kube-controller-manager kube-apiserver \
    k8s microk8s \
    kubeadm kubelet \
    ; do
    snap list | grep -F "$x"
done
```

卸载：

```
for x in \
    docker.io \
    podman-docker \
    containerd \
    rootlesskit \
    rkt \
    containernetworking-plugins \
    kubernetes \
    ; do
    apt-get purge "$x"
done

for x in \
    kubernetes-worker \
    kubectl kube-proxy kube-scheduler kube-controller-manager kube-apiserver \
    k8s microk8s \
    kubeadm kubelet \
    ; do
    snap remove --purge "$x"
done
```

Kylin

检查：

```
for x in \
    docker docker-client docker-common \
    docker-engine docker-proxy docker-runc \
    podman-docker podman \
    containernetworking-plugins \
    apptainer \
    containerd \
    kubernetes kubernetes-master kubernetes-node kubernetes-client kubernetes-kubeadm \
    ; do
    rpm -qa | grep -F "$x"
done
```

卸载：

```
for x in \
    docker docker-client docker-common \
    docker-engine docker-proxy docker-runc \
    podman-docker podman \
    containernetworking-plugins \
    apptainer \
    containerd \
    kubernetes kubernetes-master kubernetes-node kubernetes-client kubernetes-kubeadm \
    ; do
    yum remove "$x"
done
```

配置 Search Domain

在 Linux 操作系统中，`/etc/resolv.conf` 文件用于配置 DNS 客户端的域名解析设置。`search` 行指定 DNS 查询的域搜索路径。

配置要求

- 域数量： `search` 行中的域数量应小于 `domainCountLimit - 3`（默认 `domainCountLimit` 为 32）。
- 单个域名长度： 每个域名不得超过 253 个字符。
- 总字符长度： 所有域名及空格的总字符数不得超过 `MaxDNSSearchListChar`（默认 2048）。

示例

```
search domain1.com domain2.com domain3.com
```

- 域数量为 3。
- 单个域名长度，如 `domain1.com`，为 11。
- 总字符长度为 35，即 11 + 11 + 11 + 2（两个空格）。

WARNING

- 若 `/etc/resolv.conf` 文件中的 `search` 行不满足上述限制，可能导致 DNS 查询失败或性能下降。

- 修改 `/etc/resolv.conf` 文件前，建议先备份该文件。

安装

本节介绍 `global` 集群的具体安装步骤。

在开始安装前，请确保已完成先决条件检查、安装包下载与校验、节点预处理等准备工作。

目录

流程

1 上传并解压安装包

将 Core Package 安装包上传至 `global` 集群控制平面节点的任意一台机器，并按以下命令解压：

```
# 假设机器上已存在 /root/cpaas-install 文件夹
tar -xvf {Path to Core Package File}/{Core Package File Name} -C /root/cpaas-install
cd /root/cpaas-install/installer || exit 1
```

INFO

- 该机器将在 `global` 集群安装完成后成为第一个控制平面节点。
- Core Package 解压后至少需要 **100GB** 磁盘空间，请确保存储资源充足。
- 如果已下载 Extensions Package，请先解压并参照包内文档后再进行后续操作。

2 启动安装程序

执行以下安装脚本启动安装程序。安装程序启动成功后，命令行终端会输出 Web 控制台访问地址。

等待约 5 分钟后，可使用 PC 浏览器访问安装程序提供的 Web 控制台。

```
bash setup.sh
```

WARNING

请确保安装程序所在节点的 IP 地址及端口 8080 可正常访问，以保证安装程序启动成功后 Web 控制台能顺利访问。

网络模式与 IP 协议族

```
bash setup.sh --network-mode calico
```

`--network-mode` 参数影响安装程序创建的 `global` 集群的 CNI。若不指定该参数，`global` 集群的 CNI 默认为 Kube-OVN。若需使用 Calico 作为 CNI，必须显式指定 `--network-mode calico`。

```
bash setup.sh --ip-family ipv6
```

若计划创建 Single-stack Network IPv6 的 `global` 集群，启动安装程序时必须显式指定 `--ip-family ipv6`。否则，安装程序创建的 `global` 集群默认支持 Single-stack Network IPv4 和 Dual-stack Network。

3 参数配置

按页面引导完成安装参数配置后，确认安装。

参数说明 提供关键参数的详细描述，请仔细阅读并根据实际需求配置。

4 验证安装成功

安装完成后，页面会显示平台访问地址。点击 访问 按钮跳转至平台 Web UI。

在 平台管理 视图中，依次点击 集群管理 **>** 集群，找到名为 `global` 的集群。

从右侧下拉菜单中选择 `CLI Tools`，执行以下命令验证安装状态：

