

# DaoCloud Enterprise

---

**DaoCloud Enterprise Product Documentation**

## Table of contents

---

1. 博客文章	6
2. 5.0 应用工作台能力介绍	10
2.1 图形化用户交互	10
2.2 功能特性	11
2.3 常见问答	14
3. DCE 研发背景	15
3.1 云原生浪潮	15
3.2 容器 vs 虚拟化	16
3.3 DCE 与 Kubernetes	17
3.4 DCE 产品简介	18
3.5 经 CNCF 认证的 KCSP	18
4. 「DaoCloud 道客」与 Kubernetes	19
4.1 挑战	19
4.2 解决方案	19
4.3 影响	19
5. 保姆式安装 DCE 5.0 社区版	22
5.1 集群规划	22
5.2 节点系统优化	23
6. 加载模块	23
7. 刷新配置	23
7.1 安装 K8s	23
8. 清理配置文件	24
9. 初始化配置	24
10. 更新配置文件内容	24
10.1 安装 Helm	39
10.2 安装 StorageClass	40
10.3 安装 Metrics server	43
10.4 安装 DCE 5.0 社区版	46
10.5 重定向 Portal 反向代理	48
11. 反向代理地址	49
12. helm --set 参数备份文件	49
13. 获取当前 ghippo 全局管理的版本号	49
14. 边缘原生应用准则白皮书	52
14.1 目标	52
14.2 什么是边缘？	52

14.3 云原生 vs 边缘原生	52
14.4 边缘原生与云原生的相似性	53
14.5 边缘原生与云原生的区别	53
14.6 边缘原生应用	53
14.7 边缘原生准则	54
14.8 结论和后续步骤	55
14.9 如何参与	55
14.10 边缘原生开源项目和计划的工作列表	55
14.11 贡献者	55
14.12 参考资料	56
15. 5.0 多云编排能力介绍	57
15.1 功能特性	57
15.2 图形化用户交互	58
15.3 多云编排可以做什么	58
15.4 常见问答	59
16. DaoCloud 是 K8s 资深认证服务商	61
16.1 技术领先	61
16.2 CNCF 贡献排名	62
16.3 参考链接	66
17. 5.0 容器管理能力介绍	67
17.1 图形化用户交互	68
17.2 最佳实践探索	68
17.3 谁需要容器管理	69
17.4 FAQ	70
18. 介绍 KWOK	72
18.1 什么是 KWOK?	72
18.2 为什么使用 KWOK?	72
18.3 使用场景是什么?	72
18.4 有哪些限制?	73
18.5 入门指导	73
18.6 欢迎参与	73
18.7 参考链接	73
19. Linux 单机在线体验 DCE 5.0 社区版	75
19.1 准备工作	75
19.2 安装 Docker	75
19.3 kind 集群	76
19.4 安装 DCE 5.0 社区版	77

20. 通过 Docker 和 kind 在 macOS 电脑上安装社区版	78
20.1 硬件环境	78
20.2 安装和调整 Docker	78
20.3 安装 kind	79
20.4 创建 kind 配置文件	79
20.5 kind 创建 K8s 集群	79
20.6 安装 DCE 5.0 社区版	80
20.7 体验使用	81
20.8 卸载	81
21. 5.0 服务网格能力介绍	82
21.1 流量治理	83
21.2 端到端的透明安全	84
21.3 服务运行监控	85
21.4 图形化用户交互	85
22. 2022 年 DCE 5.0 攻坚语录集	86
23. DCE 5.0 涉及多少开源项目	88
23.1 容器管理	88
23.2 全局管理	96
23.3 可观测性	103
24. 5.0 资源管理能力介绍	111
24.1 资源管理的用户交互	111
24.2 资源管理要素	112
24.3 谁需要资源管理	113
24.4 FAQs	114



## 1. 博客文章

---

本页汇总 DCE 5.0 相关的博客和公众号文章， 默认按字母和拼音排序。

- 20230317 | 边缘原生应用准则白皮书

物联网边缘工作组 (IOT Edge Working Group) 一直在探索边缘原生的定义，以及“云原生”和“边缘原生”之间的异同，并发布了《边缘原生应用准则白皮书》。

- 20230315 | 在 Linux 上安装单机 DCE 5.0 社区版

讲述如何在一台 Linux 机器上使用 Docker 和 kind 在线安装 DCE 5.0 社区版。这是一种极简安装方式，便于学习和体验，性能优于 macOS 单机版。

- 20230315 | 在 macOS 上安装单机 DCE 5.0 社区版

使用 macOS 笔记本电脑创建单节点的 kind 集群，然后在线安装 DCE 5.0 社区版。适合初学者体验和学习，不适合生产环境。

- 20230301 | 开源项目 KWOK 介绍

是什么样的开源项目会在发布 5 个月内，就被 Apple、IBM、腾讯、华为纷纷使用？

KWOK 是 Kubernetes WithOut Kubelet 的缩写，即没有 Kubelet 的 Kubernetes。帮助你在几秒钟内搭建一个由数千个节点构成的集群，用少量资源模拟几千个真实的节点。

- 20230214 | DCE 5.0 社区版包含多少开源项目

时常听闻客户、社区成员、贡献者、公司内部的售前、交付、项目组在询问“DCE 到底涉及了哪些开源项目？”这篇博文详细列出了社区版所包含的开源项目。

- 20230214 | 「DaoCloud 道客」与 Kubernetes

讲述「DaoCloud 道客」如何借力 Kubernetes 打造新一代企业级云原生应用云平台—— DaoCloud Enterprise 5.0，以及如何回馈开源社区，践行云原生信仰。

- 20230201 | DCE 5.0 攻坚语录集

2022 年，上海，疫情肆虐，封城、管控、居家，程序猿们奔走在病毒  的缝隙中，那一年是 DCE 5.0 攻坚的日子，也是每个中国人艰难的一年。

- 20221209 | 保姆式安装 DCE 5.0 社区版

这是从 0 到 1 安装 DCE 5.0 社区版的真实示例，包含了 K8s 集群、依赖项、网络、存储等细节及更多注意事项。

- 20221130 | Karmada 资源解释器

Karmada 已被越来越多的企业，应用到多云和混合云场景中。在实际应用过程中，用户经常会遇到，通过 PropagationPolicy 来分发各种资源到成员集群的使用场景。这就要求分发的资源类型，不仅仅要包含常见的 Kubernetes 原生或知名扩展资源，同时也需要能支持分发用户的自定义资源。因此 Karmada 引入了内置解释器来解析常见的 Kubernetes 原生或知名扩展资源，同时又设计了自定义解释器来解释自定义资源的结构，并且在近期提出了可配置解释器方案。对无论是自定义资源，还是常见的 Kubernetes 原生资源，都可以提供更加灵活可配置的自定义方法，来提取资源的指定信息，例如副本数、状态等。

- 20221125 | KubeCon 2022 北美站 | 精彩看点回顾

2022 年 11 月落幕的云原生顶级会议 2022 KubeCon 北美站上，来自世界各地的云原生技术专家、产品或解决方案供应商和使用者带来了 300 多场精彩纷呈的演讲。主题涵盖 Kubernetes、GitOps、可观测性、eBPF、网络、服务网格和安全等。本文精心挑选了此次会议上的几个热点话题演讲，进行简单介绍，感受每一次演讲和讨论背后所蕴藏的云原生趋势。

- 20221123 | 走进车企的数字原生之路 | 论道回顾

11 月 18 日，由「DaoCloud 道客」主办的「论道原生 | 云原生数字生态私享会 · 走进车企」成功举办。本次活动从车企的具体案例出发，主要分享了云原生在汽车行业的应用与实践。让我们一起回顾一下本次活动的精彩内容。

- 20221115 | SpiderPool - 云原生容器网络 IPAM 插件

SpiderPool 来源于容器网络落地实践的经验积累，是「Daocloud 道客」开源的原生容器网络 IPAM 插件 ([github: https://github.com/spidernet-io/spiderpool](https://github.com/spidernet-io/spiderpool))，主要用于搭配 Underlay CNI 实现容器云平台精细化的管理和分配 IP。

- 20221110 | 源于热爱，始于坚持，不忘初心 — 「DaoCloud 道客」八岁生日快乐！

时光荏苒，岁月如梭。自 2014 年 11 月成立以来，在 DaoClouders 不懈的耕耘中，「DaoCloud 道客」已经走过了八个年头。在 11 月 8 日下午，「DaoCloud 道客」全体船员为「DaoCloud 道客」举办了生日会。让我们一起来看看「DaoCloud 道客」八岁生日会的盛况吧！

- 20221105 | DaoCloud 是 K8s 资深认证服务商

DaoCloud 早在 2017 年就首次顺利通过了 Kubernetes 认证，是国内最早涉足并得到 CNCF 官方认可的服务商，同时也是国内最早获得 Kubernetes Training Partner (KTP) 认证的厂商。目前经 CNCF 官方认证可支持的 K8s 版本包括：v1.25, v1.24, v1.23, v1.20, v1.18, v1.15, v1.13, v1.9, v1.7

- 20221105 | 原生思维看金融数字化转型

2022 年 11 月 5 日至 6 日，由西南财经大学、成都市地方金融监督管理局和成都市温江区人民政府联合主办的第五届国际金融科技论坛 2022 在蓉成功举办。5 日下午以“数字经济赋能金融科技创新”为主题的论坛邀请「DaoCloud 道客」创始人兼首席执行官、云原生计算基金会大使陈齐彦先生做主题演讲。本文根据演讲内容整理。

- [20221104 | HwameiStor 入选 CNCF 全景图：生产可运维的云原生本地存储系统](#)

近日，CNCF（云原生计算基金会）<sup>[^1]</sup>发布了最新版的云原生全景图<sup>[^2]</sup>。「DaoCloud 道客」自主开源的云原生本地存储系统 HwameiStor，被收录在 CNCF 云原生全景图中的 RunTime（运行时）层的 Cloud Native Storage（云原生存储）象限，成为 CNCF 推荐的云原生本地存储项目。

- [20221028 | Kubeant 集群生命周期管理](#)

Kubeant 采用 Kubespray 作为底层技术依赖，一方面简化了集群部署的操作流程，降低了用户的使用门槛。另一方面在 Kubespray 能力基础上增加了集群操作记录、离线版本记录等诸多新特性。Kubeant 还提供了界面化创建集群的能力（需要结合社区版 DCE 5.0 容器管理功能），让新手用户也能一键创建和管理集群。

- [20221028 | 激活医疗、人工智能的数字新动能 | 云原生底座](#)

10 月 28 日，由「DaoCloud 道客」主办的「论道原生 | 云原生数字生态私享会 · 上海」成功举办。本次活动从云原生思维和技术出发，分享了云原生在医疗和人工智能方面的应用与实践。让我们一起回顾一下本次活动的精彩内容吧。

- [20221026 | DCE 5.0 容器管理能力介绍](#)

说明 DCE 5.0 容器管理模块提供的能力。

- [20221018 | DCE 5.0 资源管理能力介绍](#)

说明 DCE 5.0 全局管理模块提供的能力。

- [20220925 | DCE 5.0 应用工作台能力介绍](#)

说明 DCE 5.0 应用工作台模块提供的能力。

- [20220914 | Merbridge 入选 eBPF 全景图](#)

花开自有期，绽放亦有时。在所有贡献者的共同努力和呵护下，Merbridge 的花苞徐徐绽放开来。2022 年 4 月初，Merbridge 顺利入选 CNCF 云原生全景图（Cloud Native Landscape），进入 Orchestration & Management（编排与管理）层的 Service Mesh（服务网格）象限，成为 CNCF 推荐的云原生服务网格加速器。

- [20220909 | Clusterpedia 使用心得](#)

Kubernetes 带来的云原生技术革命已席卷全球，越来越多的业务应用运行在 Kubernetes 平台上，随着业务规模的扩展，集群数量与日俱增，多集群内部资源管理和检索越来越复杂，多云管理面临巨大挑战。在多集群时代，我们可以通过 Cluster-api 来批量创建管理集群，使用 Karmada/Clusternet 来分发部署应用。不过我们貌似还是缺少了什么功能，我们要如何去统一地查看多个集群中的资源呢？

- [20220908 | 华为与「DaoCloud 道客」推出面向元宇宙的云边协同超融合一体机](#)

2022 年 9 月 2 日，在世界人工智能大会上，华为与上海道客网络科技有限公司联合推出面向元宇宙创新业务的“云边协同超融合一体机”，将云原生能力下沉至边缘，提供实时的虚拟数字世界体验，实现真正的云边一体化元宇宙。

- [20220905 | 如何打好存储基础，筑就云原生应用基石？| 论道原生](#)

应用的云原生化极大地提升了自身的可用性、稳定性、扩展性、性能等核心能力，同时也深刻改变了应用的方方面面，存储作为应用运行的基石，也不可避免地受到了冲击。云原生时代背景下，给存储带来了怎样的挑战，又该如何应对呢？第十一期论道原生，「DaoCloud 道客」携华为，分享云原生存储解决方案。

- [20220810 | Cluster API 检索从未如此简单](#)

Clusterpedia 是一个 CNCF 沙箱项目，用于跨集群复杂的资源检索。名字源于 Wikipedia，寓意是打造多集群的百科全书，可以与多个集群同步资源，并在与 Kubernetes OpenAPI 兼容的基础上，提供更强大的搜索功能，以帮助您快速、简便、有效地获取任何多集群资源。

- [20220808 | DCE 5.0 多云编排能力介绍](#)

说明 DCE 5.0 多云编排模块提供的能力。

- [20220708 | DCE 5.0 服务网格能力介绍](#)

说明 DCE 5.0 服务网格模块提供的能力。

- [20220622 | Clusterpedia 作为首个多云检索开源项目正式进入 CNCF 沙箱](#)

2022 年 6 月 15 日，云原生计算基金会（CNCF）宣布 Clusterpedia 正式被纳入 CNCF 沙箱（Sandbox）项目。Clusterpedia 是由「DaoCloud 道客」于 2021 年年底推出的开源项目，是一个通过加持 kubectl 就能完成多集群资源复杂检索的神器，也是目前 CNCF 唯一专注于多云信息检索的项目，正被广泛使用。

- [20220611 | CloudTTY：下一代云原生开源 Cloud Shell](#)

CloudTTY 是一个云原生开源项目，基于 kubernetes 之上，解决了一系列集群之上的“网页命令行”权限控制下的功能需求。

- [20220609 | KubeCon EU 热门云原生技术分享 | 精彩看点回顾](#)

在刚刚过去的年度最顶级的云原生旗舰会议 KubeCon + CloudNativeCon Europe 2022 上，全球的云原生技术专家、产品或解决方案供应商和使用者，对云原生进行广泛充分地交流和讨论。本文从云原生的外部融合、自身演进、内部相关功能特性这三个方面，分享一些大会上热门的云原生开源项目，一起走进云原生。

- [20220606 | Merbridge CNI 模式](#)

Merbridge CNI 模式的出现，旨在能够更好地适配服务网格的功能。之前没有 CNI 模式时，Merbridge 能够做的事情比较有限。其中最大的问题是不能适配注入 Istio 的 Sidecar Annotation，这就导致 Merbridge 无法排除某些端口或 IP 段的流量等。同时，由于之前 Merbridge 只处理 Pod 内部的连接请求，这就导致，如果是外部发送到 Pod 的流量，Merbridge 将无法处理。

- [20220606 | DCE 5.0 研发背景](#)

简述新一代云原生操作系统 DaoCloud Enterprise 5.0 诞生的背景。

- [20220530 | 云原生布局开始加速，市场前景如何？收好这份指南](#)

本文以 Infographic 模式图解说明了当前快速发展的云原生与云计算浪潮。

- [20220520 | 走进可观测性 | 论道原生](#)

2018 年起可观测性 (Observability) 被引入 IT 领域，CNCF-Landscape 组织创建了 Observability 的分组。自此，可观测性逐渐取代传统的系统监控，从被动监控转向主动观测应用关联的各类数据，成为云原生领域的最热门话题之一。

[^1]: CNCF：全称 Cloud Native Computing Foundation（云原生计算基金会），隶属于 Linux 基金会，成立于 2015 年 12 月，是非营利性组织，致力于培育和维护一个厂商中立的开源生态系统，来推广云原生技术，普及云原生应用。[^2]：云原生全景图：由 CNCF 从 2016 年 12 月开始维护，汇总了社区成熟和使用范围较广、具有最佳实践的产品和方案，并加以分类，为企业构建云原生体系提供参考，在云生态研发、运维领域具有广泛影响力。

## 2. 5.0 应用工作台能力介绍

如今随着云原生理念迅速普及，云原生技术给企业数字化转型的过程中带来了巨大的便利。但是企业云原生过程中面临着一些列挑战：

- 多租户设计下如何满足逻辑层面的资源隔离？如何满足物理资源的隔离？
- 频繁的持续构建，如何及时在更早的阶段找到问题、修复 bug，持续保证代码和环境符合预期？
- 云原生下的技术种类复杂，如何利用好云原生建设云原生下的 CI/CD 系统？
- 云原生技术带来了应用部署复杂性，如何降低开发者的认知负担？
- 全自动的应用部署，如何让新功能发布可控、可预测、可逆？

面对这样的挑战，DCE 5.0 基于 Kubernetes 等云原生技术打造的应用工作台应运而生，集 DaoCloud DevOps 实践、云原生社区前沿研发理念于一体。为企业建立一套自动化的从源代码构建到持续部署应用，且可以覆盖云原生下所有形态应用全生命周期管理的开发运维一体化的环境。帮助企业缩短研发周期，提供应用交付效率。

### 2.1 图形化用户交互

数字经济时代，云原生的应用已成为企业数字化转型的必选项，所以应用工作台以应用为中心，基于 DevOps 理念，解决云原生应用开发、自动化交付、运维全生命周期。降低企业应用云原生门槛，提高应用交付频率。应用工作台涉及：层级多租户资源、云原生应用、流水线CI/CD 流水线、GitOps、渐进式交付等模块。

用户在使用过程中主要存在三个阶段：开发阶段、交付阶段、运维阶段。

1. 在开发阶段：应用工作台提供了流水线功能，企业可以在流水线中定义编译代码、代码检查、构建云原生制品流程，帮助开发者及时测试代码，从而提高代码质量。另外整个流水线一次设置，多次复用，大大降低了开发人员的工作负担，减少了不必要的重复劳动。
2. 在交付阶段：应用工作台给予目前GitOps 理念，以自动化方式，频繁且持续性的将企业的应用部署到生产环境。作为上一阶段的延伸，自动化的持续部署起到了承上启下的作用，是对软件的研发的全流程管控的闭环。应用工作台可自动且持续监测代码仓库与生产环境中的应用进行比对，来确保生产环境与仓库中的期望状态保持一致，以实现全自动化部署。另外应用工作台还支持在持续部署的基础上对接外部能力，从而实现渐进式交付下的灰度发布等高级发布策略。
3. 在运维阶段：应用工作台针对多种形态的云原生应用，提供了统一的观测平面，包含监控、告警、日志等信息。还提供了对应用升级、回滚、停止和删除应用等功能。

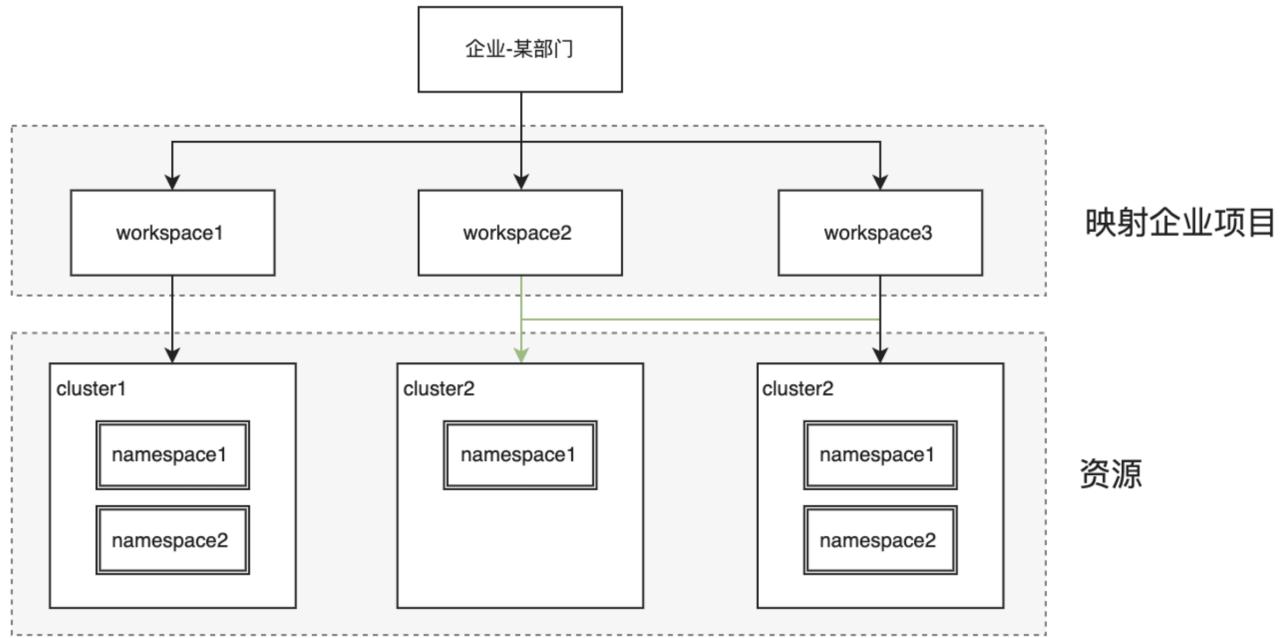


## 2.2 功能特性

应用工作台提供了层级资源管理、云原生应用、CI/CD流水线、GitOps、渐进式交付核心服务，来满足企业不同场景的需求。

### 层级资源管理

在企业环境中，企业下的每一个部门由不同的项目构成，一个一个的项目就映射到了我们的工作空间（workspace）。下图是应用工作台提供了基于 Kubernetes 的多租户管理方案。



前提条件：用户可以在5.0全局管理模块被设置为工作空间的管理员来参与工作空间的管理，从而参与项目的协同。

特点：

- 工作空间是最小的租户单元。
- 工作空间支持弱绑定集群的模式，从而使用户获得跨集群、跨命名空间共享资源的能力。
- 工作空间支持强绑定集群的模式，从而使用户获得独享集群资源的能力。
- 工作空间的管理员可以在该工作空间关联的集群中创建命名空间资源，并为命名空间进行配额管理。

支持多种形态云原生应用的部署，包括源代码、Jar 包、镜像一键完成应用部署，并且在创建过程中可使用微服务的注册发现、服务治理能力，还支持通过应用商城一键部署 Helm 应用、OLM 应用。降低了企业上云的门槛，简单易用、敏捷高效。

### 云原生应用

应用工作台以应用为中心，涵盖了云原生技术下多种形态的应用，应用工作台对云原生社区内的应用采取了更包容的态度，而不是按照自己对应用的理解，在平台中定义一套自己的应用规范。旨在帮助用户无需学习云原生知识，即可“用好云”。

目前应用工作台支持的云原生应用有：



应用工作台面向应用开发和运维，覆盖应用全生命周期，包括应用的创建、删除、配置修改、自动扩缩及自动运维。并且支持 SpringCloud、Dubbo、ServiceMesh 服务治理架构，与 5.0 微服务引擎、5.0 服务网格无缝集成：

选择框架/服务网格

Spring Cloud
 DUBBO
 Service Mesh

微服务框架  
Spring Cloud
微服务框架  
Dubbo
服务网格  
Service Mesh

选择注册中心
开启服务治理
开启可观测

NACOS.
 Sentinel
 Istio

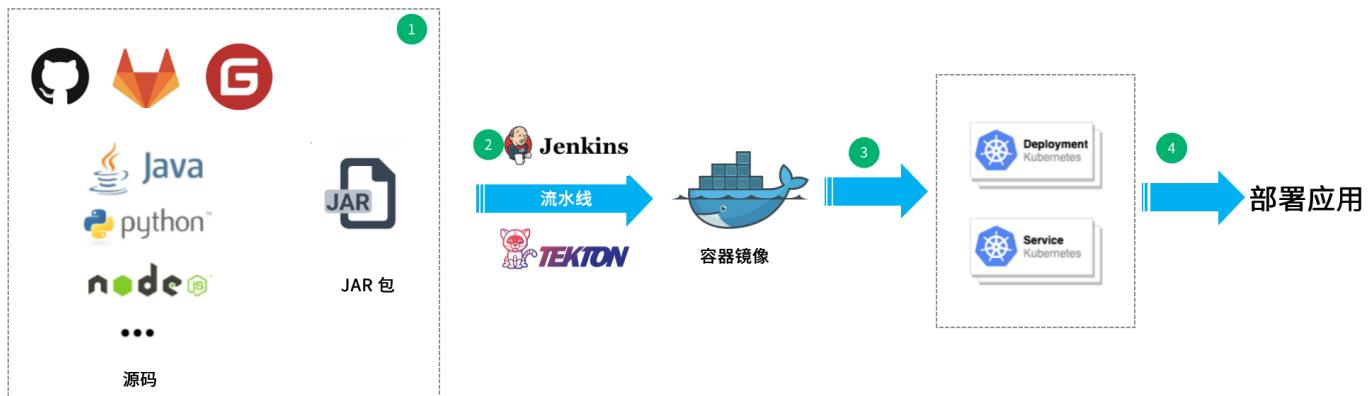
监控
日志
链路追踪

DEC 5.0 微服务引擎
 DEC 5.0 服务网格
 DEC 5.0 可观测

## CI/CD 流水线

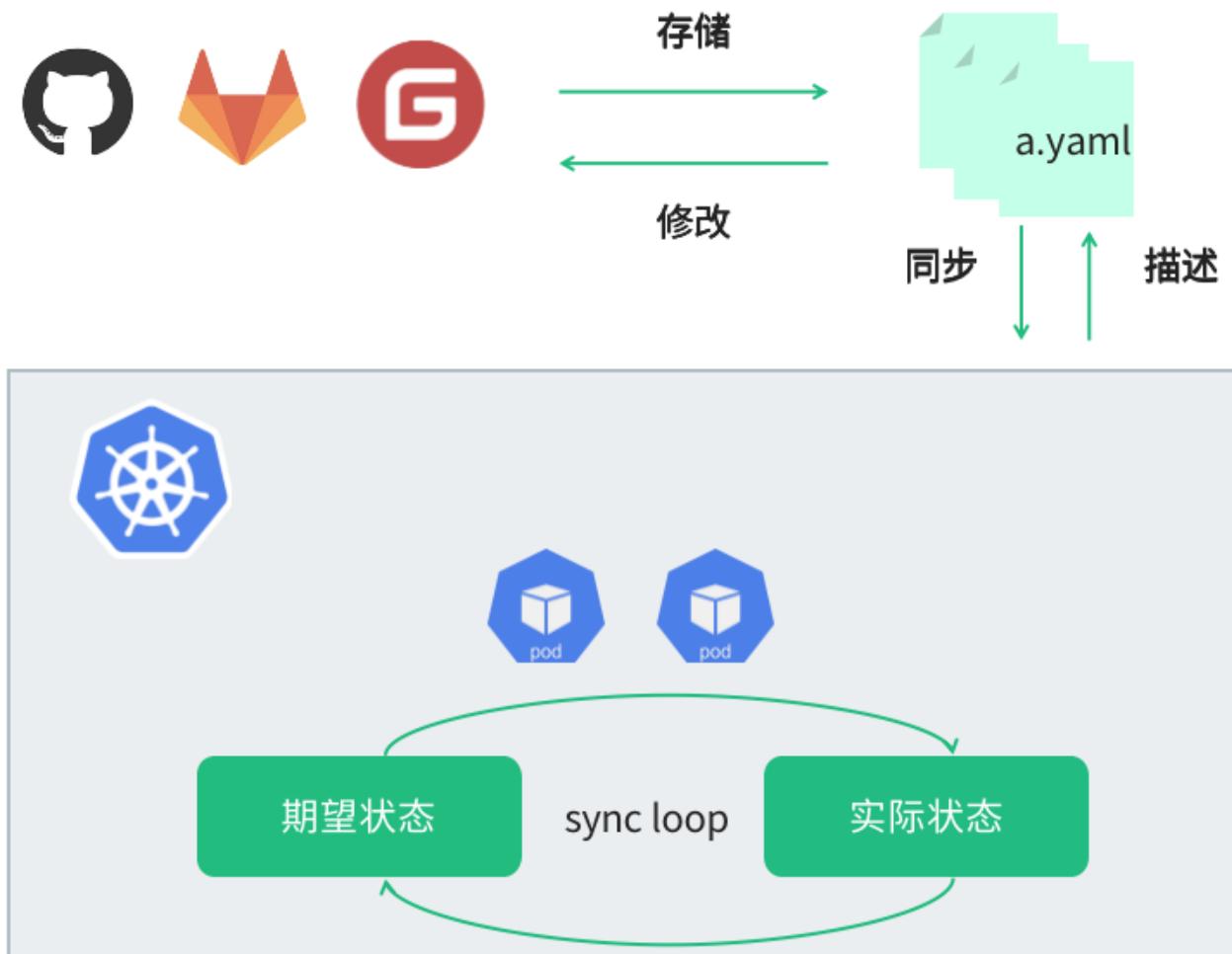
流水线是一个自定义的 CI/CD 流水线模式，定义了包含构建、测试和发布的完整构建过程。应用工作台基于流水线能力具有以下优势：

- 不同来源构建应用：支持使用源码、软件包（Jar）实现应用的一键部署。
- 双引擎：支持 Jenkins 和 Tekton 作为应用工作台的流水线系统引擎。
- 丰富的流水线模板：内置了多条流水线官方模板，大大降低了用户的使用门槛，可以适应不同业务场景，满足用户的日常需求。



## GitOps

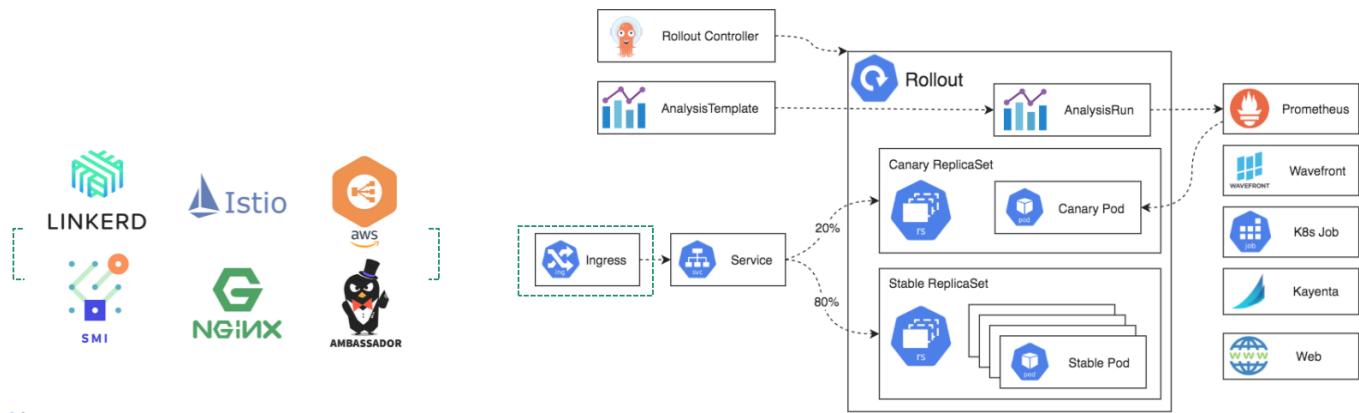
GitOps 是一种为云原生应用实现持续部署的理念。应用工作台全面拥抱 GitOps，GitOps 的核心思想是拥有一个 Git 仓库，并将应用系统的声明式基础架构和应用程序存放在 Git 仓库中进行版本控制。GitOps 结合 Kubernetes 能够利用自动交付流水线将更改应用到指定的任意多个集群中，从而解决跨云部署的一致性问题。



## 渐进式发布

为了应用部署的安全性，企业都采取了灰度发布的解决方案，这种部署策略使应用部署更加安全，但是仍然缺乏自动化。将服务发布到生产和分析指标的过程仍然是一个手动过程。这将促使我们进入下一阶段，即渐进式交付。

应用工作台基于 Argo Rollouts 实现了渐进式的灰度发布，可以让开发人员可以选择他们自己的分析指标，自定义他们的渐进式发布的步骤，甚至选择他们自己的入口或服务网格提供商来进行流量控制。



## 2.3 常见问答

### 1. 什么是 DevOps

DevOps 是开发 (Dev) 和运营 (Ops) 的综合体，是指将开发、质量保证和 IT 运营集成到统一的环境以及软件交付过程中。通过采用 DevOps 文化、做法和工具，企业能够更好地响应业务需求、更快的构建应用程序，并更快地实现业务目标。

DevOps 包括以下活动和操作：

- 持续集成 (CI) 是指如下的做法：将所有开发人员代码频繁合并到一个中央代码库中，然后执行自动化生成和测试过程。目标是快速发现和纠正代码问题、简化部署并确保代码质量。
- 持续交付 (CD) 是指如下的做法：自动生成、测试代码并将其部署到类似生产的环境。目标是确保代码始终可供部署。添加持续交付来创建完整 CI/CD 流水线的做法有助于尽早检测到代码缺陷。并确保能够在极短的时间内发布已经过适当测试的更新。
- 持续部署 是一个附加的过程，可自动引入通过 CI/CD 流水线传递的所有更新，并将其部署到生产环境。持续部署需要可靠的自动测试和高级过程规划，不一定适合所有团队。
- 持续监视 是指在 DevOps 和 IT 运营生命周期的每个阶段中整合监视功能所要采用的流程和技术。监视有助于确保应用程序在从开发环境转移到生产环境时，应用程序和基础结构保持正常的运行状况、性能和可靠性。持续监视是基于 CI 和 CD 的概念。

### 2. 什么是流水线？

流水线可以自动生成和测试代码项目，使其可供他人使用。适用于任何语言或项目类型。将持续集成 (CI) 和持续交付 (CD) 组合在一起，可以测试并生成代码，并将代码发送给任何目标。应用工作台还支持可视化的方式来编排流水线，企业可以将现有研发流程通过流水线以可视化的方式呈现出来。

