安装

安装路径选择

本节记录在 传统操作系统（例如 Ubuntu 或 RHEL）上的安装。如果您的环境使用 Alauda OS，请改为参阅 在不可变基础设施上安装 global 集群↗。

概览

概览

规划

规划

global 集群灾难恢复

global 集群灾难恢复

安装准备

前置条件 下载 节点预处理

准备外部平台镜像 **Registry**

安装

安装

安装程序参数

安装程序参数

验证

验证

安装故障排查

安装故障排查

后续步骤

后续步骤

概览

使用安装部分来规划并完成 ACP 4.4 安装。

安装会将 ACP Core 和所需的 Aligned Extensions 部署到 `global` 集群上。Core 提供平台管理平面、集群管理、用户和角色，以及 Extension 管理框架。所需的 Aligned Extensions 提供 Web Console 和其他面向平台的能力。有关平台架构以及 `global` 集群和业务集群的定义，请参阅 架构 和 术语表。

Core Package 和所需的 Aligned Extension packages 是分别下载的。启动安装程序之前，请将 Download 中列出的 8 个必需 Extension packages 复制到已解压的 Core Package 的 `plugins/` 目录中。其他 Extensions 会在平台安装完成后，通过 Extend 工作流进行安装。

> **INFO**
>
> ACP Core 安装不支持将 `global` 集群直接安装到现有的 Kubernetes 环境中。开始安装前，请准备满足硬件、网络、存储和 OS 要求的新节点。

目录

安装范围

安装会提供平台管理平面以及所需的 Aligned Extensions。安装验证通过后，您可以根据平台场景创建或连接业务集群，并安装其他 Extensions。

> **安装路径**
>
> 本节所述操作步骤会将 `global` 集群安装到 传统操作系统 上，例如 Ubuntu 或 RHEL。若 `global` 集群使用 Alauda OS，请参阅 在不可变基础设施上安装 global 集群↗。

以下工作不在本安装流程中：

- 创建业务集群。
- 连接现有的 Kubernetes 集群。
- 上传并安装除 8 个必需 Aligned Extensions 之外的其他 Extensions。
- 安装可选的 Operators 或 Cluster Plugins。
- 配置可选的平台能力，例如存储、监控、备份、registry 或身份集成。

安装工作流

安装遵循以下高层路径：

1. 规划。选择部署架构，并确认安装后无法安全更改的决策。
2. 如有需要，在准备资源之前阅读 全局集群灾难恢复。DR 会影响地址、证书、负载均衡、registry 选择以及两次安装的一致性。
3. 准备安装。满足硬件、存储、网络和节点要求。
4. 下载。下载匹配的 Core Package 和所需的 Aligned Extension packages。
5. 如果已选择，请 准备外部平台镜像 registry，并上传完整的目标版本负载。
6. 安装。解压 Core Package，准备所需的 packages，启动安装程序，并完成 Web UI 工作流。
7. 在查看安装程序中输入的值时，使用 安装程序参数。

8. 验证。仅在 Web UI、Core、所需的 Aligned Extensions、镜像源、artifacts 和 Pods 运行正常后，才接受安装。
9. 如果安装程序卡住或验证失败，请使用 安装故障排查。
10. 后续步骤。安装 Product Docs 插件，并继续执行特定场景的配置。

Customer Portal 是 ACP 安装包的官方来源。有关下载要求和 package 架构选项，请参阅 下载。

> **单独下载**
>
> Core Package 不包含所需的 Aligned Extension packages。请单独下载它们，并在启动安装程序之前，手动仅将 Download 中列出的 8 个 packages 复制到已解压的 Core Package 的 `plugins/` 目录中。

相关文档

- 架构
- 术语表
- Kubernetes 支持矩阵
- 磁盘配置要求
- Extend

规划

在开始安装程序之前，请选择部署架构，并确认会影响后续操作的安装设置。

INFO

ACP 安装会在已准备好的节点上部署 `global` 集群。它不支持将 `global` 集群直接安装到现有的 Kubernetes 环境中。

目录

选择部署架构

请选择与您的业务场景、管理模式、工作负载放置、资源规划和隔离要求相匹配的架构。

架构	适用场景	规划说明
多集群	您需要一个 `global` 集群来管理多个工作负载集群。	避免在 `global` 集群上运行非平台工作负载。请为预期管理的工作负载集群数量规划资源。
单集群	您有意使用一个集群同时承载平台组件和应用工作负载。	当符合业务场景时，单集群支持生产环境使用。由于 `global` 集群也会运行应用工作负载，因此请在安装前规划资源以及平台/工作负载隔离。
单节点	您需要测试环境或概念验证环境。	请勿将单节点用于生产环境。

有关资源需求和容量规划指导，请参见 先决条件。

确认安装时决策

在运行安装程序之前，请确认以下决策：

决策	需要确认的内容	继续阅读
安装包	选择 x86、ARM 或混合 Core Package，并下载与同一 ACP 版本和架构相匹配的所需 Aligned Extension packages。	下载
集群网络协议	在启动安装程序之前选择 IPv4、IPv6 或双栈。	安装
集群 Endpoint 和平台访问地址	准备集群组件、管理员和用户将使用的访问地址。	先决条件 和 安装程序参数
负载均衡	确定是使用外部负载均衡器还是自建 VIP。	先决条件 和 LoadBalancer 转发规则
证书	确定是使用安装程序生成的自签名证书，还是使用您提供的受信任证书。	先决条件 和 安装程序参数
平台镜像源	确定是使用平台内置 Registry，还是使用客户管理的外部平台镜像仓库。对于新的生产部署，当外部	准备外部平台镜像仓库 和 安装程序参数

决策	需要确认的内容	继续阅读
	仓库满足文档中说明的可用性和生命周期要求时，请使用外部仓库。	
节点	确认节点名称、节点角色，以及是否需要平台节点隔离。	节点预处理 和 安装程序参数
全局集群灾难恢复	确定您的环境是否需要主备 `global` 集群。	全局集群灾难恢复

如果您计划使用全局集群灾难恢复，请在安装 ACP Core 之前阅读 全局集群灾难恢复。灾难恢复规划会影响域名、VIP、证书、负载均衡器规则、镜像仓库选择、包架构以及安装参数一致性。

安装前阅读

在运行安装程序之前，请以以下页面作为事实依据：

- 先决条件：准备资源、网络和存储要求。
- 磁盘配置要求：确认磁盘容量和性能要求。
- 节点预处理：在安装前检查并准备每个节点。
- 下载：分别下载 Core Package 和 `Common ACP Extensions`，并验证它们的版本和架构是否匹配。
- 准备外部平台镜像仓库：如果已选择，请在启动安装程序之前上传完整目标版本的 Core 和 Extension 负载。
- 全局集群灾难恢复：如果需要灾难恢复，请规划主备 `global` 集群。
- 安装：上传 Core Package，准备所需包，并启动安装程序。
- 安装程序参数：在 Web UI 中配置安装选项。

global 集群灾难恢复

> **灾难恢复路径**
>
> 本页记录运行在 传统操作系统 上的 `global` 集群的灾难恢复。如果 `global` 集群使用 Alauda OS，请改为参阅 不可变基础设施上的 global 集群灾难恢复↗。

目录

概述

本方案面向 `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。为确保当 `global` 集群发生故障时平台服务持续可用，本方案部署两个 `global` 集群：主集群和备用集群。

灾难恢复机制基于从主集群到备用集群的 etcd 数据实时同步。如果主集群因故障不可用，服务可以快速切换到备用集群。

支持的灾难场景

- 主集群发生无法恢复的系统级故障，导致其无法运行；
- 托管主集群的物理机或虚拟机故障，导致其无法访问；
- 主集群所在位置发生网络故障，导致服务中断；

不支持的灾难场景

- `global` 集群内已部署应用发生故障；
- 由存储系统故障导致的数据丢失（不在 etcd 同步范围内）；

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是主集群（DNS 指向它），而待命集群是备用集群。发生故障切换后，这两个角色会互换。

