DaoCloud Enterprise

DaoCloud Enterprise 5.0 · 从零到可用

社区版

用户手册

文档版本 v1.0

内容来源 docs.daocloud.io/dce/license0

docs.daocloud.io/install/community

整理日期 2026 年 10 月 27 日

共 7 章 • 27 节

道客 DaoCloud

关于本手册

这本手册回答一个问题：怎样把 DCE 5.0 社区版从零跑起来，以及跑起来之后该点哪里。

官网上的素材是分散的——「申请社区免费体验」在 DCE 介绍里，安装又在「安装 DCE › 安装社区版」里， 中间还夹着资源规划、Kind、从零建 K8s、在线安装、离线安装五篇。真正动手的人往往一周内要把这些都走一遍， 因此本书把它们收成一条连续的路径：认清社区版是什么（第 1 章）→ 选路线、算资源、装依赖（第 2 章）→ 备一个集群（第 3 章）→ 安装 DCE（第 4 章）→ 申请并激活许可证（第 5 章）→ 用三个模块（第 6 章）， 第 7 章把常踩的坑与求助渠道收在一处。

三条阅读路线

你的情况	建议读	大概耗时
只想看看社区版长什么样	第 1 章 → 2.1 → 3.1 → 4.1 → 第 5 章	半天
手上已有一套 K8s 集群	第 1 章 → 2.2、2.3 → 4.2 或 4.3 → 第 5 章 → 第 6 章	一天
从三台裸机开始搭	全书顺序读，第 3、4 章是重点	两三天

几点约定

- **命令框**：以 `bash` / `shell` 标注的代码块可以直接复制执行；行内 等宽字体 是命令、字段或文件名。
- **界面文字**：加粗的「**立即激活**」表示界面上要点到的按钮、菜单或字段，照字面找即可。
- **提示框**：「说明」是补充信息，「注意」是容易出错的地方，「警告」是会引发故障的操作。
- **交叉引用**：指向本书内其他章节的链接已写成「见 X.Y 节」，可以按编号检索；指向站外的链接保留为可点击地址。
- **版本口径**：正文命令统一按安装器 **v0.45.0** 书写。官网各页示例版本不一（v0.28.0、v0.29.0 均有）， 实际执行时把 `VERSION` 换成你要装的版本即可；版本号变了，步骤不会变。

内容边界

第 6 章对容器管理、全局管理、可观测性只做速览——讲清每个模块管什么、第一次进去该点哪几处、 细节去哪里查。这三个模块的完整用法各自成书，本书不重复。安装与授权部分的每一个步骤都取自官方文档， 截图同样来自官方文档，实际界面可能随版本略有差异。

内容版权归 DaoCloud 所有，本手册仅用于产品学习、交付与培训参考。

目 录

第 1 章 认识 DCE 社区版 5

1.1 什么是 DCE 5

1.2 社区版包含的三个模块 5

1.3 一条完整的上手路径 6

1.4 版本口径与名词约定 7

第 2 章 装之前：路线、资源与依赖 9

2.1 三条安装路线怎么选 9

2.2 集群资源规划 10

2.3 安装依赖项 11

第 3 章 先有一个集群 13

3.1 用 Kind 起一个尝鲜集群 13

3.2 从零搭一套标准 K8s 集群 15

3.3 集群就绪自检 20

第 4 章 安装 DCE 社区版 22

4.1 在 Kind 集群上安装 22

4.2 在已有 K8s 集群上在线安装 23

4.3 在已有 K8s 集群上离线安装 26

4.4 自建集群的完整安装实例 28

4.5 安装参数速查与结果核对 30

第 5 章 申请与激活许可证 32

5.1 装完之后为什么还要授权 32

5.2 三步拿到授权 33

5.3 许可证续期 34

5.4 授权环节的常见疑问 36

第 6 章 装好之后：三个模块速览 37

6.1 三个模块的分工 37

6.2 全局管理 37

6.3 容器管理 38

6.4 可观测性 39

6.5 串起来：上线一个应用的三阶段 40

第 7 章　常见问题与排障 42

7.1　安装阶段 42

7.2　授权与访问阶段 42

7.3　拿到帮助 43

附录 A　安装前检查清单 44

附录 B　命令速查 45

附录 C　官方文档索引 46

第 1 章　认识 DCE 社区版

在动手装任何东西之前，先弄清三件事：DCE 是什么，社区版给了你哪几块能力，以及从一台机器到能用要经过几段路。 这一章只建立地图，不涉及操作；操作从第 2 章开始。

1.1　什么是 DCE

DaoCloud Enterprise（简称 DCE）是一个以 Kubernetes 为底座的云原生操作系统。官方对它的描述是：「以 K8s 作为开发底座，提供了高度可扩展、强大灵活的各项生产级功能，使得企业能够轻松构建和管理分布式应用。」 换句话说，它不替代 Kubernetes，而是在 Kubernetes 之上补齐企业真正会用到的那一层——多集群怎么管、 应用怎么发、权限怎么分、出问题怎么查。

DCE 的完整形态由十余个模块拼成：容器管理、全局管理、可观测性负责底座与运维， 镜像仓库、中间件、微服务引擎、服务网格、多云编排、应用工作台、云原生网络、云原生存储、虚拟机、云边协同、设备管理负责各类场景能力。企业按需选购、自由组合。

社区版是这套产品的最小可用集合：它只包含三个模块——全局管理、容器管理、可观测性。 三者恰好构成一条完整的闭环：谁来用、跑在哪、跑得怎么样。所以社区版虽然模块少， 却不是「只能看不能用」的试用版：装完之后你手上是一个能建集群、能发应用、能看监控、能配告警的完整平台。

> **说明**
>
> 本章出现的模块名称与界面菜单一致。第 6 章会带你把三个模块各走一遍，那里只讲「管什么、先点哪」， 具体操作请查各模块的官方文档。

1.2　社区版包含的三个模块

三个模块不是三个并列的功能菜单，而是三种视角：全局管理看「人与边界」，容器管理看「算力与应用」， 可观测性看「运行状态」。同一份资源，在三个模块里呈现的样子不同。

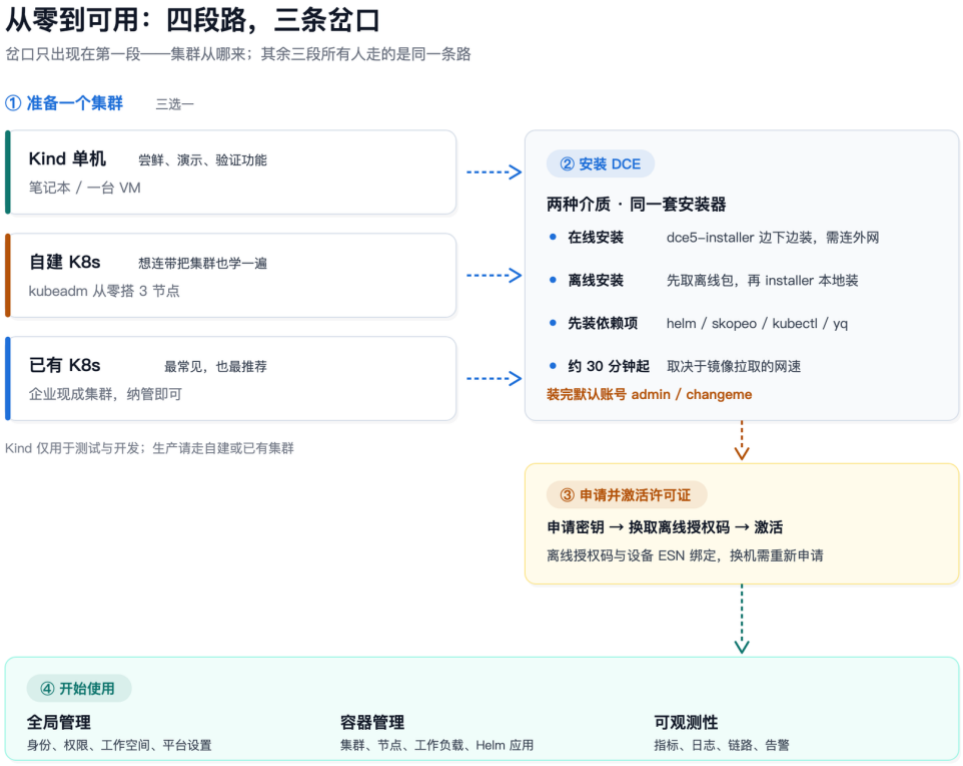
模块	官方定位	你会在里面做什么	详见
全局管理	以用户为中心的综合性服务板块，包含用户与访问控制、工作空间与层级、审计日志、平台设置等基础服务	建用户、划工作空间、查审计日志、改平台外观、做正版授权	6.2
容器管理	基于 Kubernetes 构建的面向云原生应用的容器化管理模块	建或接入集群、管节点与命名空间、发工作负载、装 Helm 应用、配	6.3

模块	官方定位	你会在里面做什么	详见
		网络与存储	
可观测性	以应用为中心、开箱即用的新一代云原生可观测性平台	看指标仪表盘、查日志、追链路、配告警规则与通知	6.4

三个模块之间的边界值得早点记住：**权限在全局管理里给，资源在容器管理里建，状态在可观测性里看**。 误把权限当成集群配置、或者到容器管理里找用户，是新手最常见的两次绕路。

1.3 一条完整的上手路径

从一台裸机到能用，要经过四段路。第一段是唯一有岔口的地方——集群从哪来；其余三段所有人走的是同一条路。



阶段	你要做的事	产物	对应章节
① 准备集群	起一个 Kind 集群，或自建 / 纳管一套标准 K8s	一个 Ready 的集群，带默认 StorageClass	第 3 章
② 安装 DCE	装依赖项，下载 dce5-installer，执行安装	能打开的 DCE 控制台	第 4 章
③ 申请授权	申请许可证密钥，换离线授权码，在平台上激活	平台显示授权已激活	第 5 章
④ 开始使用	建用户与工作空间，部署应用，配观测	一套能跑业务的平台	第 6 章

这里有个顺序上的细节容易被误解：**许可证是在装完之后申请的，不是装之前买好的**。 原因是离线授权码要与设备的 ESN 绑定，而 ESN 只能从已经装好的 DCE 里读出来（获取路径见 5.2.2）。所以流程是「先装、后授权」，装完先用默认账号登进去把 ESN 抄下来。