### 3. 流水线如何构建一个应用程序？

应用工作台提供了从 Git 导入代码、上传 Jar 包来创建应用程序。在创建的过程中，只要填写流水线所需的部分信息，应用工作台就会帮您创建一条流水线资源，自动化完成从源码构建、测试、发布应用至 Kubernetes 集群环境中。旨在帮助企业以更快的速度将软件持续交付，同时降低风险。

### 4. 支持哪些云原生应用？

Kubernetes 原生应用、Helm 应用、OAM 应用、OLM 应用。

### 5. 如何运维多种形态的云原生应用？

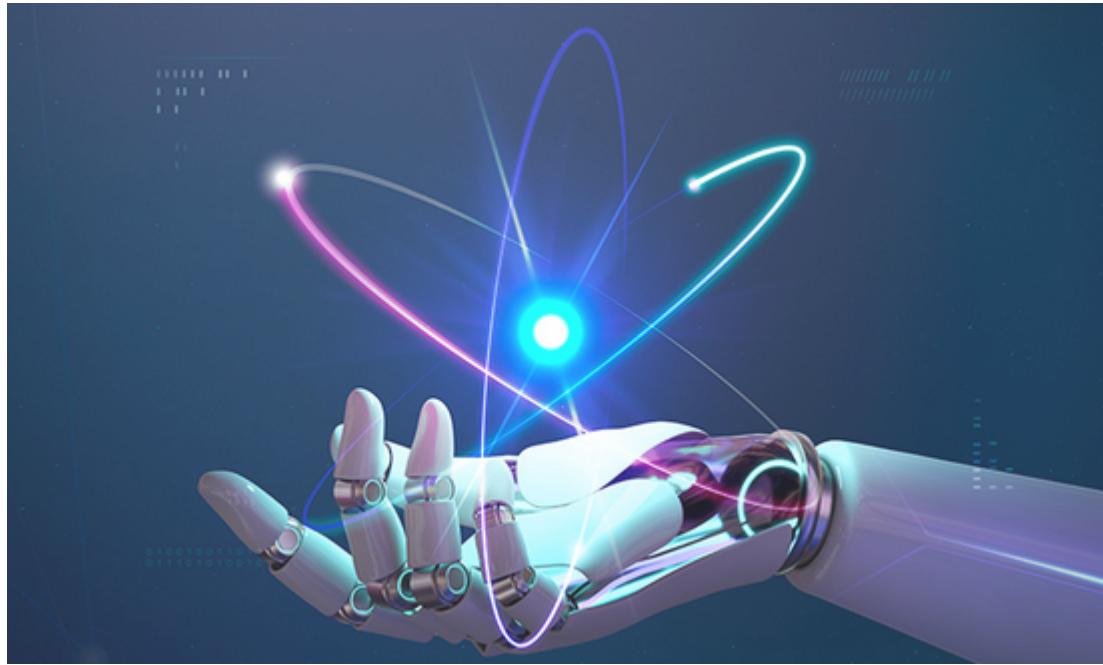
应用工作台推出以应用为中心，统一观测平面，包含监控、告警、日志等信息。让不同形态的云原生应用运维更轻松方便。

[了解应用工作台](#)

[下载 DCE 5.0](#) 安装 DCE 5.0 申请社区免费体验

### 3. DCE 研发背景

时下元宇宙概念方兴未艾，预期万物皆有虚拟化身，行业巨头正在模仿现实世界构建沉浸式数字体验，建立另一个完整的数字平行时空。现实中的一切都将在云上虚拟碎片化并被赋予真实的价值，分布式微服务微场景随处可见。以往游戏世界最大在线人数为数百万量级，但宏大逼真的数字世界将是亿级人口在活动，这将产生海量的并发数据，促使云算力、云存储和云带宽再一次升级换代，各种新兴云原生技术必将在此基础上喷涌而出，一切都在云上，一切为云而生。



云原生容器化平台正迅速成为现代应用架构的基础。据 IDC 预测，到 2022 年，将有 70% 的企业跨多个平台部署统一的虚拟机、容器、多云管理流程和工具。同样到 2022 年，全球组织 / 公司在生产环境运行容器化应用，将从今天不足 30% 的比例大幅提升到超过 75%。而据 Gartner 预测，到 2025 年，云原生平台将成为 95% 以上新数字倡议的基础，而在 2021 年这一比例只有不到 40%。

[申请社区免费体验](#)

#### 3.1 云原生浪潮

何谓云原生？英文名为 Cloud Native，这是 IT 行业资深老兵 Matt Stine 提出的一个概念，包括 DevOps、CI/CD、微服务、容器化等。从字面理解，云原生指一切都在云上。所有应用程序从设计之初就要考虑云环境，即原生为云设计，在云上以最佳姿势运行，充分利用和发挥云平台的弹性 + 分布式优势。

据艾瑞市场咨询调研报告称，2019 年开始云原生容器技术开始进入规模应用期，容器技术与国内欣欣向荣的云计算产业发展紧密结合，为企业提供更高效、敏捷和可靠的容器云服务。容器云架构的敏捷、轻简和高度兼容性使得容器成为云原生生态中最基础的一环，无论是混合云/多云，还是 DevOps、微服务应用的推进，容器都将扮演至关重要的角色，助力企业数字化转型和降本增效。

DaoCloud 多年深耕容器云技术，是国内容器领域的佼佼者。早在 2013 年中旬 Docker 在 GitHub 上开源以来，DaoCloud 几位创始人就敏锐洞察到容器作为一种充满活力和可塑性的技术，其未来的应用前景必将非常可观。几十位开发人员经过两年多日夜奋战，2016 年初推出 DaoCloud Enterprise (DCE) 容器应用云平台 1.0，以容器化平台产品赋能 DevOps、微服务、无服务器等云原生应用的快速进步，助力企业敏捷应对高频在线服务、工业互联网、金融数字化转型和边缘计算等场景。

如今，日益成熟完善的 DCE 版本 Roadmap 已演进至 5.0，成为业界领先的企业级 Kubernetes 商用平台、云原生操作系统，内含 Parcel 智能网络、UDS 统一数据存储、一体化图形监控和多集群管理等模块，经过国内外数百家金融政企生产验证，能够提供可靠、一致的 PaaS 支撑环境，让企业应用开发者专注于业务能力建设，保持竞争力并超预期达成与日俱增的客户期望。而 DaoCloud 公司也发展至近 500 名员工，研发人员比例超过 70%，汇集了国内外大厂的顶尖人才专注于容器技术，建立了从售前、产品、开发、测试、交付、售后一系列完整的容器技术流水线体系，竭诚为国内外客户服务。

# 云原生地图 全球社区影响力

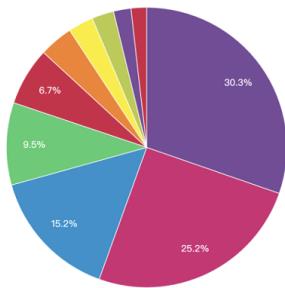
Kubernetes



Commits by Company

Show [10] entries	Search	
#	Company	Commits
1	Independent	1555
1	Google	1291
2	Red Hat	778
3	DaoCloud	343
4	ZTE Corporation	200
5	VMware	149
6	Intel	130
7	IBM	103
8	Huawei	92
9	Microsoft	88

Showing 1 to 10 of 43 entries Previous Next



Kubernetes 排名全球第 3 名 (2020.12 – 2021.12)

Kubernetes 1.22 版本增量贡献排名中国第 1

中国企业 Kubernetes 贡献数排名第 1

容器引擎



中国第 2 名  
全球第 9 名

容器引擎(替代)



中国第 2 名  
全球第 9 名

网格



中国第 2 名  
全球第 8 名

监控



中国第 1 名  
全球第 9 名

数据来源：<https://www.stackalytics.io/>

2021-12-18

2020 年 7 月，Gartner 2020 中国 ICT 技术成熟度曲线报告作为专业容器厂商提及 DaoCloud。容器技术的主体力量在开源社区，而开源社区的贡献度排名可以证明容器厂商对容器技术的深度理解。DaoCloud 近几年在全球开源社区的影响力斐然，stackalytics.io 统计数据表明，过去一年来 DaoCloud 对 Kubernetes 的全球排名第 3，国内第 1。这充分证明了 DaoCloud 开发团队对云原生技术的深刻理解，从网络、存储、算力等资源编排，到大屏可视化监控，再到多集群管理和客户场景化解决方案定制，DaoCloud 全面引领国内千行百业实现云原生数字化转型，与华为云和近百家国内厂家建立战略合作关系，响应国家号召构建信创产业生态。

## 3.2 容器 vs 虚拟化

这两者不是并行，而是此消彼长的替代关系，容器是未来。

物理机时代

虚拟机时代

现代容器化时代



多个应用跑在一台物理机上

一台物理机运行几个虚拟机  
一个虚拟机跑多个应用

一台物理机运行数百个容器  
一个容器跑应用的一个服务

在过去的 15 年内，服务器虚拟化是企业数据中心应用部署的首选方法。Hypervisor 虚拟机管理程序几乎无处不在。公有云 IaaS 服务商最常见的售卖单位是虚拟机 (VM)。然而在云原生时代，企业为了改善应用访问量和用户体验，需要寻求更快、更灵活的方法来部署和管理新的应用和服务。很多应用需要同时部署在企业内部和云环境中。另外企业为了加快新应用的上线速度，除了在虚拟机中运行单体应用外，还要部署容器化应用。

问题是如果一个企业准备采用容器，就需要努力了解什么基础设施方案最能适合业务需求。大多数企业考虑到曾投入在虚拟机方面的资金以及 IT 团队在虚拟机管理方面的多年经验，自然就想在虚拟机环境中运行容器。很多企业的 IT 团队就是这么做的。诚然这是一个熟悉容器的好方法。但是，随着容器化进程从概念验证发展到开发和测试，再到生产部署，IT 团队逐渐发现基于虚拟机部署的容器并不理想。

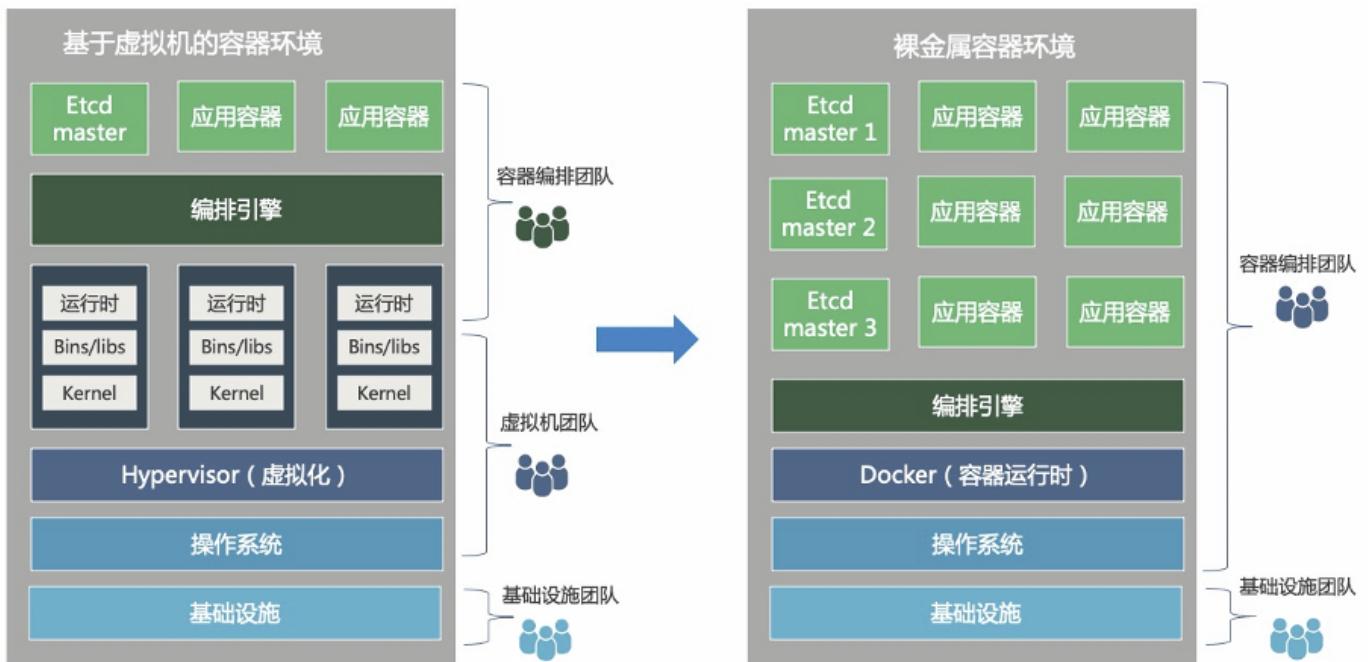
随着云原生技术的成熟和流行，企业们开始尝试将容器直接部署在物理机上，也称为裸金属容器，这种部署方式具有以下优点：

- 管理层次更少，故障排查更简便
- 效率更高
- 运行的容器密度更高
- 性能更好
- 可预测性更佳
- 总拥有成本更低

若在现有的虚拟机环境之上部署容器，容器在其中只是另一种形式的虚拟层。这样就需要两种 IT 运维人员，一种使用和管理容器环境，另一种管理虚拟机环境，运维很可能需要两套班子。而在实际工作中，无论容器和虚拟机两个团队合作得多好，都会不可避免地出现沟通问题、重复劳动，经常出现一个团队等待另一个团队的情况，不可避免会产生延迟，降低总体工作效率。

此时一旦出现故障，故障排查过程就比较复杂。衍生的问题有：故障能否在容器层面解决？是虚拟机引起的问题吗？还是物理硬件的问题？如果要打电话寻求售后支持，怎么才能说清楚哪儿出了问题？另外虚拟机是否能有效支持容器栈？

裸金属容器与基于虚拟机的容器相比，减少了需要管理的层数，而且由于裸金属的效率更高，所以运行相同数量的容器需要的硬件资源更少，减少了管理的设备总量。



从上图可看出，裸金属容器部署、管理和故障排查需要的团队人员更少，但支撑的应用密度更高。

DaoCloud 把握时代脉搏，以 DCE 容器应用云平台为核心，匠心打造的云原生一体机采用智能裸金属 + 领先容器技术的基础架构，基于原生的开源容器技术，与全球标准与规范完全同步，持续向企业输出业界最佳实践。能够在差异化的集群运行环境内，适配裸金属、虚拟机和云主机等企业多样化 IT 基础设施，支持多种虚拟平台，实现企业多元化基础设施之上的统一资源调配。

### 3.3 DCE 与 Kubernetes

DCE 包含企业需要的容器、网络、存储、监控、镜像仓库、负载均衡、DNS 服务发现、身份验证和授权解决方案。这些组件经过全面的测试，可在裸金属环境、公有云、私有云和混合云等新型动态环境中可靠运行。

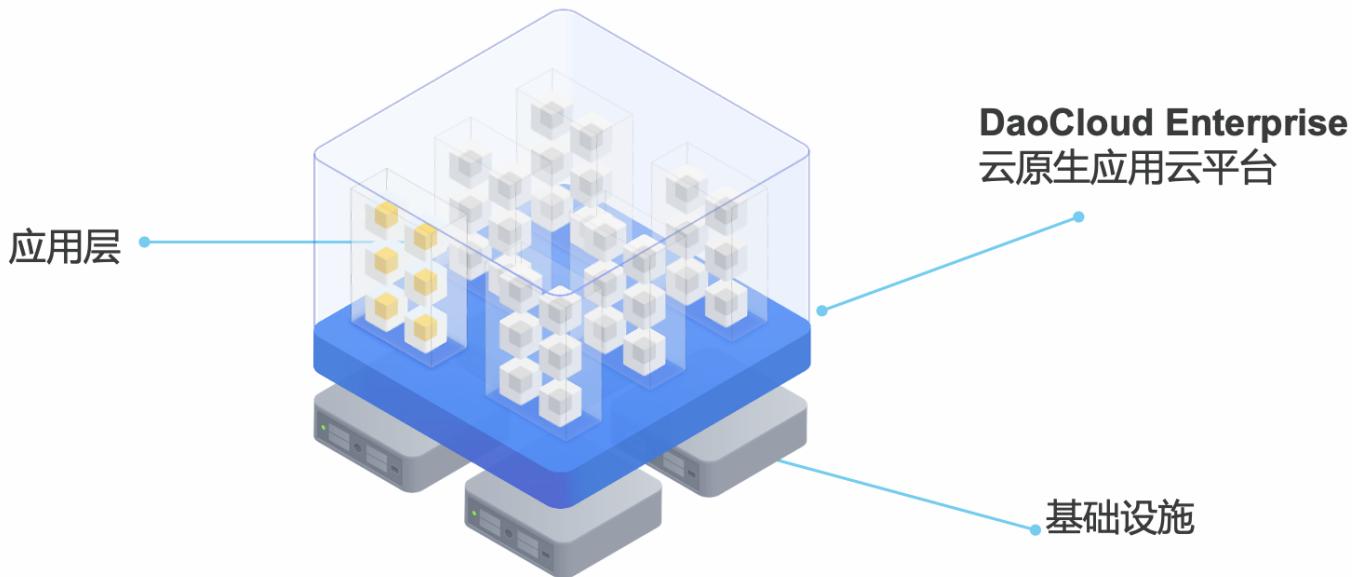
- 容器计算，适配 x86 架构以及 ARM 架构，承载混合应用的运行、调度及管理。
- 容器存储，提供面向应用的数据持久化能力，助力有状态应用负载。
- 容器网络，为应用提供高性能双态网络，兼顾传统企业基础架构和软件定义基础架构。

DCE 内置 260 项容器平台风险检查点，多维度的集群监控告警前沿实践，保障主机、集群服务、业务应用的持续运行，整体的平台高可用设计以及全面的资源保障模式（QoS），实现端到端的应用安全保障，提供高 SLA 保障，可用性达 99.99%。

DCE 以技术引领为己任，帮助企业客户落地云原生技术，并且在实践中不断加固、优化和扩展 Kubernetes，使其满足企业级、大规模、高度稳定的生产环境要求。同时，团队作为国内 Kubernetes 开源社区领导者之一，对核心开源技术有着深度理解，通过不断总结生产实践回馈开源社区，推动 Kubernetes 社区的不断发展。

### 3.4 DCE 产品简介

DCE 采用业界领先的容器编排技术，产品能力涵盖云原生应用的开发、交付、运维和运营全生命周期管理，能够帮助金融政企实现分布式业务应用的敏捷交付、弹性支撑和治理，提升企业对分布式应用的支撑和治理能力。同时，产品已经与国内主流信创合作伙伴构建「信创架构+云原生平台」的端到端云原生整体解决方案。目前已经为金融、证券、工业互联、车联网、智能制造、电商在内的数百个重点行业头部客户提供了围绕云原生技术的容器云解决方案，并且相应的信创容器云解决方案已经在政务、金融等多个行业落地。



对应用开发人员而言，可以把 DCE 看成是一个集群操作系统。DCE 提供服务发现、扩缩、负载均衡、自愈甚至选举等功能，让开发人员从基础设施相关配置等解脱出来。DCE 可以把大量的服务器看做一台巨大的服务器，在一台大服务器上面运行应用程序。无论云原生集群有多少台服务器，在 DCE 上部署应用程序的方法永远一样。

### 3.5 经 CNCF 认证的 KCSP

目前 DaoCloud 经授权合规的 Kubernetes 版本包括但不限于：

当前版本：

[1.23](#) [1.24](#) [1.25](#) [1.26](#)

历史版本：

[1.7](#) [1.9](#) [1.13](#) [1.15](#) [1.18](#) [1.20](#)

进一步了解什么是 DCE 5.0

[下载 DCE 5.0](#) 安装 [DCE 5.0](#) 申请社区免费体验

## 4. 「DaoCloud 道客」与 Kubernetes

为数字世界寻找全局最优解

### 4.1 挑战

「[DaoCloud 道客](#)」云原生领域的创新领导者，成立于 2014 年底，拥有自主知识产权的核心技术，致力于打造开放的云操作系统，为企业数字化转型赋能。这样的目标与使命决定了公司从创立之初就在云原生的世界里耕耘。与传统业务场景不同的是，云原生业务离不开容器，云操作系统更是依赖容器作为基础设施。因此「[DaoCloud 道客](#)」面临的首要挑战就是如何高效管理调度多个容器，如何保证容器间的正常通信。

此外，如今的云原生处在高速发展时期，各类云原生技术方案层出不穷且各有利弊，令人眼花缭乱。用户期望的并非解决眼前问题的局部最优解，而是要寻找全局最优解。如何整合这些独立的项目，取长补短，构建整体的云原生解决方案，这是「[DaoCloud 道客](#)」面临的又一挑战，也是云原生行业的发展难题。

### 4.2 解决方案

Kubernetes 作为容器编排的事实标准，无疑是首选的容器解决方案。「[DaoCloud 道客](#)」架构师兼开源团队负责人徐俊杰（Paco）表示“Kubernetes 是目前容器生态里面比较基础的一环，绝大多数服务都是基于 Kubernetes 部署的，应用绝大多数都是在 Kubernetes 集群中运行和管理。”

面对层出不穷的技术方案，「[DaoCloud 道客](#)」研发副总裁潘远航（Peter）认为，“在众多技术面前，坚持以 Kubernetes 为核心，整合周边最佳实践和先进技术，打造一个适合的平台和方案，才是寻找全局最优解的正确路径。”

### 4.3 影响

在拥抱云原生的过程中，「[DaoCloud 道客](#)」不断向 Kubernetes 等优秀的 CNCF 开源项目学习，逐渐形成了以 [DaoCloud Enterprise](#) 云原生应用云平台为核心的产品架构。「[DaoCloud 道客](#)」坚持以 Kubernetes 等世界领先的云原生技术为支点，为军工、金融、制造、能源、政务、零售等垂直行业提供了前沿的云原生解决方案，为浦发银行、华泰证券、富国基金、上汽集团、海尔、复旦大学、屈臣氏、吉致汽车金融、国家电网等各行各业的优秀企业都量身定制了满意的数字化转型方案。

!!! note ""

“随着 [DaoCloud Enterprise](#) 越来越强大，客户覆盖面越来越广，有些客户需要使用 Kubernetes 而不是 Swarm 进行应用编排。Kubernetes 作为一套功能强大的应用编排系统，有着强大的社区支持，也受到很多大公司的青睐。我们作为提供商需要满足用户的需求。”

[刘齐均 (Kebe) ] (<https://github.com/kebe7jun>)，「[DaoCloud 道客](#)」服务网格专家

「[DaoCloud 道客](#)」成立的初衷就是帮助传统企业进行数字化转型，实现应用上云。公司成立之后发布的首秀产品 [DaoCloud Enterprise 1.0](#) 便是一款基于 Docker 的容器引擎平台，可以轻松打包构建镜像并运行容器。随着应用数量的增加，容器越来越多，如何协调和调度这些容器逐渐成为制约产品性能的主要瓶颈。[DaoCloud Enterprise 2.0](#) 开始采用 Docker Swarm 管理容器，但随着容器调度系统越来越复杂，Docker Swarm 也开始显得力不从心。

此时恰逢 Kubernetes 崭露头角，凭借多样的功能、稳定的性能、及时的社区支持、强大的兼容性等优势迅速发展成为容器编排的业界标准。Paco 表示“企业容器平台需要容器编排来规范化应用上云的过程。Kubernetes 在 2016 – 2017 年逐渐成为容器编排的事实标准，我们在 2017 年就开始同时支持 Docker Swarm 和 Kubernetes 了。”

经过一系列的评估，2017 年发布的 [DaoCloud Enterprise 2.8](#) 版本开始正式采用 Kubernetes (v1.6.7) 作为容器编排工具。此后，2018 年发布的 [DaoCloud Enterprise 3.0](#) 采用 Kubernetes 1.10 版本，2021 年发布的 [DaoCloud Enterprise 4.0](#) 采用 Kubernetes 1.18 版本。2022 年发布的 [DaoCloud Enterprise 5.0](#) 支持 Kubernetes 1.23 至 1.26 版本。

六年时间里发布的四个主要版本一直都在坚定不移地使用 Kubernetes，这足以说明当时的选择是正确的。「[DaoCloud 道客](#)」用实际经验证明了 Kubernetes 是容器编排的最佳选择，也用自身行动证明了自己一直都是 Kubernetes 的忠实拥趸。

!!! note ""

“借力 K8S 的价值体系『自动化大于人工』，产研团队从0到1完成了从研发构建自动化，测试自动化，安全自动化，发布自动化保证了软件交付质量，其次实现了智能化协作沟通，包括产品需求及定义体系、产品多语言协作体系、产品缺陷修复协作体系、疑难杂症攻坚体系，极大的提升了产研同部门、跨部门的协作效率，这是我们走向世界一流基础设施软件产品的基石。”

叶挺，「DaoCloud 道客」产品创新副总裁

在 Kubernetes 的助力下，「DaoCloud 道客」的产品性能更优，更具竞争力。「DaoCloud 道客」坚持以 Kubernetes 为核心，整合周边最佳实践和先进技术，打造出 DaoCloud Enterprise 云原生应用云平台，提供应用商店、应用交付、微服务治理、可观测性、数据服务、多云编排、信创异构、云边协同等能力。DaoCloud Enterprise 5.0 是集云原生技术大成的完全形态。

- 「DaoCloud 道客」为上海浦发银行部署 Kubernetes 平台后，应用部署效率提升 82%，交付周期从半年缩短到一个月，交易成功率达到 99.999%；
- 四川天府银行落地基于 Kubernetes 的云原生平台，将弹性响应时间由数小时大幅缩减到平均 2 分钟，产品迭代周期从两个月缩短为两周，应用上线时间缩短 76.76%；
- 为某合资车企搭建基于 Kubernetes 的云原生平台后，将其交付周期从两个月缩短到一两周，应用部署成功率提升 53%，应用上线效率提高 24 倍；为某跨国零售集团部署基于 Kubernetes 的多个云原生平台模块，为其减少了 46% 的应用部署问题，将监控定位效率提升 90% 以上；
- 为某大型综合类券商搭建统一的云原生 PaaS 平台，使其业务流程效率提升 30%，资源成本节约 35% 左右；
- 为富国基金打造基于 Kubernetes 的新一代云原生 PaaS 平台，将标准中间件部署时间从数小时缩短至数分钟，中间件运维能力提升 50%，容器化程度提升 60%，资源利用率提升 40%。

另一方面，「DaoCloud 道客」自身的产品研发工作也是基于 Kubernetes 进行的。公司基于 Kubernetes 部署了 Gitlab，形成了“Gitlab → PR → 自动化测试 → 构建发布”的产品开发流程，显著提升了开发效率，减少了重复测试的工作量，实现了应用的自动发布。这样一来，大大节省了运维成本，技术人员可以为开发产品投入更多的时间与精力，打磨出更优秀的云原生产品。

!!! note ""

“我们的开发者很踊跃地贡献开源，沉淀技术实力，在 Kubernetes 和 Istio 社区都有越来越多的贡献。公司第五代产品走的也是开源路线，为云原生技术添砖加瓦，完善技术生态。”

徐俊杰 Paco，「DaoCloud 道客」架构师/开源 & AD 团队负责人

「DaoCloud 道客」深度参与贡献 Kubernetes 等多项云原生开源项目，在云原生开源社区中的参与度、贡献度持续增长。在过去一年里，「DaoCloud 道客」在 Kubernetes 的开源榜单累计贡献度位居全球第三（基于 Stackalytics 网站 2023/01/05 的数据）。

在 2022 年 8 月由 Kubernetes 官方组织的社区贡献者访谈活动中，接见了来自亚太地区的 4 位优秀贡献者，其中 Shiming Zhang 和 Paco Xu 都来自「DaoCloud 道客」，二人均是 SIG Node 的 Reviewer。此外在 2022 Kubecon 北美站上，「DaoCloud 道客」的 Kante Yin 荣获 Kubernetes 2022 年度贡献者奖。

此外，「DaoCloud 道客」也在坚持践行云原生信仰，持续回馈云原生社区，开源了 Clusterpedia、Kubean、CloudTTY、KLTS.io、Merbridge、HwameiStor、Spiderpool、Piraeus 等优秀项目，不断完善 Kubernetes 生态体系。其中：

- Clusterpedia 兼容 Kubernetes OpenAPI，实现了多集群资源的同步，提供了更强大的搜索功能，可以快速、轻松、有效地获取集群内所有资源信息。
- Kubean 支持快速创建 Kubernetes 集群以及其他厂商的集群。
- CloudTTY 是专为 Kubernetes 云原生环境打造的 Web 终端和 Cloud Shell Operator，可以通过一个 Web 页面随时随地管理 Kubernetes 集群。
- KLTS 为 Kubernetes 早期版本提供长期免费的维护支持。
- Piraeus 是适用于 Kubernetes 的高性能、高可用性、简单安全的存储解决方案。

「DaoCloud 道客」融合自身在各行各业的实战经验，持续贡献 Kubernetes 开源项目，致力于让以 Kubernetes 为代表的云原生技术更平稳、高效地落地到产品和生产实践中。

!!! note ""

“「DaoCloud 道客」作为首批 CNCF 官方认证的云原生技术培训伙伴，将持续开展赋能培训、项目指导等活动，携手伙伴，为客户导入云原生，共同打造云原生能力的最佳实践路径。”

郑松，「DaoCloud 道客」中国区技术总经理

「DaoCloud 道客」研发副总裁潘远航（Peter）认为“企业用户需要的是一个全局最优解，这个最优解可以理解为是涵盖多云编排、信创异构、应用交付、可观测性、云边协同、微服务治理、应用商店、数据服务等能力的最大公约数。”在如今的云原生生态体系里，这些功能都离不开 Kubernetes 作为底层的容器编排技术。这就意味着「DaoCloud 道客」在寻找数字世界最优解的过程中也离不开 Kubernetes，未来的产品研发也将继续以 Kubernetes 为基础。

此外，「DaoCloud 道客」一直致力于 Kubernetes 的培训、推广活动。2017年，公司凭借核心产品云原生应用云平台 DaoCloud Enterprise 成为全球首批通过 CNCF Kubernetes 兼容性认证的厂家。2018年，公司成为 CNCF 认证的 Kubernetes 服务提供商，并成为全球首批获得CNCF官方认证的 Kubernetes培训合作伙伴，全面拥抱 Kubernetes 技术生态。

2022 年 11 月 18 日，由 CNCF 和「DaoCloud 道客」、华为云、四川天府银行、OPPO 联合发起的「Kubernetes Community Days 成都站」成功举办，聚集了来自云原生领域开源社区的最终用户、贡献者和技术专家，分享关于云原生的多行业实践、热门开源项目、社区贡献心得等丰富内容。未来，「DaoCloud 道客」将继续为 Kubernetes 贡献自己的力量，通过项目培训、社区贡献等活动不断扩大 Kubernetes 的影响力。

## 5. 保姆式安装 DCE 5.0 社区版

作者: [SAMZONG](#)

本文完成了从 0 到 1 的 DCE 5.0 社区版安装, 包含了 K8s 集群、依赖项、网络、存储等细节及更多注意事项。

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。

### 5.1 集群规划

计划使用 3 台 UCloud 的 VM, 配置均为 8 核 16G。

角色	主机名	操作系统	IP	配置
master	master-k8s-com	CentOS 7.9	10.23.245.63	8 核 16G 300GB
node01	node01-k8s-com	CentOS 7.9	10.23.104.173	8 核 16G 300GB
node02	node02-k8s-com	CentOS 7.9	10.23.112.244	8 核 16G 300GB

计划采用的集群组件为:

- Kubernetes 1.24.8
- CRI containerd
- CNI Calico
- StorageClass HwameiStior

## 5.2 节点系统优化

---

安装前先对 3 个节点做了一些优化。

### 1. 配置主机名

```
bash
hostnamectl set-hostname master-k8s-com
```

### 2. 添加 /etc/hosts 配置

```
bash
cat <<EOF | tee /etc/hosts
10.23.245.63    master-k8s-com
10.23.104.173   node01-k8s-com
10.23.112.244   node02-k8s-com
EOF
```

### 3. 禁用 Swap

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

### 4. 禁用 SELinux

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

### 5. 关闭防火墙

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

### 6. 允许 iptables 检查桥接流量

加载 `br_netfilter` 模块:

```
```bash linenumbers="1" cat <<EOF | tee /etc/modules-load.d/kubernetes.conf br_netfilter EOF
```

## 6. 加载模块

---

```
modprobe br_netfilter ``
修改 net.bridge.bridge-nf-call-iptables 设置为 1:
````bash linenumbers="1" cat <<EOF | tee /etc/sysctl.d/kubernetes.conf net.bridge.bridge-nf-call-ip6tables = 1 net.bridge.bridge-nf-call-iptables = 1 EOF
```

## 7. 刷新配置

---

```
sysctl --system ````
```

## 7.1 安装 K8s

---

开始安装容器运行时、K8s 系统组件，初始化 Master 节点，加入 Worker 节点，安装网络 CNI。

### 安装容器运行时

本例为了后续方便拉镜像，同时安装了 Docker 和 containerd，DCE 5.0 实际用的是 containerd。

## 1. 安装 Docker 的软件源

```
bash
sudo yum install -y yum-utils
sudo yum-config-manager \
    --add-repo \
    https://download.docker.com/linux/centos/docker-ce.repo
```

## 2. 安装 Docker 和 containerd

```
bash  
sudo yum -y install docker-ce docker-ce-cli containerd.io
```

### 3. 修改 Docker 的配置

```
```bash linenums="1" sudo touch /etc/docker/daemon.json  
cat <<EOF | tee /etc/docker/daemon.json { "exec-opts": [native.cgroupdriver=systemd"] } EOF ``
```

#### 4. 修改 containerd 的配置

```
```bash linenums="1"
```

## 8. 清理配置文件

```
sudo rm -f /etc/containerd/config.toml
```

## 9. 初始配置

```
sudo containerd config default | sudo tee /etc/containerd/config.toml
```

## 10. 更新配置文件内容

```
sed -i 's/SystemdCgroup\ =\ false/SystemdCgroup\ =\ true/' /etc/containerd/config.toml sed -i 's/k8s.gcr.io/pause/registry.cn-hangzhou.aliyuncs.com/google_containers/pause/g' /etc/containerd/config.toml ```
```

## 5. 启动服务配置

```
bash linenums="1"  
sudo systemctl daemon-reload  
sudo systemctl enable --now docker  
sudo systemctl enable --now containerd
```