注意事项

- 本方案仅同步 `global` 集群的 etcd 数据，不包括 registry、chartmuseum 或其他组件的数据；
- 为便于故障排查和管理，建议使用类似 `standby-global-m1` 的方式为节点命名，以指示该节点属于哪个集群（主集群或备用集群）。
- 不支持集群内应用数据的灾难恢复；
- 两个集群之间需要稳定的网络连接，以确保可靠的 etcd 同步；
- 如果集群基于异构架构（例如 x86 和 ARM），请使用双架构安装包；

- 以下 namespace 不参与 etcd 同步。如果在这些 namespace 中创建资源，用户必须手动备份：

```
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
```

- 如果两个集群都配置为使用内置镜像仓库，则容器镜像必须分别上传到两个集群；
- 如果主集群部署了 灵雀云 **DevOps Eventing v3**（knative-operator）及其实例，则备用集群中必须预先部署相同组件。

流程概览

1. 准备一个统一的平台访问域名；
2. 将域名指向 主集群 的 VIP，并安装 主集群；
3. 临时将 DNS 解析切换到备用 VIP，以安装备用集群；
4. 将 主集群 的 ETCD 加密密钥复制到后续将作为备用集群控制平面节点的节点上；
5. 安装并启用 灵雀云容器平台 **etcd Synchronizer**；
6. 验证同步状态并进行例行检查；
7. 发生故障时，将 DNS 切换到备用集群以完成灾难恢复。

所需资源

- 一个统一域名，作为 `Platform Access Address`，以及用于在该域名上提供 HTTPS 服务的 TLS 证书和私钥；
- 每个集群各自一个专用虚拟 IP 地址——主集群一个，备用集群一个；

网络要求

统一域名是灾难恢复的前提。在正常运行期间，该域名仅解析到 主集群 的 VIP。备用集群使用该域名访问主集群上的平台服务。

在安装任一集群之前，请完成以下两项网络准备：

- 配置每个集群的负载均衡器，将其 VIP 上所需的 TCP 端口转发到该 VIP 后面的控制平面节点。仅在用户通过 HTTP 访问平台时才配置端口 `80`。
- 在主集群和备用集群网络之间的 双向 放行所需 TCP 端口。请将规则应用到防火墙、安全组、路由器 ACL 以及任何其他站点间网络控制策略。

TCP 端口	服务	为何需要
`443`	ACP 平台 HTTPS	备用集群使用平台访问地址连接到活动平台。灵雀云容器平台 **etcd Synchronizer** 使用此地址，日志和监控组件也会向其发送告警回调。
`80`	ACP 平台 HTTP	仅在用户通过 HTTP 访问平台时需要。其余情况下的连接要求与端口 `443` 相同。
`6443`	Kubernetes API server	在安装或重新安装 灵雀云容器平台 **etcd Synchronizer** 期间，备用集群会连接活动集群的 API server，以获取同步所需的 etcd CA 材料。
`11443`	内置镜像仓库	备用集群通过平台访问地址配置的仓库拉取平台和插件镜像。
`2379`	etcd	备用集群上的 灵雀云容器平台 **etcd Synchronizer** 从主集群读取 etcd 数据并将其写入本地备用 etcd。

之所以要求网络策略为双向，是因为故障切换后集群角色会反转。在故障切换前，实际生效的服务流量通常从备用集群流向主集群。故障切换后，原主集群会成为新的备用集群，并且必须通过相同端口访问新的主集群。此要求并不意味着 etcd 复制是双向的：灵雀云容器平台 **etcd Synchronizer** 仅运行在当前备用集群上，并将同步数据写入其本地 etcd。

操作步骤

步骤 1：安装主集群

> **安装 DR（Disaster Recovery Environment）时的注意事项**
>
> 在安装 DR 环境的主集群时，
>
> - 首先，记录在安装 Web UI 指引过程中设置的所有参数。安装备用集群时，需要保持其中一些选项一致。
> - 必须预先配置 **User-provisioned** 负载均衡器，以便将流量路由到虚拟 IP。`Self-built VIP` 选项不可用。
> - `Platform Access Address` 字段必须填写域名，而 `Cluster Endpoint` 必须填写虚拟 IP 地址。
> - 两个集群都必须配置为使用 `An Existing Certificate`（且必须是同一个证书）；如有必要，请申请合法证书。`Self-signed Certificate` 选项不可用。
> - 当 `Image Repository` 设置为 `Platform Deployment` 时，`Username` 和 `Password` 字段都不能为空；`IP/Domain` 字段必须设置为用作 `Platform Access Address` 的域名。
> - `Platform Access Address` 的 `HTTP Port` 和 `HTTPS Port` 字段必须分别为 80 和 443。
> - 在安装指南第二页（步骤：`Advanced`）中，`Other Platform Access Addresses` 字段必须包含当前集群的虚拟 IP。

请参阅以下文档完成安装：

- 安装前准备
- 安装

步骤 2：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，并将 etcd 加密配置复制到所有备用集群控制平面节点：

```
# Assume the primary cluster control plane nodes are 1.1.1.1, 2.2.2.2 &
3.3.3.3
# and the standby cluster control plane nodes are 4.4.4.4, 5.5.5.5 & 6.
6.6.6
for i in 4.4.4.4 5.5.5.5 6.6.6.6  # Replace with standby cluster contro
l plane node IPs
do
  ssh "<user>@$i" "sudo mkdir -p /etc/kubernetes/"
  scp /etc/kubernetes/encryption-provider.conf "<user>@$i:/tmp/encrypti
on-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/en
cryption-provider.conf"
done
```

3. 以与主集群相同的方式安装备用集群

> **安装备用集群时的注意事项**
>
> 在安装 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：请确保镜像仓库以及 ACP 管理员用户的凭据与 主集群 上设置的一致。
>
> 并且务必确保你已遵循步骤 1 中的 `安装 DR（Disaster Recovery Environment）时的注意事项`。

请参阅以下文档完成安装：

- 安装前准备
- 安装

步骤 3：启用 etcd 同步

1. 在安装插件之前，从活动 global 集群获取用于访问活动 global 集群 API server 的 bearer token。复制命令输出，并在备用 global 集群的 `cpaas-system` namespace 下创建 `etcd-`

`sync-active-cluster-token` Secret 时使用。该 Secret 会将令牌存储在数据键 `token` 中。

```
# Run this command on the active cluster.
kubectl -n cpaas-system get secret k8sadmin -o jsonpath='{.data.token}'
| base64 -d
```

复制输出值，然后在备用集群上运行此命令：

```
ACTIVE_CLUSTER_TOKEN='<paste-the-token-from-the-active-cluster>'
kubectl -n cpaas-system create secret generic etcd-sync-active-cluster-
token \
  --from-literal=token="${ACTIVE_CLUSTER_TOKEN}" \
  --dry-run=client -o yaml | kubectl apply -f -
```

2. 如适用，配置负载均衡器将端口 `2379` 转发到对应集群的控制平面节点。仅支持 TCP 模式；不支持 L7 转发。

> **INFO**
>
> 不要求通过负载均衡器进行端口转发。 如果备用集群可以直接访问活动 global 集群，请通过 **Active Global Cluster ETCD Endpoints** 指定 etcd 地址。

3. 使用备用集群的 VIP 访问 备用 **global** 集群 Web Console，并切换到 管理员 视图。
4. 导航到 **Marketplace > Cluster Plugins**，选择 `global` 集群。
5. 找到 灵雀云容器平台 **etcd Synchronizer**，点击 **Install**，并配置以下参数：

   - 将 **Active Global Cluster VIP** 设置为活动 global 集群的 VIP。
   - 当未通过负载均衡器转发端口 `2379` 时，请正确配置 **Active Global Cluster ETCD Endpoints**。
   - 将 **Standby Cluster ETCD Endpoints** 设置为备用集群的 etcd 地址。除非本地 etcd 服务通过不同端点暴露，否则请使用默认值。
   - 将 **Active Global Cluster Token Secret** 设置为 `etcd-sync-active-cluster-token`。
   - 使用 **Data Check Interval** 的默认值。
   - 除非需要排障，否则保持禁用 **Print detail logs**。