1.4 版本口径与名词约定

本书涉及四类版本号，容易混淆，先列清楚：

对象	本书取值	说明
DCE 安装器（dce5-installer）	v0.45.0	命令里的 `VERSION`，决定装哪个版本
依赖项脚本	v0.45.0	与安装器同版本号，脚本名 `install_prerequisite_v0.45.0.sh`
Kubernetes	最低 v1.22，推荐 v1.29.5	装 DCE 的载体集群版本
Kind 示例版本	kind v0.17.0、节点镜像 v1.25.3	仅第 3.1 节的尝鲜集群用得到

装 DCE 之前还会反复遇到几个名词，先给出一句话定义：

名词	含义
DCE	DaoCloud Enterprise 云原生操作系统的简称
社区版	只含全局管理、容器管理、可观测性三个模块的免费版本
dce5-installer	DCE 的安装器二进制，`install-app` 子命令负责部署
ESN	Electronic Serial Number，设备唯一标识，通常由 8 个字符组成，与离线授权码绑定
离线授权码	用许可证密钥 + ESN 换回的字符串，粘贴到平台即可激活
全局服务集群	DCE 自身组件运行的那个集群，也是控制台入口所在的集群
StorageClass	存储类，DCE 需要一套默认 SC 才能给组件分配存储卷
NodePort	把服务暴露到节点端口的方式，用于没有负载均衡器时访问控制台
metallb	裸金属环境下的负载均衡实现，可给控制台和 Insight 各分配一个 VIP

> **提示**
>
> 只需记住一句：**ESN 是机器，授权码是钥匙，许可证密钥是凭证**。三者缺一不可， 而且第一个换了，后两个都得重来。

第 2 章　装之前：路线、资源与依赖

选错路线是社区版安装最常见的返工原因：用 Kind 给生产环境做 PoC，或用在线安装去装一台不能出网的机器， 都会在第二步卡住。这一章先用三个问题把路线定下来，再把资源账算清，最后把依赖项备好。

2.1　三条安装路线怎么选

官方给出的路线其实是「两条轴」的组合：集群从哪来（Kind 自建 / 已有集群），介质怎么来（在线 / 离线）。 组合之后有四种走法：

路线	前置条件	适用场景	主要限制	详见
Kind 集群 + 在线安装	一台 8 核以上、12 GB 内存以上、100 GB 磁盘以上的机器，能连公网，装 Docker	尝鲜、演示、功能验证、培训	只能用 NodePort 访问；官方明确「仅推荐用于测试和开发」	3.1 → 4.1
已有 K8s + 在线安装	一套 3 Master + 2 Worker 的 K8s（v1.22 以上），有默认 SC 和 CoreDNS，能连公网	企业现成集群，快速上线	全程需要外网拉镜像	4.2
已有 K8s + 离线安装	同上，外加一台能上网的机器、离线包、一个可用的镜像仓库	生产、内网、隔离环境	步骤最多，需要自备 registry	4.3
从零自建 K8s + 安装	三台 8 核 16 GB 的机器（云主机或物理机）	想把集群一起学透，或没有现成集群	耗时最长，涉及 kubeadm 全套	3.2 → 4.4

选路线只需回答三个问题：

1. **手上有没有一套能用的 Kubernetes 集群？** 有 → 跳过第 3 章，直接看 4.2 或 4.3。
2. **装 DCE 的那台机器（或集群）能不能连公网？** 能 → 在线安装；不能 → 离线安装。
3. **这套环境是长期用还是只想看看？** 只想看看 → Kind 最省事；长期用 → 至少是自建集群，别用 Kind。

> **警告**
>
> Kind 集群跑在容器里，节点即容器，重启即重建。它适合验证功能、录演示、做培训， 但不要在上面跑任何有价值的业务数据。生产环境请用标准 K8s 集群。

2.2 集群资源规划

位置：安装 DCE › 安装社区版 › 资源规划

官方的资源要求分两张表：标准 Kubernetes 集群一张，Kind 集群一张。两者门槛不同， 目的也不同——前者要撑住生产，后者只求能跑起来。 您可以在标准 Kubernetes 集群上安装 DCE，也可以在 kind 集群上安装 DCE。

标准 Kubernetes 集群资源的要求如下：

资源项	要求
节点硬件	**3 Master + 2 Worker** Master 节点：4 Core，8 G；系统盘 100G Worker 节点：4 Core，8 G；系统盘 100G
K8s 版本	推荐 K8s 官方最高稳定版本，目前推荐 v1.29.5，最低支持 v1.22
支持的 CRI	Docker 和 containerd
CPU 架构	x86_64 和 ARM64

kind 集群资源的要求如下，仅推荐用于测试和开发。

资源项	要求
节点硬件	CPU > 10 核、内存 > 12 GB、磁盘空间 > 100 GB
K8s 版本	推荐 K8s 官方最高稳定版本，目前推荐 v1.29.5，最低支持 v1.22
支持的 CRI	Docker 和 containerd
CPU 架构	x86_64 和 ARM64

标准集群与 Kind 集群的差别，主要不在数字而在用途：标准集群是生产形态，三台 Master 保证控制面可用， 两台 Worker 承载业务；Kind 集群把节点跑在一个容器里，资源门槛低，但没有任何可用性可言。

> **说明**
>
> 两处官方页面对 Kind 集群的 CPU 要求写法不同：资源规划页写「CPU > 10 核」，Kind 安装页的自检脚本 按「8 核」判定。建议按 10 核准备——CPU 不足时 kind 起得来，但 DCE 组件会因为调度不上去而一直 Pending。

2.3 安装依赖项

DCE 的安装器本身不重，但它依赖一组命令行工具：装镜像、推 chart、解析 YAML、创建 kind 集群都靠它们。 安装脚本会检查这些工具，发现缺失或版本过低就自动强制更新替换。

安装位置有讲究：**社区版请在 K8s Master 节点上安装依赖项**；只有商业版才在独立的火种节点上装。

依赖清单与版本门槛

工具	用途	版本门槛
helm	渲染与安装 chart	≥ 3.11.1
skopeo	镜像搬运与复制	≥ 1.11.1
kubectl	与集群交互	≥ 1.25.6
yq	解析 YAML	≥ 4.31.1
podman	容器工具（离线包搬运）	脚本统一安装
kind	创建 Kind 集群	脚本统一安装
minio client（mc）	对象存储操作	脚本统一安装
charts-syncer	chart 同步	脚本统一安装

上表中前四个是 DCE 安装时真正会校验版本的，后四个由脚本按需装上；如果你的环境里已经有，脚本会比对版本，低了就替换成它自带的版本。

在线安装依赖项

在能连公网的 Master 节点上执行：

```
export VERSION=v0.45.0
curl -LO https://qiniu-download-
public.daocloud.io/DaoCloud_Enterprise/dce5/install_prerequisite_${VERSION}.sh
chmod +x install_prerequisite_${VERSION}.sh
bash install_prerequisite_${VERSION}.sh online community
```

最后那个 `community` 参数别写错——写 `full` 会按商业版的清单装，多出一批用不上的组件。

离线安装依赖项

离线场景要先在能上网的机器上把依赖包取下来，再连同脚本一起搬到集群节点：

```
# 在能上网的机器上
export VERSION=v0.45.0
```

```
curl -LO https://qiniu-download-
public.daocloud.io/DaoCloud_Enterprise/dce5/install_prerequisite_${VERSION}.sh
curl -LO https://qiniu-download-
public.daocloud.io/DaoCloud_Enterprise/dce5/prerequisite_${VERSION}_amd64.tar.gz
# 搬到 Master 节点后
export VERSION=v0.45.0
export BINARY_TAR=prerequisite_${VERSION}_amd64.tar.gz
chmod +x install_prerequisite_${VERSION}.sh
./install_prerequisite_${VERSION}.sh offline community
```

说明

- ARM 架构请把包名换成 `prerequisite_${VERSION}_arm64.tar.gz`。
- **离线包与脚本必须在同一个目录层级**，否则脚本找不到包。
- 离线安装的依赖项同样要装在 K8s Master 节点上。

依赖项就绪后，集群侧还剩三件事要确认，写在 3.3 的自检表里。

第 3 章　先有一个集群

社区版必须装在一个 Kubernetes 集群上。这一章给出两条得到集群的路：一条是十几分钟就能起来的 Kind 集群， 另一条是从裸机开始、用 kubeadm 搭出来的三节点标准集群。两条路的产物不同，请对照 2.1 的选择结果取用。

3.1　用 Kind 起一个尝鲜集群

位置：安装 DCE › 安装社区版 › 从零部署 Kind 集群

Kind 把 Kubernetes 的节点跑在容器里，一条命令就能起一个可用集群，是试水 DCE 最省事的方式。本节从一台干净的机器开始，装 Docker、装 kind、起集群；集群就绪后转到 4.1 安装 DCE。

> **说明**
>
> 点击在线安装社区版可观看视频演示。

环境自检

- 准备一台机器，机器资源建议：CPU > 8 核、内存 > 12 GB、磁盘空间 > 100 GB。
- 确保机器能够连通互联网。

执行如下脚本，检查系统资源和联网情况：

```
set -e
if [ $(free -g|grep Mem | awk '{print $2}')              -lt 12 ]; then (echo
"insufficient memory! (should >=12G)";); fi
if [ $(grep 'processor' /proc/cpuinfo |sort |uniq |wc -l) -lt 8 ]; then (echo
"insufficient CPU! (should >=8C)";); fi
if [ $(df -m / |tail -n 1 | awk '{print $4}') -lt 30720 ]; then (echo "insufficient
free disk space of root partition!(should >=30G)";); fi
ping daocloud.io -c 1 &> /dev/null || ( echo "no connection to internet! abort.")
echo "precheck pass.."
```

预期输出类似于：

```
precheck pass..
```

安装 Docker

> **说明**
>
> 如果您的机器已安装了 Docker 且版本高于 1.18，请跳过这一步。
>
> 装 Docker 时可使用国内源：https://developer.aliyun.com/mirror/docker-ce

如果是 CentOS

```
set -e
if  [ -x "$(command -v docker )" ] ;then
    echo "docker already installed : version = "$(docker -v);
else
```

```
    echo "docker not found, please install it first."
fi

sudo yum install -y yum-utils device-mapper-persistent-data lvm2
sudo yum-config-manager --add-repo https://mirrors.aliyun.com/docker-
ce/linux/centos/docker-ce.repo
sudo sed -i 's+download.docker.com+mirrors.aliyun.com/docker-ce+'
/etc/yum.repos.d/docker-ce.repo
sudo yum makecache fast
sudo yum -y install docker-ce
sudo service docker start
sudo systemctl enable docker
sudo yum install -y wget
```