## 6 检查配置成功

```
bash
sudo systemctl status docker containerd
sudo docker info
```

## 安装 K8s 系统组件

## 1. 安装 Kubernetes 软件源

这里采用国内阿里云的源，具有加速效果。

```
bash linenums="1"  
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. 安装 Kubernetes 组件

```
bash linenums="1"
export K8sVersion=1.24.8
sudo yum install -y kubelet-$K8sVersion kubeadm-$K8sVersion
sudo yum install -y kubectl-$K8sVersion # (1)
```

### a. 可以仅在 Master 节点安装

## 3. 启动 kubelet 系统服务

```
bash
sudo systemctl enable --now kubelet
```

此时启动后发现服务状态异常，会检查集群配置，因为没有配置会不断地重启，不影响后续操作。

## 初始化 Master 节点

注意规划集群主节点配置，初始化主节点时，规划网络配置。

指定对应的 K8s 的版本，注意与安装时的配置保持一致：

```
sudo kubeadm init --kubernetes-version=v1.24.8 \
--image-repository=registry.cn-hangzhou.aliyuncs.com/google_containers \
--pod-network-cidr 10.11.0.0/16 \
```

输出类似于：

??? note “点击查看控制台输出的内容”

```
```console
[init] Using Kubernetes version: v1.24.8
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] Generating "etcd/peer" certificate and key
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
[kubeconfig] Writing "controller-manager.conf" kubeconfig file
[kubeconfig] Writing "scheduler.conf" kubeconfig file
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Starting the kubelet
[control-plane] Using manifest folder "/etc/kubernetes/manifests"
[control-plane] Creating static Pod manifest for "kube-apiserver"
[control-plane] Creating static Pod manifest for "kube-controller-manager"
[control-plane] Creating static Pod manifest for "kube-scheduler"
[etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"
[wait-control-plane] Waiting for the kubelet to boot up the control plane as static Pods from directory "/etc/kubernetes/manifests". This can take up to 4m0s
[apiclient] All control plane components are healthy after 6.503693 seconds
[upload-config] Storing the configuration used in ConfigMap "kubeadm-config" in the "kube-system" Namespace
[kubelet] Creating a ConfigMap "kubelet-config-1.24.8" in namespace kube-system with the configuration for the kubelets in the cluster
NOTE: The "kubelet-config-1.24.8" naming of the kubelet ConfigMap is deprecated. Once the UnversionedKubeletConfigMap feature gate graduates to Beta the default name will become just "kubelet-config". Kubeadm upgrade will handle this transition transparently.
[upload-certs] Skipping phase. Please see --upload-certs
[mark-control-plane] Marking the node node1.k8s.com as control-plane by adding the labels: [node-role.kubernetes.io/master(deprecated) node-role.kubernetes.io/control-plane]
[mark-control-plane] Marking the node node1.k8s.com as control-plane by adding the taints [node-role.kubernetes.io/master:NoSchedule]
```

```
[bootstrap-token] Using token: c0wcm5.0yu9szfktsxvurza
[bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Roles
[bootstrap-token] configured RBAC rules to allow Node Bootstrap tokens to get nodes
[bootstrap-token] configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certificate credentials
[bootstrap-token] configured RBAC rules to allow the csapprover controller automatically approve CSRs from a Node Bootstrap Token
[bootstrap-token] configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy
```

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 10.23.245.63:6443 --token djdsj2.sj23js90213j323 \
--discovery-token-ca-cert-hash sha256:ewuosdjk2390rjertw32p32j43p25a70298db818ajsdjk1293jk23k23201934h
...
```

## 加入 Worker 节点

需确保完成了上述步骤：节点系统优化、安装容器运行时、安装 K8s 系统组件，并且成功初始化 Master 节点。

运行以下命令加入 Worker 节点：

```
kubeadm join 10.23.245.63:6443 --token djdsj2.sj23js90213j323 \
--discovery-token-ca-cert-hash sha256:ewuosdjk2390rjertw32p32j43p25a70298db818ajsdjk1293jk23k23201934h
```

查看新加的 Worker 节点：

```
kubectl get nodes
```

输出类似于：

NAME	STATUS	ROLES	AGE	VERSION
master01-k8s-com	Ready	control-plane	9h	v1.24.8
node01-k8s-com	Ready	<none>	9h	v1.24.8
node02-k8s-com	Ready	<none>	9h	v1.24.8

## 安装 CNI Calico

将以下 YAML 保存为 `calico.yaml`。

??? note "点击查看 `calico.yaml` 的内容"

```
```yaml
---
# Source: calico/templates/calico-config.yaml
# This ConfigMap is used to configure a self-hosted Calico installation.
kind: ConfigMap
apiVersion: v1
metadata:
  name: calico-config
  namespace: kube-system
data:
  # Typha is disabled.
  typha_service_name: "none"
  # Configure the backend to use.
  calico_backend: "bird"

  # Configure the MTU to use for workload interfaces and tunnels.
  # By default, MTU is auto-detected, and explicitly setting this field should not be required.
  # You can override auto-detection by providing a non-zero value.
  veth_mtu: "0"

  # The CNI network configuration to install on each node. The special
  # values in this config will be automatically populated.
```

```

cni_network_config: |-  

{  

  "name": "k8s-pod-network",  

  "cniVersion": "0.3.1",  

  "plugins": [  

    {  

      "type": "calico",  

      "log_level": "info",  

      "log_file_path": "/var/log/calico/cni/cni.log",  

      "datastore_type": "kubernetes",  

      "nodename": "__KUBERNETES_NODE_NAME__",  

      "mtu": __CNI_MTU__,  

      "ipam": {  

        "type": "calico-ipam"  

      },  

      "policy": {  

        "type": "k8s"  

      },  

      "kubernetes": {  

        "kubeconfig": "__KUBECONFIG_FILEPATH__"  

      }  

    },  

    {  

      "type": "portmap",  

      "snat": true,  

      "capabilities": {"portMappings": true}  

    },  

    {  

      "type": "bandwidth",  

      "capabilities": {"bandwidth": true}  

    }  

  ]  

}  

---  

# Source: calico/templates/kdd-crds.yaml  

apiVersion: apiextensions.k8s.io/v1  

kind: CustomResourceDefinition  

metadata:  

  name: bgpconfigurations.crd.projectcalico.org  

spec:  

  group: crd.projectcalico.org  

  names:  

    kind: BGPConfiguration  

    listKind: BGPConfigurationList  

    plural: bgpconfigurations  

    singular: bgpconfiguration  

  scope: Cluster  

  versions:  

  - name: v1  

    schema:  

      openAPIV3Schema:  

        description: BGPConfiguration contains the configuration for any BGP routing.  

        properties:  

          apiVersion:  

            description: 'APIVersion defines the versioned schema of this representation  

              of an object. Servers should convert recognized schemas to the latest  

              internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-  

              conventions.md#resources'  

            type: string  

          kind:  

            description: 'Kind is a string value representing the REST resource this  

              object represents. Servers may infer this from the endpoint the client  

              submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-  

              conventions.md#types-kinds'  

            type: string  

          metadata:  

            type: object  

          spec:  

            description: BGPConfigurationSpec contains the values of the BGP configuration.  

            properties:  

              asNumber:  

                description: 'ASNumber is the default AS number used by a node. [Default:  

                  64512]'  

                format: int32  

                type: integer  

              communities:  

                description: Communities is a list of BGP community values and their  

                  arbitrary names for tagging routes.  

                items:  

                  description: Community contains standard or large community value  

                    and its name.  

                  properties:  

                    name:  

                      description: Name given to community value.  

                      type: string  

                    value:  

                      description: Value must be of format `aa:nn` or `aa:nn:mm`.  

                      For standard community use `aa:nn` format, where `aa` and  

                      `nn` are 16 bit number. For large community use `aa:nn:mm`  

                      format, where `aa`, `nn` and `mm` are 32 bit number. Where,  

                      `aa` is an AS Number, `nn` and `mm` are per-AS identifier.  

                  pattern: ^(\d+):(\d+)$|^(\\d+):(\\d+):(\\d+)$

```

```

        type: string
      type: object
    type: array
  listenPort:
    description: ListenPort is the port where BGP protocol should listen.
    Defaults to 179
    maximum: 65535
    minimum: 1
    type: integer
  logSeverityScreen:
    description: 'LogSeverityScreen is the log severity above which logs
      are sent to the stdout. [Default: INFO]'
    type: string
  nodeToNodeMeshEnabled:
    description: 'NodeToNodeMeshEnabled sets whether full node to node
      BGP mesh is enabled. [Default: true]'
    type: boolean
  prefixAdVERTISEments:
    description: PrefixAdVERTISEments contains per-prefix advertisement
      configuration.
    items:
      description: PrefixAdvertisement configures advertisement properties
        for the specified CIDR.
      properties:
        cidr:
          description: CIDR for which properties should be advertised.
          type: string
      communities:
        description: Communities can be list of either community names
          already defined in `Specs.Communities` or community value
          of format `aa:nn` or `aa:nn:mm`. For standard community use
          `aa:nn` format, where `aa` and `nn` are 16 bit number. For
          large community use `aa:nn:mm` format, where `aa`, `nn` and
          `mm` are 32 bit number. Where, `aa` is an AS Number, `nn` and
          `mm` are per-AS identifier.
        items:
          type: string
        type: array
      type: object
    type: array
  serviceClusterIPs:
    description: ServiceClusterIPs are the CIDR blocks from which service
      cluster IPs are allocated. If specified, Calico will advertise these
      blocks, as well as any cluster IPs within them.
    items:
      description: ServiceClusterIPBlock represents a single allowed ClusterIP
        CIDR block.
      properties:
        cidr:
          type: string
        type: object
      type: array
  serviceExternalIPs:
    description: ServiceExternalIPs are the CIDR blocks for Kubernetes
      Service External IPs. Kubernetes Service ExternalIPs will only be
      advertised if they are within one of these blocks.
    items:
      description: ServiceExternalIPBlock represents a single allowed
        External IP CIDR block.
      properties:
        cidr:
          type: string
        type: object
      type: array
  serviceLoadBalancerIPs:
    description: ServiceLoadBalancerIPs are the CIDR blocks for Kubernetes
      Service LoadBalancer IPs. Kubernetes Service status.LoadBalancer.Ingress
      IPs will only be advertised if they are within one of these blocks.
    items:
      description: ServiceLoadBalancerIPBlock represents a single allowed
        LoadBalancer IP CIDR block.
      properties:
        cidr:
          type: string
        type: object
      type: array
    type: object
  type: object
  served: true
  storage: true
status:
  acceptedNames:
    kind: ""
    plural: ""
  conditions: []
  storedVersions: []
---  

apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  name: bgppeers.crd.projectcalico.org
spec:
  group: crd.projectcalico.org

```

```

names:
  kind: BGPPeer
  listKind: BGPPeerList
  plural: bgppeers
  singular: bgppeer
  scope: Cluster
  versions:
    - name: v1
      schema:
        openAPIV3Schema:
          properties:
            apiVersion:
              description: 'APIVersion defines the versioned schema of this representation
                            of an object. Servers should convert recognized schemas to the latest
                            internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-
                            conventions.md#resources'
              type: string
            kind:
              description: 'Kind is a string value representing the REST resource this
                            object represents. Servers may infer this from the endpoint the client
                            submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-
                            conventions.md#types-kinds'
              type: string
            metadata:
              type: object
            spec:
              description: BGPPeerSpec contains the specification for a BGPPeer resource.
              properties:
                asNumber:
                  description: The AS Number of the peer.
                  format: int32
                  type: integer
                keepOriginalNextHop:
                  description: Option to keep the original nexthop field when routes
                                are sent to a BGP Peer. Setting "true" configures the selected BGP
                                Peers node to use the "next hop keep;" instead of "next hop self;"(default)
                                in the specific branch of the Node on "bird.cfg".
                  type: boolean
                node:
                  description: The node name identifying the Calico node instance that
                                is targeted by this peer. If this is not set, and no nodeSelector
                                is specified, then this BGP peer selects all nodes in the cluster.
                  type: string
                nodeSelector:
                  description: Selector for the nodes that should have this peering. When
                                this is set, the Node field must be empty.
                  type: string
                password:
                  description: Optional BGP password for the peerings generated by this
                                BGPPeer resource.
                properties:
                  secretKeyRef:
                    description: Selects a key of a secret in the node pod's namespace.
                  properties:
                    key:
                      description: The key of the secret to select from. Must be
                                    a valid secret key.
                      type: string
                    name:
                      description: 'Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
                                    TODO: Add other useful fields. apiVersion, kind, uid?'
                      type: string
                    optional:
                      description: Specify whether the Secret or its key must be
                                    defined
                      type: boolean
                    required:
                      - key
                      type: object
                type: object
              peerIP:
                description: The IP address of the peer followed by an optional port
                                number to peer with. If port number is given, format should be `[:<IPv6>]:port` or `[:<IPv4>]:port` for IPv4. If optional port number is not set,
                                and this peer IP and ASNumber belongs to a calico/node with ListenPort
                                set in BGPConfiguration, then we use that port to peer.
                type: string
              peerSelector:
                description: Selector for the remote nodes to peer with. When this
                                is set, the PeerIP and ASNumber fields must be empty. For each
                                peering between the local node and selected remote nodes, we configure
                                an IPv4 peering if both ends have NodeBGPSpec.IPV4Address specified,
                                and an IPv6 peering if both ends have NodeBGPSpec.IPV6Address specified. The
                                remote AS number comes from the remote node's NodeBGPSpec.ASNumber,
                                or the global default if that is not set.
                type: string
              sourceAddress:
                description: Specifies whether and how to configure a source address
                                for the peerings generated by this BGPPeer resource. Default value
                                "UseNodeIP" means to configure the node IP as the source address. "None"
                                means not to configure a source address.
                type: string
              type: object
            type: object

```

```

    served: true
    storage: true
status:
  acceptedNames:
    kind: ""
    plural: ""
  conditions: []
  storedVersions: []

---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  name: blockaffinities.crd.projectcalico.org
spec:
  group: crd.projectcalico.org
  names:
    kind: BlockAffinity
    listKind: BlockAffinityList
    plural: blockaffinities
    singular: blockaffinity
    scope: Cluster
  versions:
    - name: v1
      schema:
        openAPIV3Schema:
          properties:
            apiVersion:
              description: 'APIVersion defines the versioned schema of this representation
                            of an object. Servers should convert recognized schemas to the latest
                            internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-
              conventions.md#resources'
              type: string
            kind:
              description: 'Kind is a string value representing the REST resource this
                            object represents. Servers may infer this from the endpoint the client
                            submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-
              conventions.md#types-kinds'
              type: string
            metadata:
              type: object
            spec:
              description: BlockAffinitySpec contains the specification for a BlockAffinity
                            resource.
              properties:
                cidr:
                  type: string
                deleted:
                  description: Deleted indicates that this block affinity is being deleted.
                  This field is a string for compatibility with older releases that
                  mistakenly treat this field as a string.
                type: string
              node:
                type: string
                state:
                  type: string
            required:
              - cidr
              - deleted
              - node
              - state
            type: object
          served: true
          storage: true
status:
  acceptedNames:
    kind: ""
    plural: ""
  conditions: []
  storedVersions: []

---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  name: clusterinformations.crd.projectcalico.org
spec:
  group: crd.projectcalico.org
  names:
    kind: ClusterInformation
    listKind: ClusterInformationList
    plural: clusterinformations
    singular: clusterinformation
    scope: Cluster
  versions:
    - name: v1
      schema:
        openAPIV3Schema:
          description: ClusterInformation contains the cluster specific information.
          properties:
            apiVersion:
              description: 'APIVersion defines the versioned schema of this representation
                            of an object. Servers should convert recognized schemas to the latest

```

```

internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources'
type: string
kind:
description: 'Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds'
type: string
metadata:
type: object
spec:
description: ClusterInformationSpec contains the values of describing the cluster.
properties:
calicoVersion:
description: CalicoVersion is the version of Calico that the cluster is running
type: string
clusterGUID:
description: ClusterGUID is the GUID of the cluster
type: string
clusterType:
description: ClusterType describes the type of the cluster
type: string
datastoreReady:
description: DatastoreReady is used during significant datastore migrations to signal to components such as Felix that it should wait before accessing the datastore.
type: boolean
variant:
description: Variant declares which variant of Calico should be active.
type: string
type: object
type: object
served: true
storage: true
status:
acceptedNames:
kind: ""
plural: ""
conditions: []
storedVersions: []

---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
name: felixconfigurations.crd.projectcalico.org
spec:
group: crd.projectcalico.org
names:
kind: FelixConfiguration
listKind: FelixConfigurationList
plural: felixconfigurations
singular: felixconfiguration
scope: Cluster
versions:
- name: v1
schema:
openAPIV3Schema:
description: Felix Configuration contains the configuration for Felix.
properties:
apiVersion:
description: 'APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources'
type: string
kind:
description: 'Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds'
type: string
metadata:
type: object
spec:
description: FelixConfigurationSpec contains the values of the Felix configuration.
properties:
allowIPIPPacketsFromWorkloads:
description: 'AllowIPIPPacketsFromWorkloads controls whether Felix will add a rule to drop IPIP encapsulated traffic from workloads [Default: false]'
type: boolean
allowVXLANPacketsFromWorkloads:
description: 'AllowVXLANPacketsFromWorkloads controls whether Felix will add a rule to drop VXLAN encapsulated traffic from workloads [Default: false]'
type: boolean
awsSrcDstCheck:
description: 'Set source-destination-check on AWS EC2 instances. Accepted value must be one of "DoNothing", "Enabled" or "Disabled". [Default: DoNothing]'
type: boolean

```

```

enum:
- DoNothing
- Enable
- Disable
type: string
bpfConnectTimeLoadBalancingEnabled:
description: 'BPFConnectTimeLoadBalancingEnabled when in BPF mode,
controls whether Felix installs the connection-time load balancer. The
connect-time load balancer is required for the host to be able to
reach Kubernetes services and it improves the performance of pod-to-service
connections. The only reason to disable it is for debugging purposes. [Default:
true]'
type: boolean
bpfDataInterfacePattern:
description: BPFDataInterfacePattern is a regular expression that controls
which interfaces Felix should attach BPF programs to in order to
catch traffic to/from the network. This needs to match the interfaces
that Calico workload traffic flows over as well as any interfaces
that handle incoming traffic to nodeports and services from outside
the cluster. It should not match the workload interfaces (usually
named cali...).
type: string
bpfDisableUnprivileged:
description: 'BPFDisableUnprivileged, if enabled, Felix sets the kernel.unprivileged_bpf_disabled
sysctl to disable unprivileged use of BPF. This ensures that unprivileged
users cannot access Calico's BPF maps and cannot insert their own
BPF programs to interfere with Calico's. [Default: true]'
type: boolean
bpfEnabled:
description: 'BPFEabled, if enabled Felix will use the BPF dataplane.
[Default: false]'
type: boolean
bpfExtToServiceConnmark:
description: 'BPFExtToServiceConnmark in BPF mode, control a 32bit
mark that is set on connections from an external client to a local
service. This mark allows us to control how packets of that connection
are routed within the host and how is routing interpreted by RPF
check. [Default: 0]'
type: integer
bpfExternalServiceMode:
description: 'BPFEexternalServiceMode in BPF mode, controls how connections
from outside the cluster to services (node ports and cluster IPs)
are forwarded to remote workloads. If set to "Tunnel" then both
request and response traffic is tunneled to the remote node. If
set to "DSR", the request traffic is tunneled but the response traffic
is sent directly from the remote node. In "DSR" mode, the remote
node appears to use the IP of the ingress node; this requires a
permissive L2 network. [Default: Tunnel]'
type: string
bpfKubeProxyEndpointSlicesEnabled:
description: BPKubeProxyEndpointSlicesEnabled in BPF mode, controls
whether Felix's embedded kube-proxy accepts EndpointSlices or not.
type: boolean
bpfKubeProxyIptablesCleanupEnabled:
description: 'BPKubeProxyIptablesCleanupEnabled, if enabled in BPF
mode, Felix will proactively clean up the upstream Kubernetes kube-proxy's
iptables chains. Should only be enabled if kube-proxy is not running. [Default:
true]'
type: boolean
bpfKubeProxyMinSyncPeriod:
description: 'BPKubeProxyMinSyncPeriod, in BPF mode, controls the
minimum time between updates to the dataplane for Felix's embedded
kube-proxy. Lower values give reduced set-up latency. Higher values
reduce Felix CPU usage by batching up more work. [Default: 1s]'
type: string
bpfLogLevel:
description: 'BPFLogLevel controls the log level of the BPF programs
when in BPF dataplane mode. One of "Off", "Info", or "Debug". The
logs are emitted to the BPF trace pipe, accessible with the command
`tc exec bpf debug`. [Default: Off].'
type: string
chainInsertMode:
description: 'ChainInsertMode controls whether Felix hooks the kernel's
top-level iptables chains by inserting a rule at the top of the
chain or by appending a rule at the bottom. insert is the safe default
since it prevents Calico's rules from being bypassed. If you switch
to append mode, be sure that the other rules in the chains signal
acceptance by falling through to the Calico rules, otherwise the
Calico policy will be bypassed. [Default: insert]'
type: string
dataplaneDriver:
type: string
debugDisableLogDropping:
type: boolean
debugMemoryProfilePath:
type: string
debugSimulateCalcGraphHangAfter:
type: string
debugSimulateDataplaneHangAfter:
type: string
defaultEndpointToHostAction:
description: 'DefaultEndpointToHostAction controls what happens to
traffic that goes from a workload endpoint to the host itself (after
the traffic hits the endpoint egress policy). By default Calico

```

```

blocks traffic from workload endpoints to the host itself with an
iptables "DROP" action. If you want to allow some or all traffic
from endpoint to host, set this parameter to RETURN or ACCEPT. Use
RETURN if you have your own rules in the iptables "INPUT" chain;
Calico will insert its rules at the top of that chain, then "RETURN"
packets to the "INPUT" chain once it has completed processing workload
endpoint egress policy. Use ACCEPT to unconditionally accept packets
from workloads after processing workload endpoint egress policy.
[Default: Drop]
type: string
deviceRouteProtocol:
description: This defines the route protocol added to programmed device
routes, by default this will be RTPROT_BOOT when left blank.
type: integer
deviceRouteSourceAddress:
description: This is the source address to use on programmed device
routes. By default the source address is left blank, leaving the
kernel to choose the source address used.
type: string
disableConntrackInvalidCheck:
type: boolean
endpointReportingDelay:
type: string
endpointReportingEnabled:
type: boolean
externalNodesList:
description: ExternalNodesCIDRList is a list of CIDR's of external-non-calico-nodes
which may source tunnel traffic and have the tunneled traffic be
accepted at calico nodes.
items:
type: string
type: array
failsafeInboundHostPorts:
description: 'FailsafeInboundHostPorts is a list of UDP/TCP ports
and CIDRs that Felix will allow incoming traffic to host endpoints
on irrespective of the security policy. This is useful to avoid
accidentally cutting off a host with incorrect configuration. For
back-compatibility, if the protocol is not specified, it defaults
to "tcp". If a CIDR is not specified, it will allow traffic from
all addresses. To disable all inbound host ports, use the value
none. The default value allows ssh access and DHCP. [Default: tcp:22,
udp:68, tcp:179, tcp:2379, tcp:2380, tcp:6443, tcp:6666, tcp:6667]'
```

```
items:
description: ProtoPort is combination of protocol, port, and CIDR.
Protocol and port must be specified.
```

```
properties:
net:
type: string
port:
type: integer
protocol:
type: string
required:
- port
- protocol
type: object
type: array
```

```
failsafeOutboundHostPorts:
description: 'FailsafeOutboundHostPorts is a list of UDP/TCP ports
and CIDRs that Felix will allow outgoing traffic from host endpoints
to irrespective of the security policy. This is useful to avoid
accidentally cutting off a host with incorrect configuration. For
back-compatibility, if the protocol is not specified, it defaults
to "tcp". If a CIDR is not specified, it will allow traffic from
all addresses. To disable all outbound host ports, use the value
none. The default value opens etcd's standard ports to ensure that
Felix does not get cut off from etcd as well as allowing DHCP and
DNS. [Default: tcp:179, tcp:2379, tcp:2380, tcp:6443, tcp:6666,
tcp:6667, udp:53, udp:67]'
```

```
items:
description: ProtoPort is combination of protocol, port, and CIDR.
Protocol and port must be specified.
```

```
properties:
net:
type: string
port:
type: integer
protocol:
type: string
required:
- port
- protocol
type: object
type: array
```

```
featureDetectOverride:
description: FeatureDetectOverride is used to override the feature
detection. Values are specified in a comma separated list with no
spaces, example: "SNATFullyRandom=true,MASQFullyRandom=false,RestoreSupportsLock=".
"true" or "false" will force the feature, empty or omitted values
are auto-detected.
```

```
type: string
genericXDPEnabled:
```

```
description: 'GenericXDPEnabled enables Generic XDP so network cards
that don't support XDP offload or driver modes can use XDP. This
```

```

is not recommended since it doesn't provide better performance
than iptables. [Default: false]
type: boolean
healthEnabled:
type: boolean
healthHost:
type: string
healthPort:
type: integer
interfaceExclude:
description: 'InterfaceExclude is a comma-separated list of interfaces
that Felix should exclude when monitoring for host endpoints. The
default value ensures that Felix ignores Kubernetes'' IPVS dummy
interface, which is used internally by kube-proxy. If you want to
exclude multiple interface names using a single value, the list
supports regular expressions. For regular expressions you must wrap
the value with ''/'. For example having values ''/^kube/,veth1''
will exclude all interfaces that begin with ''kube'' and also the
interface ''veth1''. [Default: kube-ipvs0]'
type: string
interfacePrefix:
description: 'InterfacePrefix is the interface name prefix that identifies
workload endpoints and so distinguishes them from host endpoint
interfaces. Note: in environments other than bare metal, the orchestrators
configure this appropriately. For example our Kubernetes and Docker
integrations set the ''cali'' value, and our OpenStack integration
sets the ''tap'' value. [Default: cali]'
type: string
interfaceRefreshInterval:
description: InterfaceRefreshInterval is the period at which Felix
rescans local interfaces to verify their state. The rescan can be
disabled by setting the interval to 0.
type: string
ipipEnabled:
type: boolean
ipipMTU:
description: 'IPIMTU is the MTU to set on the tunnel device. See
Configuring MTU [Default: 1440]'
type: integer
ipsetsRefreshInterval:
description: 'IpsetsRefreshInterval is the period at which Felix re-checks
all iptables state to ensure that no other process has accidentally
broken Calico''s rules. Set to 0 to disable iptables refresh. [Default:
90s]'
type: string
iptablesBackend:
description: IptablesBackend specifies which backend of iptables will
be used. The default is legacy.
type: string
iptablesFilterAllowAction:
type: string
iptablesLockFilePath:
description: 'IptablesLockFilePath is the location of the iptables
lock file. You may need to change this if the lock file is not in
its standard location (for example if you have mapped it into Felix''s
container at a different path). [Default: /run/xtables.lock]'
type: string
iptablesLockProbeInterval:
description: 'IptablesLockProbeInterval is the time that Felix will
wait between attempts to acquire the iptables lock if it is not
available. Lower values make Felix more responsive when the lock
is contended, but use more CPU. [Default: 50ms]'
type: string
iptablesLockTimeout:
description: 'IptablesLockTimeout is the time that Felix will wait
for the iptables lock, or 0, to disable. To use this feature, Felix
must share the iptables lock file with all other processes that
also take the lock. When running Felix inside a container, this
requires the /run directory of the host to be mounted into the calico/node
or calico/felix container. [Default: 0s disabled]'
type: string
iptablesMangleAllowAction:
type: string
iptablesMarkMask:
description: 'IptablesMarkMask is the mask that Felix selects its
IPTables Mark bits from. Should be a 32 bit hexadecimal number with
at least 8 bits set, none of which clash with any other mark bits
in use on the system. [Default: 0xffff000000]'
format: int32
type: integer
iptablesNATOutgoingInterfaceFilter:
type: string
iptablesPostWriteCheckInterval:
description: 'IptablesPostWriteCheckInterval is the period after Felix
has done a write to the dataplane that it schedules an extra read
back in order to check the write was not clobbered by another process.
This should only occur if another application on the system doesn't
respect the iptables lock. [Default: 1s]'
type: string
iptablesRefreshInterval:
description: 'IptablesRefreshInterval is the period at which Felix
re-checks the IP sets in the dataplane to ensure that no other process
has accidentally broken Calico''s rules. Set to 0 to disable IP
sets refresh. Note: the default for this value is lower than the

```

```

other refresh intervals as a workaround for a Linux kernel bug that
was fixed in kernel version 4.11. If you are using v4.11 or greater
you may want to set this to, a higher value to reduce Felix CPU
usage. [Default: 10s]
type: string
ipv6Support:
  type: boolean
kubeNodePortRanges:
  description: 'KubeNodePortRanges holds list of port ranges used for
    service node ports. Only used if felix detects kube-proxy running
    in ipvs mode. Felix uses these ranges to separate host and workload
    traffic. [Default: 30000:32767].'
  items:
    anyOf:
      - type: integer
      - type: string
      pattern: ^.*
      x-kubernetes-int-or-string: true
  type: array
logFilePath:
  description: 'LogFilePath is the full path to the Felix log. Set to
    none to disable file logging. [Default: /var/log/calico/felix.log]'
  type: string
logPrefix:
  description: 'LogPrefix is the log prefix that Felix uses when rendering
    LOG rules. [Default: calico-packet]'
  type: string
logSeverityFile:
  description: 'LogSeverityFile is the log severity above which logs
    are sent to the log file. [Default: Info]'
  type: string
logSeverityScreen:
  description: 'LogSeverityScreen is the log severity above which logs
    are sent to the stdout. [Default: Info]'
  type: string
logSeveritySys:
  description: 'LogSeveritySys is the log severity above which logs
    are sent to the syslog. Set to None for no logging to syslog. [Default:
    Info]'
  type: string
maxipsetSize:
  type: integer
metadataAddr:
  description: 'MetadataAddr is the IP address or domain name of the
    server that can answer VM queries for cloud-init metadata. In OpenStack,
    this corresponds to the machine running nova-api (or in Ubuntu,
    nova-api-metadata). A value of none (case insensitive) means that
    Felix should not set up any NAT rule for the metadata path. [Default:
    127.0.0.1]'
  type: string
metadataPort:
  description: 'MetadataPort is the port of the metadata server. This,
    combined with global.MetadataAddr (if not "'None'"), is used to
    set up a NAT rule, from 169.254.169.254:80 to MetadataAddr:MetadataPort.
    In most cases this should not need to be changed [Default: 8775].'
  type: integer
mtuIfacePattern:
  description: MTU_ifacePattern is a regular expression that controls
    which interfaces Felix should scan in order to calculate the host's
    MTU. This should not match workload interfaces (usually named cali...).
  type: string
natOutgoingAddress:
  description: NATOutgoingAddress specifies an address to use when performing
    source NAT for traffic in a natOutgoing pool that is leaving the
    network. By default the address used is an address on the interface
    the traffic is leaving on (ie it uses the iptables MASQUERADE target)
  type: string
natPortRange:
  anyOf:
    - type: integer
    - type: string
  description: NATPortRange specifies the range of ports that is used
    for port mapping when doing outgoing NAT. When unset the default
    behavior of the network stack is used.
  pattern: ^.*
  x-kubernetes-int-or-string: true
netlinkTimeout:
  type: string
openstackRegion:
  description: 'OpenstackRegion is the name of the region that a particular
    Felix belongs to. In a multi-region Calico/OpenStack deployment,
    this must be configured somehow for each Felix (here in the datamodel,
    or in felix.cfg or the environment on each compute node), and must
    match the [calico] openstack_region value configured in neutron.conf
    on each node. [Default: Empty]'
  type: string
policySyncPathPrefix:
  description: 'PolicySyncPathPrefix is used to by Felix to communicate
    policy changes to external services, like Application layer policy.
    [Default: Empty]'
  type: string
prometheusGoMetricsEnabled:
  description: 'PrometheusGoMetricsEnabled disables Go runtime metrics
    collection, which the Prometheus client does by default, when set

```

```

    to false. This reduces the number of metrics reported, reducing
    Prometheus load. [Default: true]
  type: boolean
  prometheusMetricsEnabled:
    description: 'PrometheusMetricsEnabled enables the Prometheus metrics
      server in Felix if set to true. [Default: false]'
    type: boolean
  prometheusMetricsHost:
    description: 'PrometheusMetricsHost is the host that the Prometheus
      metrics server should bind to. [Default: empty]'
    type: string
  prometheusMetricsPort:
    description: 'PrometheusMetricsPort is the TCP port that the Prometheus
      metrics server should bind to. [Default: 9091]'
    type: integer
  prometheusProcessMetricsEnabled:
    description: 'PrometheusProcessMetricsEnabled disables process metrics
      collection, which the Prometheus client does by default, when set
      to false. This reduces the number of metrics reported, reducing
      Prometheus load. [Default: true]'
    type: boolean
  removeExternalRoutes:
    description: Whether or not to remove device routes that have not
      been programmed by Felix. Disabling this will allow external applications
      to also add device routes. This is enabled by default which means
      we will remove externally added routes.
    type: boolean
  reportingInterval:
    description: 'ReportingInterval is the interval at which Felix reports
      its status into the datastore or 0 to disable. Must be non-zero
      in OpenStack deployments. [Default: 30s]'
    type: string
  reportingTTL:
    description: 'ReportingTTL is the time-to-live setting for process-wide
      status reports. [Default: 90s]'
    type: string
  routeRefreshInterval:
    description: 'RouteRefreshInterval is the period at which Felix re-checks
      the routes in the dataplane to ensure that no other process has
      accidentally broken Calico's rules. Set to 0 to disable route refresh.
      [Default: 90s]'
    type: string
  routeSource:
    description: 'RouteSource configures where Felix gets its routing
      information. - WorkloadIPs: use workload endpoints to construct
      routes. - CalicoIPAM: the default - use IPAM data to construct routes.'
    type: string
  routeTableRange:
    description: Calico programs additional Linux route tables for various
      purposes. RouteTableRange specifies the indices of the route tables
      that Calico should use.
  properties:
    max:
      type: integer
    min:
      type: integer
  required:
    - max
    - min
  type: object
  serviceLoopPrevention:
    description: 'When service IP advertisement is enabled, prevent routing
      loops to service IPs that are not in use, by dropping or rejecting
      packets that do not get DNAT''d by kube-proxy. Unless set to "Disabled",
      in which case such routing loops continue to be allowed. [Default:
      Drop]'
    type: string
  sidecarAccelerationEnabled:
    description: 'SidecarAccelerationEnabled enables experimental sidecar
      acceleration [Default: false]'
    type: boolean
  usageReportingEnabled:
    description: 'UsageReportingEnabled reports anonymous Calico version
      number and cluster size to projectcalico.org. Logs warnings returned
      by the usage server. For example, if a significant security vulnerability
      has been discovered in the version of Calico being used. [Default:
      true]'
    type: boolean
  usageReportingInitialDelay:
    description: 'UsageReportingInitialDelay controls the minimum delay
      before Felix makes a report. [Default: 300s]'
    type: string
  usageReportingInterval:
    description: 'UsageReportingInterval controls the interval at which
      Felix makes reports. [Default: 86400s]'
    type: string
  useInternalDataplaneDriver:
    type: boolean
  vxlanEnabled:
    type: boolean
  vxlanMTU:
    description: 'VXLANMTU is the MTU to set on the tunnel device. See
      Configuring MTU [Default: 1440]'
    type: integer