```
# 检查是否存在失败的 Charts
kubectl get apprelease --all-namespaces
# 检查是否存在失败的 Pods
kubectl get pod --all-namespaces | awk '{if ($4 != "Running" && $4 !=
"Completed")print}' | awk -F'[/ ]+' '{if ($3 != $4)print}'
```

5 安装产品文档插件

INFO

Alauda Container Platform Product Docs 插件提供平台内访问产品文档的功能。平台内所有帮助链接均指向该文档。若未安装此插件，点击平台中的帮助链接将导致 404 访问错误。

1. 进入 管理员。
2. 在左侧边栏点击 **Marketplace** > 集群插件，选择 `global` 集群。
3. 找到 **Alauda Container Platform Product Docs** 插件，点击 安装。

参数说明

参数	说明
Kubernetes 版本	所有可选版本均经过严格测试，确保稳定性和兼容性。 推荐：选择最新版本以获得最佳功能和支持。
集群网络协议	支持三种模式：IPv4 单栈、IPv6 单栈、IPv4/IPv6 双栈。 注意：若选择双栈模式，需确保所有节点均正确配置 IPv6 地址；设置后网络协议不可更改。
集群地址	输入预先准备好的域名。若无域名，则输入预先准备好的 `global VIP`。 `自建 VIP` 默认禁用，仅当未提供 LoadBalancer 时启用。启用后，安装程序会自动部署 `keepalived`，提供软件负载均衡支持。 注意：使用 `自建 VIP` 时必须满足以下条件， • 有可用的 VRID； • 主机网络支持 VRRP 协议； • 所有控制平面节点及 VIP 必须处于同一子网。 提示：在功能体验场景的单节点部署中，可直接输入节点 IP，无需启用 `自建 VIP` 或准备 `global VIP` 等网络资源。
平台访问地址	若无需区分 集群地址 和 平台访问地址，请输入与 集群地址 相同的地址。 若需区分，例如 `global` 集群仅供内网访问，平台需提供外网访问，则输入预先准备的域名或 `外部 IP`。 平台默认使用 HTTPS 访问，不启用 HTTP。如需启用 HTTP 访问，可在 高级设置 中开启（不推荐）。 注意：以下情况必须输入域名，

	• 计划为 `global` 集群制定灾备方案； • 平台需支持 IPv6 访问。 提示：如需配置更多平台访问地址，可在下一步的 其他设置 > 其他平台访问地址 中添加，或安装后根据用户手册在平台管理中添加。
证书	平台默认提供自签名证书以支持 HTTPS 访问。 如需使用自定义证书，可上传现有证书。
镜像仓库	默认使用 `平台部署` 镜像仓库，包含所有组件镜像。 如需使用 `外部` 镜像仓库，请先联系技术支持获取镜像同步方案后再配置。
容器网络	集群的默认子网和 Service 网络段不可重叠。 使用 Kube-OVN Overlay 网络时，确保容器网络与主机网络不在同一网段，否则可能导致网络异常。
节点名称	若选择 `主机名作为节点名`，请确保所有节点的主机名唯一。
`global` 集群平台节点隔离	仅在计划在 `global` 集群运行应用工作负载时启用。 启用后： • 节点可设置为 `平台专属`，即仅运行平台组件，确保平台与应用工作负载隔离； • 排除 DaemonSet 类型的工作负载。
添加节点	控制平面节点： • 支持添加 1 个或 3 个控制平面节点（3 个用于高可用配置）； • 若启用 `平台专属`，强制禁用 `可部署应用`，控制平面节点仅运行平台组件；

	• 若禁用 `平台专属`，可选择是否启用 `可部署应用`，允许控制平面节点运行应用工作负载。 工作节点： • 若启用 `平台专属`，强制禁用 `可部署应用`； • 若禁用 `平台专属`，强制启用 `可部署应用`。 使用 Kube-OVN 时，可通过输入网关名称指定节点网卡。 若节点可用性检查失败，请根据页面提示调整后重新添加。

安装程序清理

通常安装完成后安装程序会自动删除。若安装完成 30 分钟后安装程序未自动删除，请在安装程序所在节点执行以下命令强制删除安装程序容器：