如果是 Ubuntu

```
set -e
if  [ -x "$(command -v docker )" ] ;then
    echo "docker already installed : version = "$(docker -v);
else
    echo "docker not found, please install it first."
fi
sudo apt-get update
sudo apt-get -y install apt-transport-https ca-certificates curl software-
properties-common
curl -fsSL https://mirrors.aliyun.com/docker-ce/linux/ubuntu/gpg | sudo apt-key add
-
sudo add-apt-repository --yes "deb [arch=amd64] https://mirrors.aliyun.com/docker-
ce/linux/ubuntu $(lsb_release -cs) stable"
sudo apt-get -y update
sudo apt-get -y install docker-ce
sudo apt-get -y install wget
sudo service docker start
sudo systemctl enable docker
```

请注意，如果机器上已有 Podman，但没有 Docker，依然需要安装 Docker。 这是因为一个已知问题：Podman 虽然可以启动 kind，但是会出现文件句柄不足的问题，并且重启会出现 IP 对不上的问题。

安装 kind 并创建集群

1. 下载 kind 的二进制文件包。

   ```
   curl -Lo ./kind https://qiniu-download-public.daocloud.io/kind/v0.17.0/kind-
   linux-amd64
   chmod +x ./kind
   old_kind=$(which kind)
   if [ -f "$old_kind" ]; then mv ./kind $old_kind; else mv ./kind /usr/bin/kind ;
   fi
   ```

2. 查看 kind 版本。

   ```
   kind version
   ```

   预期输出如下：

   ```
   kind v0.17.0 go1.19.2 linux/amd64
   ```

3. 设置 `kind_cluster.yaml` 配置文件。

注意，暴露集群内的 32088 端口到 kind 对外的 8888 端口（可自行修改），配置文件示例如下：

```
apiVersion: kind.x-k8s.io/v1alpha4
kind: Cluster
nodes:
- role: control-plane
  extraPortMappings:
  - containerPort: 32088
    hostPort: 8888
```

4. 创建一个名为 `fire-kind-cluster` 的 K8s v1.25.3 示例集群。

```
kind create cluster --image release.daocloud.io/kpanda/kindest-node:v1.25.3 --
name=fire-kind-cluster --config=kind_cluster.yaml
```

预期输出如下：

```
Creating cluster "fire-kind-cluster" ...
 ✓ Ensuring node image (release.daocloud.io/kpanda/kindest-node:v1.25.3) 🖼
 ✓ Preparing nodes 📦
 ✓ Writing configuration 📜
 ✓ Starting control-plane 🕹️
 ✓ Installing CNI 🔌
 ✓ Installing StorageClass 💾
Set kubectl context to "kind-fire-kind-cluster"
```

5. 查看新创建的集群。

```
kind get clusters
```

预期输出如下：

```
fire-kind-cluster
```

3.2 从零搭一套标准 K8s 集群

位置：安装 DCE › 安装社区版 › 从零部署 K8s 集群

这一节是一份完整的实录：三台机器从裸机开始，装容器运行时、kubeadm、CNI、存储，最后装 DCE，连网络与存储的选择也一并交代。集群已经有的读者可以跳过本章，直接看第 4 章；想一次把集群也学透的，就按下面的顺序走。

> **说明**
>
> 现阶段版本迭代较快，本文的安装方式可能与最新版有所差异，请以产品文档的安装说明为准。

集群规划

使用 3 台 UCloud 的 VM，配置均为 8 核 16G。

角色	主机名	操作系统	IP	配置
control-plane	k8s-master01	CentOS 8.3	10.23.*	8 核 16G 系统盘 200GB

角色	主机名	操作系统	IP	配置
worker-node	k8s-work01	CentOS 8.3	10.23.*	8 核 16G 系统盘 200GB
worker-node	k8s-work02	CentOS 8.3	10.23.*	8 核 16G 系统盘 200GB

本示例采用的组件为：

- Kubernetes：1.31.6
- CRI：containerd（因为新版本 K8s 已经不再直接支持 Docker）
- CNI：Calico
- StorageClass：local-path
- DCE 社区版：v0.36.0

准备节点

本节所述的所有操作都是必需的。

节点配置

安装前先分别对 3 个节点做了一些必要的设置。

1. 配置主机名。修改主机名（可选），避免主机名重名即可。

```
hostnamectl set-hostname k8s-master01
hostnamectl set-hostname k8s-work01
hostnamectl set-hostname k8s-work02
```

建议修改后 exit 退出 SSH 会话，重新登录以显示新的主机名。

2. 禁用 Swap

```
swapoff -a
sed -i '/ swap / s/^/#/' /etc/fstab
```

3. 关闭防火墙（可选）

```
systemctl stop firewalld
systemctl disable firewalld
```

4. 设置内核参数并允许 iptables 进行桥接流量

加载 `br_netfilter` 模块：

```
cat <<EOF | tee /etc/modules-load.d/kubernetes.conf
br_netfilter
EOF

# 加载模块
sudo modprobe overlay
sudo modprobe br_netfilter
```

修改内核参数如 `ip_forward` 和 `bridge-nf-call-iptables`：

```
cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-iptables  = 1
```

```
net.bridge.bridge-nf-call-ip6tables = 1
net.ipv4.ip_forward                 = 1
EOF

# 刷新配置
sysctl --system
```

安装容器运行时（containerd）

1. 如果是 CentOS 8.x，要先卸载系统预装的 Podman，否则会版本冲突

```
yum erase podman buildah -y
```

2. 安装依赖

```
sudo cd /etc/yum.repos.d/
sudo mkdir bak
sudo mv CentOS-*.repo ./bak
sudo curl -o CentOS-base.repo http://mirrors.aliyun.com/repo/Centos-8.repo
sudo yum clean all
sudo yum install -y yum-utils device-mapper-persistent-data lvm2
```

3. 安装 containerd，可以用二进制也可以用 yum 包（yum 是 docker 社区维护的，本例中使用 yum 包）

```
sudo yum-config-manager --add-repo http://mirrors.aliyun.com/docker-
ce/linux/centos/docker-ce.repo
sudo yum makecache
yum install containerd.io -y
ctr -v  # (1)
```

注 1 显示安装的版本，例如 ctr containerd.io 1.6.20

1. 修改 containerd 的配置文件

```
# 删除自带的 config.toml，避免后续 kubeadm 出现错误 CRI v1 runtime API is not
implemented for endpoint
mv /etc/containerd/config.toml /etc/containerd/config.toml.old

# 重新初始化配置
sudo containerd config default | sudo tee /etc/containerd/config.toml

# 更新配置文件内容: 使用 systemd 作为 cgroup 驱动，并且替代 pause 镜像地址
sed -i 's/SystemdCgroup\ =\ false/SystemdCgroup\ =\ true/'
/etc/containerd/config.toml
sed -i 's/k8s.gcr.io\/pause/k8s-gcr.m.daocloud.io\/pause/g'
/etc/containerd/config.toml # (1)
sed -i 's/registry.k8s.io\/pause/k8s-gcr.m.daocloud.io\/pause/g'
/etc/containerd/config.toml
sudo systemctl daemon-reload
sudo systemctl restart containerd
sudo systemctl enable containerd
```

注 1 老的 pause 地址

1. 安装 CNI（可选）

```
curl -JLO
https://github.com/containernetworking/plugins/releases/download/v1.2.0/cni-
plugins-linux-amd64-v1.2.0.tgz
mkdir -p /opt/cni/bin &&  tar Cxzvf /opt/cni/bin cni-plugins-linux-amd64-
v1.2.0.tgz
```

2. 安装 nerdctl（可选）

```
curl -LO
https://github.com/containerd/nerdctl/releases/download/v1.2.1/nerdctl-1.2.1-
linux-amd64.tar.gz
tar xzvf nerdctl-1.2.1-linux-amd64.tar.gz
mv nerdctl /usr/local/bin
nerdctl -n k8s.io ps # (1)
```

注 1 查看容器

安装 k8s 集群

安装 k8s 二进制组件

在三个节点上都需要如下操作：

1. 安装 Kubernetes 软件源 (这里采用国内阿里云的源加速)

```
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://mirrors.aliyun.com/kubernetes/yum/repos/kubernetes-el7-x86_64/
enabled=1
gpgcheck=1
repo_gpgcheck=1
gpgkey=https://mirrors.aliyun.com/kubernetes/yum/doc/yum-key.gpg
https://mirrors.aliyun.com/kubernetes/yum/doc/rpm-package-key.gpg
EOF
```

2. 将 SELinux 设置为 permissive 模式（相当于将其禁用）

```
sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
```

3. 安装 Kubernetes 组件，版本以 1.25.8 为例

```
export K8sVersion=1.25.8
sudo yum install -y kubelet-$K8sVersion
sudo yum install -y kubeadm-$K8sVersion
sudo yum install -y kubectl-$K8sVersion
sudo systemctl enable --now kubelet
```

用 kubeadm 初始化第一个 master 节点

1. 预先下载镜像以加速安装，使用 DaoCloud 的加速仓库

```
# 指定 K8s 版本，并拉取镜像
kubeadm config images pull --image-repository k8s-gcr.m.daocloud.io --
kubernetes-version=v1.25.8
```

2. 调用 kubeadm 初始化第一个节点（使用 DaoCloud 加速仓库）

> **说明**
>
> 如下 Pod CIDR 不能与宿主机物理网络的网段重合（该 CIDR 未来还需要跟 Calico 的配置一致）。

```
sudo kubeadm init --kubernetes-version=v1.25.8 --image-repository=k8s-
gcr.m.daocloud.io --pod-network-cidr=192.168.0.0/16
```

经过十几分钟，你能看到打印成功的信息如下（请记住最后打印出的 `kubeadm join` 命令和相应 token，后续会用到 ）

```
Your Kubernetes control-plane has initialized successfully!
To start using your cluster, you need to run the following as a regular user:

mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

Alternatively, if you are the root user, you can run:
export KUBECONFIG=/etc/kubernetes/admin.conf
You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as
root:

kubeadm join 10.23.207.16:6443 --token p4vw62.shjjzm1ce3fza6q7 \
--discovery-token-ca-cert-hash
sha256:cb1946b96502cbd2826c52959d0400b6e214e06cc8462cdd13c1cb1dc6aa8155
```

1. 配置 kubeconfig 文件，以便用 kubectl 更方便管理集群

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
kubectl get no # (1)
```

注 1 你能看到第一个节点，但是仍然 NotReady

1. 安装 CNI，以 Calico 为例

> **说明**
>
> 在安装 Calico 前，请先创建 `calico-system` 命名空间。可参考创建命名空间。

请以官方安装方案为准。参考官方 Calico 安装文档。

1. 下载 Calico 的部署清单文件:

```
wget
https://raw.githubusercontent.com/projectcalico/calico/v3.26.1/manifests/calic
o.yaml
```

2. 使用以下命令加速镜像拉取:

```
sed -i 's?docker.io?docker.m.daocloud.io?g' calico.yaml
```

3. 使用以下命令安装 Calico:

```
kubectl apply -f calico.yaml
```

4. 等待部署成功

```
kubectl get po -n calico-system -w # (1)
kubectl get no # (2)
```

注 1 等待 Pod 都 Running

注 2 可以看到第一个节点变为 ready 状态了

接入其他 worker 节点

最后在其他 worker 节点执行 join 命令。 在上述 master 节点执行 `kubeadm init` 时最后会在屏幕打出（注意三个参数都是跟环境相关的，请勿直接拷贝）

```
kubeadm join $第一台 master 的 IP:6443 --token p...7 --discovery-token-ca-cert-hash
s....x
```

成功 join 之后，输出类似于：

```
This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

在 master 节点确认节点都被接入，并且等待其都变为 Ready 状态。

```
kubectl get no -w
```

安装默认存储 CSI（本地存储）

```
# 参考:  https://github.com/rancher/local-path-provisioner
wget https://raw.githubusercontent.com/rancher/local-path-
provisioner/v0.0.24/deploy/local-path-storage.yaml
sed -i "s/image: rancher/image: docker.m.daocloud.io\/rancher/g" local-path-
storage.yaml # (1)
sed -i "s/image: busybox/image: docker.m.daocloud.io\/busybox/g" local-path-
storage.yaml
kubectl apply -f local-path-storage.yaml
kubectl get po -n local-path-storage -w # (2)

# 把 local-path 设置为默认 SC
kubectl patch storageclass local-path -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
kubectl get sc # (3)
```

注 1　替换 docker.io 为实际镜像

注 2　等待 Pod 都 running

注 3　可以看到形如: local-path (default)

3.3　集群就绪自检

无论集群是自己搭的还是现成的，装 DCE 之前请把下面六项逐条过一遍。这六项就是 4.2、4.3 里「准备工作」反复强调的内容，提前查完能省掉一次失败的安装。

#	检查项	命令	期望结果
1	节点全部就绪	`kubectl get no`	所有节点 `STATUS` 为 `Ready`
2	CoreDNS 运行	`kubectl get po -n kube-system -l k8s-app=kube-dns`	Pod 全部 `Running`
3	有默认 StorageClass	`kubectl get sc`	至少一个名字带 `(default)`
4	单节点集群已去污点	`kubectl`	没有 `node-`

#	检查项	命令	期望结果
		`describe no <节点名> | grep Taints`	`role.kubernetes.io/control-plane:NoSchedule`
5	依赖项版本达标	`helm version` / `skopeo -v` / `kubectl version` / `yq --version`	达到 2.3 表中的门槛
6	控制台端口可达	`curl -k https://<节点IP>:32088`	能从浏览器所在网段访问到该端口

第 6 项容易被忽略：DCE 控制台通过 NodePort 或负载均衡 VIP 暴露，如果目标机器在公有云上，安全组里没放行 32088，会出现「装成功了但打不开」的情况。4.4 会专门讲这一步。

> **提示**
>
> 自检通过后就可以安装了。Kind 集群走 4.1，已有集群走 4.2（在线）或 4.3（离线）， 自建集群走 4.4。四节里的安装器是同一个，差别只在参数与介质。

第 4 章　安装 DCE 社区版

四节对应四种走法，但底层是同一个程序：`dce5-installer`。它的 `install-app` 子命令负责把 DCE 的组件 部署进集群，区别只在于参数——要不要指定配置文件、要不要指定访问地址、要不要最小化安装。 先看一遍 4.5 的参数速查表，再读对应的小节，会顺一些。

4.1　在 Kind 集群上安装

安装步骤

1. 安装依赖项（见 2.3 节）。

> **说明**
>
> 如果集群中已安装所有依赖项，请确保依赖项版本符合要求：

- helm ≥ 3.11.1
- skopeo ≥ 1.11.1
- kubectl ≥ 1.25.6
- yq ≥ 4.31.1

1. 在 kind 主机下载 dce5-installer 二进制文件。

   假定 VERSION 为 v0.45.0

```
export VERSION=v0.45.0
curl -Lo ./dce5-installer https://qiniu-download-
public.daocloud.io/DaoCloud_Enterprise/dce5/dce5-installer-$VERSION
chmod +x ./dce5-installer
```

2. 获取 kind 所在主机的 IP，然后开始安装 DCE。

```
myIP=$(ip -o route get 1.1.1.1 | cut -d " " -f 7)
./dce5-installer install-app -z -k $myIP:8888
```

> **说明**
>
> kind 集群仅支持 NodePort 模式。 安装过程持续 30 分钟以上，具体取决于镜像拉取的网速。 可通过以下命令观察安装过程中的 Pod 启动情况：

```
docker exec -it fire-kind-cluster-control-plane kubectl get po -A -w
```

1. 安装完成后，命令行会提示安装成功。恭喜您！ 现在可以通过屏幕提示的 URL 使用 **默认的账号和密码（admin/changeme）** 探索全新的 DCE 啦！

完成

- 请记录好提示的 URL，方便下次访问。
- 成功安装 DCE 社区版后，请申请社区免费体验（见 5.2 节）。
- 如果安装过程中遇到什么问题，欢迎扫描二维码，与开发者畅快交流：

4.2 在已有 K8s 集群上在线安装

位置：安装 DCE › 安装社区版 › 在线安装到 K8s 集群

已经有一套 Kubernetes 集群时，装 DCE 就是往集群里部署一批组件。在线安装从公网仓库边拉边装，前置条件只有三样：集群本身、一套默认 StorageClass、能连外网。

说明

- 点击在线安装社区版可观看视频演示。
- 如果需要离线安装，请查阅离线安装步骤（见 4.3 节）。

准备工作

- 准备一个 K8s 集群，集群配置请参考文档集群资源规划（见 2.2 节）。
  - 提前准备好 StorageClass，并设置为默认 SC
  - 确保集群已安装 CoreDNS
  - 如果是单节点集群，确保已移除该节点的污点
- 安装依赖项（见 2.3 节）。

  如果集群中已安装所有依赖项，请确保依赖项版本符合要求：

  - helm ≥ 3.11.1
  - skopeo ≥ 1.11.1
  - kubectl ≥ 1.25.6
  - yq ≥ 4.31.1

下载和安装

1. 在 K8s 集群控制平面节点（Controller Node）下载 dce5-installer 二进制文件（也可以通过浏览器下载）。

   假定 VERSION 为 v0.45.0：

```
export VERSION=v0.45.0

# 如果是 ARM 架构请更新 `dce5-installer-$VERSION` 为 `dce5-installer-$VERSION-linux-arm64`

curl -Lo ./dce5-installer https://qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/dce5-installer-$VERSION
chmod +x ./dce5-installer
```

2. 设置配置文件 `clusterConfig.yaml`

   - 如果使用 NodePort 暴露控制台（仅推荐 PoC 使用），直接执行下一步。
   - 如果是非公有云环境（虚拟机、物理机），请启用负载均衡 (metallb)，以规避 NodePort 因节点 IP 变动造成的不稳定。请仔细规划您的网络，设置 2 个必要的 VIP，配置文件范例如下：

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
spec:
  loadBalancer:
    type: metallb
    istioGatewayVip: 10.6.229.10/32 # (1)
    insightVip: 10.6.229.11/32      # (2)
```

注 1 这是 Istio gateway 的 VIP，也是 DCE 控制台的浏览器访问 IP

注 2 全局服务集群的 Insight-Server 采集所有子集群监控指标的网络路径所用的 VIP

- 如果是公有云环境，并通过预先准备好的 Cloud Controller Manager 的机制提供了公有云的 K8s 负载均衡能力, 配置文件范例如下:

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
spec:
  loadBalancer:
    type: cloudLB
```

3. 安装 DCE。

```
./dce5-installer install-app -c clusterConfig.yaml -z
```

说明

- 有关 clusterConfig.yaml 文件设置，请参考在线安装第 2 步。
- `-z` 最小化安装
- `-c` 指定集群配置文件。使用 NodePort 暴露控制台时不需要指定 `-c`。
- `-d` 开启 debug 模式
- `--serial` 指定后所有安装任务串行执行

4. 安装完成后，命令行会提示安装成功。恭喜您！ 现在可以通过屏幕提示的 URL 使用 **默认的账号和密码（admin/changeme）** 探索全新的 DCE 啦！

完成

请记录好提示的 URL，方便下次访问。

5. 另外，安装 DCE 成功之后，您需要正版授权后使用，请参考申请社区免费体验（见 5.2 节）。

4.3 在已有 K8s 集群上离线安装

位置：安装 DCE › 安装社区版 › 离线安装到 K8s 集群

内网环境拿不到公网镜像，得先在能上网的机器上把离线包整个拉下来，再搬到集群里安装。离线包体量大、步骤多，换来的是可重复、可审计的交付过程。

点击社区版部署 Demo 可观看视频演示。

准备工作

- 准备一台能连通外网的机器
- 准备一个 Kubernetes 集群，集群配置请参考文档资源规划（见 2.2 节）。
  - 存储：需要提前准备好 StorageClass，并设置为默认 SC
  - 确保集群已安装 CoreDNS
  - 如果是单节点集群，请确保您已移除该节点的污点
- 安装依赖项（见 2.3 节）。

  如果集群中已安装所有依赖项，请确保依赖项版本符合要求：

  - helm ≥ 3.11.1
  - skopeo ≥ 1.11.1
  - kubectl ≥ 1.25.6
  - yq ≥ 4.31.1

下载和安装

1. 先找一台能连通外网的机器，运行以下命令下载社区版离线包并解压（也可以从网页下载中心下载离线包并解压）：

   假定版本 VERSION=v0.45.0

```
export VERSION=v0.45.0
wget https://qiniu-download-
public.daocloud.io/DaoCloud_Enterprise/dce5/offline-community-$VERSION-
amd64.tar
tar -xvf offline-community-$VERSION-amd64.tar
```

2. 将解压的文件上传到 K8s 集群控制平面节点（Controller Node），在此节点上设置 clusterConfig.yaml
   - 如果是非公有云环境（虚拟机、物理机），请启用负载均衡 (metallb)，以规避 NodePort 因节点 IP 变动造成的不稳定。请仔细规划您的网络，设置 2 个必要的 VIP，配置文件范例如下：