```

```

vxlanPort:
  type: integer
vxlanVNI:
  type: integer
wireguardEnabled:
  description: 'WireguardEnabled controls whether Wireguard is enabled.
  [Default: false]'
  type: boolean
wireguardInterfaceName:
  description: 'WireguardInterfaceName specifies the name to use for
  the Wireguard interface. [Default: wg.calico]'
  type: string
wireguardListeningPort:
  description: 'WireguardListeningPort controls the listening port used
  by Wireguard. [Default: 51820]'
  type: integer
wireguardMTU:
  description: 'WireguardMTU controls the MTU on the Wireguard interface.
  See Configuring MTU [Default: 1420]'
  type: integer
wireguardRoutingRulePriority:
  description: 'WireguardRoutingRulePriority controls the priority value
  to use for the Wireguard routing rule. [Default: 99]'
  type: integer
xdpEnabled:
  description: 'XDPEnabled enables XDP acceleration for suitable untracked
  incoming deny rules. [Default: true]'
  type: boolean
xdpRefreshInterval:
  description: 'XDPRefreshInterval is the period at which Felix re-checks
  all XDP state to ensure that no other process has accidentally broken
  Calico's BPF maps or attached programs. Set to 0 to disable XDP
  refresh. [Default: 90s]'
  type: string
  type: object
  type: object
served: true
storage: true
status:
  acceptedNames:
    kind: ""
    plural: ""
  conditions: []
  storedVersions: []

---
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  name: globalnetworkpolicies.crd.projectcalico.org
spec:
  group: crd.projectcalico.org
  names:
    kind: GlobalNetworkPolicy
    listKind: GlobalNetworkPolicyList
    plural: globalnetworkpolicies
    singular: globalnetworkpolicy
  scope: Cluster
  versions:
  - name: v1
    schema:
      openAPIV3Schema:
        properties:
          apiVersion:
            description: 'APIVersion defines the versioned schema of this representation
              of an object. Servers should convert recognized schemas to the latest
              internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-
            conventions.md#resources'
            type: string
          kind:
            description: 'Kind is a string value representing the REST resource this
              object represents. Servers may infer this from the endpoint the client
              submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-
            conventions.md#types-kinds'
            type: string
        metadata:
          type: object
        spec:
          properties:
            applyOnForward:
              description: ApplyOnForward indicates to apply the rules in this policy
              on forward traffic.
              type: boolean
            doNotTrack:
              description: DoNotTrack indicates whether packets matched by the rules
              in this policy should go through the data plane's connection tracking,
              such as Linux conntrack. If True, the rules in this policy are
              applied before any data plane connection tracking, and packets allowed
              by this policy are marked as not to be tracked.
              type: boolean
        egress:
          description: The ordered set of egress rules. Each rule contains
          a set of packet match criteria and a corresponding action to apply.
          items:

```

```

description: "A Rule encapsulates a set of match criteria and an
action. Both selector-based security Policy and security Profiles
reference rules - separated out as a list of rules for both ingress
and egress packet matching. \n Each positive match criteria has
a negated version, prefixed with \"Not\". All the match criteria
within a rule must be satisfied for a packet to match. A single
rule can contain the positive and negative version of a match
and both must be satisfied for the rule to match."
properties:
  action:
    type: string
  destination:
    description: Destination contains the match criteria that apply
      to destination entity.
    properties:
      namespaceSelector:
        description: "NamespaceSelector is an optional field that
          contains a selector expression. Only traffic that originates
          from (or terminates at) endpoints within the selected
          namespaces will be matched. When both NamespaceSelector
          and Selector are defined on the same rule, then only workload
          endpoints that are matched by both selectors will be selected
          by the rule. \n For NetworkPolicy, an empty NamespaceSelector
          implies that the Selector is limited to selecting only
          workload endpoints in the same namespace as the NetworkPolicy.
          \n For NetworkPolicy, `global()` NamespaceSelector implies
          that the Selector is limited to selecting only GlobalNetworkSet
          or HostEndpoint. \n For GlobalNetworkPolicy, an empty
          NamespaceSelector implies the Selector applies to workload
          endpoints across all namespaces."
        type: string
  nets:
    description: Nets is an optional field that restricts the
      rule to only apply to traffic that originates from (or
      terminates at) IP addresses in any of the given subnets.
    items:
      type: string
      type: array
  notNets:
    description: NotNets is the negated version of the Nets
      field.
    items:
      type: string
      type: array
  notPorts:
    description: NotPorts is the negated version of the Ports
      field. Since only some protocols have ports, if any ports
      are specified it requires the Protocol match in the Rule
      to be set to "TCP" or "UDP".
    items:
      anyOf:
        - type: integer
        - type: string
        pattern: ^.*
        x-kubernetes-int-or-string: true
      type: array
  notSelector:
    description: NotSelector is the negated version of the Selector
      field. See Selector field for subtleties with negated
      selectors.
    type: string
  ports:
    description: "Ports is an optional field that restricts
      the rule to only apply to traffic that has a source (destination)
      port that matches one of these ranges/values. This value
      is a list of integers or strings that represent ranges
      of ports. \n Since only some protocols have ports, if
      any ports are specified it requires the Protocol match
      in the Rule to be set to \"TCP\" or \"UDP\"."
    items:
      anyOf:
        - type: integer
        - type: string
        pattern: ^.*
        x-kubernetes-int-or-string: true
      type: array
  selector:
    description: "Selector is an optional field that contains
      a selector expression (see Policy for sample syntax).
      \ Only traffic that originates from (terminates at) endpoints
      matching the selector will be matched. \n Note that: in
      addition to the negated version of the Selector (see NotSelector
      below), the selector expression syntax itself supports
      negation. The two types of negation are subtly different.
      One negates the set of matched endpoints, the other negates
      the whole match: \n \tSelector = '\"has(my_label)\"' matches
      packets that are from other Calico-controlled \tendpoints
      that do not have the label '\"my_label\"'. \n \tNotSelector
      = '\"has(my_label)\"' matches packets that are not from
      Calico-controlled \tendpoints that do have the label '\"my_label\"'.
      \n The effect is that the latter will accept packets from
      non-Calico sources whereas the former is limited to packets
      from Calico-controlled endpoints."
    type: string

```

```

serviceAccounts:
  description: ServiceAccounts is an optional field that restricts
    the rule to only apply to traffic that originates from
    (or terminates at) a pod running as a matching service
    account.
  properties:
    names:
      description: Names is an optional field that restricts
        the rule to only apply to traffic that originates
        from (or terminates at) a pod running as a service
        account whose name is in the list.
    items:
      type: string
      type: array
  selector:
    description: Selector is an optional field that restricts
      the rule to only apply to traffic that originates
      from (or terminates at) a pod running as a service
      account that matches the given label selector. If
      both Names and Selector are specified then they are
      AND'ed.
    type: string
    type: object
  type: object
http:
  description: HTTP contains match criteria that apply to HTTP
    requests.
  properties:
    methods:
      description: Methods is an optional field that restricts
        the rule to apply only to HTTP requests that use one of
        the listed HTTP Methods (e.g. GET, PUT, etc.) Multiple
        methods are OR'd together.
    items:
      type: string
      type: array
  paths:
    description: 'Paths is an optional field that restricts
      the rule to apply to HTTP requests that use one of the
      listed HTTP Paths. Multiple paths are OR'd together.
      e.g: - exact: /foo - prefix: /bar NOTE: Each entry may
      ONLY specify either a `exact` or a `prefix` match. The
      validator will check for it.'
    items:
      description: 'HTTPPath specifies an HTTP path to match.
        It may be either of the form: exact: <path>; which matches
        the path exactly or prefix: <path-prefix>; which matches
        the path prefix'
      properties:
        exact:
          type: string
        prefix:
          type: string
      type: object
    type: array
  type: object
icmp:
  description: ICMP is an optional field that restricts the rule
    to apply to a specific type and code of ICMP traffic. This
    should only be specified if the Protocol field is set to "ICMP"
    or "ICMPv6".
  properties:
    code:
      description: Match on a specific ICMP code. If specified,
        the Type value must also be specified. This is a technical
        limitation imposed by the kernel's iptables firewall,
        which Calico uses to enforce the rule.
      type: integer
    type:
      description: Match on a specific ICMP type. For example
        a value of 8 ``.
...
```

```

执行初始化 Calico 命令:

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

## 10.1 安装 Helm

本例使用了 Helm 3，注意安装脚本会下载最新版本:

```
curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3
chmod 700 get_helm.sh
./get_helm.sh
```

## 10.2 安装 StorageClass

!!! tip

注意，HwameiStor 需要使用未格式化文件系统的裸盘直接来进行存储管理

默认安装时将 K8s localdisk 作为默认存储，本例采用开源项目 [HwameiStor](#) 用作本地磁盘管理。

1. 下载并解压 HwameiStor Repo

```
bash linenums="1"
helm repo add hwameistor http://hwameistor.io/hwameistor
helm repo update hwameistor
helm pull hwameistor/hwameistor --untar
```

2. 安装 HwameiStor

使用镜像仓库安装，要切换镜像仓库的镜像，请使用 `--set` 更改这两个参数值：`global.k8sImageRegistry` 和 `global.hwameistorImageRegistry`。如果无法访问默认的镜像仓库 `quay.io` 和 `ghcr.io`，请尝试使用 DaoCloud 提供的镜像源 `quay.m.daocloud.io` 和 `ghcr.m.daocloud.io`。

```
bash linenums="1"
helm install hwameistor ./hwameistor \
-n hwameistor --create-namespace \
--set global.k8sImageRegistry=k8s-gcr.m.daocloud.io \
--set global.hwameistorImageRegistry=ghcr.m.daocloud.io
```

3. 检查 HwameiStor 的全部 Pod 状态

```
bash
kubectl -n hwameistor get pod
```

输出类似于：

| NAME                                                     | READY | STATUS  |
|----------------------------------------------------------|-------|---------|
| hwameistor-local-disk-csi-controller-665bb7f47d-6227f    | 2/2   | Running |
| hwameistor-local-disk-manager-5ph2d                      | 2/2   | Running |
| hwameistor-local-disk-manager-jhj59                      | 2/2   | Running |
| hwameistor-local-disk-manager-k9cvj                      | 2/2   | Running |
| hwameistor-local-disk-manager-kxwww                      | 2/2   | Running |
| hwameistor-local-storage-csi-controller-667d949fbb-k488w | 3/3   | Running |
| hwameistor-local-storage-csqqv                           | 2/2   | Running |
| hwameistor-local-storage-gcrzm                           | 2/2   | Running |
| hwameistor-local-storage-v8g7t                           | 2/2   | Running |
| hwameistor-local-storage-zkwmn                           | 2/2   | Running |
| hwameistor-scheduler-58dfcf79f5-lswkt                    | 1/1   | Running |
| hwameistor-webhook-986479678-278cr                       | 1/1   | Running |

`local-disk-manager` 和 `local-storage` 都是 DaemonSet。在每个 K8s 节点上都应该有一个 DaemonSet Pod。

等到全部状态正常后，可进行后续操作，检查 `StorageClass`：

```
bash
kubectl get storageclass hwameistor-storage-lvm-hdd
```

输出类似于：

| NAME                                 | PROVISIONER       | RECLAIMPOLICY |
|--------------------------------------|-------------------|---------------|
| hwameistor-storage-lvm-hdd (default) | lvm.hwameistor.io | Delete        |

4. 检查 `localdisknodes`，默认在使用的磁盘 `PHASE` 为 `Bound`。

```
bash
kubectl get localdisknodes
```

输出类似于：

| NAME             | NODEMATCH        | PHASE |
|------------------|------------------|-------|
| master01-k8s-com | master01-k8s-com | Bound |
| node01-k8s-com   | node01-k8s-com   | Bound |
| node02-k8s-com   | node02-k8s-com   | Bound |

5. 检查 `localdisks`，默认在使用的磁盘 `PHASE` 为 `Bound`。

```
bash
kubectl get localdisks
```

输出类似于：

```
none
NAME      NODEMATCH   CLAIM     PHASE
master01-k8s-com-vda master01-k8s-com   Bound
master01-k8s-com-vdb master01-k8s-com   Bound
node01-k8s-com-vda   node01-k8s-com   Bound
node01-k8s-com-vdb   node01-k8s-com   Bound
node02-k8s-com-vda   node02-k8s-com   Bound
node02-k8s-com-vdb   node02-k8s-com   Bound
```

## 6. 设置默认的 StorageClass

为 storageclasses 增加标识为默认的 annotations。

```
bash
kubectl patch storageclasses.storage.k8s.io hwameistor-storage-lvm-hdd -p '{"metadata": {"annotations": {"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

## 7. 创建存储池，注意替换对应的 nodeName

```
bash linenums="1"
helm template ./hwameistor \
-s templates/post-install-claim-disks.yaml \
--set storageNodes='{master01-k8s-com,node01-k8s-com,node02-k8s-com}' \
| kubectl apply -f -
```

创建成功后，查看本地磁盘的 ldc，应该全部是 Bound。

```
bash
kubectl get ldc
```

## 10.3 安装 Metrics server

接下来需要安装 metrics-server，方便后续使用 Insight 可观测性模块。将以下 YAML 保存为 metrics-server.yaml。

??? note "点击查看 metrics-server.yaml 内容"

```
```yaml title="metrics-server.yaml"
apiVersion: v1
kind: ServiceAccount
metadata:
  labels:
    k8s-app: metrics-server
    name: metrics-server
    namespace: kube-system
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  labels:
    k8s-app: metrics-server
    rbac.authorization.k8s.io/aggregate-to-admin: "true"
    rbac.authorization.k8s.io/aggregate-to-edit: "true"
    rbac.authorization.k8s.io/aggregate-to-view: "true"
  name: system:aggregated-metrics-reader
rules:
- apiGroups:
  - metrics.k8s.io
  resources:
  - pods
  - nodes
  verbs:
  - get
  - list
  - watch
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  labels:
    k8s-app: metrics-server
    name: system:metrics-server
```

```

rules:
- apiGroups:
  - ""
  resources:
  - nodes/metrics
  verbs:
  - get
- apiGroups:
  - ""
  resources:
  - pods
  - nodes
  verbs:
  - get
  - list
  - watch
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  labels:
    k8s-app: metrics-server
  name: metrics-server-auth-reader
  namespace: kube-system
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: extension-apiserver-authentication-reader
subjects:
- kind: ServiceAccount
  name: metrics-server
  namespace: kube-system
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  labels:
    k8s-app: metrics-server
    name: metrics-server:system:auth-delegator
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: system:auth-delegator
subjects:
- kind: ServiceAccount
  name: metrics-server
  namespace: kube-system
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  labels:
    k8s-app: metrics-server
    name: system:metrics-server
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: system:metrics-server
subjects:
- kind: ServiceAccount
  name: metrics-server
  namespace: kube-system
---
apiVersion: v1
kind: Service
metadata:
  labels:
    k8s-app: metrics-server
    name: metrics-server
    namespace: kube-system
spec:
  ports:
  - name: https
    port: 443
    protocol: TCP
    targetPort: https
  selector:
    k8s-app: metrics-server
---
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    k8s-app: metrics-server
    name: metrics-server
    namespace: kube-system
spec:
  selector:
    matchLabels:
      k8s-app: metrics-server
  strategy:
    rollingUpdate:
      maxUnavailable: 0
  template:

```

```

metadata:
  labels:
    k8s-app: metrics-server
spec:
  containers:
    - args:
        - --cert-dir=/tmp
        - --secure-port=4443
        - --kubelet-preferred-address-types=InternalIP,ExternalIP,Hostname
        - --kubelet-use-node-status-port
        - --metric-resolution=15s
        - --kubelet-insecure-tls
      image: registry.cn-hangzhou.aliyuncs.com/chenby/metrics-server:v0.6.2
      imagePullPolicy: IfNotPresent
      livenessProbe:
        failureThreshold: 3
        httpGet:
          path: /livez
          port: https
          scheme: HTTPS
        periodSeconds: 10
      name: metrics-server
      ports:
        - containerPort: 4443
          name: https
          protocol: TCP
      readinessProbe:
        failureThreshold: 3
        httpGet:
          path: /readyz
          port: https
          scheme: HTTPS
        initialDelaySeconds: 20
        periodSeconds: 10
      resources:
        requests:
          cpu: 100m
          memory: 200Mi
      securityContext:
        allowPrivilegeEscalation: false
        readOnlyRootFilesystem: true
        runAsNonRoot: true
        runAsUser: 1000
      volumeMounts:
        - mountPath: /tmp
          name: tmp-dir
    nodeSelector:
      kubernetes.io/os: linux
  priorityClassName: system-cluster-critical
  serviceAccountName: metrics-server
  volumes:
    - emptyDir: {}
      name: tmp-dir
---
apiVersion: apiregistration.k8s.io/v1
kind: APIService
metadata:
  labels:
    k8s-app: metrics-server
  name: v1beta1.metrics.k8s.io
spec:
  group: metrics.k8s.io
  groupPriorityMinimum: 100
  insecureSkipTLSVerify: true
  service:
    name: metrics-server
    namespace: kube-system
  version: v1beta1

```

```
versionPriority: 100
```

```

### 1. 执行安装命令:

```
bash
kubectl apply -f metrics-server.yaml
```

### 2. 安装成功后，运行以下命令查看 node 的资源使用情况

```
bash
kubectl top node
```

输出类似于：

| NAME             | CPU (cores) | CPU% | MEMORY (bytes) | MEMORY% |
|------------------|-------------|------|----------------|---------|
| master01-k8s-com | 218m        | 5%   | 5610Mi         | 35%     |
| node01-k8s-com   | 445m        | 11%  | 9279Mi         | 59%     |
| node02-k8s-com   | 464m        | 11%  | 9484Mi         | 60%     |

## 10.4 安装 DCE 5.0 社区版

现在一切准备就绪，开始安装 DCE 5.0 社区版。

### 安装基础依赖

DCE 已经提供了一键安装的离线工具依赖包，经过测试能够比较稳定地运行在 CentOS 7 和 CentOS 8 上。如果您也是这两个操作系统，可以直接运行下面的命令。

```
```bash linenum=1 curl -L0 https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/install_prerequisite.sh chmod +x install_prerequisite.sh sudo bash install_prerequisite.sh online community
```

当然也可以选择手动安装这些依赖项：

```
- podman
- helm > 3.9.4
- skopeo > 1.9.2
- kind
- kubectl > 1.22.0
- yq > 4.27.5
- minio client

### 下载 dce5-installer
```

注意，需要在 Master 节点执行 dce5-installer 的安装，建议[直接下载此 installer] ([https://docs.daocloud.io/download/dce5/\\_1](https://docs.daocloud.io/download/dce5/_1))。

下载并解压 tar 包之后，假定 `VERSION` 为 v0.3.28：

```
```bash linenum=1
export VERSION=v0.3.28
curl -Lo ./dce5-installer https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/dce5-installer-$VERSION
chmod +x dce5-installer
```

如果 proxy-qiniu-download-public.daocloud.io 链接失效，可使用 qiniu-download-public.daocloud.io。

### 设置配置文件 [可选]

将以下配置文件内容保存为 clusterConfig.yaml，如果使用 NodePort 的方式安装则不需要指定该配置文件。

```
yaml title="clusterConfig.yaml"
apiVersion: provision.daocloud.io/v1alpha1
kind: ClusterConfig
spec:
  loadBalancer: metallb
```

```
istioGatewayVip: 10.6.229.10/32 # (1)
insightVip: 10.6.229.11/32 # (2)
```

1. 这是 Istio gateway 的 VIP，也是 DCE 5.0 控制台的浏览器访问 IP
2. 这是全局服务集群的 Insight-Server 采集子集群监控指标的网络路径用的 VIP

如果使用配置文件，注意需要事先安装 `MetaLB`，这一部分需要自行完成。

## 执行安装

根据实际情况，运行以下命令之一：

- 无配置文件时

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

- 指定配置文件和 loadBalancer 时

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

## 安装完成

如果看到下方的页面，说明安装成功了；默认账号密码为：`admin/changeme`

安装成功

## 10.5 重定向 Portal 反向代理

通常安装后的集群访问地址是一个内网地址。当需要集群页面开放到公网访问时，会发现登录页会自动重定向内网地址，这是因为 DCE 5.0 反向代理服务器地址默认是安装时的配置，需要进行以下修改。

## 1. 设置环境变量

```
```bash
```

## 11. 反向代理地址

```
export DCE_PROXY="https://domain:port"
```

## 12. helm --set 参数备份文件

```
export GHIPPO_VALUES_BAK="ghippo-values-bak.yaml"
```

## 13. 获取当前 ghippo 全局管理的版本号

```
export GHIPPO_HELM_VERSION=$(helm get notes ghippo -n ghippo-system | grep "Chart Version" | awk -F ':' '{ print $2 }')````
```

## 2. 更新 Helm repo

```
bash
helm repo update ghippo
```

## 3. 备份 --set 参数

```
bash
helm get values ghippo -n ghippo-system -o yaml > ${GHIPPO_VALUES_BAK}
```

## 4. 使用 vim 命令编辑并保存

```
bash
vim ${GHIPPO_VALUES_BAK}
```

只需修改一行：

```
yaml
USER-SUPPLIED VALUES:
...
global:
...
reverseProxy: ${DCE_PROXY} # (1)
```

## a. 只需要修改这一行

注意这里需要把 `$(DCE_PROXY)` 替换为实际访问地址；需要配置完整的路径，包含 https 或 http、IP 或域名以及端口；如果是默认 80/443，则可以省略。

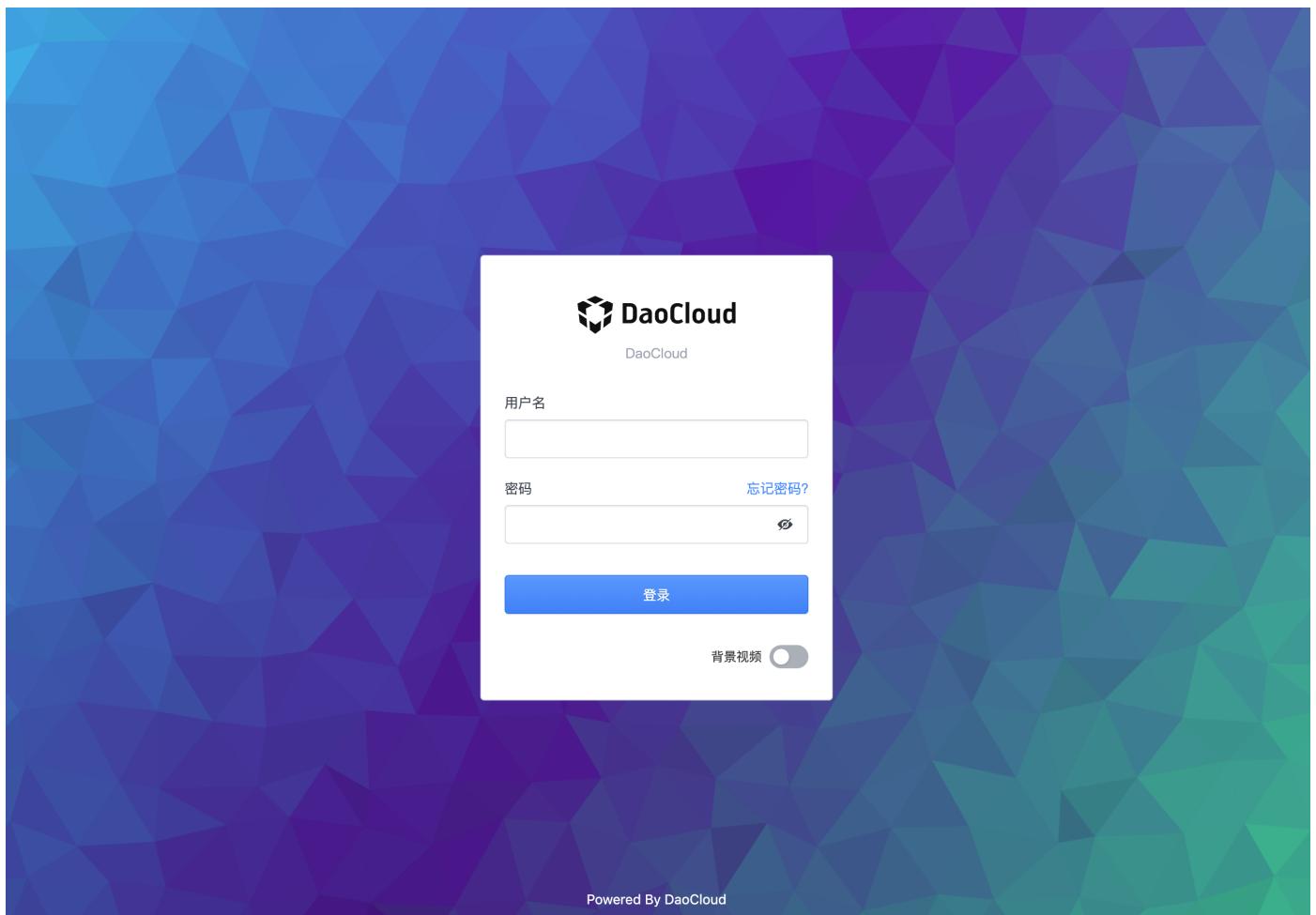
## 5. 使用 helm upgrade 更新配置

```
bash
helm upgrade ghippo ghippo/ghippo \
-n ghippo-system \
-f ${GHIPPO_VALUES_BAK} \
--version ${GHIPPO_HELM_VERSION}
```

## 6. 使用 kubectl 重启全局管理 Pod，使配置生效