```
docker rm -f minialauda-control-plane
```

Global Cluster 灾难恢复

目录

Overview

该方案针对 `global` 集群的灾难恢复场景设计。 `global` 集群作为平台的控制平面，负责管理其他集群。为确保在 `global` 集群故障时平台服务的持续可用性，本方案部署两个 `global` 集群：主集群和备用集群。

灾难恢复机制基于主集群到备用集群的 etcd 数据实时同步。当主集群因故障不可用时，服务可快速切换至备用集群。

支持的灾难场景

- 主集群发生不可恢复的系统级故障，导致无法正常运行；
- 托管主集群的物理机或虚拟机故障，导致无法访问；
- 主集群所在位置的网络故障，导致服务中断；

不支持的灾难场景

- `global` 集群内部部署的应用故障；
- 存储系统故障导致的数据丢失（etcd 同步范围之外）；

主集群和备用集群的角色是相对的：当前为平台提供服务的集群为主集群（DNS 指向该集群），备用集群为备用集群。故障切换后，角色互换。

注意事项

- 本方案仅同步 `global` 集群的 etcd 数据；不包含 registry、chartmuseum 或其他组件的数据；
- 为便于排查和管理，建议节点命名风格如 `standby-global-m1`，以标明节点所属集群（主集群或备用集群）。
- 不支持集群内应用数据的灾难恢复；
- 两个集群间需保持稳定的网络连接，确保 etcd 同步可靠；
- 若集群基于异构架构（如 x86 和 ARM），需使用双架构安装包；
- 以下命名空间不参与 etcd 同步，若在这些命名空间创建资源，需用户自行备份：

```
cpaas-system
cert-manager
default
global-credentials
cpaas-system-global-credentials
kube-ovn
kube-public
kube-system
nsx-system
cpaas-solution
kube-node-lease
kubevirt
nativestor-system
operators
```

- 若两个集群均使用内置镜像仓库，容器镜像需分别上传；
- 若主集群部署了 灵雀云 **DevOps Eventing v3**（knative-operator）及其实例，备用集群必须预先部署相同组件。

流程概述

1. 准备统一的域名作为平台访问地址；
2. 将域名指向主集群的 VIP 并安装主集群；
3. 临时将 DNS 解析切换至备用 VIP，安装备用集群；
4. 将主集群的 ETCD 加密密钥复制到备用集群后续作为控制平面节点的节点上；
5. 安装并启用 etcd 同步插件；
6. 验证同步状态并定期检查；
7. 发生故障时，将 DNS 切换至备用集群完成灾难恢复。

所需资源

- 统一域名作为 `Platform Access Address`，以及该域名的 TLS 证书和私钥，用于 HTTPS 服务；
- 每个集群专用的虚拟 IP 地址——一个用于主集群，另一个用于备用集群；
  - 预先配置负载均衡器，将端口 `80`、`443`、`6443`、`2379` 和 `11443` 的 TCP 流量路由到对应 VIP 后端的控制平面节点。

流程

第 1 步：安装主集群

DR（灾难恢复环境）安装注意事项

安装 DR 环境的主集群时，

- 首先，记录安装 Web UI 指南中设置的所有参数。安装备用集群时需保持部分选项一致。
- 必须预配置用户自建的负载均衡器，将流量路由至虚拟 IP。不可使用“自建 VIP”选项。
- `Platform Access Address` 字段必须为域名，`Cluster Endpoint` 必须为虚拟 IP 地址。
- 两个集群必须配置使用“已有证书”（且为同一证书），必要时申请合法证书。不可使用“自签名证书”选项。
- 当 `Image Repository` 设置为 `Platform Deployment` 时，`Username` 和 `Password` 字段均不能为空；`IP/Domain` 字段必须设置为作为 `Platform Access Address` 使用的域名。
- `Platform Access Address` 的 `HTTP Port` 和 `HTTPS Port` 必须分别为 80 和 443。
- 安装指南第二页（步骤：`Advanced`）中，`Other Platform Access Addresses` 字段必须包含当前集群的虚拟 IP。

参考以下文档完成安装：

- 安装准备
- 安装指南

第 2 步：安装备用集群

1. 临时将域名指向备用集群的 VIP；

2. 登录主集群的第一个控制平面节点，将 etcd 加密配置复制到所有备用集群控制平面节点：