在安装过程中，系统会在 `etcd-sync` Deployment 启动之前先运行 `etcd-sync-bootstrap` Job。只有当该 Job 准备好 `remote-etcd-ca`、`remote-etcd-issuer` 和 `remote-etcd-client` 后，插件安装才会继续。

验证 bootstrap Job 和运行时资源：

```
kubectl get job -n cpaas-system etcd-sync-bootstrap
kubectl logs -n cpaas-system job/etcd-sync-bootstrap
kubectl get secret -n cpaas-system remote-etcd-ca
kubectl get issuer -n cpaas-system remote-etcd-issuer
kubectl get certificate -n cpaas-system remote-etcd-client
kubectl get secret -n cpaas-system remote-etcd-client
```

验证同步 Pod 是否已在备用集群上运行，并确认当前 leader：

```
kubectl get po -n cpaas-system -l app=etcd-sync
kubectl get lease -n cpaas-system etcd-sync-mirror
leader_pod=$(kubectl get lease -n cpaas-system etcd-sync-mirror -o jsonpa
th='{.spec.holderIdentity}')
kubectl logs -n cpaas-system "$leader_pod" | grep -E "Acquired leader lea
se|Start Sync update"
```

如果需要重新同步依赖 `ownerReference` 的资源，请在出现 `Start Sync update` 后重新创建当前 leader Pod：

```
leader_pod=$(kubectl get lease -n cpaas-system etcd-sync-mirror -o jsonpa
th='{.spec.holderIdentity}')
kubectl delete po -n cpaas-system "$leader_pod"
```

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 =~ $ipv6_regex ]]; then
  mirror_host="[$mirror_svc]"
else
  mirror_host="$mirror_svc"
fi
curl -g "http://${mirror_host}/check"
```

输出说明：

- `LOCAL ETCD missed keys`：这些键存在于主集群中，但在备用集群中缺失。通常由同步期间资源顺序导致的 GC 引起。重启当前 etcd-sync leader Pod 即可修复；
- `LOCAL ETCD surplus keys`：这些多余键仅存在于备用集群中。请在从备用集群删除这些键之前先与运维团队确认。

如果安装了以下组件，请重启其服务：

- 灵雀云容器平台 Log Storage for Elasticsearch：

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

- 灵雀云容器平台 Monitoring for VictoriaMetrics：

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

灾难恢复过程

> **卸载 Alauda Container Platform etcd Synchronizer 之前验证主/备用一致性**
>
> 本操作会卸载 灵雀云容器平台 **etcd Synchronizer**。在卸载之前，请确认备用集群数据与主集群一致。如果在备用集群缺少主集群拥有的数据时卸载插件，可能会导致 owner reference 解析错误，并且业务集群节点 `Machine` 对象——包括不可变操作系统集群在内，在这些场景下其背后的虚拟机

将被销毁——可能会被删除。如果一致性检查报告存在缺失键或多余键，请不要卸载插件；应先解决不一致问题，或联系技术支持。

1. 如有必要，先在备用集群上重启 Elasticsearch：

```
# Copy installer/res/packaged-scripts/for-upgrade/ensure-asm-template.s
h to /root:
# DO NOT skip this step

# switch to the root user if necessary
sudo -i

# check whether the Log Storage for Elasticsearch is installed on globa
l cluster
_es_pods=$(kubectl get po -n cpaas-system | grep cpaas-elasticsearch |
awk '{print $1}')
if [[ -n "${_es_pods}" ]]; then
    # In case the script returned the 401 error, restart Elasticsearch
    # then execute the script to check the cluster again
    bash /root/ensure-asm-template.sh

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

2. 验证备用集群中的数据一致性（与 步骤 3 中的检查相同）。如果检查报告存在缺失键或多余键，说明备用集群与主集群不一致：不要继续下一步。应先解决不一致问题，或联系技术支持。在 两个 集群上，还要确认没有任何 `Machine` 节点处于非运行状态；继续之前先处理所有异常节点：

```
kubectl get machines.platform.tkestack.io
```

3. 卸载 灵雀云容器平台 **etcd Synchronizer**；
4. 移除两个 VIP 上的端口转发 `2379`；
5. 将平台域名的 DNS 切换到备用 VIP，此时该 VIP 将成为主集群；
6. 验证 DNS 解析：

```
kubectl exec -it -n cpaas-system deployments/sentry -- nslookup <platfo
rm access domain>
# If not resolved correctly, restart coredns Pods and retry until succe
ss
```

7. 清除浏览器缓存并访问平台页面，确认其显示的是原备用集群；
8. 重启以下服务（如果已安装）：
   - 灵雀云容器平台 Log Storage for Elasticsearch：

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

   - 灵雀云容器平台 Monitoring for VictoriaMetrics：

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

   - cluster-transformer：

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

9. 如果业务集群向主集群发送监控数据，请在业务集群中重启 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 中的步骤，将其转换为新的备用集群。

例行检查

定期检查备用集群上的同步状态：

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

如果存在缺失键或多余键，请根据输出中的说明进行处理。

上架软件包

有关 `violet push` 子命令的详细信息，请参阅 上架软件包。

安装准备

前置条件 下载 节点预处理

准备外部平台镜像 **Registry**

前置条件

在安装 `global` 集群之前，需要先准备满足要求的硬件、网络和 OS。

INFO

1. 当前平台不支持在已有 Kubernetes 环境中直接安装 `global` 集群。如果你的环境中已经存在 Kubernetes 集群，请先备份数据并清理环境后再安装。
2. 如果你计划使用 global 集群容灾，请先阅读 global 集群容灾。
3. 请确保所有新节点都满足 节点要求。
4. 磁盘性能和容量也必须满足 磁盘配置要求。

目录

资源规划

本节提供在安装 ACP 之前的资源规划指导。 请根据你的环境和使用需求选择合适的部署场景，并据此准备资源。

> **INFO**
>
> 以下建议仅覆盖成功安装 **global** 集群 所需的最小资源。
>
> 它们不包括在 global 集群上部署的任何额外扩展或组件所需的资源。
>
> 有关各扩展的详细要求，请参阅对应的组件文档。

部署架构

> **WARNING**
>
> 对于 ARM 架构（例如 Kunpeng 920），建议将配置提升到 x86 最低配置的 **2** 倍，但不低于 **1.5** 倍。
>
> 例如：如果 x86 需要 8 核 16GB，那么 ARM 至少应达到 12 核 24GB，推荐配置为 16 核 32GB。

在安装之前，你必须先确定最适合你的使用场景的部署架构。 ACP 支持以下三种常见部署架构：

- 多集群
  如果你需要集中管理多个 Kubernetes 集群，请选择此架构。 在此模式下，ACP 由一个 **global** 集群 和多个 业务集群 组成。 在 global 集群上运行非平台业务负载可能会降低平台稳定性和性能，应尽量避免。
- 单集群
  如果你计划只安装一个集群并直接在其上运行工作负载，请选择此架构。 在此模式下，global 集群同时充当业务集群，因此与多集群模式下纯粹的 global 集群部署相比，需要更多资源。
- 单节点