```
bash
kubectl rollout restart deploy/ghippo-apiserver -n ghippo-system
kubectl rollout restart statefulset/ghippo-keycloak -n ghippo-system
```

## 7. 登录 DCE



8. DCE 登录成功

The screenshot shows the DaoCloud management dashboard with the following sections:

- 容器统计 (Container Statistics):** Displays a chart of container usage over time (21:00 to 21:30) and a summary table:
 

容器组	119	运行中	106	异常	13
-----	-----	-----	-----	----	----

 A legend indicates usage ranges: 90~100% (green), 80~90% (light red), 60~80% (blue), 40~60% (light blue), 0~40% (yellow), and Unknown (grey).
- CPU 用量 (CPU Usage):** Line chart showing CPU usage over time.
- 健康状态 (Health Status):** Shows a green "健康" (Healthy) status indicator.
- 告警 (Alerts):** Summary of alert counts:
 

紧急	4	警告	10	提示	4
----	---	----	----	----	---

 A log table lists recent alerts:
 

时间	描述
2022-12-26T12:31:28	etcdMembersDown:etcd clu...
2022-12-26T13:29:59	etcdInsufficientMembers:etc...
2022-12-26T12:29:59	KubeControllerManagerDown:...
2022-12-26T12:29:59	KubeSchedulerDown:KubeS...
- 集群统计 (Cluster Statistics):** Summary of clusters and nodes:
 

集群	1	节点	4
----	---	----	---

 Cluster name: kpanda-global-cluster
- 资源用量 (Resource Usage):** Circular progress charts for CPU, 内存 (Memory), 容器组 (Container Group), and 磁盘 (Disk):
 

资源	使用量 (%)	总计
CPU	2%	已用 1.895 core / 总计 64 core
内存	16%	已用 21.14 GB / 总计 125.04 GB
容器组	27%	已用 119 / 总计 440
磁盘	16%	已用 172.94 GB / 总计 1.03 TB
- 功能一览 (Function Overview):** Links to Container Management, Observability, and Global Management.

下载 DCE 5.0 { .md-button .md-button--primary } 安装 DCE 5.0 { .md-button .md-button--primary } 申请社区免费体验 { .md-button .md-button--primary }

# 14. 边缘原生应用准则白皮书

发布日期：2023 年 1 月 17 日（第一版），2022 年 10 月 24 日（草稿）

原文链接：[边缘原生应用准则白皮书](#)

## 14.1 目标

“边缘原生”一词已在行业博文等许多地方被提及，比如 Gartner、Macrometa、FutureCIO。而 State of the Edge 和 Linux Foundation (LF) 这些组织也在讨论边缘原生应用，但还没有关注边缘原生应用准则。

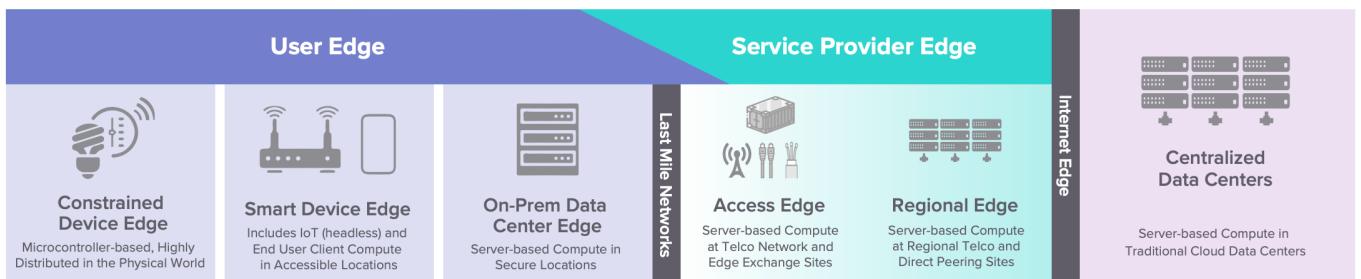
本白皮书侧重于边缘原生应用以及如何定义这些应用准则。

## 14.2 什么是边缘？

边缘计算使数据处理更接近源头，例如在工厂中控制机器人。未来五年，边缘计算将变得更加普遍，该行业预计从 2022 年到 2030 年将增长 38.9%。很多公司正在看到以下将计算能力放在边缘的好处：

- 减少延迟
- 带宽管理
- 增强隐私数据的安全
- 不可靠网络下的稳定运行

有多种边缘计算定义存在，但本文将重点关注基于数据处理所处的地理位置的边缘计算。基于地理位置的边缘被分类为多个类别，具体取决于与用户的距离。下图显示了根据 [Linux 基金会边缘白皮书](#) 定义的类别。



边缘原生准则与云原生准则存在许多相似之处，但也有一些关键的区别。

## 14.3 云原生 vs 边缘原生

根据 [云原生基金会\(CNCF\)](#) 的定义，云原生技术是：

“云原生技术赋予组织在现代动态环境中构建和运行可扩展的应用的能力，例如公有云、私有云和混合云。容器、服务网格、微服务、不可变基础架构和声明式 API 是这种方法的典范。”

这些技术使得松耦合的系统具有弹性、可管理和可观测性。结合强大的自动化，它们允许工程师频繁且可预测的进行高影响的更改，最大限度的减少繁琐工作。”

这一广泛的使命对于边缘应用仍然适用，因为[开放边缘计算术语表](#)指出，“边缘原生应用”参考了云原生准则：

“边缘原生应用是利用边缘计算能力构建的应用，不适合在中心云运行。边缘原生应用参考云原生准则，同时需要考虑边缘的独特特性，例如资源限制、安全性、延迟和自治性。边缘原生应用是利用云计算能力，并能与上游资源协同工作的方式构建的。不关心集中式资源管理、远程管理、编排、CI/CD 的边缘应用并不真正“原生”，而更类似于传统的本地应用。”

随着云原生用例涉及传统云以外的边缘位置的数据和事件，新的工具和技术正在不断发展，以实现松耦合的系统，具有弹性、可管控和可观测性，同时管理边缘的独特性。

## 14.4 边缘原生与云原生的相似性

边缘原生与云原生有许多相似之处，本节将描述这些相似之处。

属性	云原生 与 边缘原生
应用和服务的可移植性	应用和服务将它们与基础设施的耦合分离。一个良好编写的应用不需要知道它运行的位置，可以支持在平台之间移植。
可观测性	平台配备了一套良好的文档接口和工具选项，以便检测问题和收集指标。这使得开发人员可以构建具有弹性和高效管理的系统。
可管理性	提供接口和工具选项以规模化管理应用和资源。平台还具有插件机制，以提供基础网络连接、服务和管理功能。
支持多种语言和框架	应用和服务可以使用各种流行的语言和框架实现。

## 14.5 边缘原生与云原生的区别

边缘原生和云原生的广泛使命有相似之处，但开发人员应该意识到不同之处。

属性	云原生	边缘原生
应用模型	大多数微服务组件是支持水平扩展的无状态服务。	虽然服务提供的边缘应用非常相似，但用户边缘应用可能是单独的单体程序；在这些情况下，状态与应用相关联。
数据模型	常见的是支持无状态组件的集中式模型。	常常采用缓存、流式处理、实时和分布式模型。
弹性	快速启动和关闭；通常将底层资源视为无限制。	由于边缘设备的硬件资源受限，弹性受到限制；如果需要更多资源，会通过请求云端进行“垂直”扩展。
稳定性	将稳定性外包给云提供商，使用分布在不同地域的冗余节点。	通常依赖于经过强化的基础设施，具有面向有状态组件的恢复架构；在许多情况下，稳定性可能比云上的稳定性低。
规模	通常限于少数区域和实例	可以支持大规模区域（高达数万个），支持大量外部设备（高达数十万个）
编排	大型公有或私有云中的编排旨在通过在集中池化的主机上运行工作负载（以水平方式调度）来实现效率和可用性。	边缘是分散的，工作负载以分布式方式部署，通常以指定区域的方式调度。
管控	虽然云原生和边缘原生都是可管控的，但机制有所不同；云原生依赖于集中管控和自动化。	边缘原生需要远程和集中管控的混合方式以及硬件和软件的零接触部署（ZTP）。边缘的运维人员可能没有接受过培训，人数很少，甚至不存在。升级流程需要具有原子性和一致性，防止设备升级失败导致不可用。
网络	应用可以依赖高速网络。	应用需要考虑各种网速（不稳定、比较差、非常好）和功能。包括基于移动和无线的，集成来自非 IP 协议网络的数据和事件。
安全	安全管控的基础设施。	不可信不安全环境中的“零信任”。
硬件配置	很少需要关注硬件配置，能适用于大多数应用。	应用可能具有更高的实时要求，对硬件平台、位置和安全意识有要求。开发人员需要了解更广泛的硬件和接口。
与外部资源交互	应用很少需要与本地硬件资源交互。	在边缘部署的服务通常需要与本地环境交互：相机、传感器、执行器、用户等。

## 14.6 边缘原生应用

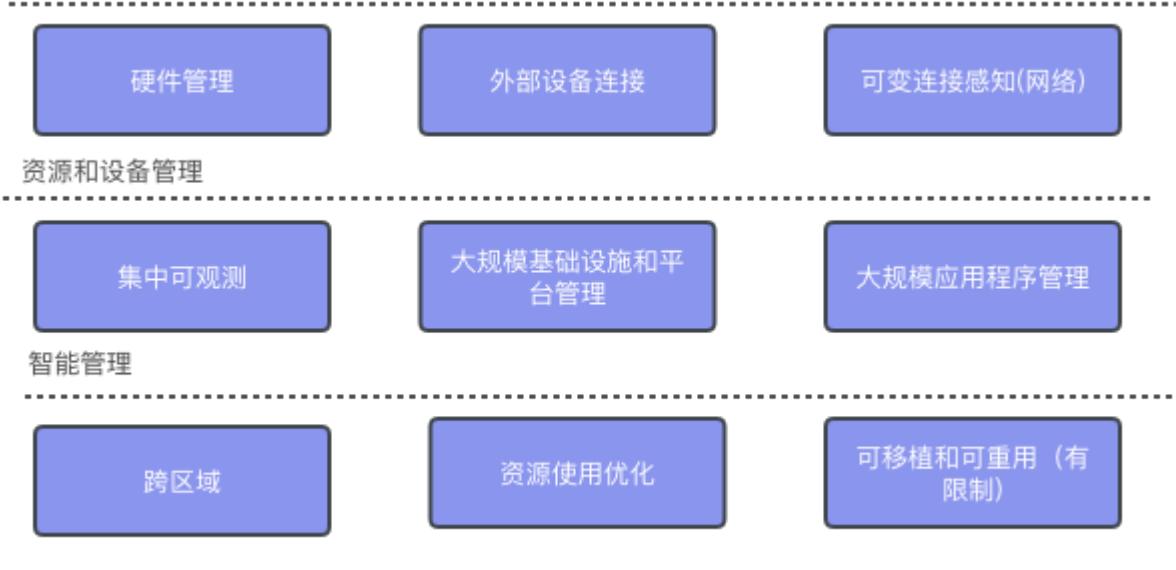
边缘原生应用是为边缘设计的应用和服务。它们参考以上相似性和差异性编写。以下是这些应用的核心准则。

## 14.7 边缘原生准则

为了实现本文前面提到的边缘原生的使命，边缘原生应用应遵循以下准则。

准则	描述
硬件管理能力	开发人员需要了解广泛的硬件平台和接口，而不是只有同质化的硬件平台。
外部设备连接	应用必须知道如何连接其环境中的设备，并了解运行时的功能变化。例如，它们在初始化配置后，能响应传感器连接/断开或新设备接入的情况。功能并不固定，需要考虑应用的环境，因此编排器需要能够协调应用状态和功能变化。
可变连接感知	应用必须适应不可靠甚至无法使用（完全隔离）的网络连接，使用异步通信、排队和缓存等机制。当边缘从中心站点获取配置时，可能需要使用“拉取”机制来克服规模、网络连接和安全问题。
集中可观测	虽然边缘和云原生应用都需要集中可观测，但边缘原生应用具有独特的考虑因素。边缘原生应用可能被部署到大规模实例中，存在运维人员和现场支持受限的情况。因此，需要采用数据分布式收集和集中聚合、开放式环路（人员可观察/可操作）和闭合环路（机器自动化）等技术。可观测性包括指标、日志、数字孪生、警报（事件和警报）和健康监测。
大规模基础设施和平台管理	基础设施和平台的管理在大规模边缘应用上非常重要，需要支持声明式的管理。此外，可能存在一些特殊的要求，例如设备接入、横向扩展限制、管理裸机环境等。在平台层面，部署或管理 Kubernetes 或虚拟化层以及各种插件也是一个问题；需要保持平台层面的供应商中立，以实现应用的可移植性。
大规模应用管理	应用的数量和这些应用的实例数量在边缘可能非常大，需要基于声明式的规则和条件来配置、通过自动化服务生效，以及跨多个应用实例的聚合管理视图。应用也可能有实时需求，这意味着应用和基础设施平台之间的联系（例如使用 GPU、DPU、FPGA、CPU 架构、内核优化、Kubernetes 插件）可能比云应用更紧密。换句话说，应用编排可能触发底层的基础设施和平台编排。
跨区域	应用不止部署在一个区域，存在跨区域的延迟和故障。事实上，边缘应用也可能跨公有云、私有云。
资源使用优化	由于边缘计算资源受限，应用必须持续优化资源使用。按需调整应用，基于部署位置和可用性意图的迁移和伸缩。这意味着在一天的时间里存在不同的运行工作负载。
应用可移植和可重用 (有限制)	抽象层试图通过供应商中立的 PaaS 提供与基础设施和平台无关的可移植性。但由于本地资源、硬件平台、安全、移动网络等限制，配置选项需要适应本地差异。

### 边缘原生准则分组



这九项准则可以归纳为一组较小的五项准则。硬件管理、外部设备连接、可变连接感知(网络)都可以在更广泛的资源和硬件管理准则下考虑。同样，边缘应用可大规模管理、可集中观测以及具有可管理的基础设施和平台，这些都可以归类为大规模管理的准则。以下是扩展的五个准则：跨区域、资源使用优化、可移植和可重用限制、资源和硬件管理以及规模管理。

## 14.8 结论和后续步骤

本文为第一版，可能会进行修订。后面会有一些与本文子内容相关的论文。

## 14.9 如何参与

CNCF IoT Edge 工作组有定期会议、邮件列表和 Slack。有关最新信息，请参阅工作组 GitHub 页面的[通信部分](#)。我们欢迎读者参与，介绍 Edge 相关项目，为小组的工作领域提出想法，或帮助修改本白皮书和起草后续文件。

## 14.10 边缘原生开源项目和计划的工作列表

作为本文的一部分，CNCF IoT Edge 工作组正在收集开源项目的工作列表，这些项目可帮助应用开发人员实现本文中概述的边缘原生应用准则。

可以在此[电子表格](#)中或通过二维码找到该列表。要获得添加项目的编辑权限，请加入 [IoT Edge Working Group Google 群组](#)。



## 14.11 贡献者

### 作者

- Amar Kapadia, Aarna Networks
- Brandon Wick, Aarna Networks
- Joel Roberts, Cisco
- Kate Goldenring, Fermion
- Dejan Bosanac, Red Hat
- Tomoya Fujita, Sony US Lab
- Ravi Chunduru, Verizon
- Natalie Fisher, VMware
- Steven Wong, VMware

### 审稿人

- Frédéric Desbiens, Eclipse Foundation
- Prakash Ramchandran, eOTF
- Mark Abrams, SUSE

## 14.12 参考资料

Linux 基金会边缘白皮书: [https://www.lfedge.org/wp-content/uploads/2020/07/LFedge\\_Whitepaper.pdf](https://www.lfedge.org/wp-content/uploads/2020/07/LFedge_Whitepaper.pdf)

开放边缘计算术语表 [v2.1.0] State of the Edge: <https://github.com/State-of-the-Edge/glossary/blob/master/edge-glossary.md#edge-native-application>

云原生组织(CNCF)章程: <https://github.com/cncf/foundation/blob/main/charter.md>

Gartner “云原生不等于边缘原生”: [https://blogs.gartner.com/thomas\\_bittman/2020/04/17/cloud-native-isnt-edge-native/](https://blogs.gartner.com/thomas_bittman/2020/04/17/cloud-native-isnt-edge-native/)

Macrometa “边缘原生非云原生”: <https://www.macrometa.com/blog/edge-native-is-not-cloud-native>

Future CIO “云原生和边缘原生区别”: <https://futurecio.tech/cloud-native-versus-edge-native-know-the-difference/>

《边缘计算市场规模、份额和趋势分析报告》，按组件（硬件、软件、服务、边缘管理平台）、应用、行业垂直、地区和细分预测，2022 年至 2030 年: <https://www.grandviewresearch.com/industry-analysis/edge-computing-market>

## 15.5.0 多云编排能力介绍

在过去的两年里，随着公有云服务的成熟、云供应商产品的差异化，许多企业为了在数字化转型的队伍中，提高效率并保持竞争优势，开始采用混合IT和多云解决方案，从而加速多云战略的步伐。以前位于本地数据中心的集中式应用和工作负载逐渐变得分散，这一趋势正在重新定义传统数据中心的边界，并迫使应用现代化和基础设施的扩展，以便适应日益重要的分布式工作负载。依靠私有云、多个公有云和传统基础设施平台的组合，企业可以根据自己的特定优势自由选择云，从每个云提供商的特定优势中受益。通过这种方式，构建最适合企业的应用架构，从而保证业务高可用、增强应用弹性、提升负载能力。当然，企业也可以通过构建冗余部署，实现快速的故障转移，来降低单个云供应商的宕机的风险。



在面对复杂的多云环境，5.0多云编排将给我们带来一致的集群运维，降低管理负担；跨集群应用分发能力，打破数据孤岛，实现应用跨集群级别的容灾部署。

### 15.1 功能特性

多云编排的主要功能如下：

多云集群实例管理

- 多云集群实例一键部署，通过 Web 界面快速部署企业级 Karmada 集群，支持创建多个 Karmada 实例，它们之间同时工作，实例之间互不感知、互不影响。
- 工作集群一键导入，支持从 Kpanda 现有纳管集群中一键导入集群到 Karmada 实例中，并实时同步 Kpanda 集群最新信息。
- 实例信息概览，支持查看实例内已接入的工作集群的 CPU、内存的使用率等情况。
- 工作集群管理，支持动态将新集群加入到 Karmada 实例中；支持动态将特定集群从 Karmada 移除。
- CloudShell，支持获取 KubeConfig 链接信息，用户可在本地终端管理 Karmada 实例。
- Karmada 原生 API 支持，支持所有 Karmada 原生 API。

### 多云工作负载

- 支持界面化创建无状态工作负载，添加差异化配置、分发策略
- 支持查看无状态负载的部署详情，单 Pod 监控、日志信息
- 应用故障转移，内置提供应用多云故障转移能力
- 跨集群工作负载分发，丰富的多云应用分发策略、覆盖策略
- 可观测性，丰富的审计、Metrics 暴露

### 资源管理

- 支持多云命名空间管理
- 支持多云存储声明管理
- 支持多云配置、密钥管理
- 支持多云 Service 和 Ingress 资源管理

### 策略中心

支持部署策略、差异化策略管理，查看策略关联的工作负载，提供删除空闲的部署策略、差异化策略的功能。

## 15.2 图形化用户交互

多云编排模块是一个多集群管理的聚合平台，支持用户接入不同厂商、不同地域的集群进行统一管理。帮助企业快速跨集群部署工作负载，满足用户跨集群管理、容灾部署等使用场景。在层级结构上，采用两级架构的形式，最上层提炼出多云集群列表，选择进入多云集群管理中心，查看到概览、工作集群管理、多云工作负载、资源管理、策略中心。

用户在多云集群实例列表界面，能够清晰的感知每一个多云集群实例的运行状态、资源用量、集群数等实时信息。并且能够基于多云集群实例名称进入详情。同时如果用户想使用 kubectl 的方式管理集群，也可以在界面上使用基于 Cloudtty 桌面控制台工具。

在 概览 界面，用户能够查阅到详细的集群概览信息、跨集群工作负载的健康状态、资源信息等。

在 工作集群管理 界面，用户能够对多云集群的接入集群，进行管理，并能够查看到每个集群的状态、发行版本等信息。

在 多云工作负载 界面，用户能够感知所拥有权限的当前多云集群下的工作负载状态、实例数等信息。并通过名称进入工作负载详情，查看日志、监控等。

在 资源管理 界面，用户能够管理服务与路由、命名空间、存储声明、配置与密钥模块。这些资源都是跨集群创建，以满足跨集群工作负载的部署所依赖的资源。

在 策略中心，用户可以对通过工作负载所关联创建出来的部署策略、差异化策略进行查看，但不允许对正在使用中的策略进行删除。另外，用户也可以自定义创建部署策略、差异化策略，解锁高级使用模式。

## 15.3 多云编排可以做什么

您能够通过 Web UI 的方式，对多云编排进行管理。您可以借助多云编排模块实现创建多云集群实例、接入集群、应用跨集群部署等功能操作。其中，多云工作负载的能力，完全对齐 Kubernetes 的工作负载。

### 创建多云集群

在多云编排模块，用户可以一键创建多云集群实例，然后在实例创建成功之后，在详情页面，进行工作集群的既接入、移除。同时可以通过给工作集群打上污点，实现工作负载的驱逐。

### 接入成员集群

多云编排会同步所有已经接入容器管理模块的集群，用户只需要在多云集群实例的工作集群处点击 接入集群 按钮，选择需要接入的集群，点击 确认，既完成了对集群的接入操作。

### 部署多云工作负载

多云编排模块支持基于 Kubernetes 原生的工作负载类型部署和管理能力，包括：创建、配置、扩容、删除等全生命周期管理。另外支持集群差异化配置，让用户轻松应对不同集群的配置管理。

- 实例调度策略

目前多云编排支持以复制的方式，将实例数部署到用户所选择的集群上，也就是说，实例数配置成 2，并选择了 Cluster A 和 Cluster B 两个工作集群，那么，会在 A 和 B 上同时部署一个实例数为 2 的工作负载。后续会逐步支持更多的实例调度策略。

- 集群亲和性和反亲和性

用户可以通过地域、可用区进行集群的选择，也可以通过亲和性和反亲和性配置，实现更精细的集群选择。

- 差异化配置

用户可以选择需要差异化配置的集群，然后会看到前面步骤已经配好的通用配置，在差异步骤修改配置，实现不同集群上的工作负载使用不同配置运行。

- 资源管理

对于跨集群的工作负载创建时，集群间的配置差异不可避免。所以，我们可以在资源管理模块，进行配置与密钥、存储声明的创建，从而在创建工作负载的时候，快速进行差异化配置。资源管理模块还有对命名空间、Service 和 Ingress 的管理，这些资源均会跨集群创建。

- 策略中心

在策略中心，用户可以管理通过工作负载关联创建出来的部署策略和差异化策略，提供对空闲策略的删除功能。也支持以 YAML 的方式维护创建和编辑部署策略、差异化策略。

## 15.4 常见问答

- 问：多云编排兼容哪些厂商的集群？

答：5.0 多云编排兼容 5.0 容器管理支持的厂商集群，包括 AWS、谷歌云、信创架构、边云融合等任何基于标准 Kubeneters API Server 的发行版容器集群。

- 问：如何实现对其他厂商集群的管理？

答：通过获取其它厂商集群的 KubeConfig 文件，在容器管理模块执行接入集群操作，填入目标集群的 KubeConfig 文件，即完成了对一个容器集群在容器管理的接入管理。然后在多云编排模块，可以在多云集群实例的工作集群列表中，接入新加入容器管理模块的集群。

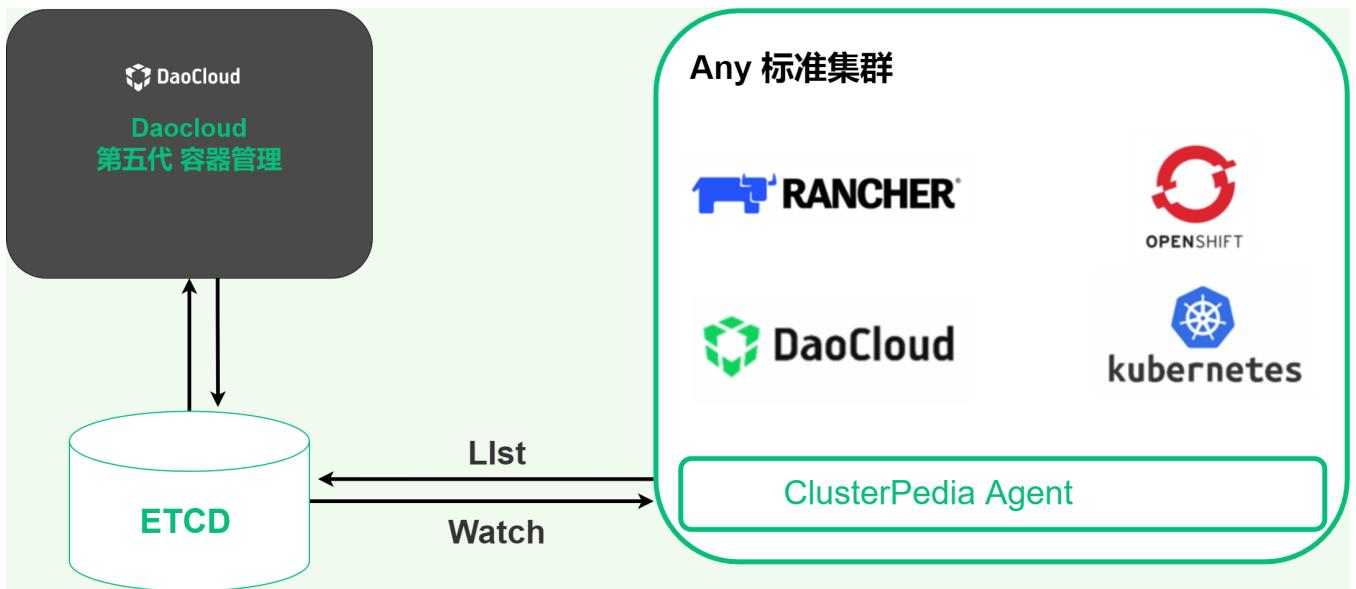


- 问：应用如何实现跨集群通信？

答：目前，多云编排模块没有内置应用跨集群通信的实现。我们可以集成 Submariner，将连接的集群之间的网络扁平化，实现集群间 Pod 和服务的 IP 可达性，Submariner 还通过 Lighthouse 提供跨集群的服务发现能力。

- 问：如何获取多集群资源的实时状态？

答：借助自研开源组件 ClusterPedia，通过在目标集群上安装 Agent 组件，结合 Watch 机制能够实时将目标集群的资源变动信息同步到容器管理集群的 ETCD 内。多云编排直接查询容器管理的 ETCD，从而获取多集群资源的实时状态。



- 问：5.0多云编排在第五代产品中的定位？

答：在第五代产品中，多云编排模块处于承上启下的核心位置，对上：对接 DCE 5.0 应用工作台。对上层的场景化管理模块而言，多云编排模块能够把任何一个标准的 Kubernetes API 对上暴露，轻松实现应用跨集群分发、容灾部署。对下：对接容器管理平台 Kpanda，而 Kpanda 对接 Any Kubernetes，例如边缘的 K3S、信创环境、DCE、DKG、DKE、以及外部的 Openshift, Tanzu、CCE等等，多云编排基于容器管理模块，快速实现多集群管理。



- 问：谁能够创建多云集群？

答：拥有 Global Admin、Kairship Owner 角色的用户能够去创建多云集群。

[了解多云编排](#)

[下载 DCE 5.0](#) [安装 DCE 5.0](#) [申请社区免费体验](#)

## 16. DaoCloud 是 K8s 资深认证服务商

在云原生领域有这样一个词：KCSP，英文全称 Kubernetes Certified Service Provider，即 CNCF 官方认证的 Kubernetes 服务提供商。

DaoCloud 早在 2017 年就首次顺利通过了 Kubernetes 认证，是国内最早涉足并得到 CNCF 官方认可的服务商，同时也是国内最早获得 Kubernetes Training Partner (KTP) 认证的厂商。DaoCloud 的开发者们常年积极参与 CNCF 相关社区贡献，以引领云原生技术潮流为己任，致力于推动云原生社区发展壮大。

每次 Kubernetes 新版本发布，DaoCloud 都会率先适配增强新特性，并将其推向实际生产环境。作为一家经过官方严格审核的 Kubernetes 服务商及合作伙伴，DaoCloud 在帮助企业采用 Kubernetes 成功部署大规模集群方面具有深厚的生产经验。

DaoCloud 作为一家久经认证的 KCSP 意味着：

- DaoCloud 是 [Kubernetes 官方推荐的资深合作伙伴](#)
- DaoCloud 是经 CNCF 社区公认、有专家资质、能充分胜任 Kubernetes 集群部署专业工作的优良企业
- 最终用户选用 Kubernetes 部署集群时，DaoCloud 是 CNCF 云原生社区公认并推荐的品牌供应商
- 常年位列 KCSP 的 DaoCloud 掌握云原生领域领先技术，时刻引领云原生社区发展方向

目前 DaoCloud 经认证合规且完美支持的 Kubernetes 版本包括：

当前版本：

[1.23](#) [1.24](#) [1.25](#) [1.26](#)

历史版本：

[1.7](#) [1.9](#) [1.13](#) [1.15](#) [1.18](#) [1.20](#)

### 16.1 技术领先

DaoCloud 的研发团队崇尚技术为先，广受 CNCF 社区肯定，担任了很多云原生项目的 Member、Reviewer、Approver/Maintainer 助力社区繁荣发展，还有大咖加入 Steering Committee 负责推进各项底层特性的研发和拓展。

头衔	数量
Member	52
Reviewer	15
Approver/Maintainer	6
Technical Steering Committee	1
CNCF Ambassador	1
Certified Kubernetes Administrator (CKA)	65
Certified Kubernetes Application Developer (CKAD)	26
Certified Kubernetes Security Specialist (CKS)	8

在 2022 年 8 月由 Kubernetes 官方组织的社区贡献者访谈活动中，接见了来自亚太地区的 4 位优秀贡献者，其中 2 位都来自 DaoCloud，他们为 Kubernetes 的发展做出了卓越贡献。

DaoCloud 在推动社区技术演进的同时，对 Kubernetes 为首的云原生技术保持敏锐的嗅觉。以 Kubernetes 1.25 为例，第一个小版本 1.25.1 于 2022-09-14 发布，DaoCloud 经过一个多月的适配调试后于 2022-10-25 提交了[合规认证申请 PR](#)，附带 5 万多行真实的 E2E 测试报告、完善的[中英文档站](#)、详尽的资料素材，于 2022-11-04 成功通过了 v1.25 的 KCSP 审查。

目前能对 Kubernetes 1.25 完美支持并经官方认证的 国内厂商不超过 5 家。有鉴于此，华为云在[元宇宙云边协同超融合一体机](#)方面与 DaoCloud 建立战略合作关系，双方在多云编排、智能存储等方面也有深度合作关系。

DaoCloud 围绕 Kubernetes 生态自主开源的社区项目有：

- [Clusterpedia](#): Kubernetes 多集群资源一站式检索百科全书，已入选 CNCF 全景图和 CNCF 沙箱
- [CloudTTY](#): 社区首个开源的 Kubernetes 网页版控制台
- [Ferry](#): Kubernetes 多集群通信组件，消除多集群复杂度，如同管理一个单集群
- [HwameiStor](#): 高可用的本地存储方案，已入选 CNCF 全景图
- [KLTS](#): 对 Kubernetes 10 多个版本的长期维护
- [Kubean](#): 基于 kubespray 构建的集群生命周期管理工具
- [kwok](#): 用一台笔记本以最小资源模拟成千上万的 kubelet
- [Merbridge](#): 使用 eBPF 加速服务网格，已入选 CNCF 全景图
- [Spiderpool](#): 云原生网络 IPAM 自动化管理软件，适用于 Underlay CNI

## 16.2 CNCF 贡献排名

---

考量一家云原生企业的指标之一是社区贡献，DaoCloud 在 CNCF 社区各云原生项目的总体贡献全球排名领先，在国内首屈一指。

过去 365 天内 DaoCloud 对 Kubernetes 社区的贡献 全球排名第 3



OpenInfra Foundation

CNCF

Other Projects

about

DATE

365 days

PROJECT TYPE

CNCF

PROJECT

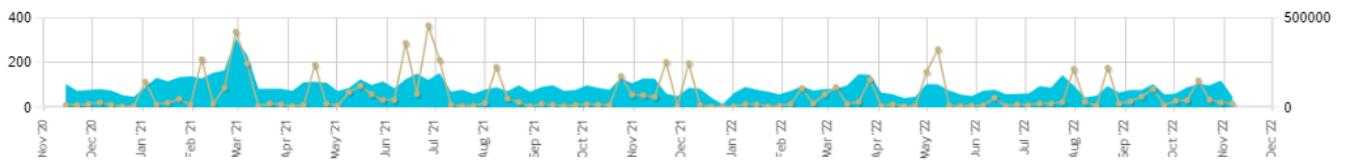
kubernetes

COMPANY

Any company

CONTRIBUTOR

Any contributor



## Kubernetes

Repo: <https://github.com/kubernetes/kubernetes.git>

### Commits by Company

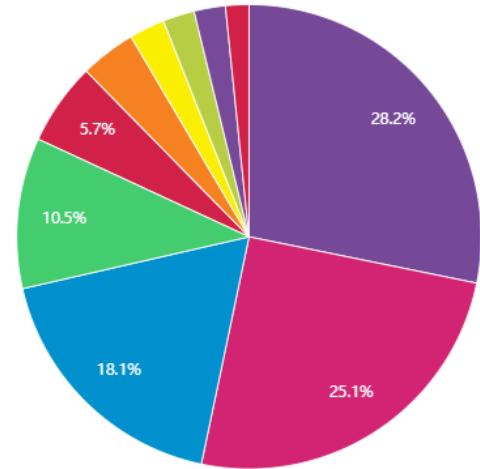
Show 10 entries

Search

#	Company	Commits
	*Independent	1164
1	Google	1036
2	Red Hat	748
3	DaoCloud	237
4	Intel	160
5	Futurewei	102
6	BoCloud	91
7	VMware	90
8	ARM	67
9	ZTE Corporation	65

Showing 1 to 10 of 46 entries

Previous Next



过去 365 天内 DaoCloud 对 containerd 社区的贡献 全球排名第 5



OpenInfra Foundation

CNCF

Other Projects

about

DATE

PROJECT TYPE

PROJECT

COMPANY

CONTRIBUTOR

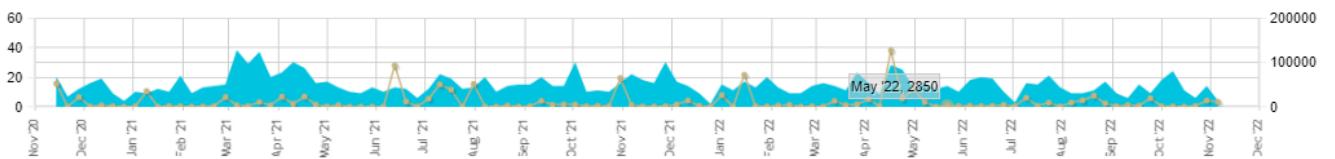
365 days

CNCF

containerd

Any company

Any contributor



### Containerd

Repo: <https://github.com/containerd/containerd.git>

### Commits by Company

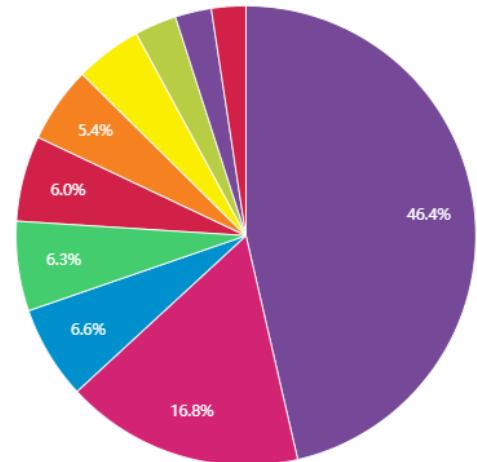
Show 10 entries

Search

#	Company	Commits
	*independent	346
1	Amazon	125
2	NTT	49
3	Cloudbase Solutions	45
4	Microsoft	40
5	DaoCloud	35
6	Google	22
7	Intel	19
8	IBM	18
9	Docker	15

Showing 1 to 10 of 21 entries

Previous Next



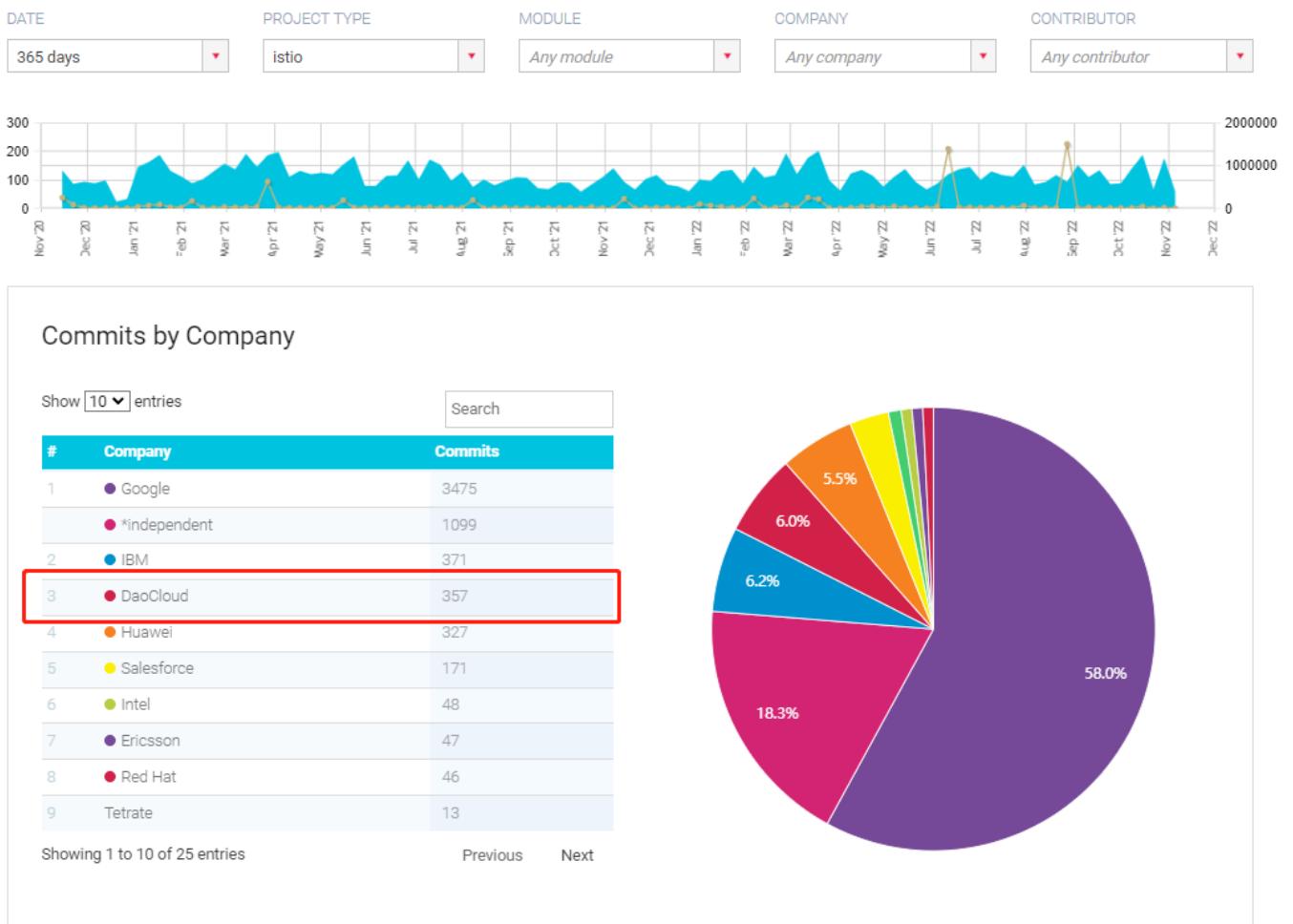
过去 365 天内 DaoCloud 对 Istio 社区的贡献 全球排名第 3



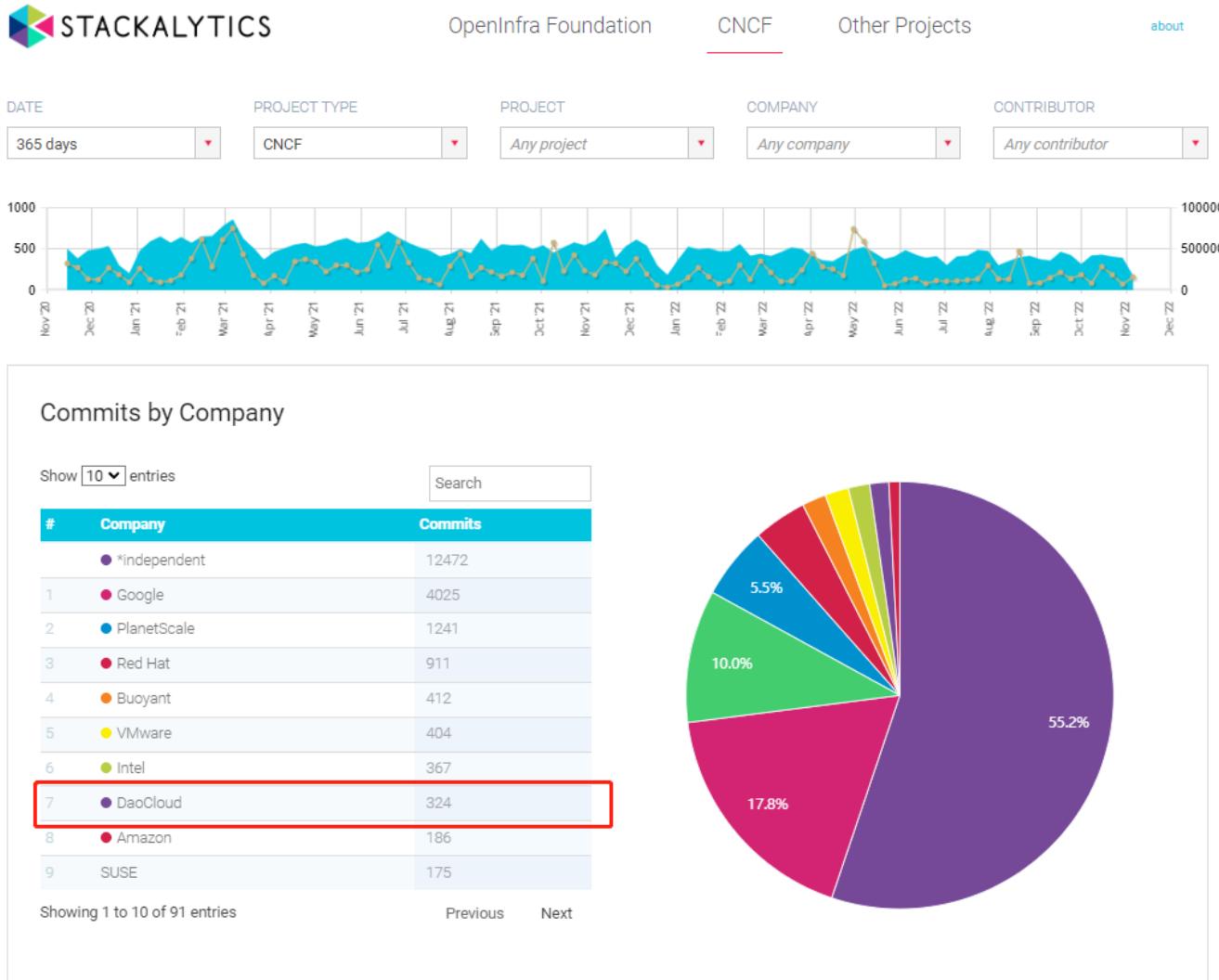
OpenInfra Foundation

CNCF

Other Projects

[about](#)

过去 365 天内 DaoCloud 对 CNCF 社区所有项目 的累计贡献 全球排名第 7



## 16.3 参考链接

- <https://www.cncf.io/>
- <https://github.com/cncf/k8s-conformance>
- <https://kubernetes.io/partners/>
- <https://www.stackalytics.io/>

下载 DCE 5.0{ .md-button .md-button--primary } 安装 DCE 5.0{ .md-button .md-button--primary } 申请社区免费体验{ .md-button .md-button--primary }

## 17. 5.0 容器管理能力介绍

自 2013 年 Docker 开源以来，以 Docker 为代表的容器技术为软件行业带来了颠覆性的变革。数字经济成为当代经济发展的重要驱动力量。基础设施云化，软件云原生化已成了整个社会科技发展的必然趋势。以金融行业为代表，传统制造业为基石的各行各业都在积极拥抱和探索这一技术发展的新趋势。云原生技术抽象了底层基础设施，使开发人员能够专注于应用本身开发，而无需担心底层系统架构等运维问题，让应用更加弹性、高效。 DaoCloud 第五代容器管理（Kpanda）是基于 Kubernetes 开源技术构建的面向云原生工作负载的容器管理平台，基于原生多云架构，完全解耦底层基础设施平台，实现多云与多集群统一化管理，进一步简化企业的应用上云流程，降低运维管理和人力成本。支持一键式创建 Kubernetes 集群，帮助企业快速搭建企业级的容器云平台。

容器管理的主要功能如下：

The screenshot shows the DaoCloud Kpanda management interface. At the top, there is a navigation bar with the DaoCloud logo and the text "第五代容器管理平台". Below the navigation bar, there is a horizontal menu bar with several buttons: "集群管理" (Cluster Management), "节点管理" (Node Management), "负载管理" (Load Management), "插件管理" (Plugin Management), "网络访问" (Network Access), "存储管理" (Storage Management), "权限管理" (Permission Management), "集群运维" (Cluster Operations), "开放式 API" (Open API), and "CloudShell". Below the menu bar, there is a section titled "IaaS 基础设施" (IaaS Infrastructure) which lists various cloud providers and Kubernetes distributions. The listed providers include: 自建集群 (Self-built Cluster), 边缘集群 (Edge Cluster), 信创架构 (X86 Architecture), 裸金属架构 (Bare Metal Architecture), 其它厂商集群 (Other Vendor Clusters) (aws, Google Cloud, OPENSHIFT, RANCHER, HUAWEI, 腾讯云, 阿里云, kubernetes), 华为云 (Huawei Cloud), 腾讯云 (Tencent Cloud), 阿里云 (Aliyun), and kubernetes.

### 集群管理

- 多云多集群统一纳管，支持所有特定版本范围内的任意 Kubernetes 集群纳入容器管理管理范围，实现云上云下多云、混合云容器云平台的统一纳管。
- 集群快速部署，通过 Web 界面快速部署企业级的 Kubernetes 集群，快速搭建企业级容器云台，适配物理机和虚拟机底层环境。
- 集群一键升级，支持一键升级集群 Kubernetes 版本。
- 集群高可用，内置集群容灾、备份能力，支持业务系统在主机故障、机房中断、自然灾害等情况下快速恢复，进一步提高生产环境的稳定性，降低业务中断风险。

### 节点管理

- 支持对管理集群和基于管理集群创建工作集群内的节点进行全生命周期管理。
- 实时监控节点状态。支持实时对节点和节点上运行的容器组状态进行监控。
- 支持采用界面 CloudShell 的方式登录并管理节点。
- 提供对节点的调度策略、标签、污点、注解等进行管理的基础云原生能力。

### 负载管理

- 一站式部署，解耦底层 Kubernetes 平台，一站式部署和运维业务应用，实现工作负载的全生命周期管理。
- 工作负载负载的扩缩容，通过界面可实现工作负载手动/自动扩缩容能力，自定义扩缩容策略，应对流量高峰。
- 工作负载的全生命周期，支持工作负载查看，更新，删除，回滚，时间查看以及升级等全生命周期管理。
- 跨集群负载统一管理能力。

### 插件管理

- 内置 DaoCloud 自研核心插件库，支持引入 Helm 官方库和私有插件仓库，为用户提供海量的插件。
- 提供对插件进行增删改查的全生命周期管理能力。
- 对插件的安装、更新、删除等操作进行记录，并以运维人员更容易理解的控制台日志进行展现，使插件运维更加云原生化。

### 网络访问

提供云原生的负载均衡能力，通过固定的 IP 地址和端口进行访问。目前支持的 网络访问模式包括：

- 集群内访问（ClusterIP）：仅在集群内访问服务。
- 节点访问（NodePort）：使用节点 IP 进行访问。
- 负载均衡（LoadBlance）：基于公有云厂商的外部负载进行访问。

#### 存储管理

- 数据卷管理：支持本地存储、文件存储、块存储，通过 CSI 能力接入，并提供给工作负载工作负载使用。
- 数据卷动态创建：可支持存储池实现动态创建数据卷
- 存储池管理：兼容业内各类主流存储技术和符合标准 Kubernetes CNI 架构的存储插件。

#### 权限管理

支持多集群权限的统一管理，结合 Kubernetes 原生的 RBAC 角色访问控制和全局管理模块的用户身份认证，提供工作空间级别、集群级别和命名空间级别的权限控制。基于以往金融、银行、传统制造等多行业的生产运维管理经验，提供开箱即用的多种企业权限控制方案，帮助不同组织架构的企业快速对用户/用户组进行授权。

#### 集群运维

支持集群级、节点级、负载级、容器级的全方位指标监控及告警，集成各类资源状态统一监控看板。提供集群自动/手动巡检能力，帮助运维和开发人员实时了解集群资源状态，保障业务连续性。

#### 开放式 API

提供云原生的 Kubernetes OpenAPI 能力，打通容器管理与不同系统的界限，快速赋能用户业务上云。

#### 对 Kubernetes 友好的 Web 终端 CloudTTY

很多企业在使用云平台管理 Kubernetes 时，出于安全考虑，不支持使用 SSH 的方式直连业务主机，因此需要云 Shell 能力。Daocloud 第五代容器管理模块引入了自研开源组件 CloudTTY，支持通过 CloudShell 直连集群。并提供通过 Kubectl 工具对集群进行运维操作的能力。

## 17.1 图形化用户交互

容器管理模块是一个多集群管理的聚合平台，支持用户对不同厂商、不同区域的集群进行统一的纳管。帮助企业快速上云，满足用户多云多集群管理的使用场景。在层级结构上，采用了三级架构的形式，最上层将用户常用的使用场景进行抽象，提炼出集群列表、命名空间、工作负载、权限管理几个模块。

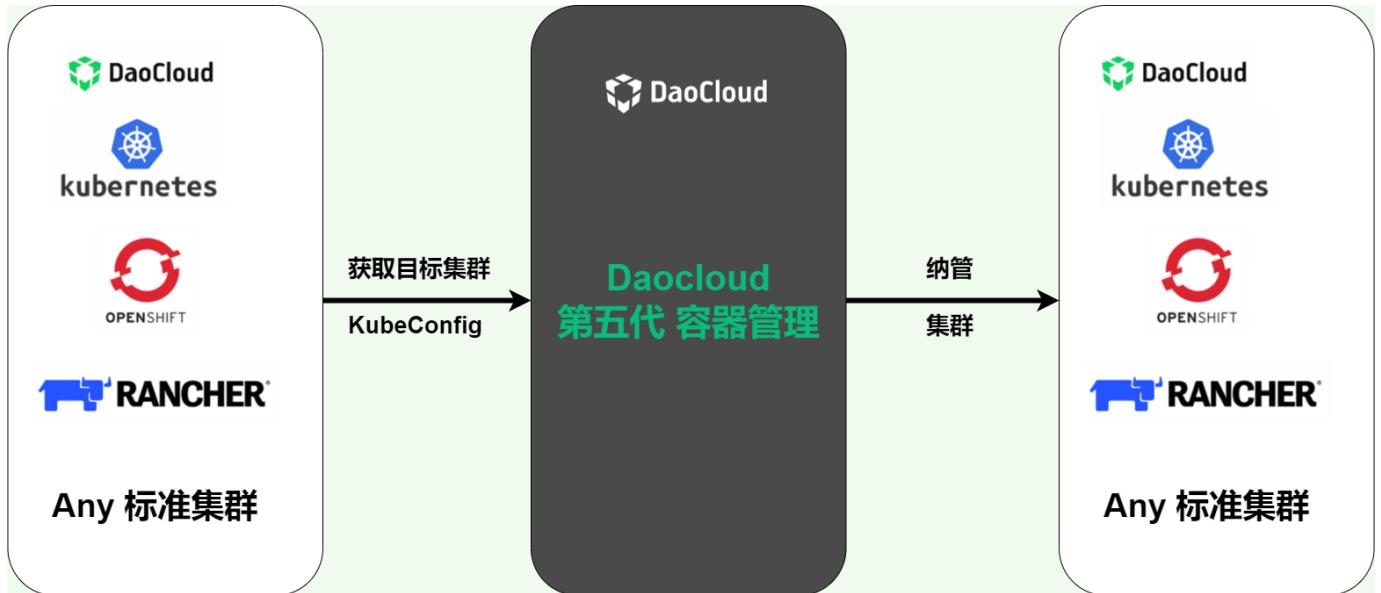
- 用户通过 集群列表 界面，能够清晰的感知每一个集群的运行状态、资源用量、节点/容器组数量等实时信息。并且能够基于集群名称进入集群的详情之中，对集群的节点、工作负载、服务与路由、运维管理等功能进行精细化管理。同时如果用户想使用 kubectl 的方式管理集群，也可以在界面上使用基于 Cloudtty 桌面控制台工具。
- 在 命名空间 界面，用户能够感知所拥有权限的全部集群下的命名空间的状态，并且能够对不同的命名空间进行配额设置和绑定/解绑工作空间。
- 在 工作负载 界面，用户能够感知所拥有权限的全部集群下的工作负载状态、资源使用率等工作负载的全局概览信息。
- 在 权限管理 界面，用户能够基于集群、命名空间两个层级对用户/用户组的权限进行查看以及授权。
- 在 集群详情 界面，用户能够查阅到详细的集群概览信息，并且能够对集群内的资源进行全生命周期的管理。包括节点管理、服务与路由管理、工作负载和容器组管理、命名空间管理、Helm 插件管理、存储管理、运维管理等功能都能够在集群详情界面内进行操作。

## 17.2 最佳实践探索

在本地构建起第一个全局服务集群（Kpanda）后，您就能够通过 Web UI 的方式，对容器集群进行管理了。您可以借助容器管理模块实现接入集群、创建集群、应用一键部署等功能操作了。

#### 接入集群

首先前往目标集群的控制节点上获取它的 `KubeConfig` 文件，然后在容器管理的 集群列表 处点击接入集群按钮，并将 `KubeConfig` 文件内的内容作为参数复制粘贴到 接入参数 的表单内，既完成了对一个集群的接入操作。



#### 创建集群

容器管理模块支持创建工作集群和管理集群两种集群类型，用户可根据自身业务需要，选择相应的集群模式。管理集群和工作集群的差别是在管理集群只运行系统组件，不部署业务，以保障集群的高可用。工作集群只运行业务相关的负载，同时管理集群支持纳管工作集群，能够基于管理集群创建工作集群，并在创建过程中为工作集群配置网络、节点、插件等操作。

#### 部署工作负载

容器管理模块支持基于 Kubernetes 原生的工作负载类型部署和管理能力，包括：创建、配置、监控、扩容、升级、回滚、删除等全生命周期管理。

- 弹性扩缩：通过界面可实现工作负载的手动/自动扩缩容能力，自定义扩缩容策略，应对流量高峰。
- 容器的生命周期设置：支持创建工作负载时设置回调函数、启动后参数、停止前参数，满足特定场景需求。
- 容器就绪检查、存活检查设置：支持部署工作负载时设定工作负载就绪检查及存活检查：

  - 工作负载就绪检查：用于检测用户业务是否就绪，如未就绪，则不转发流量至当前实例。
  - 工作负载存活检查：用户检测容器是否正常，如异常，集群会执行容器重启操作

- 容器的环境变量设置：可为业务容器运行环境设定指定环境变量。
- 容器的自动调度：支持工作负载服务调度管理，自动按照主机资源用量进行容器调度，可以指定具体的部署主机，以及通过标签策略进行容器调度。
- 亲和性和反亲和性：支持定义容器组间调度的亲和性和反亲和性；以及容器组与节点间的亲和性和反亲和性，满足业务自定义调度需求。
- 容器安全用户设置：支持设定容器运行用户，如以 Root 权限运行则填写 Root 用户 ID 0。
- 自定义资源（CRD）支持：支持自定义资源的创建、配置、删除等全生命周期管理及基于自定义资源创建实例（CR）的能力。

## 17.3 谁需要容器管理

业务庞大、或跨地区或跨部门需要统一运维管理的大型金融、银行、航空航天、传统制造业等类型的企业。

### 解决问题

1. 在现代企业的软件生成过程中，虽然一般来讲，最佳做法是使用尽可能少的集群，但企业出于各种原因会选择部署多个集群来实现各种技术和业务目标。大多数企业至少将各种服务放在不同的集群中，将非生产服务与生产服务分开。在更复杂的场景中，企业可能会选择多个集群来跨层级、语言区域、团队或基础架构提供商分隔服务。
2. 超大规模项目部署在一个集群上，为了保证工作负载的可用性，只能手动不断的往集群上新增节点，但是 Kubernetes 的最大 Pod 数有边界，单纯依靠增加物理节点并不能解决根本问题，且运维难度和成本极速增加。
3. 混合云场景，集群类型复杂，既有远程的边缘融合集群，还有本地自建机房集群和信创集群及公有云集群，集群的 Kubernetes 版本可能还存在差异，如何统一管理这些不同架构的集群问题。

4. 由于企业希望在更多传统工作环境（例如仓库、工厂车间、零售店等）中采用应用现代化改造做法，因此这需要管理更多基础架构组件上的更多工作负载。
5. 当企业担心潜在就地升级问题（即升级自动化故障、应用不稳定或回滚功能）时，他们可以选择将其服务副本部署到新集群中。以这种方式升级需要规划或自动化才能实现，从而确保在升级过程中处理流量管理和状态复制。

## 应用场景

1. 探索一种针对多机房、跨区域的容器调度方式，实现不同区域集群的统一纳管，减少人力物力消耗，缩减成本。
2. 探索集群级别的弹性扩缩，支持创建集群，并打通集群间的边界，探索跨集群的工作负载部署方式。
3. 探索不同集群类型间的兼容性统一管理，形成资源的统一管理平面。

## 17.4 FAQ

- 问：容器管理兼容哪些厂商的集群？

答：第五代容器管理兼容包括 AWS、谷歌云、信创架构、边云融合等任何基于标准 Kubenetes API Server 的发行版容器集群。



- 问：如应对跨集群的网络中断问题？

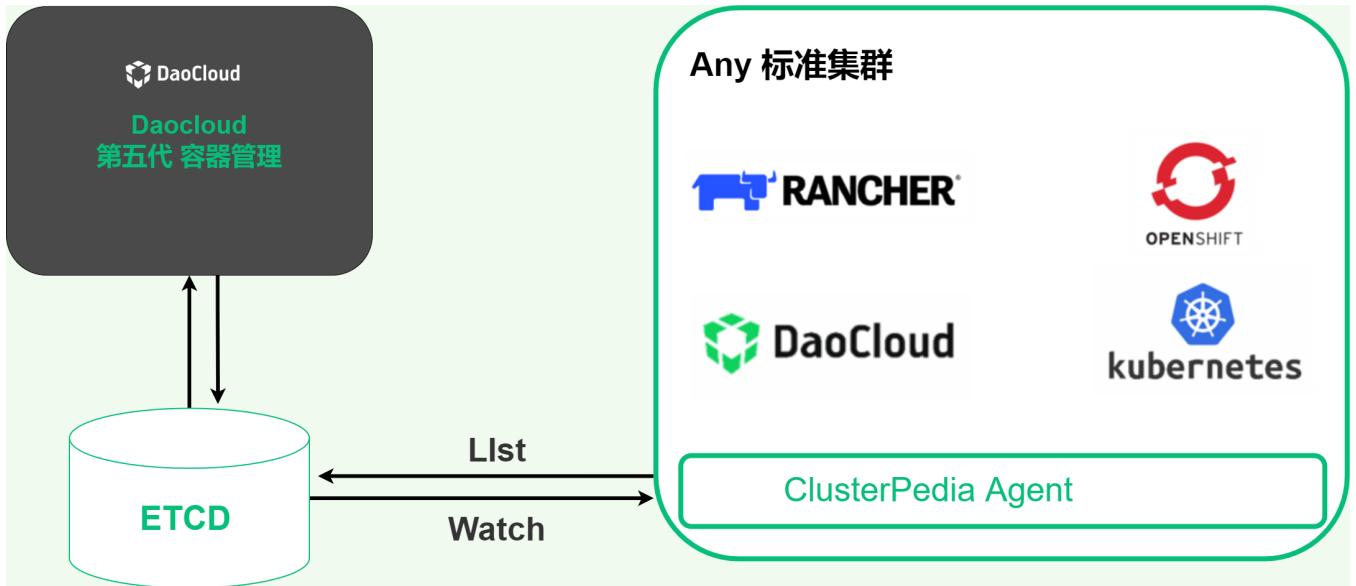
答：为保证业务高可用，同时将业务部署在不同地域的多个云容器平台上，帮助应用实现多地域的流量分发，并且实现跨云应用在同一平台管理，减少运维成本。另外当某个云容器平台发生故障时，通过统一流量分发的机制，自动的将业务流量切换到其他云容器平台上。

- 问：如何实现对其它厂商集群的管理？

答：通过获取其它厂商集群的 KubeConfig 文件，在容器管理模块执行接入集群操作，填入目标集群的 KubeConfig 文件，即完成了对一个容器集群的接入管理。

- 问：如何获取多集群资源的实时状态？

答：借助自研开源组件 ClusterPedia 通过在目标集群上安装 Agent 组件，结合 Watch 机制能够实时将目标集群的资源变动信息同步到容器管理集群的 ETCD 内。



- 问：容器管理在第五代产品中的定位？

答：在第五代产品中，容器管理模块处于承上启下的核心位置。对上：对接 DCE 5.0 产品模块及未来定制化模块。对上层的场景化管理模块而言，容器管理模块能够把任何一个标准的 Kubernetes API 对上暴露，无论是制品中心、Karmada、还是应用商店等，只要需要调用 Kubernetes API，就能够通过容器管理这个模块去调用底层的任何平台的 Kubernetes API。对下：对接 Any Kubernetes，例如边缘的 K3S 信创环境、DCE、DKG、DKE、以及外部的 Openshift、Tanzu、CCE 等等都是基于容器管理这个模块。

- 问：谁能够创建集群？

答：拥有 Global Admin、Kpanda Owner 角色的用户能够去创建集群。

- 问：谁能管理集群？

答：取决于用户所采用的权限控制解决方案，一般来说除了拥有 Global Admin、Kpanda Owner、Cluster Admin、Cluster Admin 角色的用户外，拥有 Cluster Edit 角色的用户及拥有强绑定的 Floder Admin、WorkerSpace Admin 角色的用户也能够实现对集群内资源的增删改查操作，以及对用户进行授权。

[了解容器管理](#)

[下载 DCE 5.0](#) [安装 DCE 5.0](#) [申请社区免费体验](#)

## 18. 介绍 KWOK

作者: Shiming Zhang (DaoCloud), Wei Huang (Apple), Yibo Zhuang (Apple)

译者: Michael Yao (DaoCloud)

KWOK logo

是什么项目会被 Apple、IBM、腾讯、华为纷纷使用？你是否曾想过在几秒钟内搭建一个由数千个节点构成的集群，如何用少量资源模拟真实的节点，如何不耗费太多基础设施就能大规模地测试你的 Kubernetes 控制器？

如果你曾有过这些想法，那你可能会对 KWOK 有兴趣。KWOK 是一个工具包，能让你在几秒钟内创建数千个节点构成的集群。

### 18.1 什么是 KWOK?

KWOK 是 Kubernetes WithOut Kubelet 的缩写，即没有 Kubelet 的 Kubernetes。到目前为止，KWOK 提供了两个工具：

`kwok` : `kwok` 是这个项目的基石，负责模拟伪节点、Pod 和其他 Kubernetes API 资源的生命周期。

`kwokctl` : `kwokctl` 是一个 CLI 工具，设计用于简化创建和管理由 `kwok` 模拟节点组成的集群。

### 18.2 为什么使用 KWOK?

KWOK 具有下面几点优势：

- 速度：你几乎可以实时创建和删除集群及节点，无需等待引导或制备过程。
- 兼容性：KWOK 能够与兼容 Kubernetes API 的所有工具或客户端（例如 `kubectl`、`helm`、`kui`）协同作业。
- 可移植性：KWOK 没有特殊的软硬件要求。一旦安装了 Docker 或 Nerdctl，你就可以使用预先构建的镜像来运行 KWOK。另外，二进制文件包适用于所有平台，安装简单。
- 灵活：你可以配置不同类型的节点、标签、污点、容量、条件等，还可以配置不同的 Pod 行为和状态来测试不同的场景和边缘用例。
- 性能：你在自己的笔记本电脑上就能模拟数千个节点，无需大量消耗 CPU 或内存资源。

### 18.3 使用场景是什么？

KWOK 可用于各种用途：

- 学习：你可以使用 KWOK 学习 Kubernetes 概念和特性，无需顾虑资源浪费或其他后果。
- 开发：你可以使用 KWOK 为 Kubernetes 开发新特性或新工具，无需接入真实的集群，也不需要其他组件。
- 测试：
  - 你可以衡量自己的应用程序或控制器在使用节点和 Pod 时的扩缩表现如何。
  - 你可以用不同的资源请求或限制创建大量 Pod 或服务，在集群上营造高负载的环境。
  - 你可以通过更改节点状况或随机删除节点来模拟节点故障或网络分区。
  - 你可以通过启用不同的特性门控或 API 版本来测试控制器如何与其他组件交互。

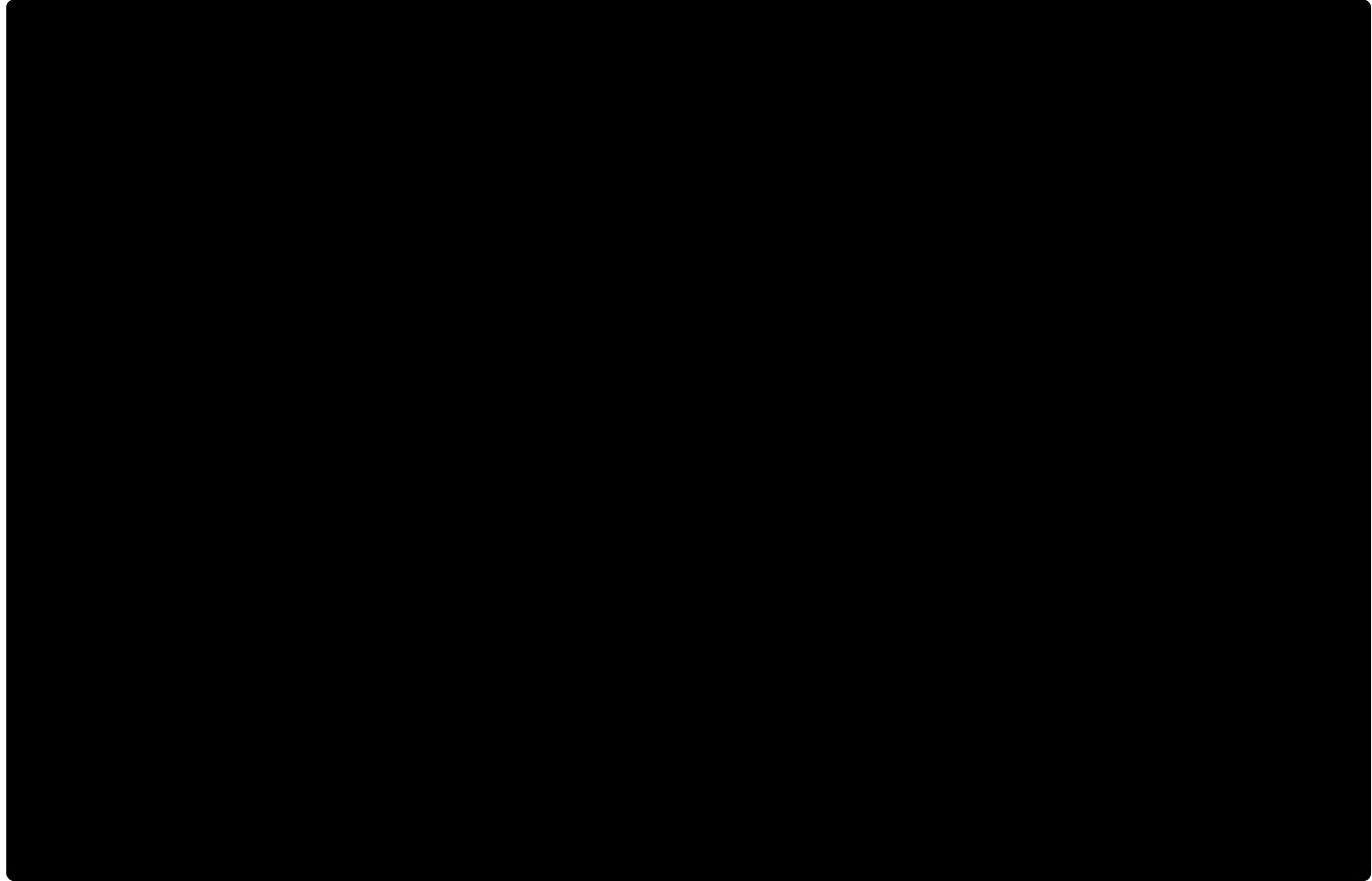
## 18.4 有哪些限制？

KWOK 并非试图完整替代其他什么。当然也有一些限制需要你多加注意：

- 功能性：KWOK 不是 kubelet。KWOK 在 Pod 生命周期管理、卷挂载和设备插件方面所展现的行为与 kubelet 不同。KWOK 的主要功能是模拟节点和 Pod 状态的更新。
- 准确性：需要重点注意 KWOK 还不能确切地反映各种工作负载或环境下真实节点的性能或行为。KWOK 只能使用一些公式来逼近真实的节点行为。
- 安全性：KWOK 没有对模拟的节点实施任何安全策略或安全机制。KWOK 假定来自 kube-apiserver 的所有请求都是经过授权且是有效的。

## 18.5 入门指导

如果你对试用 KWOK 感兴趣，请查阅 [KWOK 文档](#)了解详情。



## 18.6 欢迎参与

如果你想参与讨论 KWOK 的未来或参与开发，可通过以下几种方式参与进来：

- Slack [#kwok](#) 讨论一般用法，Slack [#kwok-dev](#) 讨论开发问题（访问 [slack.k8s.io](https://slack.k8s.io) 获取 KWOK 工作空间的邀请链接）
- 在 [sigs.k8s.io/kwok](https://sigs.k8s.io/kwok) 上提出 Issue/PR/Discussion

我们欢迎所有想要加入这个项目的贡献者，欢迎任何形式的反馈和贡献。

## 18.7 参考链接

- KWOk 代码仓库：<https://github.com/kubernetes-sigs/kwok>
- KWOk 文档：<https://kwok.sigs.k8s.io/>

- KWOK slack 频道: <https://slack.k8s.io/>

# 19. Linux 单机在线体验 DCE 5.0 社区版

本页说明如何通过 Docker 和 kind 在一台 Linux 机器上在线安装单机 DCE 5.0 社区版。

!!! tip

这是一种极简安装方式，便于学习和体验，性能优于 [macOS 单机版](./macos.md)。原文作者是 [panpan0000](https://github.com/panpan0000)。

## 19.1 准备工作

- 准备一台 Linux 机器，资源建议：CPU > 8 核、内存 > 12 GB、磁盘空间 > 100 GB
- 确保这台机器能够连通公网
- 操作系统：CentOS 7 或 Ubuntu 22.04

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

预期输出如下：