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
```

```
    spec:
      loadBalancer:
        type: metallb
        istioGatewayVip: 10.6.229.10/32
        insightVip: 10.6.229.11/32
      fullPackagePath: absolute-path-of-the-offline-directory # (1)
      imagesAndCharts:        # (2)
        type: external
        externalImageRepo: your-external-registry # (3)
        # externalImageRepoUsername: admin
        # externalImageRepoPassword: Harbor123456
```

注 1 解压离线包后的路径

注 2 镜像仓库

注 3 镜像仓库地址，必须是 http 或者 https

- 如果是公有云环境，并通过预先准备好的 Cloud Controller Manager 的机制提供了公有云的 k8s 负载均衡能力, 配置文件范例如下:

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
spec:
  loadBalancer:
    type: cloudLB
  fullPackagePath: absolute-path-of-the-offline-directory # (1)
  imagesAndCharts:        # (2)
    type: external
    externalImageRepo: your-external-registry # (3)
    # externalImageRepoUsername: admin
    # externalImageRepoPassword: Harbor123456
```

注 1 解压离线包后的路径

注 2 镜像仓库

注 3 镜像仓库地址，必须是 http 或者 https

- 如果使用 NodePort 暴露控制台（仅推荐 PoC 使用），配置文件范例如下:

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
spec:
  loadBalancer:
    type: NodePort
  fullPackagePath: absolute-path-of-the-offline-directory # (1)
  imagesAndCharts:        # (2)
    type: external
    externalImageRepo: your-external-registry # (3)
    # externalImageRepoUsername: admin
    # externalImageRepoPassword: Harbor123456
```

注 1 解压离线包后的路径

注 2 镜像仓库

注 3 镜像仓库地址，必须是 http 或者 https

1. 在 K8s 集群控制平面节点安装 DCE。

```
./dce5-installer install-app -c clusterConfig.yaml
```

说明

- 有关 clusterConfig.yaml 文件设置，请参考离线安装第 2 步。
- `-z` 最小化安装
- `-c` 指定集群配置文件。使用 NodePort 暴露控制台时不需要指定 `-c`。
- `-d` 开启 debug 模式
- `--serial` 指定后所有安装任务串行执行

1. 安装完成后，命令行会提示安装成功。恭喜您！ 现在可以通过屏幕提示的 URL 使用默认的账号和密码（admin/changeme）探索全新的 DCE 啦！

完成

请记录好提示的 URL，方便下次访问。

1. 另外，安装 DCE 成功之后，您需要正版授权后使用，请参考申请社区免费体验（见 5.2 节）。 在线与离线的差别只有三处：介质从网络换成离线包、配置文件多了 `fullPackagePath` 与 `imagesAndCharts`、 安装命令去掉 `-z`（离线包自带完整清单）。如果照 4.2 的命令在离线环境执行，会在拉镜像那一步超时。

4.4 自建集群的完整安装实例

这一节是官方一份完整的落地实录，和 4.2 的步骤本质相同，差异在两处：安装前要确认节点的 **外部可达 IP**（公有云场景），安装时用 `-k` 把它传给安装器；依赖项在这里用官方脚本一次装齐，写法和 2.3 略有出入——脚本名没带版本号，换成你实际下载的名字即可。

安装步骤

现在一切准备就绪，开始安装 DCE 社区版。

安装基础依赖

```
curl -LO https://qiniu-download-
public.daocloud.io/DaoCloud_Enterprise/dce5/install_prerequisite.sh
bash install_prerequisite.sh online community
```

下载 dce5-installer