```
# 假设主集群控制平面节点为 1.1.1.1、2.2.2.2 和 3.3.3.3
# 备用集群控制平面节点为 4.4.4.4、5.5.5.5 和 6.6.6.6
for i in 4.4.4.4 5.5.5.5 6.6.6.6  # 替换为备用集群控制平面节点 IP
do
  ssh "<user>@$i" "sudo mkdir -p /etc/kubernetes/"
  scp /etc/kubernetes/encryption-provider.conf "<user>@$i:/tmp/encryption-
provider.conf"
  ssh "<user>@$i" "sudo install -o root -g root -m 600 /tmp/encryption-provider.conf
/etc/kubernetes/encryption-provider.conf && rm -f /tmp/encryption-provider.conf"
done
```

3. 按照主集群相同方式安装备用集群

> **安装备用集群注意事项**
>
> 安装 DR 环境的备用集群时， 以下选项必须与主集群保持一致：
>
> - `Platform Access Address` 字段；
> - `Certificate` 的所有字段；
> - `Image Repository` 的所有字段；
> - 重要：确保镜像仓库凭据和 ACP 管理员用户与主集群设置一致。
>
> 并且务必遵循第 1 步中 `DR（灾难恢复环境）安装注意事项` 。

参考以下文档完成安装：

- 安装准备
- 安装指南

第 3 步：启用 etcd 同步

1. 配置负载均衡器，将端口 `2379` 转发至对应集群的控制平面节点。仅支持 TCP 模式，不支持 L7 转发。
2. 使用备用 global 集群的 VIP 访问 Web 控制台，切换至 **Administrator** 视图；
3. 进入 **Marketplace > Cluster Plugins**，选择 `global` 集群；
4. 找到 灵雀云容器平台 **etcd Synchronizer**，点击 **Install**，配置参数：
   - 使用默认同步间隔；
   - 除非排查问题，否则关闭日志开关。

验证同步 Pod 在备用集群运行：

```
kubectl get po -n cpaas-system -l app=etcd-sync
kubectl logs -n cpaas-system $(kubectl get po -n cpaas-system -l app=etcd-sync --no-
headers | head -1) | grep -i "Start Sync update"
```

出现 “Start Sync update” 后，重建一个 Pod 以重新触发带有 ownerReference 依赖的资源同步：

```
kubectl delete po -n cpaas-system $(kubectl get po -n cpaas-system -l app=etcd-sync --no-
headers | head -1)
```

检查同步状态：

```
mirror_svc=$(kubectl get svc -n cpaas-system etcd-sync-monitor -o
jsonpath='{.spec.clusterIP}')
ipv6_regex="^[0-9a-fA-F:]+$"
if [[ $mirror_svc =~ $ipv6_regex ]]; then
  export mirror_new_svc="[$mirror_svc]"
else
  export mirror_new_svc=$mirror_svc
fi
curl $mirror_new_svc/check
```

输出说明：

- `LOCAL ETCD missed keys`：主集群存在但备用集群缺失的键，通常因同步时资源顺序导致的 GC。重启一个 etcd-sync Pod 可修复；
- `LOCAL ETCD surplus keys`：备用集群独有的多余键，删除前请与运维确认。

若安装了以下组件，重启其服务：

- 灵雀云容器平台 Elasticsearch 日志存储：

```
kubectl delete po -n cpaas-system -l service_name=cpaas-elasticsearch
```

- 灵雀云容器平台 VictoriaMetrics 监控：

```
kubectl delete po -n cpaas-system -l 'service_name in
(alertmanager,vmselect,vminsert)'
```

灾难恢复流程

1. 如有必要，重启备用集群的 Elasticsearch：

```
# 将 installer/res/packaged-scripts/for-upgrade/ensure-asm-template.sh 复制到 /root：
# 不可跳过此步骤

# 如有必要，切换至 root 用户
sudo -i

# 检查 global 集群是否安装 Elasticsearch 日志存储
_es_pods=$(kubectl get po -n cpaas-system | grep cpaas-elasticsearch | awk '{print $1}')
if [[ -n "${_es_pods}" ]]; then
    # 若脚本返回 401 错误，重启 Elasticsearch
    # 然后执行脚本再次检查集群
    bash /root/ensure-asm-template.sh

    # 重启 Elasticsearch
    xargs -r -t -- kubectl delete po -n cpaas-system <<< "${_es_pods}"
fi
```

2. 验证备用集群数据一致性（同第 3 步检查）；

3. 卸载 etcd 同步插件；

4. 取消两个 VIP 上的 `2379` 端口转发；

5. 将平台域名 DNS 切换至备用 VIP，备用集群成为新的主集群；

6. 验证 DNS 解析：

```
kubectl exec -it -n cpaas-system deployments/sentry -- nslookup <platform access domain>
# 若解析不正确，重启 coredns Pod 并重试，直至成功
```

7. 清理浏览器缓存，访问平台页面，确认显示为原备用集群；

8. 重启以下服务（如已安装）：

   - 灵雀云容器平台 Elasticsearch 日志存储：

     ```
     kubectl delete po -n cpaas-system -l service_name=cpaas-elasticsearch
     ```

- 灵雀云容器平台 VictoriaMetrics 监控：