> **WARNING**
>
> 此架构仅用于测试或概念验证，不应用于生产环境。

单节点

下表列出了在 单节点 模式下安装 ACP 的最低硬件要求。

资源	最低要求
CPU	12 核
内存	24GB
存储	存储容量

单集群

在此模式下，global 集群同时充当控制平面和业务集群。 global 集群的控制平面节点数必须为 3。

总资源需求由两部分组成：

- global 集群自身所需的基础资源
- 在同一集群上运行工作负载所需的额外资源

高可用 **global** 集群 所需资源如下：

资源	最低要求
CPU	8 核
内存	16GB
存储	存储容量

如需估算工作负载所需的额外资源，请参阅 评估业务集群资源

多集群

在管理多个业务集群时，global 集群的资源占用会随着所管理集群数量的增加而线性上升。 额外开销主要来自集群注册、监控以及控制平面同步。

如需根据所管理集群数量估算 global 集群所需资源，请参阅 评估 **global** 集群资源

网络

在安装之前，请确保已准备好所需的网络资源。

如果你的环境中有硬件负载均衡器，推荐使用它。如果没有，可以启用 `Self-built VIP`，它使用 keepalived 提供软件负载均衡。

注意： `Self-built VIP` 不支持域名配置。

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，并配置在负载均衡设备中以确保高可用性。 该 IP 也可用作平台 Web UI 的访问地址。 与 `global` 集群处于同一网络中的业务集群也可以通过该 IP 访问 `global` 集群。
外部 IP	可选	按需	当存在与 `global` 集群不在同一网络中的业务集群时，例如混合云场景，必须提供该 IP。其他网络中的业务集群通过该 IP 访问 `global` 集群。 该 IP 需要配置在负载均衡设备中以确保高可用性。 该 IP 也可用作平台 Web UI 的访问地址。

资源	是否必需	数量	说明
域名	推荐	按需	推荐用于集群端点和平台访问地址。请提前提供，并确保域名解析正确。 推荐使用域名，因为如果在集群安装后需要变更 VIP，可以轻松更新 DNS 记录而不影响集群运行。 在以下情况下，域名是必需的： • `global` 集群需要支持 IPv6 访问； • 计划为 `global` 集群配置容灾方案。
证书	可选	按需	建议使用受信任的证书，以避免浏览器安全警告；如果未提供，安装程序将生成自签名证书，但在使用 HTTPS 时可能存在安全风险。

> **注意**
>
> 如果平台需要配置多个访问地址（例如内网和外网地址），请根据上表提前准备相应的 IP 地址或域名。你可以稍后在安装参数中进行配置，或者在安装后根据产品文档进行添加。

网络配置

类型	要求说明
网络带宽	集群内带宽必须 ≥1 Gbps（推荐 10 Gbps）。 跨集群带宽必须 ≥100 Mbps（推荐 1 Gbps）。 带宽不足可能会显著降低数据查询性能。

类型	要求说明
网络时延	集群内时延必须 ≤10 ms。 跨集群时延必须 ≤100 ms（推荐 ≤30 ms）。
网络策略	请参考 负载均衡器转发规则，确保必要端口已开放。
IP 地址范围	`global` 集群节点应避免使用 172.17-18 网段。如果已经使用，请调整 nerdctl 配置（添加 bip 参数）以避免冲突。

负载均衡器转发规则

此规则旨在确保 `global` 集群能够正常接收来自负载均衡器的流量。请根据下表检查网络策略，确保相关端口已开放。

源 **IP**	协议	目标 **IP**	目标端口	说明
`global` VIP, External 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`。如果你需要使用自定义 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你的自定义端口号。 • 然后在 Web UI 安装参数中

源 IP	协议	目标 IP	目标端口	说明
				填写你的自定义端口号。
`global` VIP, External IP	TCP	所有控制平面节点 IP	6443	此端口为集群内节点提供访问 Kubernetes API Server 的能力。
`global` VIP, External IP	TCP	所有控制平面节点 IP	11443	此端口为集群内节点提供访问镜像仓库的能力。注意：如果你计划使用外部平台镜像仓库，而不是 `global` 集群提供的 Registry，则无需配置此端口。有关外部仓库网络要求，请参阅 准备外部平台镜像仓库。

提示

- 建议在负载均衡器上配置健康检查，以监控端口状态。
- global 集群容灾要求在两个集群 VIP 上都配置负载均衡监听，并且主备集群网络之间具备双向网络访问能力。请参阅 网络要求。
- 平台默认仅支持 HTTPS。如果需要支持 HTTP，你需要为所有控制平面节点开放 HTTP 端口。

外部平台镜像仓库

对于新的生产部署，当你的组织需要独立的仓库可用性、容量、备份和生命周期管理时，请准备一个由客户自主管理的外部平台镜像仓库。

在运行 `setup.sh` 之前，请完成 准备外部平台镜像仓库。该页面定义了所需的地址、网络访问、存储以及推送和拉取权限；将这些要求与生产建议区分开来；并说明如何上传完整的目标版本负载。

下载

在安装之前，请从 灵雀云 **customer portal** 分别下载 **Core Package** 和所需的 Aligned Extension packages。

目录

关于 **customer portal**

灵雀云 **customer portal** 是 ACP 产品资源的官方服务和交付门户。可通过它获取经过验证的软件包，以及用于安装、升级和维护活动的官方技术指导。

customer portal 提供以下资源：

- 产品下载，包括安装、升级和扩展包。
- 产品文档和技术指导。

- 用于提交、跟踪和管理支持请求的支持资源。
- 用于扩展平台能力的 Marketplace 包。
- 用于部署和维护规划的许可证相关资源。

请使用已授权账户下载适用于你的环境的包。如果你没有注册账户，请联系技术支持。

对于客户交付和生产环境，请使用 customer portal 上发布的包版本和文档作为官方部署和维护基线。

下载安装包

1. 下载与目标架构匹配的 ACP 4.4 **Core Package**。
2. 下载与相同 ACP 版本和架构匹配的 **Common ACP Extensions**。推荐使用 Violet CLI，因为它可以直接定位并下载场景包。

使用 **Violet CLI** 下载（推荐）

安装 **Violet v4.2.13** 或更高版本，运行 `violet version` 验证版本，并使用 `violet ac login` 登录 灵雀云 customer portal。有关安装和登录说明，请参见 下载包。

将 `<target-platform-version>` 设置为 Core Package 的确切 ACP Distribution Version，例如 `v4.4.0`。将 `<arch>` 设置为 `amd64`、`arm64` 或 `hybrid`，以匹配 Core Package。先列出场景，然后下载 **Common ACP Extensions**：

```
violet ac scenarios --arch=<arch> --platformVersion=<target-platform-vers
ion>
violet ac download-app --arch=<arch> --platformVersion=<target-platform-v
ersion> --scenario="Common ACP Extensions"
```

在下载之前，请确认 **Common ACP Extensions** 已出现在场景列表中。

从 **customer portal** 下载

作为 Violet 的替代方案，在 customer portal 中，进入 **Marketplace > Batch Download >** 安装。在页面上，从 **Your ACP Version** 中选择 Core Package 版本，选择匹配的

Architecture，从 **Scenarios** 中选择 **Common ACP Extensions**，然后下载这些包。

> 独立包
>
> Core Package 只包含 Core 制品，不包含通过 **Common ACP Extensions** 下载的任何 Extension packages。解压 Core Package 后，在启动安装器之前，必须手动将所需的 Extension packages 复制到其 `plugins/` 目录中。

必需的 **Aligned Extensions**

将以下应用的包复制到已解压 Core Package 的 `plugins/` 目录中：

- **Alauda Container Platform Web Console**
- **Alauda Container Platform Web Console Central**
- **Alauda Container Platform Marketplace Web Console**
- **Alauda Container Platform Helm Application Catalog**
- **Alauda Container Platform Observability Essentials**
- **Alauda Container Platform Essentials**
- **Alauda Container Platform Cluster Essentials**
- **Alauda Container Platform GatewayAPI Plugin**

不要将 **Common ACP Extensions** 中的其他包复制到已解压 Core Package 的 `plugins/` 目录中。请按照 扩展 在平台安装完成后安装这些 Extensions。

选择包架构

包支持 **x86**、**ARM** 和 **hybrid** 架构。hybrid 包同时包含 x86 和 ARM 的镜像，因此包体积更大。请选择最适合你环境的包，并使用与 Core Package 架构匹配的 Extension packages。