```
precheck pass..
```

## 19.2 安装 Docker

如果主机上已有 Docker，并且版本高于 1.18，则可跳过此步骤。

==== "CentOS"

依次执行以下命令，大概需要 5 分钟左右：

```
```bash
set -e
if [ -x "$(command -v docker)" ] ;then
  echo "docker already installed : version = \"$(docker -v)\"";
  exit 0
fi
```
```bash
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
```

```

==== "Ubuntu"

依次执行以下命令，大概需要 5 分钟左右：

```
```bash
set -e
if [ -x "$(command -v docker)" ] ;then
  echo "docker already installed : version = \"$(docker -v)\"";
  exit 0
fi
```
```bash
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
...
```

!!! note

- 如果机器上已有 Podman，但是没有 Docker，仍需要安装 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 对不上问题。
- Docker 安装问题请参阅 [Docker 官方安装说明] (<https://docs.docker.com/desktop/install/linux-install/>)。

## 19.3 kind 集群

### 1. 下载 kind 的二进制文件包。

```
bash
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
```

查看 kind 版本：

```
bash
kind version
```

预期输出如下：

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

### 2. 创建 kind\_cluster.yaml 配置文件。暴露集群内的 32088 端口到 kind 对外的 8888 端口（可自行修改）。

```
bash
cat > kind_cluster.yaml << EOF
apiVersion: kind.x-k8s.io/v1alpha4
kind: Cluster
nodes:
- role: control-plane
  extraPortMappings:
  - containerPort: 32088
    hostPort: 8888
EOF
```

### 3. 通过 kind 创建一个名为 fire-kind-cluster 的 K8s 集群，以 k8s 1.25.3 为例。

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

此步骤大概需要 3~5 分钟，取决于镜像下载的网速。预期输出如下：

```
console
Creating cluster "fire-kind-cluster" ...
✓ Ensuring node image (docker.m.daocloud.io/kindest/node:v1.25.3) [?] [?]
✓ Preparing nodes 📦
✓ Writing configuration 📜
✓ Starting control-plane 🏰
✓ Installing CNI 🐳
✓ Installing StorageClass 🏷
Set kubectl context to "kind-fire-kind-cluster"
```

### 4. 查看新创建的 kind 集群。

```
console
kind get clusters
```

预期输出如下：

```
console  
fire-kind-cluster
```

## 19.4 安装 DCE 5.0 社区版

### 1. 安装依赖项，另请参阅[依赖项安装说明](#)

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

### 2. 在主机下载 dce5-installer 二进制文件（也可以通过浏览器下载）

```
shell  
export VERSION=v0.5.0  
curl -Lo ./dce5-installer https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/dce5-installer-$VERSION  
chmod +x ./dce5-installer
```

### 3. 执行以下命令开始安装。

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

kind 集群仅支持 NodePort 模式。

### 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（默认为 `https://${主机 IP}:8888`），使用 默认的账户和密码（admin/changeme）探索全新的 DCE 5.0 啦！

安装成功

## 20. 通过 Docker 和 kind 在 macOS 电脑上安装社区版

本页说明如何使用 macOS 笔记本电脑创建单节点的 kind 集群，然后在线安装 DCE 5.0 社区版。

!!! tip

这是针对初学者的简化安装体验步骤，实际生产很少会使用 macOS，  
原文作者是 [panpan0000] (<https://github.com/panpan0000>)。