```
kubectl delete po -n cpaas-system -l 'service_name in
(alertmanager,vmselect,vminsert)'
```

- cluster-transformer：

```
kubectl delete po -n cpaas-system -l service_name=cluster-transformer
```

9. 若业务集群向主集群发送监控数据，重启业务集群中的 warlock：

```
kubectl delete po -n cpaas-system -l service_name=warlock
```

10. 在原主集群上重复启用 etcd 同步步骤，将其转为新的备用集群。

日常检查

定期检查备用集群的同步状态：

```
curl $(kubectl get svc -n cpaas-system etcd-sync-monitor -o
jsonpath='{.spec.clusterIP}')/check
```

若发现缺失或多余键，按输出提示处理。

上传软件包

使用 `violet` 向备用集群上传软件包时，必须指定 `--dest-repo` 参数为备用集群的 VIP。 若省略该参数，软件包将上传至主集群的镜像仓库，导致备用集群无法安装或升级对应扩展。

常见问题

- 若备用集群的 ETCD 加密密钥未与主集群同步，安装备用集群前请按以下步骤操作。

1. 在任一备用集群控制平面节点获取 ETCD 加密密钥：

```
ssh <user>@<STANDBY cluster control plane node> sudo cat /etc/kubernetes/encryption-provider.conf
```

示例内容：

```
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
  - resources:
    - secrets
    providers:
    - aescbc:
        keys:
        - name: key1
          secret: MTE0NTE0MTkxOTgxMA==
```

2. 在任一主集群控制平面节点（如 1.1.1.1）备份 ETCD 加密密钥：

```
# 登录主集群控制平面节点 1.1.1.1
ssh "<user>@1.1.1.1"
sudo install -o root -g root -m 600 /etc/kubernetes/encryption-provider.conf /etc/kubernetes/encryption-provider.conf.bak
```

3. 将备用集群的 ETCD 加密密钥合并至节点 `1.1.1.1` 的 `/etc/kubernetes/encryption-provider.conf` 文件，确保密钥名称唯一。 例如，若主集群密钥为 `key1` ，则将备用集群密钥重命名为 `key2` ：

```
apiVersion: apiserver.config.k8s.io/v1
kind: EncryptionConfiguration
resources:
  - resources:
    - secrets
    providers:
    - aescbc:
        keys:
        - name: key1
          secret: My4xNDE1OTI2NTM1ODk3
        - name: key2
          secret: MTE0NTE0MTkxOTgxMA==
```

4. 确保新的 `/etc/kubernetes/encryption-provider.conf` 文件覆盖两个集群控制平面节点的所有副本：

```
# 假设主集群控制平面节点为 1.1.1.1、2.2.2.2 和 3.3.3.3
# 备用集群控制平面节点为 4.4.4.4、5.5.5.5 和 6.6.6.6

# 由于 1.1.1.1 已配置使用两个 ETCD 加密密钥，
# 登录节点 1.1.1.1，执行以下命令：
for i in \
    2.2.2.2 3.3.3.3 \
    4.4.4.4 5.5.5.5 6.6.6.6 \
; do
    scp /etc/kubernetes/encryption-provider.conf "<user>@${i}:/tmp/encryption-
provider.conf"
    ssh "<user>@${i}" '
#!/bin/bash
set -euo pipefail

sudo install -o root -g root -m 600 \
    /etc/kubernetes/encryption-provider.conf /etc/kubernetes/encryption-
provider.conf.bak
sudo install -o root -g root -m 600 \
    /tmp/encryption-provider.conf            /etc/kubernetes/encryption-provider.conf
sudo rm -f /tmp/encryption-provider.conf

_pod_name="kube-apiserver"
_pod_id=$(sudo crictl ps --name "${_pod_name}" --no-trunc --quiet)
if [[ -z "${_pod_id}" ]]; then
    echo "FATAL: could not find pod `kube-apiserver` on node $(hostname)"
    exit 1
fi
sudo crictl rm --force "${_pod_id}"
sudo systemctl restart kubelet.service
'
done
```

5. 重启节点 1.1.1.1 上的 kube-apiserver

```
_pod_name="kube-apiserver"
_pod_id=$(sudo crictl ps --name "${_pod_name}" --no-trunc --quiet)
if [[ -z "${_pod_id}" ]]; then
    echo "FATAL: could not find pod `kube-apiserver` on node $(hostname)"
    exit 1
fi
sudo crictl rm --force "${_pod_id}"
sudo systemctl restart kubelet.service
```