从单架构迁移到 **hybrid**

如果最初使用 x86 或 ARM Core Package 进行安装，但后来需要支持另一种架构，则必须重新下载 **hybrid Core Package**，并执行以下步骤：

1. 将新下载的 hybrid Core Package 上传到 global 集群的任一控制平面节点。
2. 解压该包，并使用其中包含的 `upgrade.sh` 脚本将多架构镜像同步到你的镜像仓库：

   ```
   bash upgrade.sh --only-sync-image=true
   ```

3. 脚本执行完成后，检查 `cluster.platform.tkestack.io` 资源，确认是否存在标签 `cpaas.io/node-arch-constraint`。如果存在，则必须将其删除：

   ```
   kubectl get cluster.platform.tkestack.io global -oyaml | grep cpaas.io/node-arch-constraint
   # If there is output, edit the resource to remove the label; otherwise, you can skip this step.
   kubectl edit cluster.platform.tkestack.io global   ### Edit the labels field and delete cpaas.io/node-arch-constraint
   ```

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）都必须完成预处理。

> **INFO**
>
> 本页适用于运行 传统操作系统 的节点，例如 Kylin Linux Advanced Server、RHEL 或 Ubuntu，ACP 会通过 SSH 为其进行 provisioning。下面的一些检查——例如 SSH 用户以及 `/etc/ssh/sshd_config` 设置——是为了确保基于 SSH 的节点加入流程能够正常工作。如果您的集群使用 Alauda OS，则本页中的节点预处理步骤不适用。请改为参考 Immutable Infrastructure 文档中的 在 Immutable Infrastructure 上安装 global 集群↗。

目录

支持的 OS 和 Kernel 版本

下表列出了受支持的操作系统、已验证版本以及对应的已测试 Kernel 版本。

平台会强制执行支持列表中声明的版本匹配粒度：

- **OS** 版本：使用下方列出的发行版版本。不支持其他 major 或 minor 版本，除非它也在列表中。
- **Kernel** 版本：使用操作系统厂商提供的官方 Kernel。若列表给出了精确的 `x.y.z-build` 值，则 build 后缀可以不同，但 `x.y.z` 必须匹配。若列表给出了 Kernel 系列，请使用该系列中已针对已安装的 ACP patch 验证过的 Kernel。不要假设任意较新的 Kernel 都受支持。

> **INFO**
>
> - 对于运行 Kubernetes 1.35 或更高版本的 ACP 集群，节点必须使用 cgroup v2 和 Linux kernel 5.8 或更高版本。仅使用下方列出的、已针对已安装的 ACP patch 验证过的 OS 和 Kernel 组合。
> - `x86-64-v2` 是 CPU 指令集基线，但 ACP 不会将其作为运行用户提供的传统操作系统的 x86 节点的通用要求。CPU 仍必须满足所选操作系统和待部署组件的要求。如果无法确认兼容性，请联系技术支持。

x86

Red Hat Enterprise Linux (RHEL)

- RHEL 9.6: `5.14.0-570.12.1`

Ubuntu

- Ubuntu 22.04 LTS: `5.15.0-56-generic`

注意： 不支持 Ubuntu HWE（Hardware Enablement）版本。

Kylin Linux Advanced Server

- Kylin Linux Advanced Server V11: 适用于已安装的 ACP patch 的 6.6 系列官方 Kernel

注意： Kylin Linux Advanced Server V11 不支持 Flannel。

ARM

Kylin Linux Advanced Server

- Kylin Linux Advanced Server V11: 适用于已安装的 ACP patch 的 6.6 系列官方 Kernel

注意： ARM 架构仅支持 `Kunpeng 920`，且不支持 Flannel。对于其他 CPU 型号，请联系技术支持。

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，可在 `res` 目录中获得 `init.sh` 脚本文件。将该脚本文件复制到各节点，并确保您具有 `root` 权限。

执行脚本：

```
bash init.sh
```

WARNING

`init.sh` 无法保证以下所有检查都已正确处理。您仍然需要继续执行下面的步骤。

节点检查

以下列出了节点上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，某些检查仅适用于控制平面节点。

检查分为两类：

- ✅ 表示该检查必须通过。
- ⚠️ 表示该检查仅在特定场景下必须满足。请根据说明判断对应条件是否成立。如果成立，您必须解决该问题。

以下是检查列表：

- **OS** 和 **Kernel**
  - ✅ 机器的 grub 启动配置中必须包含 `transparent_hugepage=never` 参数。
  - ✅ 检查 Kernel 模块 `ip_vs`、`ip_vs_rr`、`ip_vs_wrr` 和 `ip_vs_sh` 是否已启用。
  - ⚠️ 如果 `global` 集群计划使用 `Kube-OVN` CNI，则必须启用 Kernel 模块 `geneve` 和 `openvswitch`。
  - ✅ 禁用 apparmor/selinux 和 firewall。
  - ✅ 禁用 `swap` 。
- 用户和权限
  - ✅ 节点的 SSH 用户必须具有 `root` 权限，并且可以免密码使用 `sudo`。
  - ✅ `/etc/ssh/sshd_config` 中的 `UseDNS` 参数必须设置为 `no`。
  - ✅ 在添加节点之前，将 `/etc/ssh/sshd_config` 中的 `UsePAM` 参数设置为 `no` ，然后重启 `sshd`。否则，PAM session 策略（例如强制修改密码、`pam_access`、`faillock` 或 `pam_limits`）可能会阻止基于 SSH 的节点加入流程。节点达到 `Ready` 状态后，您可以恢复为 `UsePAM yes`。在启用 SELinux 强制模式且使用密码认证的系统上，`UsePAM no` 本身可能会破坏 SSH 登录；在这种情况下，请使用基于密钥的认证。
  - ✅ `systemctl show --property=DefaultTasksMax` 必须返回 `infinity`；较低的限制会导致繁忙的容器无法创建线程。如果不是 `infinity` ，请在 `/etc/systemd/system.conf` 中设置 `DefaultTasksMax=infinity` ，并运行 `systemctl daemon-reexec`。
- 节点网络
  - ✅ `hostname` 必须符合以下规则：
    - 不超过 36 个字符。
    - 以字母或数字开头并结尾。
    - 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` 文件必须存在并包含 `nameserver` 配置，但不得包含以 172 开头的地址（禁用 systemd-resolved）。
- ⚠️ `/etc/resolv.conf` 文件不应配置 search domains（如果必须配置，请参见 配置 Search Domain）。
- ✅ 机器的 IP 地址不能是 loopback、multicast、link-local、all-0 或 broadcast 地址。
- ✅ 执行 `ip route` 必须返回默认路由，或者返回指向 `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`
- ✅ 如果集群使用 **Kube-OVN** 或 **Calico**，请确保以下端口未被占用：
  - **Kube-OVN**：`6641`、`6642`
  - **Calico**：`179`
- ⚠️ 确保 nerdctl 所需的 `172.17.x.x` ~ `172.18.x.x` 网络段中的 IP 地址未被占用。如果该网络段中的 IP 已被占用且无法更改，请联系技术支持。

- 软件和目录要求：
  - ✅ 必须已安装以下组件：`ip`、`ss`、`tar`、`swapoff`、`modprobe`、`sysctl`、`md5sum`，以及 `scp` 或 `sftp`。
  - ⚠️ 如果您计划使用本地存储 **TopoLVM** 或 **Rook**，则需要安装 `lvm2`。
  - ✅ `/etc/systemd/system/kubelet.service` 文件不得存在。
  - ✅ `/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（参见 删除冲突软件包）。
  - ✅ 如果存在以下文件，则必须将其删除：
    - `/var/lib/docker`
    - `/var/lib/nerdctl`
    - `/opt/nerdctl/`

    - `/var/lib/containerd`
    - `/var/log/pods`
    - `/var/lib/kubelet/pki`
- 跨节点检查
    - ✅ `global` 集群中节点之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。
    - ✅ 所有节点的时区必须统一，且时间同步误差必须 ≤ 10 秒。