```
export VERSION=v0.45.0
curl -Lo ./dce5-installer https://qiniu-download-
public.daocloud.io/DaoCloud_Enterprise/dce5/dce5-installer-$VERSION
chmod +x ./dce5-installer
```

确认节点的外部可达 IP

1. 如果你的浏览器跟 master 节点的 IP 是可以直通的，无需额外操作。
2. 如果 master 节点的 IP 是内网（比如本示例的公有云机器）：
   - 请在公有云中为其创建外网可达的 IP
   - 请在公有云配置中，在该主机的的防火墙规则中，允许 32088 端口的进出
   - 如上的 32088 端口是 `kubectl -n istio-system get svc istio-ingressgateway` 的 NodePort 端口

执行安装

1. 如果你的浏览器跟 master 节点的 IP 是可以直通的，直接执行

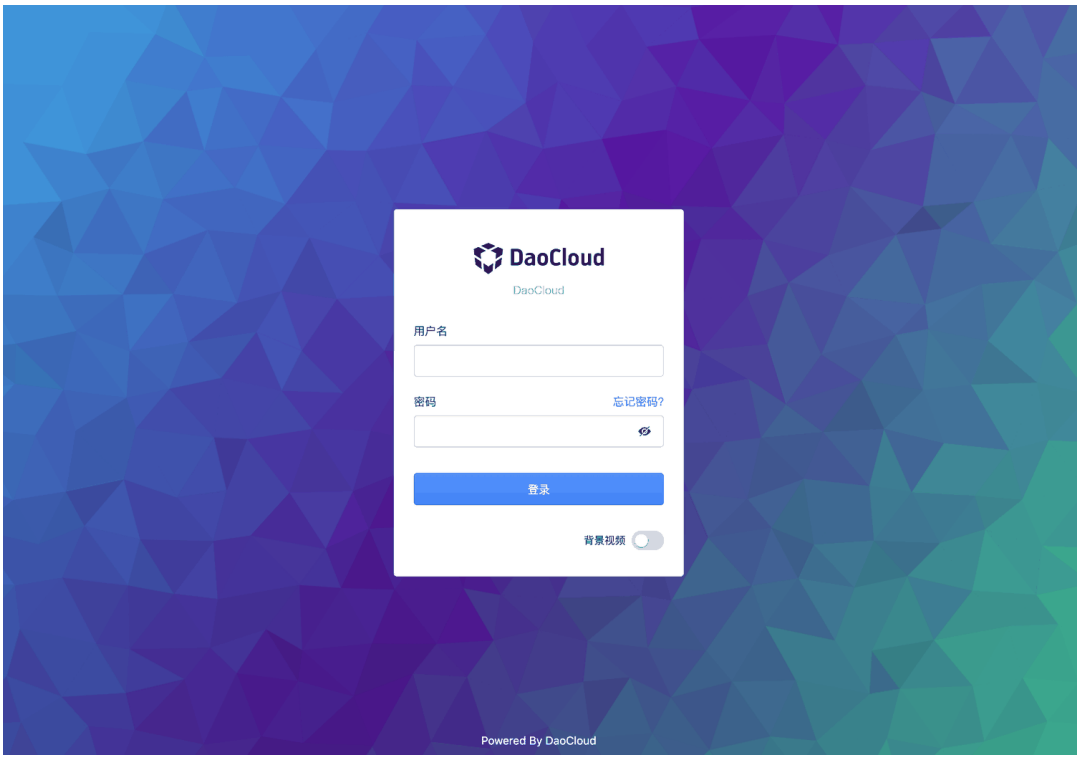
```
./dce5-installer install-app -z
```

2. 如果 master 节点的 IP 是内网（比如本示例的公有云机器），请确认上述外部 IP 和防火墙配置完毕，然后执行如下命令：

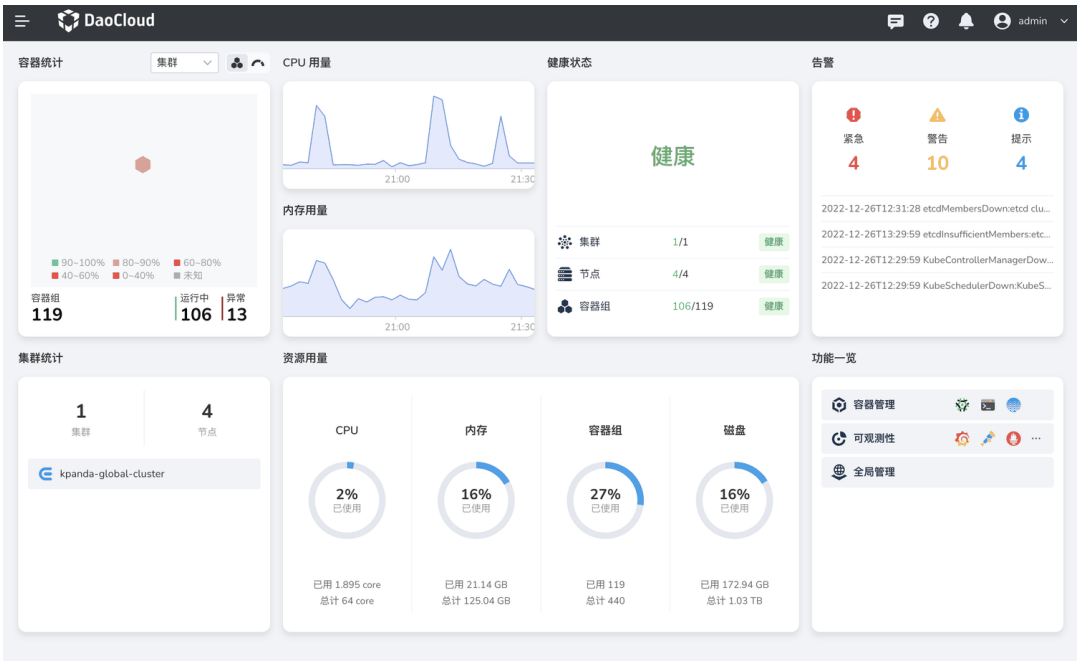
```
./dce5-installer install-app -z -k $外部 IP:32088
```

注意：上述的 32088 是 `kubectl -n istio-system get svc istio-ingressgateway` 的 NodePort 端口

3. 在浏览器中打开登录界面。



4. 以用户名 admin 密码 changeme 登录 DCE。



4.5 安装参数速查与结果核对

四节里反复出现的参数只有五个，汇总如下：

参数	作用	什么时候用
`-c`	指定集群配置文件（负载均衡类型、	需要 metallb VIP 或离线安装时；用

参数	作用	什么时候用
`clusterConfig.yaml`	VIP、离线包路径、镜像仓库）	NodePort 暴露控制台时不需要
`-z`	最小化安装，只装社区版三个模块	社区版默认都带这个参数
`-k <IP>:<端口>`	指定控制台的对外访问地址	集群在公有云、节点 IP 是内网 IP 时（如 `-k 1.2.3.4:32088`）
`-d`	开启 debug 模式	安装卡住、需要看详细日志时
`--serial`	所有安装任务串行执行	资源紧张、并行安装容易失败时

安装成功的标志是命令行打印出成功提示，并给出一个访问 URL。用它给出的地址打开控制台，用默认账号 **admin**、默认密码 **changeme** 登录：

> **警告**
>
> 默认口令是公开写在文档里的，任何知道地址的人都能用它登进来。**请在第一次登录后立刻修改 admin 密码**， 并尽快按 6.2 的说明建好日常使用的用户，把 admin 留给应急。

登录进去还不算完，最后一步是授权：

> **完成**

- 请记录好提示的 URL，方便下次访问。
- 安装 DCE 社区版成功后，请按第 5 章申请并激活许可证；未激活时可以登录，但不算正式授权使用。
- 如果安装过程中遇到问题，第 7 章按阶段整理了常见故障。

第 5 章　申请与激活许可证

社区版免费，但仍然需要一张许可证。整件事在浏览器里完成，只有最后一步要回到 DCE 控制台。

5.1　装完之后为什么还要授权

不少人以为社区版「免费就不用授权」，于是装完看到能登录就以为结束了。实际上安装器并不拦你—— 用 admin / changeme 可以立刻登进去——但「能登录」和「已授权」是两件事：**许可证绑定的是设备，不是账号**， 所以必须先有设备（先装完），才能换到绑定这台设备的授权码。

三个容易混淆的东西，先分清：

名称	谁产生的	作用	丢了怎么办
许可证密钥	官网申请表提交后，邮件发放	凭它换取离线授权码；也是续期凭证	联系 DaoCloud 交付团队重新获取
ESN	你的 DCE 平台自己生成	设备身份标识，会被写进授权码	在平台里重新查看即可，不会变
离线授权码	license.daocloud.io 用「密钥 + ESN」换回	粘贴到平台完成激活	用密钥重新换一次

因此授权是**设备级**而非用户级的：换机器、重装系统、重建集群，只要 ESN 变了，旧授权码就失效。 这也解释了为什么本书把授权放在第 5 章而不是第 2 章——顺序上它一定在安装之后。

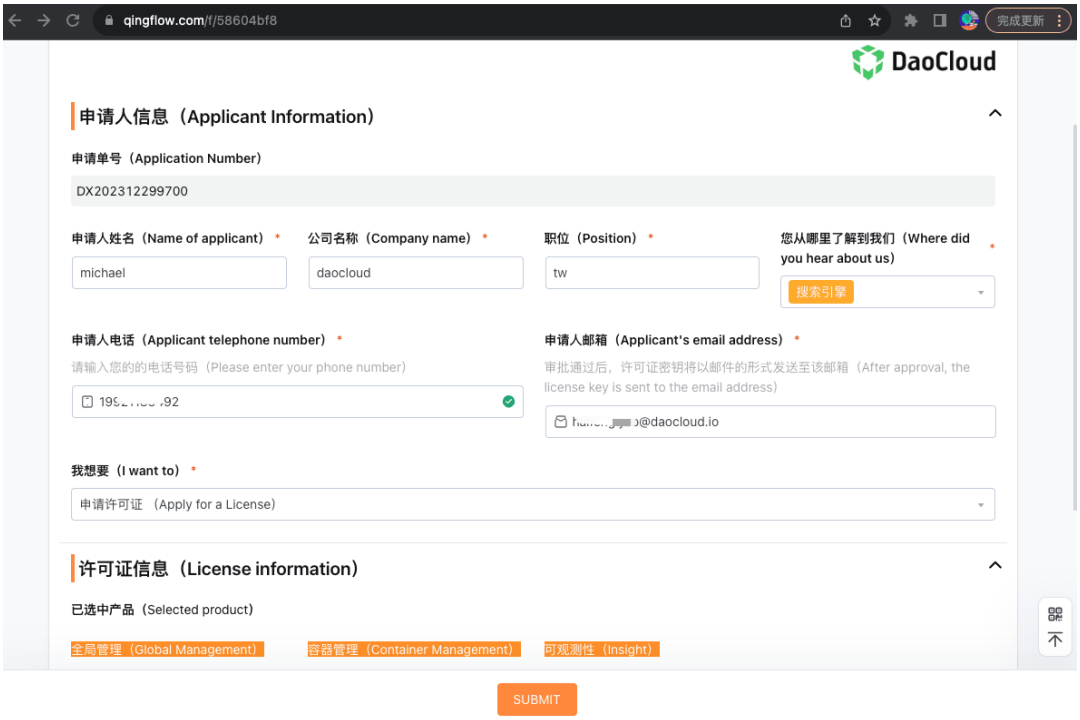
5.2 三步拿到授权

位置：DCE › DCE 介绍 › 申请社区免费体验

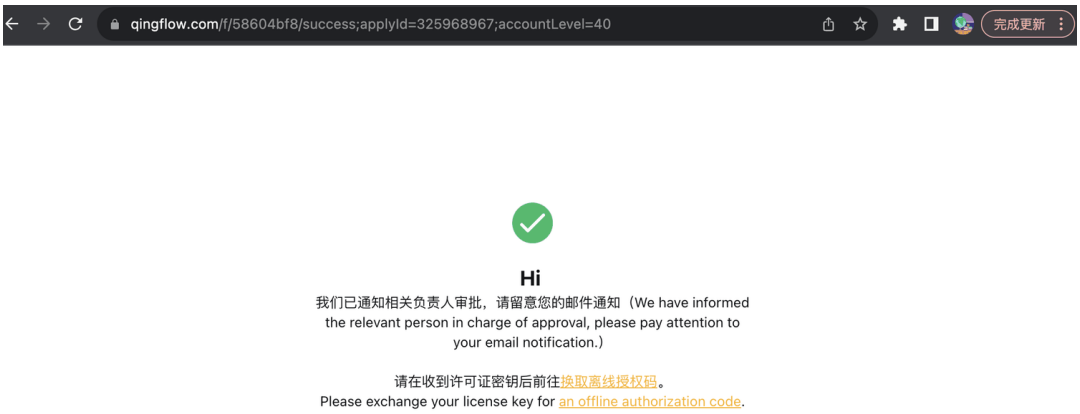
从申请到激活：四个动作

申请许可证的具体步骤如下：

1. 点击 申请许可证密钥 ，填写表单后点击 **提交**。

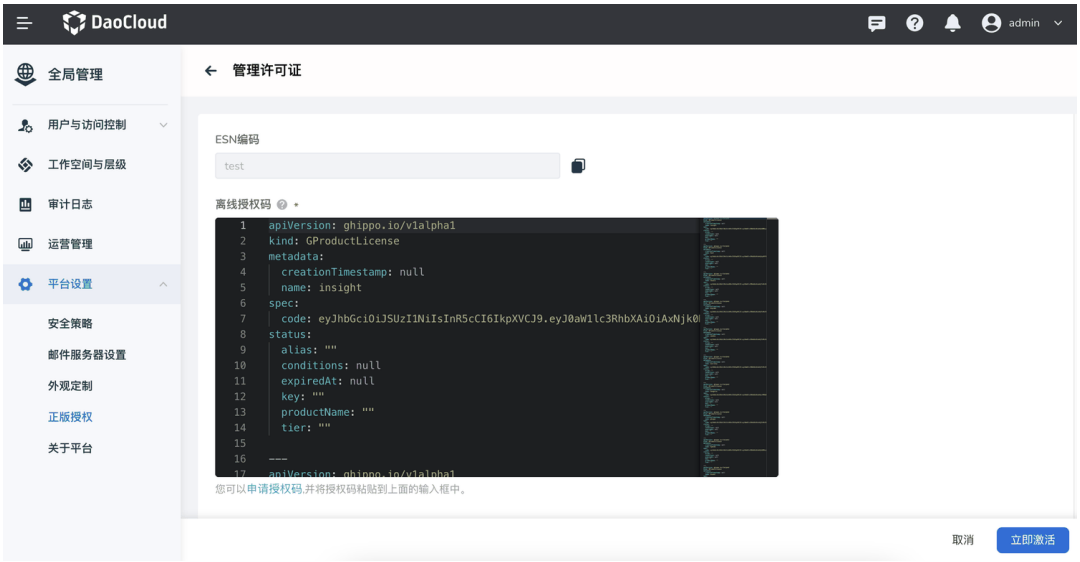


2. 按屏幕提示，打开邮箱查看包含许可证密钥的邮件，然后点击 换取离线授权码 。



3. 输入上一步的许可证密钥，输入设备独有的 ESN 后，点击 **获取离线授权码**。

其中 ESN 是集群系统独有的设备编码。 具体获取方式为：打开 DCE，点击左侧导航栏 **全局管理 → 平台设置 → 正版授权**，点击 **管理许可证** 按钮，打开 **管理许可证** 页面，复制 ESN 编码。



4. 复制并粘贴上一步换取的离线授权码，点击 **立即激活**。恭喜您！现在可以探索全新的社区版啦！

> **说明**
>
> 离线授权码与设备 ESN 绑定，请妥善保存好许可证密钥邮件。 您可以随时联系 DaoCloud 交付团队重新申请，我们已为您保存好所有许可密钥。

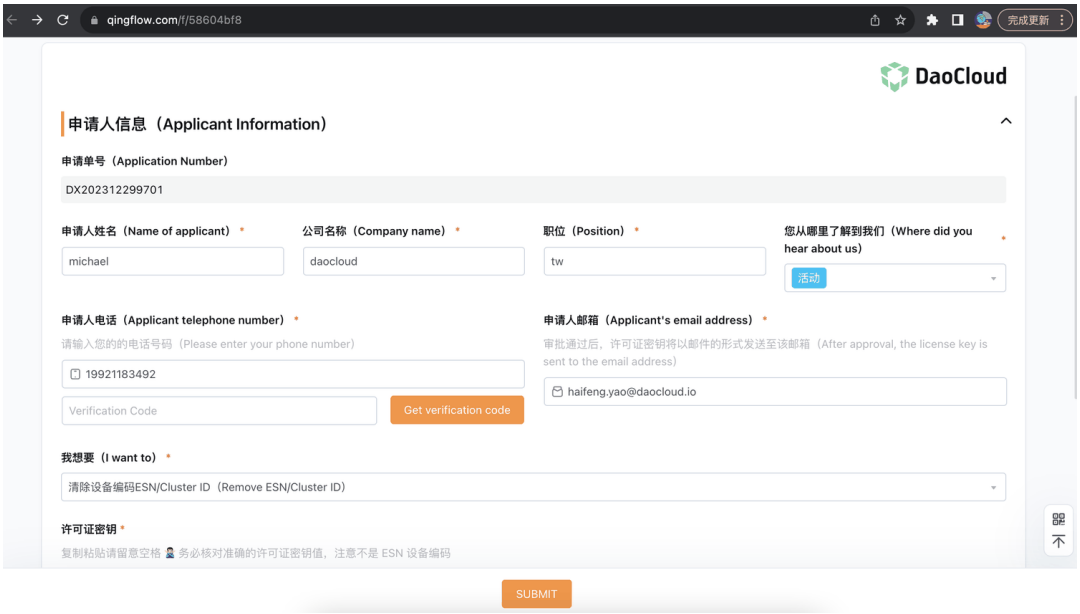
5.3 许可证续期

两种续期做法

如果您的许可证因某些原因到期了，可参考以下步骤续期。

推荐方案

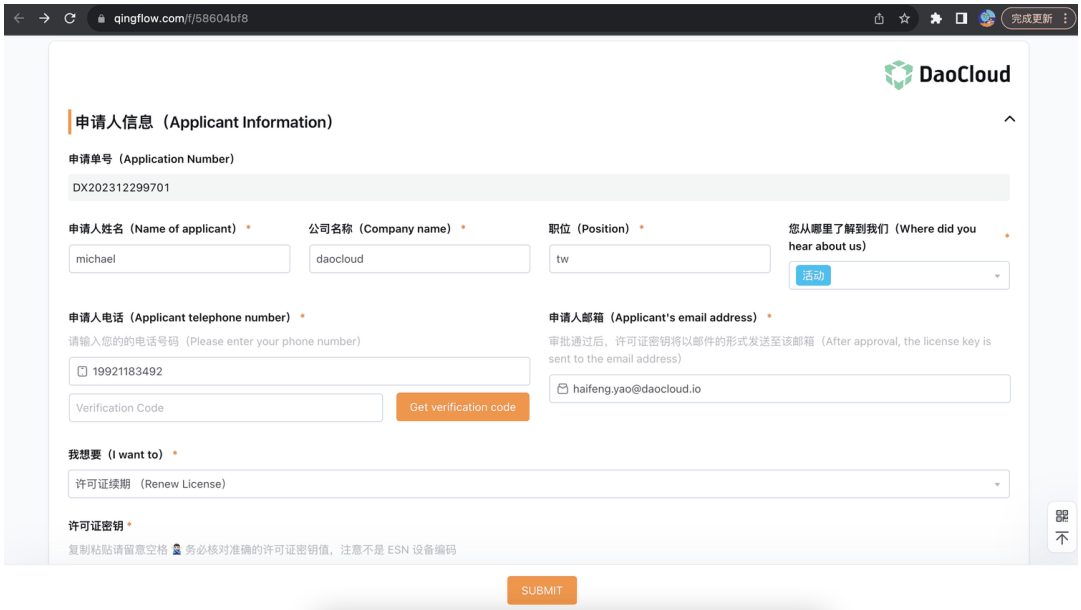
1. 首先清除设备编码，在表单中选择 **清除设备编码（ESN/Cluster ID）**，输入先前的许可证密钥，填写完成后点击 **提交**。



2. 参照上一节重新申请许可证。

按模块延续许可证

这种方法需要对安装的每个模块延续许可证。 例如社区版包含了容器管理、全局管理、可观测性 3 个模块，每个模块有 1 个许可证。 这样就需要在许可证页面，为每个模块选择 **许可证续期**。



5.4 授权环节的常见疑问

ESN 到底在哪看？ 登录 DCE，点左侧导航栏 **全局管理→平台设置→正版授权**，点 **管理许可证** 按钮， 在打开的页面里就能看到并复制 ESN 编码（截图见 5.2.2）。

换了机器或重装了系统，授权还能用吗？ 不能。离线授权码与设备 ESN 绑定，ESN 变了授权码就失效。处理办法有两个：按 5.3 的推荐方案 先清除旧设备编码再重新申请；或者联系 DaoCloud 交付团队协助处理。

许可证密钥的邮件找不到了怎么办？ 官方会保存所有许可密钥，可以联系 DaoCloud 交付团队重新获取。建议第一次收到邮件后就归档到共享位置。

社区版有三个模块，是一张证还是三张证？ 按模块延续许可证时的说法是：社区版包含容器管理、全局管理、可观测性 3 个模块，每个模块有 1 个许可证。 所以续期时可以整机重申请，也可以逐个模块续。

许可证到期了会怎样？ 平台会提示许可证到期，需要按 5.3 续期。续期不影响已部署的业务负载，但授权状态会显示为未授权。

常用链接速查

用途	地址
申请许可证密钥	https://qingflow.com/f/58604bf8
查看已有许可证密钥	https://license.daocloud.io/dce5-licenses
换取离线授权码	https://license.daocloud.io/dce5-license

> **提示**
>
> 这三个地址在激活与续期时会反复用到，建议连同第 5.2 的步骤一起收藏。 申请表单里除「申请许可证」，还有「清除设备编码（ESN/Cluster ID）」「许可证续期」两个入口。

第 6 章　装好之后：三个模块速览

这一章不教操作，只解决「进去之后不知道点哪」的问题：每个模块管什么、第一次进去该做哪三件事、 细节去哪里查。三个模块各自的完整用法都有专门的文档，本书不重复。

6.1　三个模块的分工

同一套集群，三个模块看它的角度不一样：

模块	核心对象	一句话职责	你不看它会怎样
全局管理	用户、用户组、工作空间、层级	决定「谁能进、能看哪一块」	所有人共用 admin，权限无法收敛，审计无记录
容器管理	集群、节点、命名空间、工作负载、Helm 应用	决定「应用跑在哪、怎么跑」	只能看控制台空转，业务发不出去
可观测性	指标、日志、链路、告警规则	决定「跑得好不好、出事找谁」	故障靠猜，扩容凭感觉

三者有一条明确的依赖顺序：先有身份与边界（全局管理），资源才有归属；先有资源（容器管理）， 才谈得上观测（可观测性）。反过来做，会出现「装好了却没有权限看」的尴尬。

6.2　全局管理

官方定义：全局管理是以用户为中心的综合性服务板块，包含用户与访问控制、工作空间与层级、审计日志、 平台设置等基础服务模块。四块内容各管一件事：

服务	它解决的问题	官方描述
用户与访问控制	谁可以登录、能做什么	帮助用户安全管理资源的访问权限。可创建、管理、删除用户/用户组，并灵活配置权限，完成职能权限的划分
工作空间与层级	资源怎么分给不同的部门或项目	具有层级结构和访问权限控制的资源隔离单元，可按开发环境、部门结构设置层级并控制访问
审计日志	谁在什么时候改了什么	提供资源的操作记录，可用于安全分析、资源变更、问题定位

服务	它解决的问题	官方描述
平台设置	平台长什么样、怎么发通知、授权状态	通过平台安全策略、邮件服务器、外观定制等，实现用户信息的安全性和平台的个性化

第一次进去，按这个顺序做三件事

1. **建用户与用户组并授权**。在 **用户与访问控制** 里创建用户、创建用户组，把用户放进组，再给组授权。日常操作不要都用 admin——admin 是超级管理员，出事时留作应急。
2. **建层级与工作空间**。在 **工作空间** 里先建层级（对应企业的部门或环境划分），再在工作空间里绑定资源。工作空间是租户的边界，容器管理里的命名空间最终要落到某个工作空间下。
3. **配平台设置**。在 **平台设置** 里配置安全策略（密码规则、会话超时）、邮件服务器（告警通知要靠它）、外观定制（登录页与导航栏），并在这里查看 **正版授权** 状态。

> **说明**
>
> 第 5 章的许可证激活也在这一块：**全局管理→平台设置→正版授权→管理许可证**。换句话说，装完 DCE 后第一个该进的模块就是全局管理——既为了取 ESN，也为了建第一个业务用户。