### 20.1 硬件环境

确认 MacBook 的性能和资源是否满足需求。最低配置为：

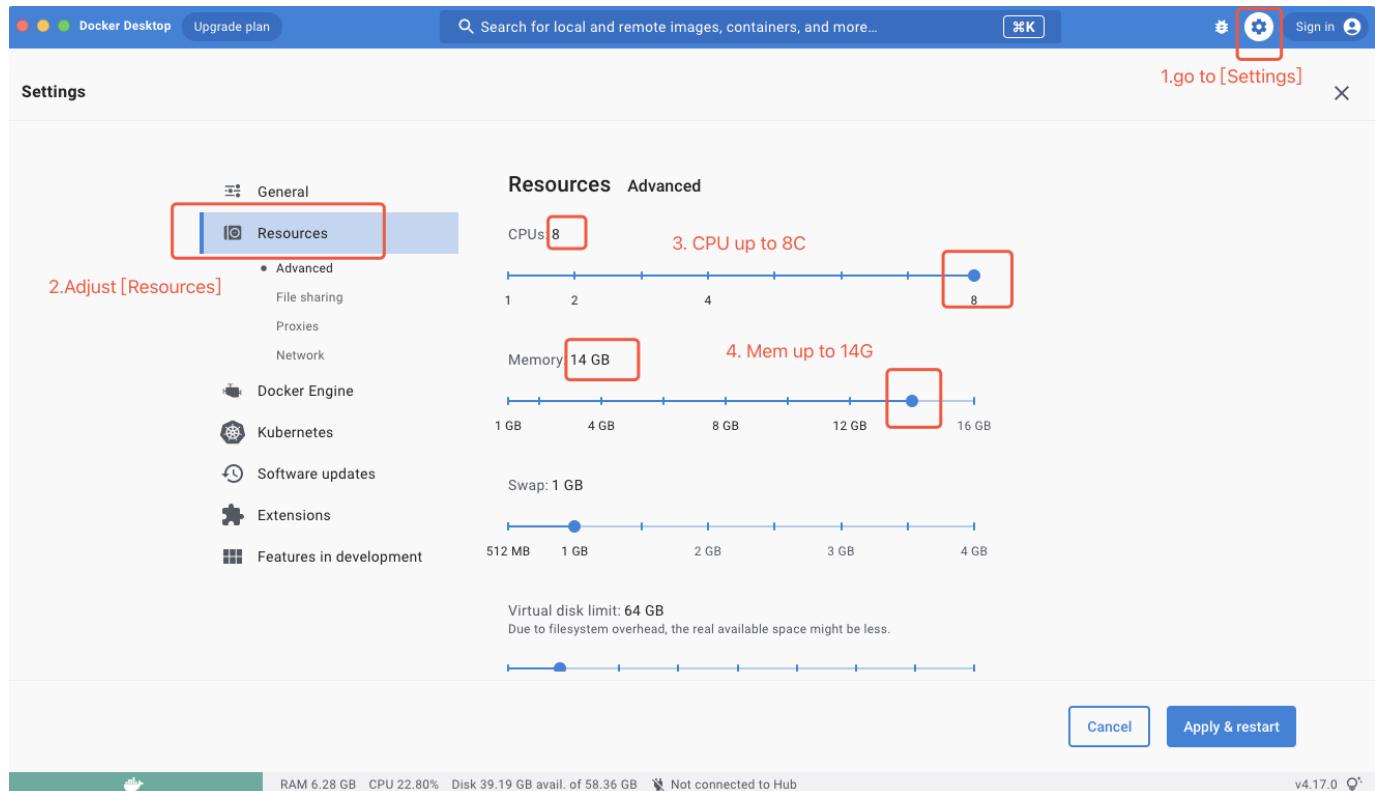
- CPU：8 核
- 内存：16G
- 磁盘剩余空间：大于 20G

### 20.2 安装和调整 Docker

根据 MacBook 的芯片（Intel 或 M1），安装 Docker Desktop。

调整容器资源上限：

1. 启动 Docker。
2. 点击右上角的 ，以打开 Settings 页面。
3. 点击左侧的 Resources，将启动容器的资源上限调节到 8C14G，点击 Apply & Restart 按钮。



## 20.3 安装 kind

按照实际电脑情况，以下任选其一，安装 kind。如果遇到其他问题，请参阅 [kind 官方安装说明](#)。

==== “Mac 是 Intel 芯片”

```
```shell
[ $(uname -m) = x86_64 ] && curl -Lo ./kind https://kind.sigs.k8s.io/dl/v0.17.0/kind-darwin-amd64
```

```

==== “Mac 是 M1/ARM 芯片”

```
```shell
[ $(uname -m) = arm64 ] && curl -Lo ./kind https://kind.sigs.k8s.io/dl/v0.17.0/kind-darwin-arm64
chmod +x ./kind
sudo mv kind /usr/local/bin/kind
```

```

==== “通过 Homebrew 安装 kind”

```
安装 Homebrew:
```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

安装 kind
```shell
brew install kind
```

```

最后执行以下命令，确认 kind 安装成功：

```
kind version
```

## 20.4 创建 kind 配置文件

暴露集群内的 32088 端口到 kind 对外的 8888 端口（可自行修改）：

```
cat > kind_cluster.yaml << EOF
apiVersion: kind.x-k8s.io/v1alpha4
kind: Cluster
nodes:
- role: control-plane
  extraPortMappings:
  - containerPort: 32088
    hostPort: 8888
EOF
```

## 20.5 kind 创建 K8s 集群

以 K8s 1.25.3 版本为例，执行以下命令创建一个 K8s 集群：

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

确认 kind 集群创建是否成功：

```
docker exec -it fire-kind-cluster-control-plane kubectl get no
```

期望输出：

| NAME                            | STATUS | ROLES         | AGE | VERSION |
|---------------------------------|--------|---------------|-----|---------|
| fire-kind-cluster-control-plane | Ready  | control-plane | 18h | v1.25.3 |

## 20.6 安装 DCE 5.0 社区版

### 1. 安装依赖项

```
shell
cat <<EOF | docker exec -i fire-kind-cluster-control-plane bash
curl -LO https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/install_prerequisite.sh
bash install_prerequisite.sh online community
apt-get update && apt-get install -y wget
EOF
```

### 2. 下载 dce5-installer 二进制文件

```
shell
docker exec -it fire-kind-cluster-control-plane /bin/bash

假定 VERSION=v0.5.0

shell
export VERSION=v0.5.0;
curl -Lo ./dce5-installer https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/dce5-installer-$VERSION
chmod +x ./dce5-installer
exit
```

### 3. 安装 DCE5 社区版

#### a. 先获取本机 IP

```
shell
myIP=$(ip r get 1.1.1.1| awk '{print $NF}')
如果报错 zsh: command not found: ip, 有 2 个方案:
• 执行 myIP=$(ifconfig en0| grep "inet[ ]" | awk '{print $2}')
• 或通过 brew install iproute2mac 这类命令安装 iproute2mac 后重试。
```

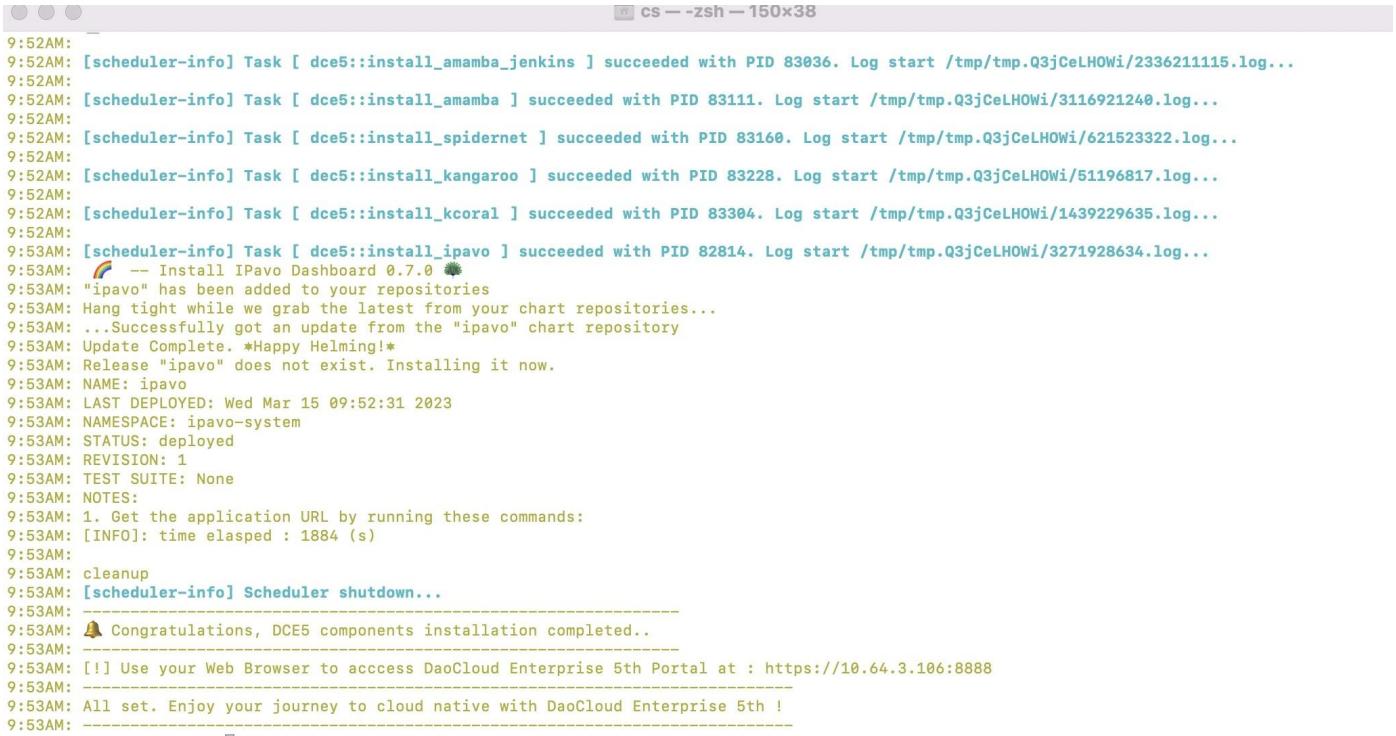
#### b. 开始安装, 大概用时 30 分钟, 取决于镜像拉取的网速

```
shell
docker exec -it fire-kind-cluster-control-plane bash -c "./dce5-installer install-app -z -k $myIP:8888"
```

### 4. 在安装过程中可以另起一个终端窗口, 执行如下命令, 观察 Pod 启动情况。

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

当看到以下提示, 则表示 DCE 5.0 社区版安装成功了。



```

9:52AM: [scheduler-info] Task [ dce5::install_amamba_jenkins ] succeeded with PID 83036. Log start /tmp/tmp.Q3jCeLHOWi/2336211115.log...
9:52AM: [scheduler-info] Task [ dce5::install_amamba ] succeeded with PID 83111. Log start /tmp/tmp.Q3jCeLHOWi/3116921240.log...
9:52AM: [scheduler-info] Task [ dce5::install_spidernet ] succeeded with PID 83160. Log start /tmp/tmp.Q3jCeLHOWi/621523322.log...
9:52AM: [scheduler-info] Task [ dce5::install_kangaroo ] succeeded with PID 83228. Log start /tmp/tmp.Q3jCeLHOWi/51196817.log...
9:52AM: [scheduler-info] Task [ dce5::install_kcoral ] succeeded with PID 83304. Log start /tmp/tmp.Q3jCeLHOWi/1439229635.log...
9:52AM: [scheduler-info] Task [ dce5::install_ipavo ] succeeded with PID 82814. Log start /tmp/tmp.Q3jCeLHOWi/3271928634.log...
9:53AM: 🎉 -- Install IPavo Dashboard 0.7.0 🎉
9:53AM: "ipavo" has been added to your repositories
9:53AM: Hang tight while we grab the latest from your chart repositories...
9:53AM: ...Successfully got an update from the "ipavo" chart repository
9:53AM: Update Complete. *Happy Helm-ing!*
9:53AM: Release "ipavo" does not exist. Installing it now.
9:53AM: NAME: ipavo
9:53AM: LAST DEPLOYED: Wed Mar 15 09:52:31 2023
9:53AM: NAMESPACE: ipavo-system
9:53AM: STATUS: deployed
9:53AM: REVISION: 1
9:53AM: TEST SUITE: None
9:53AM: NOTES:
9:53AM: 1. Get the application URL by running these commands:
9:53AM: [INFO]: time elasped : 1884 (s)
9:53AM:
9:53AM: cleanup
9:53AM: [scheduler-info] Scheduler shutdown...
9:53AM: -----
9:53AM: 🎉 Congratulations, DCE5 components installation completed..
9:53AM: -----
9:53AM: [!] Use your Web Browser to access DaoCloud Enterprise 5th Portal at : https://10.64.3.106:8888
9:53AM: -----
9:53AM: All set. Enjoy your journey to cloud native with DaoCloud Enterprise 5th !
9:53AM: -----

```

- 输入默认用户和密码 (admin/changeme) 登录后，系统将提示申请许可密钥。

## 20.7 体验使用

申请到许可证后，将进入 DCE 5.0 主界面，显示当前安装的组件、集群/节点/资源、告警等信息。

您可以尝试：

- 创建一个[用户](#)，加入一个[用户组](#)，赋予[角色权限](#)
- 定制软件界面
- 接入一个集群
- 管理你的节点
- 创建一个负载
- 更多请查阅[文档站页面](#)

## 20.8 卸载

- 卸载 DCE 5.0 社区版。

- 删除 kind 集群。

```
kind delete cluster --name=fire-kind-cluster
```

- 卸载 kind 本身。

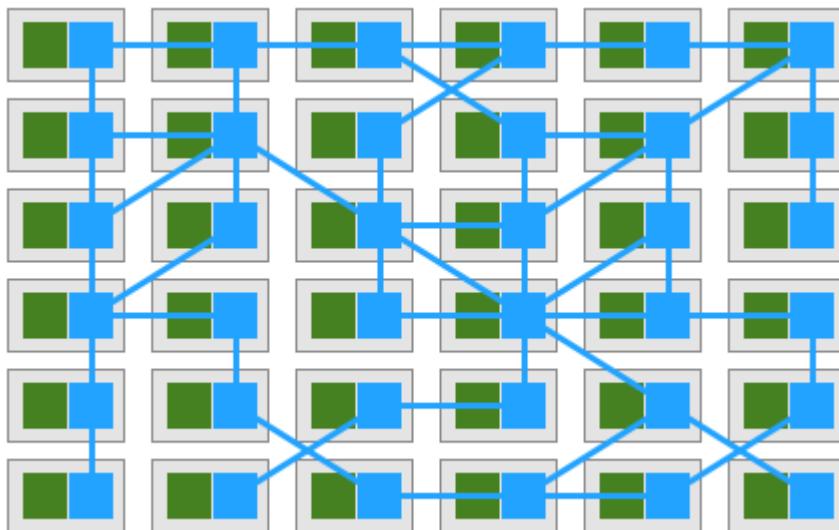
```
rm -f $(which kind)
```

- 在应用列表中卸载 Docker。

至此您的 MacBook 恢复到了最初状态 😊

## 21. 5.0 服务网格能力介绍

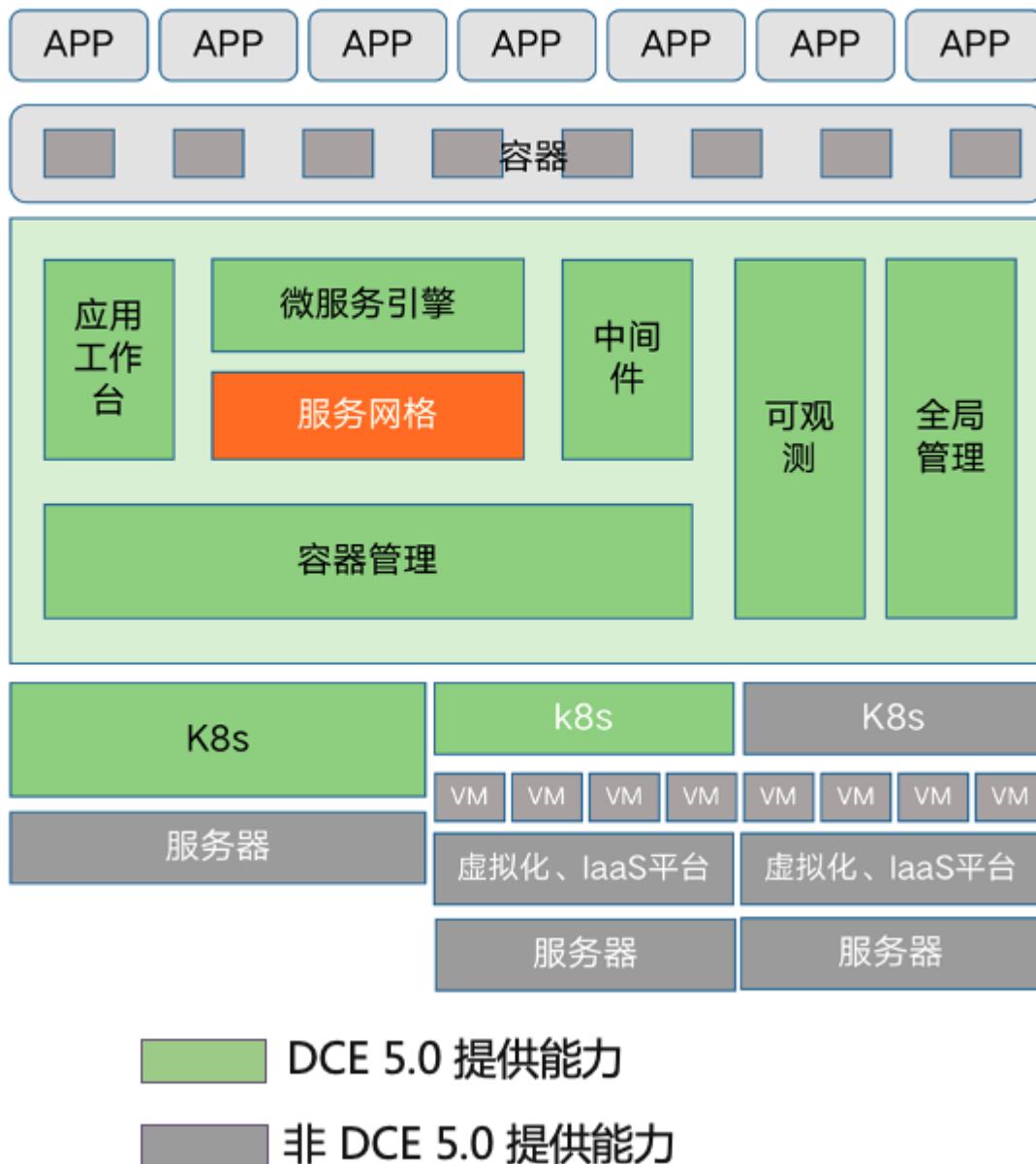
下图是一个网格状服务的正常模型，如何管理这样复杂的网格服务？



DaoCloud 服务网格是一种具备高性能、高易用性的全托管服务网格产品，通过提供完整的非侵入式的微服务治理解决方案，能够很好的支持多云多集群复杂环境的统一治理，以基础设施的方式为用户提供服务流量治理、安全性治理、服务流量监控、以及对传统微服务（SpringCloud、Dubbo）接入支持。并兼容社区原生 Istio 开源服务网格，提供原生 Istio 接入管理能力。较高的层次上，服务网格有助于降低服务治理的复杂性，并减轻开发运维团队的压力。

道客服务网格作为 DCE 5.0 产品的体系一员，无缝对接容器管理平台，可以为用户提供开箱即用的上手体验，并作为基础设施为微服务引擎提供容器微服务治理支持，方便用户通过单一平台对各类微服务系统做统一管理。

# 在DCE 5.0中的产品定位



## 21.1 流量治理

流量治理是一个非常宽泛的话题，例如：

- 动态修改服务间访问的负载均衡策略，如根据某个请求特征做会话保持。
- 同一个服务有两个版本在线，将一部分流量切到某个版本上。
- 对服务进行保护，例如限制并发连接数、限制请求数、隔离故障服务实例等。
- 动态修改服务中的内容，或者模拟一个服务运行故障等。

在服务网格中实现这些服务治理功能时无须修改任何应用的代码。服务网格可以为管理的服务提供非侵入的流量治理能力。根据服务的协议，提供策略化、场景化的网络连接管理。在应用拓扑上对选定服务的选定端口，根据需要配置各种不同的治理规则。

## 场景优势

---

### 重试

服务访问失败自动重试，从而提高总体访问成功率和质量。支持配置 HTTP 请求重试次数、重试超时时间和重试条件。

### 超时

服务访问超时自动处理，快速失败，从而避免资源锁定和请求卡顿。支持配置 HTTP 请求超时时间。

### 连接池

通过连接池管理，可以对四层协议配置 TCP 的最大连接数、连接超时时间、最大无响应次数、最短空闲时间和健康检查间隔，对七层协议配置 HTTP 最大请求数、最大重试次数、最大等待请求数、每连接最大请求数、连接最大空闲时间，从而防止一个服务的失败级联影响到整个应用。

### 熔断

通过熔断配置实例被驱逐前的连续错误次数、驱逐间隔时长、最小驱逐时间、最大驱逐比例等参数，从而定期考察被访问的服务实例的工作情况，如果连续出现访问异常，则将服务实例标记为异常并进行隔离，在一段时间内不为其分配流量。过一段时间后，被隔离的服务实例会再次被解除隔离，尝试处理请求。如果还不正常，则被隔离更长的时间。从而实现异常服务实例的故障隔离和自动故障恢复。

### 负载均衡

配置各种负载均衡策略，如随机、轮询、最少连接，还可以配置一致性哈希将流量转发到特定的服务实例上。

### HTTP 头域

灵活增加、修改和删除指定 HTTP 头域，包括将 HTTP 请求转发到目标服务之前对 Headers 的操作，以及将 HTTP 响应回复给客户端前，对 Headers 的操作，以非侵入方式管理请求内容。

### 故障注入

通过对选定的服务注入中断故障、延时故障来构造故障场景，无需修改代码即可进行故障测试。

## 21.2 端到端的透明安全

---

众所周知，将传统的单体应用拆分为一个个微服务固然带来了各种好处，包括更好的灵活性、可扩缩性、重用性，但微服务也同样面临着特殊的安全需求：

- 为了抵御中间人攻击，需要用到流量加密。
- 为了提供灵活的服务访问控制，需要用到 TLS 和细粒度访问策略。
- 为了决定哪些人在哪些时间可以做哪些事，需要用到审计工具。

面对这些需求服务网格提供全面的安全解决方案，包括身份验证策略、透明的 TLS 加密以及授权和审计工具。

## 场景优势

---

### 非侵入安全

服务网格是以一种安全基础设施的方式向用户提供透明的安全能力，让不涉及安全问题的代码安全运行，让不太懂安全的人可以开发和运维安全的服务，不用修改业务代码就能提供服务访问安全。应用服务网格提供了一个透明的分布式安全层，并提供了底层安全的通信通道，管理服务通信的认证、授权和加密，提供 Pod 到 Pod、服务到服务的通信安全。开发人员在这个安全基础设施层上只需专注于应用程序级别的安全性。

### 多集群安全

在多集群场景下服务网格提供了全局的服务访问安全。多个集群的网格共享一套根证书，给数据面的服务实例分发密钥和证书对，并定期替换密钥证书，根据需要撤销密钥证书。在服务间访问时，网格的数据面代理就会代理本地服务和对端进行双向认证、通道加密。这里的双向认证的服务双方可以来自两个不同的集群，从而做到跨集群的透明的端到端双向认证。

### 细粒度授权

在认证的基础上，就可以进行服务间的访问授权管理，可以控制某个服务，或者服务的一个特定接口进行授权管理。如只开放给特定的一个 Namespace 下的服务，或者开放给某个特定的服务。源服务和目标服务可以在不同的集群，甚至源服务的不同实例在不同的集群，目标服务的不同实例在不同的集群。

## 21.3 服务运行监控

运营容器化的基础设施带来了一系列新的挑战。我们需要增强容器、评估 API 端点的性能以及识别出基础设施中的有害部分。服务网格可在不修改代码的情况下实现 API 增强，并且不会带来服务延迟。

服务网格为网格内的所有服务通信生成详细的遥测，这种遥测技术提供了服务行为的可观察性，允许运营商对其应用程序进行故障排除、维护和优化，而不会给服务开发人员带来任何额外负担。通过服务网格，运营商可以全面了解被监控的服务如何与其他服务以及组件本身进行交互。

### 场景优势

非侵入监控数据采集

在复杂应用的场景下，服务间的访问拓扑、调用链、监控等都是对服务整体运行状况进行管理，以及服务访问异常时进行定位定界的必要手段。服务网格技术的一项重要能力就是以应用非侵入的方式提供这些监控数据的采集，用户只需关注自己的业务开发，无需额外关注监控数据的生成。

丰富性能监控能力

基于网格生成服务访问数据，集成各种不同的性能监控服务，提供跨集群智能的服务运行管理。包括跨集群的服务调用链、服务访问拓扑和服务运行健康状态。通过跨集群的全局视图来关联服务间的访问状况。

## 21.4 图形化用户交互

在服务网格中，为用户实现以下图形化的交互性功能：

1. 对服务网格下的网格资源创建、管理、设置能力，用户可以根据需要把集群资源分配至合理的网格中，实现聚合管理；
2. 为用户接入集群资源，或把用户已部署ISTIO治理组件的集群资源置于服务网格平台管控下；
3. 提供对网格下管控资源的工作空间绑定功能，通过工作空间实现网格资源的权限分享，方便各子产品间互动；
4. 以聚合管理方式，为用户提供命名空间、服务的统一查看、操作功能；
5. 针对命名空间、服务的流量治理功能，并提供表单和YAML两种创建、编辑形式，用户几乎可以使用原生ISTIO提供的所有流量治理功能（路由、分流、熔断等）；
6. 针对命名空间、服务的安全治理功能，并提供表单和YAML两种创建、编辑形式，用户可以配合流量治理实现网格内零信任环境和访问授权功能；
7. 对网格内服务通信的流量拓扑查看功能和状态图表查看功能；流量拓扑会为用户提供各类网格内节点间通信状态、使用协议、性能参数等各项信息展示，用户可以通过搜索筛选功能迅速定位问题；状态图表可以从全局、服务、工作负载等多个纬度为用户提供系统内各状态信息；

[了解服务网格](#)

[下载 DCE 5.0](#) [安装 DCE 5.0](#) [申请社区免费体验](#)

## 22. 2022 年 DCE 5.0 攻坚语录集

Inspire People, Enable People! 致程序猿里超有文采的 [panpan0000](#) 同学。

\*[panpan0000]: 技术圈的诗人，被代码耽误的文学家

!!! tip

这是一个没有硝烟的时间战场，是一场开疆拓土的光荣远征，是一次振奋人心的技术攀登，是一回一往无前的破釜沉舟 :material-sail-boat:

=扬起旗帜，吹响号角，夺下那个山头！For the DaoCloud 5^th!=

- :leaves:{ .lg .middle } \_3 月春风里\_ --- 在上海肆虐的疫情下，

每个人都在用自己的方式守护，

感谢每一位坚强隐忍、静待花开的你们。

所有的冬天都必将过去，

没有一个春天不会回来。 - :material-virus:{ .lg .middle } \_4 月封控中\_ --- 这个 4 月，就着泡面 🍜，穿着睡衣 🛌，头势飘逸 😱 敲着代码 🖥，上班开会 📺，下班抢菜 🥬

做过的抗原比 bug 还多，git 的记录比你的乱发还长。

即使疫情阴霾，即使周边喧嚣，你还能保持阳光 🌟

坚持，热爱 105°C 的你。

- :desktop:{ .lg .middle } \_5 月居家办公\_ --- 5 月万箭齐发，第五代开发已渐入佳境。

通过细细的网线，四地连成一心，这归功于：

```yaml title="封城中的坚守" 你的坚守：这魔幻的上海两月，终会载入这历史性产品的历史性一页 你的风采：一张张活起来的页面，联而贯之，梦想逐渐开始兑现 你的努力：新的技术跃然纸上，新的能力呼之欲出，振奋人心 ``` - :hugging:{ .lg .middle } \_6 月解封了\_ --- 那些波澜不惊的日复一日，会在某天让你看到坚持的意义。当打开 demo-dev 站点发出赞叹“真漂亮啊，秒杀友商”的时候；

当“关于”的界面中，五彩斑斓地闪耀出你的名字的时候；

当你用一个个实打实的功能，回应几个月期待的时候；

当不经意种下自主开源种子，某天突然发现花香满园的时候。我的伙伴：你所热爱的东西 / 真得有一天 / 会反过来拥抱你！

- :love\_letter:{ .lg .middle } \_7 月阳光下星辰里\_ --- 念念不忘，必有回响，功不唐捐，玉汝于成。世界上大部分事，都没有太大意义；真理和热爱除外。![动态星空] ./images/stars.gif) - :material-run-fast:{ .lg .middle } \_8 月炎夏之际\_ --- “人定胜天”一直是一个违心的误读；

“人定”实指志向坚定、战胜自己。

所以哪有什么天生如此，只有我们的天天坚持。8 个月过去了，夺旗团队迎来更多的伙伴，

老同志们引领着前路，新同学们奋力跟上。

每个团队都有着厚积薄发的高光，也经历过韬光养晦的低谷。

这 8 个月，时间回答了成长，成长回答梦想，而梦想决定了你现在的模样：```yaml title="程序猿的日常" 你在半夜两点才合上的电脑 你在文档里字字谨慎的推敲 你在床上还辗转萦绕的思考 你在台风里仍惦记的 git 提交 一切热情和荣耀 在 9 月即将而来的实践面前 将赢得检验和回报 走吧！去发光 / 而不是被照亮！```

- :stars:{ .lg .middle } \_9 月核酸之光\_ --- 夜深，不服输的你，还在痛苦的抗争，

犹如宇宙中漂浮的无力。在技术的巨壑中前行，宛如广袤宇宙之中的渺小和孤独；

你只拥有模糊的漆黑世界和清晰的自己。但只有如此黑暗之时，你才能发现星辰的灌灌生辉；

倘若不曾见过黑夜，那日出也不会如此明媚动人。何去何从，信念使然... ![宇宙之光] ./images/light.png) - :material-flower-tulip:{ .lg .middle } \_10 月秋风起\_ --- 与其抱怨身处黑暗，不如提灯前行；

因为人生最重要的，不是你现在所处的位置，而是你面朝的方向。董宇辉总说：“苦难，能给予我们坚持的力量”，

其实，力量的源泉还有对明天的期待。秋风已深，凛冬将至。

但是我知道：你们心里 / 有一个 / 不可战胜的夏天。![那朵花] ./images/flower.png)

- :sunny:{ .lg .middle } \_11 月我还行\_ --- > 海明威说：>> “生活总是让我们遍体鳞伤，但到后来，那些受伤的地方一定会变成我们最强壮的地方。”在这个结痂上，愿你长出羽翼，去拥抱星河万里。也许你未必 / 一直光芒万丈 / 但始终 / 温暖有光 / ! [move forward] ./images/runner.png) - :fontawesome-solid-hourglass-end:{ .lg .middle } \_12 月放开管控\_ --- 2022 年画上了句号。

这个句号，带着病毒带来的精神肉体的双重挑战，

也带着对大疫三年后百废待兴、否极泰来的期许。5.0 夺旗转眼又是一年了，你还记得一年前的自己么？

淞沪路边被你日夜来回踏过的砖，

新江湾城里被你从绿看到黄再到秃的树，  
 多少次发誓明天再也不吃老乡鸡，  
 园区前将永远消失的核酸亭... 打开电脑，面对紫色悦动的登录界面，你也会眼眶湿润；  
 是为了末世电影一般的 2022，  
 也是为了这一年日夜熬过的 12 个里程碑。 面对不完美的世界，总有人选择挺身而出。  
 去奔赴，去热爱，去见证。  
 是你，带着跨越瘟疫与巨变的勇气，  
 像贺炜说的“请相信，世界上总有一些美好，值得我们全力以赴。”  
 艰难困苦，玉汝于成！

!!! quote

生活总是让我们遍体鳞伤，但到后来，那些受伤的地方一定会变成我们最强壮的地方 🔥



[下载 DCE 5.0](#) { .md-button .md-button--primary } 安装 DCE 5.0 { .md-button .md-button--primary } 申请社区免费体验 { .md-button .md-button--primary }

## 23. DCE 5.0 涉及多少开源项目

时常听闻客户、社区成员、贡献者、公司内部的售前、交付、项目组在询问“DCE 到底涉及了哪些开源项目？”

为此，我们做了一个简单的汇总。下文列出了截止到 2023-02-14，DCE 5.0 社区版所涉及的开源项目。

!!! info

集成和调用的所有相关项目将跟随社区版本持续演进，动态更新。

### 23.1 容器管理

涉及的开源项目有直接集成和间接调用两种。

## 直接集成

容器管理模块直接集成的项目包括但不限于：

1. DATA-DOG/go-sqlmock v1.5.0
2. Miniredis v2.22.0
3. cloudtpty v0.0.3-0.20220525041218-c0b20763ee9f
4. clusterpedia-io/api v0.0.0-20220802044336-d3ea49998d11
5. clusterpedia-io/client-go v0.0.0-20220721030555-ba14329a2df5
6. clusterpedia-io/fake-apiserver v0.0.0-20220726040952-67538a31d6c2
7. Docker v2.8.1+incompatible
8. fsnotify v1.5.4
9. yaml v1.0.0
10. go-openapi/runtime v0.24.1
11. go-openapi/strfmt v0.21.3
12. go-redis/redis v8.11.5
13. golang/mock v1.6.0
14. google/go-cmp v0.5.8
15. google/uuid v1.3.0
16. gorilla/mux v1.8.0
17. grpc-ecosystem/go-grpc-middleware v1.3.0
18. grpc-ecosystem/grpc-gateway v2.10.3
19. heroku/docker-registry-client v0.0.0-20211012143308-9463674c8930
20. kubernetes-csi/external-snapshotter v4.2.0
21. moby v20.10.17+incompatible
22. onsi/ginkgo v2.1.4
23. onsi/gomega v1.20.1
24. patrickmn/go-cache v2.1.0+incompatible
25. prometheus/client\_golang v1.13.0
26. russross/blackfriday v2.1.0
27. shirou/gopsutil v3.22.7
28. soheilhy/cmux v0.1.5
29. spf13/cobra v1.5.0
30. spf13/pflag v1.0.5
31. spf13/viper v1.10.1
32. stretchr/testify v1.8.0
33. golang.org/x/sync v0.0.0-20220601150217-0de741cfad7f
34. golang.org/x/term v0.0.0-20210927222741-03fcf44c2211
35. golang.org/x/text v0.3.7
36. golang.org/x/tools v0.1.10
37. google.golang.org/grpc v1.46.2
38. google.golang.org/protobuf v1.28.1
39. gopkg.in/yaml.v2 v2.4.0
40. gopkg.in/yaml.v3 v3.0.1
41. gorm.io/driver/mysql v1.3.2
42. gorm.io/driver/postgres v1.3.1
43. gorm.io/gorm v1.23.2