附录

删除冲突软件包

在安装之前，节点上可能已经在 docker/nerdctl/containerd 环境中运行了应用，或者已经安装了与 `global` 集群冲突的软件。因此，有必要检查并卸载冲突软件包。

> **DANGER**
>
> - 为避免应用中断或数据丢失，请务必确认是否存在冲突的软件包。发现冲突时，请先制定应用切换方案并备份数据，然后再进行卸载。
> - 卸载冲突软件包后，仍需检查 `/usr/local/bin/` 等目录下是否存在其他可能冲突的二进制文件（例如与 docker、nerdctl、containerd、runc、podman、容器网络、容器运行时或 Kubernetes 相关的软件）。

可参考以下命令。

RHEL

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-ap
iserver \
    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-ap
iserver \
    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 ku
bernetes-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 ku
bernetes-kubeadm \
    ; do
    yum remove "$x"
done
```

配置 Search Domain

在 Linux OS 中，`/etc/resolv.conf` 文件用于配置 DNS 客户端的域名解析设置。`search` 行指定了 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`（默认值为 2048）。

示例

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

- 域的总数为 3。
- 单个域名长度，例如 `domain1.com`，为 11。
- 总字符长度为 35，即 11 + 11 + 11 + 2（两个空格）。

> **WARNING**

- 如果 `/etc/resolv.conf` 文件中的 `search` 行不满足上述限制，可能会导致 DNS 查询失败或性能下降。
- 在修改 `/etc/resolv.conf` 文件之前，建议先备份该文件。

准备外部平台镜像 Registry

对于新的生产部署，你可以使用客户自行管理的外部镜像 Registry 作为 ACP Core、受管集群和 Extensions 的中央镜像源。在启动安装程序之前，Registry 必须包含完整的目标版本负载。

> **安装前准备 Registry**
>
> 当选择外部 Registry 时，安装程序不会部署平台内置 Registry，也不会将缺失的镜像或软件包复制到外部 Registry。在运行 `setup.sh` 之前，请完成本页上的两种上传模式。

目录

了解 Registry 角色

以下 Registry 角色彼此独立：

角色	用途	生命周期所有者
平台内置 Registry	当安装 `global` 集群并将其镜像仓库设置为 **Platform Deployment** 时，用于存储平台 Core 镜像和 Extension 工件。	ACP 部署该服务；平台管理员负责其容量和备份。
外部平台镜像 Registry	替代平台内置 Registry，作为 Core、受管集群和 Extensions 的镜像源。	由你提供并运行该 Registry，并上传所需的每个目标版本负载。

本页配置的是 ACP 组件和集群使用的平台镜像源。它不配置应用工作负载的镜像服务。

平台内置 Registry 仍然支持现有环境和非生产评估。本页不描述从内置 Registry 到外部 Registry 的在线迁移。

Registry 要求

必需能力和访问权限

在上传目标版本负载之前，请确认 Registry：

- 支持 Core Package 工具执行的标准容器镜像 push 和 pull 操作。
- 使用如下形式的地址：`<host-or-IP>[:port]`，例如 `registry.example.com:5000`。在安装程序的 Registry 地址中不要包含 `http://` 或 `https://`。
- 可通过安装节点以及每个拉取平台镜像的集群节点所需的防火墙、代理和网络路由访问。
- 提供可创建或更新所需仓库路径并推送镜像的上传凭据。传递给 `setup.sh` 的凭据必须具有 pull 权限。上传和 pull 凭据可以属于不同的账户；在对 `setup.sh` 进行身份验证时，请同时提供其用户名和密码。
- 具有足够的可用存储空间，以容纳你上传的目标版本 Core 和 Extension 负载。

生产环境建议

对于生产 Registry：

- 使用 HTTPS，并确保其证书对于 Registry 地址有效。确保安装节点以及每个拉取平台镜像的集群节点都信任签发该证书的 CA。
- 使用稳定的 DNS 名称，并确保安装节点和所有集群节点都能解析。
- 为每一种所需 CPU 架构以及用于升级、回滚和节点恢复时保留的版本规划容量。
- 根据你的环境提供可用性、监控、备份和恢复能力。当新 Pod 启动、节点被替换、创建或升级集群，以及安装或升级 Extensions 时，都需要 Registry 可用。
- 配置保留策略时，不要删除仍被已安装的平台版本、集群或 Extensions 引用的 manifest 或 digest。

> 客户自行管理的服务
>
> ACP 会使用外部 Registry，但不会安装、升级、备份、监控或修复它。对于这些操作，请遵循你的 Registry 供应商提供的管理文档。

准备目标版本负载

使用与你将要安装的 ACP Distribution Version 和 CPU 架构完全匹配的 Core Package。不要重复使用其他版本或架构的负载。

1. 完成 下载，解压 Core Package，并切换到其 `installer/` 目录。
2. 按照 安装 中的说明，将所需的 Aligned Extension 软件包复制到 `plugins/` 中。`all` 上传模式会处理当前位于此目录中的软件包。
3. 在当前 shell 中设置 Registry 地址和上传凭据。读取密码时不要将其写入 shell history：

   ```
   export REGISTRY_ADDRESS="registry.example.com"
   export REGISTRY_USERNAME="<registry-username>"
   read -rsp "Registry password: " REGISTRY_PASSWORD
   export REGISTRY_PASSWORD
   printf '\n'
   ```

   对于允许匿名 push 的 Registry，请将 `REGISTRY_USERNAME` 和 `REGISTRY_PASSWORD` 都设置为空值。
4. 从解压后的 `installer/` 目录运行两种上传模式。你可以按任意顺序执行这些命令，但两者都必须成功完成：

```
bash res/upload.sh all \
  "$REGISTRY_ADDRESS" "$REGISTRY_USERNAME" "$REGISTRY_PASSWORD"

bash res/upload.sh necessary \
  "$REGISTRY_ADDRESS" "$REGISTRY_USERNAME" "$REGISTRY_PASSWORD"
```

这两种模式有重叠，但彼此都不会替代对方：

- `all` 会上传产品镜像、cluster version operator 镜像，以及当前 `plugins/` 中的每个 `*.tgz` 软件包。它不会上传安装程序工件。
- `necessary` 会上传产品镜像、cluster version operator 镜像、安装程序工件，以及创建 `global` 集群所需的 bootstrap plugins。它不会上传 `plugins/` 中的所有软件包。

5. 在 Registry 管理界面或 API 中，确认新上传的仓库和 manifest 已存在，并且不会立即被清理。

故障排除

症状	可能原因	操作
`x509: certificate signed by unknown authority`	Registry CA 或中间证书不受信任。	在安装节点以及每个拉取镜像的集群节点上安装完整的 CA 链，然后重试。不要在生产环境中用不安全的 Registry 代替 HTTPS 作为变通方案。
Certificate name mismatch	Registry 地址与证书的 subject alternative names 不匹配。	使用证书所覆盖的 Registry DNS 名称，或替换证书。
`unauthorized` or `authentication required`	上传凭据不正确，或缺少 push 权限。	验证 `res/upload.sh` 使用的账户，并确认其可以 push 到所需的仓库路径。
`denied` or repository creation fails	上传账户无法创建或更新所需的仓库路径。	为该账户授予所需的仓库权限，然后重新运行两种上传模式。

症状	可能原因	操作
报告找不到必需的软件包	Core Package 不完整、解压了错误的版本或架构，或者所需的 Extension 软件包没有复制到 `plugins/` 中。	在重试之前，请验证软件包校验和、版本、架构以及 `plugins/` 内容。

下一步

在两种上传模式都成功完成且 Registry 中包含目标负载后，继续执行 安装。

安装

本节介绍如何安装 ACP Core 和所需的 Aligned Extensions，并部署 `global` 集群。

> **安装路径**
>
> 此安装路径适用于运行在传统操作系统上的集群。如果你的环境使用 Alauda OS，请改为参见 在不可变基础设施上安装 global 集群↗。

在开始安装之前，请确保你已完成先决条件检查、已下载并验证 Core Package 和所需的 Aligned Extension 包、已完成节点预处理，并完成了其他准备工作。如果你选择了外部平台镜像仓库，则在运行 `setup.sh` 之前，必须确保其完整的目标版本 payload 已可用。

目录

流程

1 上传并解压安装包

将 Core Package 安装包上传到 `global` 集群控制平面的任意一台机器上，并按以下命令进行解压：