细节去哪查：全局管理的用户与访问控制、工作空间、审计日志、平台设置各有专页，从 `docs.daocloud.io/ghippo/` 进入。本书不展开操作步骤。

6.3 容器管理

官方定义：容器管理是基于 Kubernetes 构建的面向云原生应用的容器化管理模块。基于原生多集群架构，解耦底层基础设施，实现多云与多集群统一化管理，大幅简化企业的应用上云流程。

先认识四类集群角色。这是社区版装完后最容易绕晕的地方——控制台里能看到的集群，角色并不相同：

集群角色	作用	谁建的
全局服务集群	运行 DCE 自身组件（容器管理、全局管理、可观测性等），一般不承载业务负载	安装 DCE 时自动产生
管理集群	用于管理工作集群，一般不承载业务负载；简约模式下与全局服务集群合并	安装 DCE 时按模式产生
工作集群	承载业务负载，由管理集群管理	在容器管理里创建
接入集群	接入已有的标准 K8s 集群（自建、公有云、私有云、边缘、信创、异构均可），承载业务负载	从外部接入，不支持全生命周期管理

社区版最常见的形态是：安装时得到一个全局服务集群，之后在容器管理里再建（或接入）工作集群。 **业务应用请发到工作集群或接入集群，不要发到全局服务集群**。

能力地图。容器管理覆盖的功能可以归成八类，社区版全部可用：

类别	包含内容
集群管理	创建集群、接入集群、升级集群、集群角色与状态、选择容器运行时
节点管理	节点调度、标签与注解、污点管理、节点扩容/缩容、节点认证与可用性检查
命名空间	创建/删除命名空间、命名空间独享节点、Pod 安全策略
工作负载	Deployment、StatefulSet、DaemonSet、Job、CronJob
弹性伸缩	HPA、VPA、定时伸缩（需先装 metrics-server 等插件）
应用交付	Helm 模板、Helm 应用、Helm 仓库（即应用商店的三层结构）
网络与存储	Service、Ingress、NetworkPolicy；PVC、PV、StorageClass（原生 CSI）
安全与配置	合规性/权限/漏洞扫描、ConfigMap、Secret、集群巡检、集群运维

第一次进去，做这三件事

1. 在 **集群列表** 里确认全局服务集群状态正常，再创建一个工作集群（或接入一套已有集群）。
2. 在工作集群下建一个命名空间，把它绑到 6.2 建好的工作空间上。
3. 部署第一个工作负载：从 **工作负载 → 无状态负载**建一个 Deployment，或用 **Helm 应用** 装一个现成模板。

细节去哪查：集群、节点、工作负载、网络、存储、安全等逐项操作见 `docs.daocloud.io/kpanda/`。

6.4 可观测性

官方定义：可观测模块（Insight）是以应用为中心、开箱即用的新一代云原生可观测性平台。能够实时监控 应用及资源，采集指标、日志及事件等数据来分析应用健康状态，提供告警能力以及多维度的数据可视化能力。

它把三样东西统一采集到一起，这也是理解 Insight 的最短路径：

数据类型	回答什么问题	典型入口

数据类型	回答什么问题	典型入口
指标（Metric）	现在忙不忙、用了多少资源	仪表盘（Dashboard）
日志（Log）	刚才那一步为什么报错	日志查询
链路（Trace）	一次请求慢在哪个环节	链路追踪

Insight 与 Insight Agent 是两个东西：Insight 是平台侧的组件，装在全局服务集群； Insight Agent 需要部署到**每一个被观测的 Kubernetes 集群**上，负责采集那个集群的指标与日志。 社区版装好后，全局服务集群自带 Insight，但新建的工作集群若要被观测，需要单独装 Agent。

五个必会概念

概念	含义
监控目标（Target）	被监控的对象；系统定时向监控点抓取指标
指标（Metric）	用 OpenMetrics 格式描述的、衡量系统某种属性的标准
仪表盘（Dashboard）	可视化面板，把实时情况和各项性能指标画成图
告警规则（Rule）	返回布尔值的 PromQL 表达式，不满足阈值即产生告警事件
告警消息（Alert）	规则被触发时的记录，包含规则、触发时间和当时状态，并可触发通知

第一次进去，做这三件事

1. 打开 **集群监控**，看全局服务集群的 CPU、内存、磁盘曲线，确认采集正常。
2. 给一个工作集群装上 Insight Agent，再看它的指标是否出现。
3. 在 **告警中心** 配一条最简单的规则（例如某节点 CPU 持续高于 80%），并把通知渠道指向 6.2 里配好的邮件服务器。

细节去哪查：安装 Agent、仪表盘、日志、链路、告警的逐项操作见 `docs.daocloud.io/insight/`。

6.5 串起来：上线一个应用的三阶段

把三个模块放在一条时间线上，社区版的日常使用就是这三段：

阶段	在哪个模块	关键动作	产出
① 身份与边界	全局管理	建用户与用户组、建层级与工作空间、授权	明确「谁能在哪个空间里做事」

阶段	在哪个模块	关键动作	产出
② 部署与运行	容器管理	建命名空间、部署工作负载或 Helm 应用、配 Service 与 Ingress	应用可被访问
③ 观测与告警	可观测性	看指标与日志、配告警规则与通知	出问题时能定位、能提前预警

这条链上有两个常见的顺序错误：

- **先发应用、后配权限**：应用发出去了，但业务同事登录后看不到，只能临时给 admin，权限从此收不回来。
- **先配告警、后装 Agent**：规则建了一堆，却没有任何数据源，告警永远不触发。

正确的做法是按 ①→②→③ 走，每一步都有可验证的产出，出了问题也知道停在哪一段。

第 7 章　常见问题与排障

社区版的问题相对集中：要么卡在安装，要么卡在授权，要么打不开控制台。这一章按阶段给出「现象 → 原因 → 处理」的对照表，末尾列出求助渠道。

7.1　安装阶段

现象	原因	处理
Kind 重装时，Redis Pod 长时间 0/4 running，提示 `primary ClusterIP can not unset`	上一次安装残留的 Service 与新的 ClusterIP 冲突	删除 `mcamel-system` 命名空间下的 `rfs-mcamel-common-redis` 服务（`kubectl delete svc rfs-mcamel-common-redis -n mcamel-system`），再重跑安装命令
安装报 `DaemonSet is not ready: insight-system/insight-agent-fluent-bit. 0 out of 2 expected pods are ready`	fluent-bit 的已知缺陷，日志里能看到 `Could not contact DNS servers`，解析的是 `mcamel-common-es-cluster-masters-es-http.mcamel-system.svc.cluster.local`	先确认该 Service 是否存在、CoreDNS 是否正常；这是 aws-for-fluent-bit 的已知 Issue（aws/aws-for-fluent-bit#233），按 Issue 中的说明处理
安装器提示依赖项缺失或版本过低	环境里的 helm / skopeo / kubectl / yq 未达门槛	按 2.3 重装依赖项，脚本会自动替换低版本
安装成功但浏览器打不开控制台	控制台走 NodePort 或 VIP，安全组/防火墙未放行	放行 32088 端口（或你的 VIP），见 4.4；公有云还需确认外网 IP 绑定
安装过程中某个任务反复重试	镜像拉取慢或并行任务抢占资源	加 `-d` 看详细日志；必要时加 `--serial` 串行执行（见 4.5）

7.2　授权与访问阶段

现象	原因	处理
不知道 ESN 在哪里	ESN 只在已安装的平台上	全局管理 → 平台设置 → 正版授权 → 管理许可证，见 5.2.2

现象	原因	处理
粘贴授权码提示无效	授权码对应的 ESN 与当前设备不一致（换机、重装、重建集群）	按 5.3 的推荐方案清除设备编码后重新申请
平台显示许可证到期	授权有效期结束	按 5.3 续期；两个方案任选其一
登录后看不到任何集群或菜单	当前用户缺少授权	用 admin 检查该用户的角色与工作空间绑定，见 6.2

7.3 拿到帮助

安装或使用中遇到问题，按下面的顺序找答案：先查本章与对应章节，再查官网 FAQ，最后找社区或交付团队。

- **官方文档**：https://docs.daocloud.io/（社区版入口为 `docs.daocloud.io/install/community/`）
- **安装 FAQ**：`docs.daocloud.io/install/faq`，按 UI 访问、火种节点、证书、操作系统、社区版问题分类
- **下载中心**：https://docs.daocloud.io/download/，可取社区版安装器历史版本与离线包
- **社区交流群**：扫描下方二维码加入社区版交流群，与开发者直接沟通

> **说明**
>
> 如果问题与授权、交付相关（例如密钥邮件遗失、需要批量部署、要评估商业版模块）， 可以直接联系 DaoCloud：电邮 `info@daocloud.io` 或致电 400 002 6898。

附录 A　安装前检查清单

打印这一页，逐项打勾。任何一项没勾上，都不要急着执行安装命令。

#	检查项	达标标准	☐
1	路线已选定	明确走 Kind / 已有 K8s 在线 / 已有 K8s 离线 / 自建集群	☐
2	机器数量与规格	标准集群 3 Master + 2 Worker，每台 4C8G、系统盘 100G；Kind 至少 8C12G、100G	☐
3	K8s 版本	≥ v1.22，推荐 v1.29.5	☐
4	CPU 架构确认	x86_64 或 ARM64，ARM 需换安装器与依赖包名	☐
5	容器运行时	Docker 或 containerd 已装并运行	☐
6	默认 StorageClass	`kubectl get sc` 能看到 `(default)`	☐
7	CoreDNS	kube-system 下 coredns Pod 全部 Running	☐
8	单节点污点	单节点集群已移除 control-plane 的 NoSchedule 污点	☐
9	依赖项	helm ≥3.11.1、skopeo ≥1.11.1、kubectl ≥1.25.6、yq ≥4.31.1	☐
10	外网连通	在线安装需能访问公网仓库；Kind 需能 `ping daocloud.io`	☐
11	离线物料	离线安装需备好离线包、依赖包与镜像仓库地址	☐
12	访问地址	已规划控制台访问方式：NodePort 端口 32088，或 metallb 两个 VIP	☐
13	安全组/防火墙	32088 或 VIP 对应端口已放行	☐

#	检查项	达标标准	☐
14	时间同步	各节点时间一致（否则证书与 Token 校验会失败）	☐
15	许可证准备	知道去哪里申请密钥，邮箱可用	☐
16	记录本	准备记录安装结束输出的 URL 与账号信息	☐

附录 B　命令速查

环境自检（Kind 场景）