```
# Assume that the /root/cpaas-install folder already exists on the machine
tar -xvf {Path to Core Package File}/{Core Package File Name} -C /root/cpaas-install
cd /root/cpaas-install/installer || exit 1
```

> 信息
>
> - 在 `global` 集群安装完成后，这台机器将成为第一个控制平面节点。
> - Core Package 解压后，至少需要 **100GB** 的磁盘空间。请确保有足够的存储资源。
> - Core Package 不包含所需的 Aligned Extension 包。请按照 下载 中的说明单独准备这些包。

2 准备所需的 **Aligned Extension** 包

将 Required Aligned Extensions 中列出的 8 个包分别复制到解压后的 Core Package 的 `plugins/` 目录中。每个包执行一次以下命令：

```
cp <path-to-extension-package> /root/cpaas-install/installer/plugins/
```

在启动安装程序之前，请确认该目录中包含这 8 个必需的包：

```
ls -1 /root/cpaas-install/installer/plugins/
```

请勿将 **Common ACP Extensions** 中的其他包复制到该目录。安装程序只会处理你手动放入 `plugins/` 中的包；它不会从 Customer Portal 下载包。

3 启动安装程序

根据平台镜像源选择对应的命令。安装程序成功启动后，终端会输出 web 控制台访问地址。

等待大约 5 分钟后，你可以使用 PC 上的浏览器访问安装程序提供的 web 控制台。

对于平台内置 Registry：

```
bash setup.sh
```

对于允许匿名拉取的外部 Registry：

```
bash setup.sh --registry "<registry-address>"
```

对于需要身份验证的外部 Registry，请同时提供拉取用户名和密码。`setup.sh` 使用的拉取账号可以与用于上传 payload 的推送账号不同：

```
export REGISTRY_ADDRESS="registry.example.com"
export REGISTRY_USERNAME="<registry-pull-username>"
read -rsp "Registry password: " REGISTRY_PASSWORD
export REGISTRY_PASSWORD
printf '\n'

bash setup.sh \
  --registry "$REGISTRY_ADDRESS" \
  --username "$REGISTRY_USERNAME" \
  --password "$REGISTRY_PASSWORD"

unset REGISTRY_PASSWORD
```

Registry 地址必须采用 `<host-or-IP>[:port]` 这种格式，且不要包含 `http://` 或 `https://`。不要只提供 `--username` 和 `--password` 中的其中一个。

> **警告**
>
> 请确保安装程序所在节点的 IP 地址以及端口 8080 可以正常访问，以便安装程序成功启动后能够顺利访问其提供的 web 控制台。

IP Family

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

如果你计划创建使用单栈网络 IPv6 的 `global` 集群，启动安装程序时必须显式指定 `--ip-family ipv6`。如果不指定该参数，安装程序创建的 `global` 集群将默认支持单栈网络 IPv4 和双栈网络。

4 参数配置

在 Web UI 中配置安装，检查每个值，然后确认安装。有关逐字段说明，请参见 安装程序参数。

验证安装

安装程序报告安装完成后，在继续执行安装后任务之前，请先验证 Core 和所需的 Aligned Extensions。有关检查清单和命令，请参见 验证。

安装程序参数参考

有关详细的字段说明，请参见 安装程序参数。

安装故障排除

有关安装程序卡住、资源失败、镜像拉取失败、缺失制品以及清理的说明，请参见 安装故障排除。

其他资源

- 上传并安装扩展
- 安装程序参数
- 验证
- 安装故障排除
- 下一步

安装程序参数

在安装程序 Web UI 中配置 `global` 集群时，请参考此说明。在提交安装前，请确认网络、访问、镜像源和节点相关决策。

参数	说明
Kubernetes 版本	所有可选版本都经过严格测试，以确保稳定性和兼容性。 建议： 选择最新版本，以获得最佳功能和支持。
集群网络协议	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；网络协议设置后无法更改。
集群端点	输入预先准备好的域名。 如果没有可用域名，请输入预先准备好的 `global VIP`。 默认情况下已禁用 `Self-built VIP`，仅当未提供 LoadBalancer 时才启用它。 使用 `Self-built VIP` 时必须满足以下条件： • 已有可用的 VRID； • 主机网络支持 VRRP 协议； • 所有控制平面节点和 VIP 必须位于同一子网。

	提示：对于功能体验场景中的单节点部署，可以直接输入节点 IP。无需启用 `Self-built VIP` 或准备诸如 `global VIP` 之类的网络资源。
平台访问地址	如果不需要区分 集群端点 和 平台访问地址，请输入与 集群端点 相同的地址。 如果需要区分，例如 `global` 集群仅用于内网访问，而平台需要提供外网访问，请输入预先准备好的域名或 `External IP`。 平台默认使用 HTTPS 访问，不启用 HTTP。如果需要启用 HTTP 访问，请在 高级设置 中启用（不推荐）。 注意：在以下情况下必须输入域名， • 已规划 `global` 集群的灾难恢复方案； • 平台需要支持 IPv6 访问。 提示：如果需要配置更多平台访问地址，可以在下一步的 其他设置 > 其他平台访问地址 中添加。或者，在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。 如果需要使用自定义证书，可以上传现有证书。
镜像仓库	`Platform Deployment` 使用随 `global` 集群部署的 Registry。 `External` 使用客户自主管理的平台镜像仓库。其地址和拉取凭证必须与传递给 `setup.sh` 的值一致，并且该仓库中必须已包含目标版本完整的 Core 和 Extension 负载。
容器网络	集群的默认子网和 Service 网段不能重叠。 使用 Kube-OVN Overlay 网络时，请确保容器网络和主机网络不在同一网段，否则可能引起网络异常。

节点名称	如果选择 `Host Name as Node Name`，请确保所有节点的主机名唯一。
`global` 集群平台节点隔离	仅当计划在 `global` 集群中运行应用工作负载时启用。 启用后： • 可将节点设置为 `Platform Exclusive`，即仅运行平台组件，从而确保平台和应用工作负载隔离； • 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）； • 如果启用了 `Platform Exclusive`，则会强制禁用 `Deployable Applications`，控制平面节点仅运行平台组件； • 如果未启用 `Platform Exclusive`，可以选择是否启用 `Deployable Applications`，从而允许控制平面节点运行应用工作负载。 工作节点： • 如果启用了 `Platform Exclusive`，则会强制禁用 `Deployable Applications`； • 如果未启用 `Platform Exclusive`，则会强制启用 `Deployable Applications`。 使用 Kube-OVN 时，可以通过输入网关名称来指定节点网卡。 如果节点可用性检查失败，请根据页面提示进行调整后重新添加。

查看完这些参数后，请返回 安装 提交安装。

验证

在安装程序完成后、继续执行后续安装任务之前，验证 ACP Core 和所需的 Aligned Extensions。

目录

前提条件

- ACP 安装已完成。
- 你可以访问安装程序显示的平台 Web UI 地址。
- 你可以在安装节点上运行 `kubectl`。

验证 **Web UI** 访问

安装完成后，安装程序会显示平台访问 URL。单击 **Access** 打开平台 Web UI，并验证平台登录页面是否可访问。

验证 **Core** 应用、扩展和 **Pod**

在新的 `global` 集群的控制平面节点上运行以下命令，以确认已安装状态：

```
# All nodes are Ready
kubectl get nodes

# ClusterModule/global reports the base module as healthy
kubectl get clustermodule global -o jsonpath='{.status.phase}'

# No AppRelease is in a failed state
kubectl get apprelease -A

# Core and Aligned modules report their current phase and version
kubectl get moduleinfo -l cpaas.io/cluster-name=global

# No Pod is stuck outside Running or Completed
kubectl get pod --all-namespaces | awk '{if ($4 != "Running" && $4 != "Co
mpleted")print}' | awk -F'[/ ]+' '{if ($3 != $4)print}'
```

当满足以下条件时，安装状态正常：

- 所有 `global` 集群节点均为 `Ready`。
- `ClusterModule/global` 报告健康的阶段。
- 每个 AppRelease 都处于非失败状态。
- 已安装以下应用，并且其对应的 `ModuleInfo` 资源在目标版本上报告为 `Running`：
  - **Alauda Container Platform Web Console**
  - **Alauda Container Platform Web Console Central**
  - **Alauda Container Platform Marketplace Web Console**
  - **Alauda Container Platform Helm Application Catalog**
  - **Alauda Container Platform Observability Essentials**
  - **Alauda Container Platform Essentials**

- **Alauda Container Platform Cluster Essentials**
- **Alauda Container Platform GatewayAPI Plugin**

- `cpaas-system` 中的关键 Pod 处于 `Running` 或 `Completed` 状态。

在 Web Console 中，打开 **Administrator > Marketplace > Cluster Plugins**，并确认所需应用可用，且没有安装失败。

验证平台镜像源

确认 `ProductBase` 记录了预期的平台镜像源：

```
kubectl get productbase base \
  -o jsonpath='{.spec.registry.address}{"\t"}{.spec.registry.external}
{"\n"}'
```

对于外部 registry，地址必须与准备好的 registry 匹配，且第二个值必须为 `true`。

对于经过认证的外部 registry，请确认 pull Secret 存在，但不要输出其内容：

```
kubectl -n cpaas-system get secret global-registry-auth
```

确认 `global` 集群记录了相同的 registry 地址：

```
kubectl -n cpaas-system get cluster global \
  -o jsonpath='{.metadata.annotations.cpaas\.io/registry-address}{"\n"}'
```

`ProductBase` 还包含可选或未安装 Extensions 的 catalog 条目。不要要求 `ProductBase.status.artifacts` 中的每个条目都为 `Ready`；一个无关的 `Absent` 条目本身并不表示安装失败。

确认所需的 Extension catalog 条目存在，并且 Pod 未被镜像拉取阻塞：

```
kubectl get moduleplugins

kubectl get pods --all-namespaces \
  | awk '$4 ~ /ImagePullBackOff|ErrImagePull/'
```

Pod 命令必须没有任何输出。安装验收基于所需应用、其 `ModuleInfo` 和 `AppRelease` 状态，以及其运行中的 Pod。如果某个必需的已安装应用不可用，请按照 Installation Troubleshooting 进行排查，以区分是缺少 manifest，还是 DNS、路由、防火墙、CA 信任、拉取凭证或 registry 可用性问题。

下一步

验证成功后，继续执行 下一步。

安装故障排查

当安装程序没有继续执行、安装资源失败，或者安装后验证报告镜像或制品问题时，请使用此页面。

目录

常见卡住场景及排查位置

在更改环境或创建支持工单之前，请先收集第二列中的信息。

症状	首先查看的位置	你要查找的内容
Web UI 无法加载	安装节点上的 `setup.sh` 和 `nerdctl ps` 的终端输出	`minialauda-control-plane` 容器处于 Running 状态，并且浏览器可以访问端口 `8080`。

症状	首先查看的位置	你要查找的内容
安装程序日志停止前进	`tail -f /var/cpaas/data/installer.log`	上一个已完成的阶段，以及第一个重复或失败的操作。
卡在控制平面预配	`kubectl get machines -A`，以及在一个非 `Ready` 的机器上执行 `kubectl describe`	`Bootstrap` 和 `Infrastructure` 条件，以及 `status.addresses` 中的节点地址。
卡在网络和 Core 附加组件	新集群上的 `kubectl -n kube-system get pods`	Kube-OVN、CoreDNS 和 kube-proxy Pod 处于 `Running` 状态；镜像拉取失败通常表明 registry 路径存在问题。
AppRelease 显示 `Failed`	`kubectl describe apprelease <name> -n <namespace>`	能够标识 chart 或依赖失败的状态条件和最近的 Events。
`ClusterModule/global` 未变为健康状态	`kubectl describe clustermodule global`	能够标识阻塞的 Core 或 Aligned 模块的状态条件。

镜像拉取与制品失败

症状	检查	恢复
Pod 处于 `ImagePullBackOff` 或 `ErrImagePull`	运行 `kubectl describe pod <pod> -n <namespace>`，并从失败节点测试已配置的 registry。	修正报告的 DNS、路由、防火墙、CA 信任、拉取凭证、registry 可用性，或缺少 manifest 的问题。
安装后所需的 Core 或 Aligned 应用不可用	检查对应的 `AppRelease`、`ModuleInfo`、`ModulePlugin` 和 Pod Events。如果包可用性仍然不明确，请检查 `kubectl get productbase base -o yaml` 中的	如果 registry 确认缺少所需的 manifest，请上传匹配的安装版本包。对于授权、CA、DNS、路由或连接错误，请修正报告的 registry 问题，而不

症状	检查	恢复
	匹配条目，以及 registry 中精确的 `repository:tag`。	是重新发布包。忽略可选或未安装 catalog 包中无关的 `Absent` 条目。
恢复后的节点或替换节点无法拉取已安装版本镜像	检查失败的 Pod 和 registry 保留记录。	恢复所引用的 digest，然后在重试恢复之前修正 registry 可用性、备份或保留策略。

对于在安装程序启动之前发生的上传命令失败，请使用 External Platform Image Registry 故障排查。

安装程序清理

安装程序通常会在安装完成后自行移除。如果安装完成 30 分钟后安装程序容器仍然存在，请在安装节点上运行以下命令：

```
nerdctl rm -f minialauda-control-plane
```

升级前收集证据

请收集 `/var/cpaas/data/installer.log`、相关的 `kubectl describe` 输出、失败资源的条件，以及准确的 Core Package 版本和架构。不要包含 registry 密码或 Secret 内容。

后续步骤

在 Core 和必需的 Aligned Extensions 通过验证后，请根据你的场景完成以下安装后任务。

优先级	任务	适用场景	继续执行
建议立即执行	安装 Product Docs 插件。	安装此插件后，控制台内的帮助链接可以打开产品文档。	安装 Product Docs 插件
有条件	创建 workload 集群或连接现有集群。	当你的架构要求 workload 集群与 `global` 集群分离时，使用此路径。	Clusters
有条件	上传并安装其他 Extensions Packages。	当你需要来自 **Common ACP Extensions** 的、未包含在八个必需安装包中的包，或者需要其他平台能力时，使用此路径。	Extend
建议在大规模用户入驻前执行	配置身份提供程序、用户、角色、项目以及项目成员关系。	在入驻管理员、项目用户或应用团队之前，使用此路径。	Users and Roles 和 Projects
建议在应用 workload 之前执行	准备存储、网络、registry 访问以及 workload 前置条件。	在应用团队部署 workload 之前，使用此路径。	Storage、Networking、Registry 和 Clusters
建议在生产运行前执行	查看备份、升级、监控和运维文档。	在将平台用于生产 workload 之前，使用此路径。	Backup、Upgrade 和 Observability

这些任务并不适用于每次安装。请按照与你的部署架构和运维要求相匹配的路径执行。

目录

安装 **Product Docs** 插件

灵雀云容器平台 产品文档 插件提供了在平台内访问产品文档的能力。平台中的所有帮助链接都会将用户引导到这些文档。如果未安装此插件，点击平台中的帮助链接会导致 404 访问错误。

1 打开 **Cluster Plugins**

导航到 **Administrator**。在左侧边栏中，点击 **Marketplace** > **Cluster Plugins**，然后选择 `global` 集群。

2 安装插件

找到 灵雀云容器平台 产品文档 插件，然后点击 **Install**。