```
set -e
if [ $(free -g|grep Mem | awk '{print $2}')              -lt 12 ]; then (echo
"insufficient memory! (should >=12G)";); fi
if [ $(grep 'processor' /proc/cpuinfo |sort |uniq |wc -l) -lt 8 ]; then (echo
"insufficient CPU! (should >=8C)";); fi
if [ $(df -m / |tail -n 1 | awk '{print $4}') -lt 30720 ]; then (echo "insufficient
free disk space of root partition!(should >=30G)";); fi
ping daocloud.io -c 1 &> /dev/null || ( echo "no connection to internet! abort.")
echo "precheck pass.."
```

Kind 相关

命令	说明
`kind version`	查看 kind 版本
`kind create cluster --image <镜像> --name=<名字> --config=kind_cluster.yaml`	创建集群
`kind get clusters`	列出本机所有 kind 集群
`docker exec -it <集群名>-control-plane kubectl get po -A -w`	在 kind 节点里观察 Pod 启动

依赖项与安装器

命令	说明
`bash install_prerequisite_v0.45.0.sh online community`	在线装依赖项（社区版）
`./install_prerequisite_v0.45.0.sh offline community`	离线装依赖项（社区版）
`curl -Lo ./dce5-installer https://qiniu-download-`	下载安装器

命令	说明
`public.daocloud.io/DaoCloud_Enterprise/dce5/dce5-installer-$VERSION`	
`./dce5-installer install-app -z`	最小化安装（NodePort 场景）
`./dce5-installer install-app -c clusterConfig.yaml -z`	带配置文件安装
`./dce5-installer install-app -z -k <IP>:<端口>`	指定控制台访问地址安装

装完之后常用

命令	说明
`kubectl get no`	查看节点状态
`kubectl get sc`	查看 StorageClass 与默认标记
`kubectl -n istio-system get svc istio-ingressgateway`	查看控制台 NodePort 端口
`kubectl get po -A | grep -v Running`	快速找出非 Running 的 Pod

附录 C　官方文档索引

本书的每一节都能在官网找到对应页面。想核对原文或看后续更新，按此表访问：

本书章节	官方页面
1.1、1.2、1.3	https://docs.daocloud.io/dce/（DCE 概览与模块地图）
2.1、2.2	https://docs.daocloud.io/install/community/resources/
2.3	https://docs.daocloud.io/install/install-tools/
3.1、4.1	https://docs.daocloud.io/install/community/kind/online/
3.2、4.4	https://docs.daocloud.io/install/community/k8s/k8s-dce-deploy/
4.2	https://docs.daocloud.io/install/community/k8s/online/
4.3	https://docs.daocloud.io/install/community/k8s/offline/
第 5 章	https://docs.daocloud.io/dce/license0/

本书章节	官方页面
6.2	https://docs.daocloud.io/ghippo/intro/
6.3	https://docs.daocloud.io/kpanda/intro/ 与 /kpanda/user-guide/clusters/cluster-role/
6.4	https://docs.daocloud.io/insight/intro/
7.1	https://docs.daocloud.io/install/faq/

说明

官网版本迭代较快，页面内容可能已领先于本书。凡涉及具体版本号的命令，请以官网当前页面为准； 本书的组织方式（谁先谁后、每步的产物是什么）不受版本变化影响。