44. [helm.sh/helm/v3](#) v3.8.0
45. [k8s.io/api](#) v0.24.3
46. [k8s.io/apiextensions-apiserver](#) v0.24.0
47. [k8s.io/apimachinery](#) v0.24.3
48. [k8s.io/apiserver](#) v0.24.0
49. [k8s.io/client-go](#) v0.24.3
50. [k8s.io/code-generator](#) v0.24.1
51. [k8s.io/component-base](#) v0.24.0
52. [k8s.io/component-helpers](#) v0.24.0
53. [k8s.io/klog](#) v2.60.1
54. [k8s.io/kubectl](#) v0.24.0
55. [k8s.io/kubernetes](#) v1.24.0
56. [k8s.io/metrics](#) v0.24.0
57. [k8s.io/utils](#) v0.0.0-20220210201930-3a6ce19ff2f9
58. [kubean.io/api](#)
59. [sigs.k8s.io/cli-utils](#) v0.32.0
60. [sigs.k8s.io/cluster-api](#) v1.1.3
61. [sigs.k8s.io/controller-runtime](#) v0.12.1
62. [sigs.k8s.io/yaml](#) v1.3.0

## 间接调用

容器管理模块间接调用的项目包括但不限于：

1. github.com/Azure/go-ansiterm v0.0.0-20210617225240-d185dfc1b5a1
2. github.com/Masterminds/semver/v3 v3.1.1
3. github.com/NYTimes/gziphandler v1.1.1
4. github.com/PuerkitoBio/purell v1.1.1
5. github.com/PuerkitoBio/urlesc v0.0.0-20170810143723-de5bf2ad4578
6. github.com/alicebob/gopher-json v0.0.0-20200520072559-a9ecdc9d1d3a
7. github.com/asaskevich/govalidator v0.0.0-20210307081110-f21760c49a8d
8. github.com/beorn7/perks v1.0.1
9. github.com/blang/semver v3.5.1+incompatible
10. github.com/blang/semver/v4 v4.0.0
11. github.com/cespare/xxhash/v2 v2.1.2
12. github.com/containerd/containerd v1.5.13
13. github.com/coreos/go-semver v0.3.0
14. github.com/coreos/go-systemd/v22 v22.3.2
15. github.com/davecgh/go-spew v1.1.1
16. github.com/dgryski/go-rendezvous v0.0.0-20200823014737-9f7001d12a5f
17. github.com/docker/cli v20.10.11+incompatible
18. github.com/docker/docker v20.10.12+incompatible
19. github.com/docker/docker-credential-helpers v0.6.4
20. github.com/docker/go-connections v0.4.0
21. github.com/docker/go-metrics v0.0.1
22. github.com/docker/go-units v0.4.0
23. github.com/docker/libtrust v0.0.0-20160708172513-aabc10ec26b7
24. github.com/emicklei/go-restful v2.9.5+incompatible
25. github.com/envoyproxy/protoc-gen-validate v0.6.2
26. github.com/evanhph/json-patch v4.12.0+incompatible
27. github.com/felixge/httpsnoop v1.0.1
28. github.com/go-errors/errors v1.0.1
29. github.com/go-logr/logr v1.2.0
30. github.com/go-ole/go-ole v1.2.6
31. github.com/go-openapi/analysis v0.21.2
32. github.com/go-openapi/errors v0.20.2
33. github.com/go-openapi/jsonpointer v0.19.5
34. github.com/go-openapi/jsonreference v0.19.6
35. github.com/go-openapi/loads v0.21.1
36. github.com/go-openapi/spec v0.20.4
37. github.com/go-openapi/swag v0.21.1
38. github.com/go-openapi/validate v0.22.0
39. github.com/go-sql-driver/mysql v1.6.0
40. github.com/gobuffalo/flect v0.2.4
41. github.com/gogo/protobuf v1.3.2
42. github.com/golang/groupcache v0.0.0-20210331224755-41bb18bfe9da
43. github.com/golang/protobuf v1.5.2

44. github.com/google/btree v1.0.1  
45. github.com/google/gnostic v0.5.7-v3refs  
46. github.com/google/gofuzz v1.2.0  
47. github.com/google/shlex v0.0.0-20191202100458-e7afc7fb510  
48. github.com/gregjones/httpcache v0.0.0-20180305231024-9cad4c3443a7  
49. github.com/grpc-ecosystem/go-grpc-prometheus v1.2.0  
50. github.com/grpc-ecosystem/grpc-gateway v1.16.0  
51. github.com/hashicorp/hcl v1.0.0  
52. github.com/imdario/mergo v0.3.12  
53. github.com/inconshreveable/mousetrap v1.0.0  
54. github.com/jackc/chunkreader/v2 v2.0.1  
55. github.com/jackc/pgconn v1.11.0  
56. github.com/jackc/pgio v1.0.0  
57. github.com/jackc/pgpassfile v1.0.0  
58. github.com/jackc/pgproto3/v2 v2.2.0  
59. github.com/jackc/pgservicefile v0.0.0-20200714003250-2b9c44734f2b  
60. github.com/jackc/pgtype v1.10.0  
61. github.com/jackc/pgx/v4 v4.15.0  
62. github.com/jhump/protoreflect v1.12.0  
63. github.com/jinzhu/inflection v1.0.0  
64. github.com/jinzhu/now v1.1.4  
65. github.com/josharian/intern v1.0.0  
66. github.com/json-iterator/go v1.1.12  
67. github.com/klauspost/compress v1.13.6  
68. github.com/liggitt/tabwriter v0.0.0-20181228230101-89fcab3d43de  
69. github.com/lufia/plan9stats v0.0.0-20211012122336-39d0f177cc0  
70. github.com/magiconair/properties v1.8.5  
71. github.com/mailru/easyjson v0.7.7  
72. github.com/matttproud/golang\_protobuf\_extensions v1.0.2-0.20181231171920-c182affec369  
73. github.com/mitchellh/mapstructure v1.4.3  
74. github.com/moby/locker v1.0.1  
75. github.com/moby/term v0.0.0-20210619224110-3f7ff695adc6  
76. github.com/modern-go/concurrent v0.0.0-20180306012644-bacd9c7ef1dd  
77. github.com/modern-go/reflect2 v1.0.2  
78. github.com/monochromegane/go-gitignore v0.0.0-20200626010858-205db1a8cc00  
79. github.com/morikuni/aec v1.0.0  
80. github.com/munnerz/goautoneg v0.0.0-20191010083416-a7dc8b61c822  
81. github.com/mxk/go-flowrate v0.0.0-20140419014527-cca7078d478f  
82. github.com/oklog/ulid v1.3.1  
83. github.com/opencontainers/go-digest v1.0.0  
84. github.com/opencontainers/image-spec v1.0.2  
85. github.com/opentracing/opentracing-go v1.2.0  
86. github.com/pelletier/go-toml v1.9.4

87. github.com/peterbourgon/diskv v2.0.1+incompatible  
88. github.com/pkg/errors v0.9.1  
89. github.com/pmezard/go-difflib v1.0.0  
90. github.com/power-devops/perfstat v0.0.0-20210106213030-5aafc221ea8c  
91. github.com/prometheus/client\_model v0.2.0  
92. github.com/prometheus/common v0.37.0  
93. github.com/prometheus/procfs v0.8.0  
94. github.com/sirupsen/logrus v1.8.1  
95. github.com/spf13/afero v1.6.0  
96. github.com/spf13/cast v1.4.1  
97. github.com/spf13/jwalterweatherman v1.1.0  
98. github.com/subosito/gotenv v1.2.0  
99. github.com/tklauser/go-sysconf v0.3.10  
100. github.com/tklauser/numcpus v0.4.0  
101. github.com/xlab/treeprint v1.1.0  
102. github.com/yuin/gopher-lua v0.0.0-20220504180219-658193537a64  
103. github.com/yusufpapurcu/wmi v1.2.2  
104. go.etcd.io/etcd/api/v3 v3.5.1  
105. go.etcd.io/etcd/client/pkg/v3 v3.5.1  
106. go.etcd.io/etcd/client/v3 v3.5.1  
107. go.mongodb.org/mongo-driver v1.10.0  
108. go.opentelemetry.io/contrib v0.20.0  
109. go.opentelemetry.io/contrib/instrumentation/google.golang.org/grpc/otelgrpc v0.20.0  
110. go.opentelemetry.io/contrib/instrumentation/net/http/otelhttp v0.20.0  
111. go.opentelemetry.io/otel v0.20.0  
112. go.opentelemetry.io/otel/exporters/otlp v0.20.0  
113. go.opentelemetry.io/otel/metric v0.20.0  
114. go.opentelemetry.io/otel/sdk v0.20.0  
115. go.opentelemetry.io/otel/sdk/export/metric v0.20.0  
116. go.opentelemetry.io/otel/sdk/metric v0.20.0  
117. go.opentelemetry.io/otel/trace v0.20.0  
118. go.opentelemetry.io/proto/otlp v0.7.0  
119. go.starlark.net v0.0.0-20200306205701-8dd3e2ee1dd5  
120. go.uber.org/atomic v1.9.0  
121. go.uber.org/multierr v1.6.0  
122. go.uber.org/zap v1.19.1  
123. golang.org/x/crypto v0.0.0-20220622213112-05595931fe9d  
124. golang.org/x/mod v0.6.0-dev.0.20220106191415-9b9b3d81d5e3  
125. golang.org/x/net v0.0.0-20220425223048-2871e0cb64e4  
126. golang.org/x/oauth2 v0.0.0-20220411215720-9780585627b5  
127. golang.org/x/sys v0.0.0-20220520151302-bc2c85ada10a  
128. golang.org/x/time v0.0.0-20220210224613-90d013bbcef8  
129. golang.org/x/errors v0.0.0-20200804184101-5ec99f83aff1

130. gomodules.xyz/jsonpatch/v2 v2.2.0  
131. google.golang.org/appengine v1.6.7  
132. google.golang.org/genproto v0.0.0-20220519153652-3a47de7e79bd  
133. gopkg.in/inf.v0 v0.9.1  
134. gopkg.in/ini.v1 v1.66.2  
135. gorm.io/datatypes v1.0.6  
136. k8s.io/cli-runtime v0.24.0  
137. k8s.io/gengo v0.0.0-2021129171323-c02415ce4185  
138. k8s.io/kube-openapi v0.0.0-20220328201542-3ee0da9b0b42  
139. oras.land/oras-go v1.1.0  
140. sigs.k8s.io/apiserver-network-proxy/konnectivity-client v0.0.30  
141. sigs.k8s.io/json v0.0.0-20211208200746-9f7c6b3444d2  
142. sigs.k8s.io/kustomize/api v0.11.4  
143. sigs.k8s.io/kustomize/kyaml v0.13.6  
144. sigs.k8s.io/structured-merge-diff/v4 v4.2.1

## 23.2 全局管理

---

涉及的开源项目有直接集成和间接调用两种。

## 直接集成

全局管理模块直接集成的项目包括但不限于：

1. ghippo.io/api v0.0.0
2. github.com/DATA-DOG/go-sqlmock v1.5.0
3. github.com/Nerzal/gocloak/v10 v10.0.1
4. github.com/agiledragon/gomonkey/v2 v2.8.0
5. github.com/coreos/go-oidc v2.2.1+incompatible
6. github.com/dgrijalva/jwt-go v3.2.0+incompatible
7. github.com/gin-gonic/gin v1.8.1
8. github.com/go-logr/zapr v1.2.3
9. github.com/go-playground/validator/v10 v10.11.0
10. github.com/go-sql-driver/mysql v1.6.0
11. github.com/golang-jwt/jwt/v4 v4.4.3
12. github.com/golang-migrate/migrate/v4 v4.15.2
13. github.com/google/uuid v1.3.0
14. github.com/grpc-ecosystem/go-grpc-middleware v1.3.0
15. github.com/grpc-ecosystem/grpc-gateway/v2 v2.10.0
16. github.com/grpc-ecosystem/protoc-gen-grpc-gateway-ts v1.1.1
17. github.com/jackc/pgconn v1.13.0
18. github.com/jackc/pgerrcode v0.0.0-20201024163028-a0d42d470451
19. github.com/jackc/pgx/v4 v4.17.2
20. github.com/jinzhu/configor v1.2.1
21. github.com/lestrrat-go/jwx/v2 v2.0.4
22. github.com/lib/pq v1.10.5
23. github.com/nicksnyder/go-i18n/v2 v2.2.0
24. github.com/onsi/ginkgo/v2 v2.3.1
25. github.com/pkg/errors v0.9.1
26. github.com/robfig/cron/v3 v3.0.1
27. github.com/smartystreets/goconvey v1.7.2
28. github.com/soheilhy/cmux v0.1.5
29. github.com/stretchr/testify v1.8.1
30. github.com/uptrace/opentelemetry-go-extra/otelgorm v0.1.17
31. github.com/urfave/cli/v2 v2.3.0
32. go.opentelemetry.io/contrib/instrumentation/github.com/gin-gonic/gin/otelgin v0.36.3
33. go.opentelemetry.io/contrib/instrumentation/google.golang.org/grpc/otelgrpc v0.32.0
34. go.opentelemetry.io/otel v1.11.0
35. go.opentelemetry.io/otel/exporters/otlp/otlptrace v1.10.0
36. go.opentelemetry.io/otel/exporters/otlp/otlptrace/otlptracegrpc v1.10.0
37. go.opentelemetry.io/otel/exporters/prometheus v0.30.0
38. go.opentelemetry.io/otel/metric v0.32.1
39. go.opentelemetry.io/otel/sdk v1.10.0
40. go.opentelemetry.io/otel/sdk/metric v0.31.0
41. go.uber.org/zap v1.19.1
42. golang.org/x/oauth2 v0.0.0-20220309155454-6242fa91716a
43. golang.org/x/sync v0.1.0

44. golang.org/x/text v0.6.0
45. google.golang.org/grpc v1.50.1
46. google.golang.org/grpc/cmd/protoc-gen-go-grpc v1.1.0
47. google.golang.org/protobuf v1.28.1
48. gopkg.in/natefinch/lumberjack.v2 v2.0.0
49. gorm.io/driver/mysql v1.4.4
50. gorm.io/driver/postgres v1.4.4
51. gorm.io/gorm v1.23.10
52. istio.io/api v0.0.0-20220304230428-0cdff8a2764f
53. istio.io/client-go v1.13.2
54. k8s.io/apimachinery v0.24.3
55. k8s.io/client-go v0.24.3
56. k8s.io/code-generator v0.24.3
57. k8s.io/klog/v2 v2.90.0
58. kpanda.io/api v0.0.0
59. sigs.k8s.io/controller-runtime v0.12.3

## 间接调用

全局管理模块间接调用的项目包括但不限于：

1. github.com/cenkalti/backoff/v4 v4.1.3
2. github.com/decred/dcrd/dcrec/secp256k1/v4 v4.0.1
3. github.com/emicklei/go-restful/v3 v3.10.1
4. github.com/go-logr/stdr v1.2.2
5. github.com/go-openapi/swag v0.22.3
6. github.com/goccy/go-json v0.9.10
7. github.com/google/gnostic v0.6.9
8. github.com/gopherjs/gopherjs v0.0.0-20181017120253-0766667cb4d1
9. github.com/jtolds/gls v4.20.0+incompatible
10. github.com/lestrrat-go/blackmagic v1.0.1
11. github.com/lestrrat-go/httpcc v1.0.1
12. github.com/lestrrat-go/httprc v1.0.4
13. github.com/lestrrat-go/iter v1.0.2
14. github.com/lestrrat-go/option v1.0.0
15. github.com/munnerz/goautoneg v0.0.0-20191010083416-a7dc8b61c822
16. github.com/pelletier/go-toml/v2 v2.0.1
17. github.com/smartystreets/assertions v1.2.0
18. github.com/uptrace/opentelemetry-go-extra/otelsql v0.1.17
19. go.opentelemetry.io/otel/exporters/otlp/internal/retry v1.10.0
20. go.opentelemetry.io/otel/trace v1.11.0
21. go.opentelemetry.io/proto/otlp v0.19.0
22. k8s.io/kube-openapi v0.0.0-20230127205639-68031ae9242a
23. cloud.google.com/go v0.99.0
24. github.com/Azure/go-autorest v14.2.0+incompatible
25. github.com/Azure/go-autorest/autorest v0.11.18
26. github.com/Azure/go-autorest/autorest/adal v0.9.16
27. github.com/Azure/go-autorest/autorest/date v0.3.0
28. github.com/Azure/go-autorest/logger v0.2.1
29. github.com/Azure/go-autorest/tracing v0.6.0
30. github.com/BurntSushi/toml v1.2.0
31. github.com/ModernMinds/goutils v1.1.0
32. github.com/ModernMinds/semver v1.5.0
33. github.com/ModernMinds/sprig v2.22.0+incompatible
34. github.com/beorn7/perks v1.0.1
35. github.com/cespare/xxhash/v2 v2.1.2
36. github.com/cpuguy83/go-md2man/v2 v2.0.1
37. github.com/davecgh/go-spew v1.1.1
38. github.com/envoyproxy/protoc-gen-validate v0.6.2
39. github.com/evanphx/json-patch v4.12.0+incompatible
40. github.com/fsnotify/fsnotify v1.5.1
41. github.com/gin-contrib/sse v0.1.0
42. github.com/go-logr/logr v1.2.3
43. github.com/go-openapi/jsonpointer v0.19.6

44. github.com/go-openapi/jsonreference v0.20.2  
45. github.com/go-playground/locales v0.14.0  
46. github.com/go-playground/universal-translator v0.18.0  
47. github.com/go-resty/resty/v2 v2.6.0  
48. github.com/gogo/protobuf v1.3.2  
49. github.com/golang/glog v1.0.0  
50. github.com/golang/groupcache v0.0.0-20210331224755-41bb18bfe9da  
51. github.com/golang/protobuf v1.5.2  
52. github.com/google/go-cmp v0.5.9  
53. github.com/google/gofuzz v1.2.0  
54. github.com/hashicorp/errwrap v1.1.0  
55. github.com/hashicorp/go-multierror v1.1.1  
56. github.com/huandu/xstrings v1.3.2  
57. github.com/iancoleman/strcase v0.2.0  
58. github.com/imdario/mergo v0.3.12  
59. github.com/jackc/chunkreader/v2 v2.0.1  
60. github.com/jackc/pgio v1.0.0  
61. github.com/jackc/pgpassfile v1.0.0  
62. github.com/jackc/pgproto3/v2 v2.3.1  
63. github.com/jackc/pgservicefile v0.0.0-20200714003250-2b9c44734f2b  
64. github.com/jackc/pgtype v1.12.0  
65. github.com/jinzhu/inflection v1.0.0  
66. github.com/jinzhu/now v1.1.5  
67. github.com/josharian/intern v1.0.0  
68. github.com/json-iterator/go v1.1.12  
69. github.com/leodido/go-urn v1.2.1  
70. github.com/mailru/easyjson v0.7.7  
71. github.com/matttn/go-isatty v0.0.14  
72. github.com/matttproud/golang\_protobuf\_extensions v1.0.2-0.20181231171920-c182affec369  
73. github.com/mitchellh/copystructure v1.0.0  
74. github.com/mitchellh/reflectwalk v1.0.0  
75. github.com/modern-go/concurrent v0.0.0-20180306012644-bacd9c7ef1dd  
76. github.com/modern-go/reflect2 v1.0.2  
77. github.com/opentracing/opentracing-go v1.2.0  
78. github.com/pmezard/go-difflib v1.0.0  
79. github.com/pquerna/cachecontrol v0.0.0-20171018203845-0dec1b30a021  
80. github.com/prometheus/client\_golang v1.12.1  
81. github.com/prometheus/client\_model v0.2.0  
82. github.com/prometheus/common v0.32.1  
83. github.com/prometheus/procfs v0.7.3  
84. github.com/russross/blackfriday/v2 v2.1.0  
85. github.com/segmentio/ksuid v1.0.4  
86. github.com/sirupsen/logrus v1.8.1

87. github.com/spf13/pflag v1.0.5  
88. github.com/stretchr/objx v0.5.0  
89. github.com/tjfoc/gmsm v1.4.2-0.20220114090716-36b992c51540  
90. github.com/ugorji/go/codec v1.2.7  
91. go.uber.org/atomic v1.7.0  
92. go.uber.org/multierr v1.8.0  
93. golang.org/x/crypto v0.0.0-20220722155217-630584e8d5aa  
94. golang.org/x/mod v0.7.0  
95. golang.org/x/net v0.5.0  
96. golang.org/x/sys v0.4.0  
97. golang.org/x/term v0.4.0  
98. golang.org/x/time v0.0.0-20220224211638-0e9765cccd65  
99. golang.org/x/tools v0.5.0  
100. gomodules.xyz/jsonpatch/v2 v2.2.0  
101. google.golang.org/appengine v1.6.7  
102. google.golang.org/genproto v0.0.0-20220317150908-0efb43f6373e  
103. gopkg.in/inf.v0 v0.9.1  
104. gopkg.in/square/go-jose.v2 v2.5.1; indi7509080b3f373e  
105. gopkg.in/yaml.v2 v2.4.0  
106. gopkg.in/yaml.v3 v3.0.1  
107. istio.io/gogo-genproto v0.0.0-20211208193508-5ab4acc9eb1e  
108. k8s.io/api v0.24.3  
109. k8s.io/apiextensions-apiserver v0.24.2  
110. k8s.io/component-base v0.24.2  
111. k8s.io/gengo v0.0.0-20221011193443-fad74ee6edd9  
112. k8s.io/utils v0.0.0-20220210201930-3a6ce19ff2f9  
113. sigs.k8s.io/json v0.0.0-20211208200746-9f7c6b3444d2  
114. sigs.k8s.io/structured-merge-diff/v4 v4.2.3  
115. sigs.k8s.io/yaml v1.3.0

### 23.3 可观测性

---

涉及的开源项目有直接集成和间接调用两种。

## 直接集成

可观测性模块直接集成的项目包括但不限于：

1. ghippo.io/api v0.0.0
2. github.com/DATA-DOG/go-sqlmock v1.5.0
3. github.com/elastic/go-elasticsearch/v7 v7.16.0
4. github.com/gorilla/mux v1.8.0
5. github.com/json-iterator/go v1.1.12
6. github.com/mwitkow/go-conctrack v0.0.0-20190716064945-2f068394615f
7. github.com/oklog/ulid/v2 v2.1.0
8. github.com/pkg/errors v0.9.1
9. github.com/prometheus-community/prom-label-proxy v0.4.0
10. github.com/prometheus/client\_golang v1.14.0
11. github.com/prometheus/common v0.39.0
12. github.com/prometheus/prometheus v1.8.2-0.20210811141203-dcb07e8eac34
13. github.com/soheilhy/cmux v0.1.5
14. github.com/spf13/cobra v1.6.1
15. github.com/spf13/pflag v1.0.5
16. github.com/spf13/viper v1.14.0
17. github.com/stretchr/testify v1.8.1
18. go.opentelemetry.io/contrib/instrumentation/google.golang.org/grpc/otelgrpc v0.37.0
19. go.opentelemetry.io/otel v1.11.2
20. go.opentelemetry.io/otel/exporters/otlp/otlptrace v1.11.2
21. go.opentelemetry.io/otel/exporters/otlp/otlptrace/otlptracegrpc v1.4.1
22. go.opentelemetry.io/otel/exporters/prometheus v0.30.0
23. go.opentelemetry.io/otel/metric v0.34.0
24. go.opentelemetry.io/otel/sdk v1.11.2
25. go.opentelemetry.io/otel/sdk/metric v0.31.0
26. go.opentelemetry.io/otel/trace v1.11.2
27. go.uber.org/zap v1.24.0
28. google.golang.org/grpc v1.51.0
29. google.golang.org/protobuf v1.28.1
30. gorm.io/driver/mysql v1.3.4
31. gorm.io/driver/postgres v1.3.7
32. gorm.io/gorm v1.23.8
33. gotest.tools v2.2.0+incompatible
34. insight.io/api v0.0.0
35. k8s.io/apimachinery v0.24.6
36. k8s.io/client-go v0.24.6
37. kpanda.io/api v0.0.0
38. sigs.k8s.io/yaml v1.3.0
39. github.com/VictoriaMetrics/metricsql v0.43.0
40. github.com/VictoriaMetrics/operator/api v0.0.0-20220623030345-4957cfea49fb
41. github.com/aquasecurity/esquery v0.2.0
42. github.com/deckarep/golang-set/v2 v2.1.0
43. github.com/dgrijalva/jwt-go v3.2.0+incompatible

44. github.com/gocarina/gocsv v0.0.0-20220729221910-a7386ae0b221
45. github.com/golang-migrate/migrate/v4 v4.15.2
46. github.com/grpc-ecosystem/go-grpc-middleware v1.3.0
47. github.com/grpc-ecosystem/grpc-gateway/v2 v2.10.3
48. github.com/hashicorp/golang-lru v0.5.4
49. github.com/kr/pretty v0.3.1
50. github.com/muesli/cache2go v0.0.0-20211005105910-8e46465cca4a
51. github.com/prometheus-operator/prometheus-operator/pkg/apis/monitoring v0.57.0
52. github.com/prometheus-operator/prometheus-operator/pkg/client v0.57.0
53. github.com/robfig/cron/v3 v3.0.1
54. github.com/uptrace/opentelemetry-go-extra/otelgorm v0.1.15
55. gopkg.in/gomail.v2 v2.0.0-20160411212932-81ebce5c23df
56. moul.io/zapgorm2 v1.1.3
57. sigs.k8s.io/controller-runtime v0.12.0
58. github.com/agiledragon/gomonkey/v2 v2.7.0
59. github.com/gogo/protobuf v1.3.2
60. gopkg.in/yaml.v2 v2.4.0

## 间接调用

可观测性模块间接调用的项目包括但不限于：

1. github.com/evanhx/json-patch v5.6.0+incompatible
2. github.com/google/gofuzz v1.2.0
3. github.com/pelletier/go-toml v1.9.5
4. github.com/pelletier/go-toml/v2 v2.0.5
5. github.com/rogpeppe/go-internal v1.9.0
6. github.com/VictoriaMetrics/VictoriaMetrics v1.77.1
7. github.com/VictoriaMetrics/fasthttp v1.1.0
8. github.com/VictoriaMetrics/metrics v1.18.1
9. github.com/asaskevich/govalidator v0.0.0-20210307081110-f21760c49a8d
10. github.com/beorn7/perks v1.0.1
11. github.com/cenkalti/backoff/v4 v4.2.0
12. github.com/cespare/xxhash/v2 v2.1.2
13. github.com/davecgh/go-spew v1.1.1
14. github.com/efficientgo/tools/core v0.0.0-20210201224146-3d78f4d30648
15. github.com/emicklei/go-restful/v3 v3.9.0
16. github.com/envoyproxy/protoc-gen-validate v0.6.8
17. github.com/fatih/structs v1.1.0
18. github.com/fsnotify/fsnotify v1.6.0
19. github.com/go-kit/log v0.2.1
20. github.com/go-logfmt/logfmt v0.5.1
21. github.com/go-logr/logr v1.2.3
22. github.com/go-logr/stdr v1.2.2
23. github.com/go-openapi/analysis v0.21.2
24. github.com/go-openapi/errors v0.20.2
25. github.com/go-openapi/jsonpointer v0.19.5
26. github.com/go-openapi/jsonreference v0.20.0
27. github.com/go-openapi/loads v0.21.1
28. github.com/go-openapi/runtime v0.24.1
29. github.com/go-openapi/spec v0.20.6
30. github.com/go-openapi/strfmt v0.21.3
31. github.com/go-openapi/swag v0.22.3
32. github.com/go-openapi/validate v0.22.0
33. github.com/go-sql-driver/mysql v1.6.0
34. github.com/gogo/googleapis v1.4.1
35. github.com/golang/groupcache v0.0.0-20210331224755-41bb18bfe9da
36. github.com/golang/protobuf v1.5.2
37. github.com/google/gnostic v0.6.9
38. github.com/google/go-cmp v0.5.9
39. github.com/google/uuid v1.3.0
40. github.com/hashicorp/errwrap v1.1.0
41. github.com/hashicorp/go-multierror v1.1.1
42. github.com/hashicorp/hcl v1.0.0
43. github.com/imdario/mergo v0.3.13

44. github.com/inconshreveable/mousetrap v1.0.1  
45. github.com/jackc/chunkreader/v2 v2.0.1  
46. github.com/jackc/pgconn v1.13.0  
47. github.com/jackc/pgio v1.0.0  
48. github.com/jackc/pgpassfile v1.0.0  
49. github.com/jackc/pgproto3/v2 v2.3.1  
50. github.com/jackc/pgservicefile v0.0.0-20200714003250-2b9c44734f2b  
51. github.com/jackc/pgtype v1.12.0  
52. github.com/jackc/pgx/v4 v4.17.0  
53. github.com/jinzhu/inflection v1.0.0  
54. github.com/jinzhu/now v1.1.5  
55. github.com/josharian/intern v1.0.0  
56. github.com/jpillora/backoff v1.0.0  
57. github.com/klauspost/compress v1.15.11  
58. github.com/kr/text v0.2.0  
59. github.com/lib/pq v1.10.6  
60. github.com/magiconair/properties v1.8.6  
61. github.com/mailru/easyjson v0.7.7  
62. github.com/mattlproud/golang\_protobuf\_extensions v1.0.4  
63. github.com/mitchellh/mapstructure v1.5.0  
64. github.com/modern-go/concurrent v0.0.0-20180306012644-bacd9c7ef1dd  
65. github.com/modern-go/reflect2 v1.0.2  
66. github.com/munnerz/goautoneg v0.0.0-20191010083416-a7dc8b61c822  
67. github.com/oklog/ulid v1.3.1  
68. github.com/opentracing/opentracing-go v1.2.0  
69. github.com/pmezard/go-difflib v1.0.0  
70. github.com/prometheus/alertmanager v0.23.0  
71. github.com/prometheus/client\_model v0.3.0  
72. github.com/prometheus/procfs v0.8.0  
73. github.com/shopspring/decimal v1.3.1  
74. github.com/spf13/afero v1.9.2  
75. github.com/spf13/cast v1.5.0  
76. github.com/spf13/jwalterweatherman v1.1.0  
77. github.com/stretchr/objx v0.5.0  
78. github.com/subosito/gotenv v1.4.1  
79. github.com/uber/jaeger-client-go v2.30.0+incompatible  
80. github.com/uber/jaeger-lib v2.4.1+incompatible  
81. github.com/uptrace/opentelemetry-go-extra/otelsql v0.1.15  
82. github.com/valyala/bytbufferpool v1.0.0  
83. github.com/valyala/fastrand v1.1.0  
84. github.com/valyala/fasttemplate v1.2.1  
85. github.com/valyala/histogram v1.2.0  
86. github.com/valyala/quicktemplate v1.7.0

87. go.mongodb.org/mongo-driver v1.10.0
88. go.opentelemetry.io/otel/exporters/otlp/internal/retry v1.11.2
89. go.opentelemetry.io/proto/otlp v0.19.0
90. go.uber.org/atomic v1.10.0
91. go.uber.org/multierr v1.8.0
92. golang.org/x/crypto v0.0.0-20220722155217-630584e8d5aa
93. golang.org/x/net v0.4.0
94. golang.org/x/oauth2 v0.3.0
95. golang.org/x/sys v0.3.0
96. golang.org/x/term v0.3.0
97. golang.org/x/text v0.5.0
98. golang.org/x/time v0.0.0-20220609170525-579cf78fd858
99. gomodules.xyz/jsonpatch/v2 v2.2.0
100. google.golang.org/appengine v1.6.7
101. gopkg.in/alexcesaro/quotedprintable.v3 v3.0.0-20150716171945-2caba252f4dc
102. gopkg.in/inf.v0 v0.9.1
103. gopkg.in/ini.v1 v1.67.0
104. gopkg.in/yaml.v3 v3.0.1
105. k8s.io/api v0.24.6
106. k8s.io/apiextensions-apiserver v0.24.0
107. k8s.io/component-base v0.24.0
108. k8s.io/klog/v2 v2.80.1
109. k8s.io/kube-openapi v0.0.0-20220928191237-829ce0c27909
110. k8s.io/utils v0.0.0-20220728103510-ee6ede2d64ed
111. sigs.k8s.io/json v0.0.0-20211208200746-9f7c6b3444d2
112. sigs.k8s.io/structured-merge-diff/v4 v4.2.3

未完待续（例如网络、存储等基础模块还未包含在上述列表）

## 24. 5.0 资源管理能力介绍

随着企业规模的扩大，管控的资源必定日渐庞杂复：成千上万台的服务器、机器设备、电子器件等有形固定资产；日常办公软件、专业财务/人事/OA/项目管理/开发等专业软件；管理层、雇员、职员等资源。

如何高效管理这些资源？



基于资源的管理能力本质上是建立一套与企业相关的，基于资源使用的关系结构。在 DCE 5.0 中主要通过工作空间与层级模块实现，该模块具有全局一致性的特点，方便您基于此关系结构对平台上多个子模块内的各种资源进行高效规划和管理。

工作空间与层级模块中有两个概念：层级和工作空间。层级可以映射为企业中的项目、环境、供应商等概念，方便您基于企业的业务或生态环境构建出体现资源关系的企业层级关系。同时层级具有权限继承能力，上级层级管理员能够创建和管理下级层级及工作空间，有效解决层级结构中的一次性授权问题。工作空间与资源直接关联，支持资源独享、资源共享等多维度，跨集群的资源管理能力，满足数百个集群下的资源快速授权和资源灵活调配需求。同时，工作空间与层级模块还支持移动层级和移动工作空间功能，随时应对企业层级关系调整等带来的资源调整。

### 24.1 资源管理的用户交互

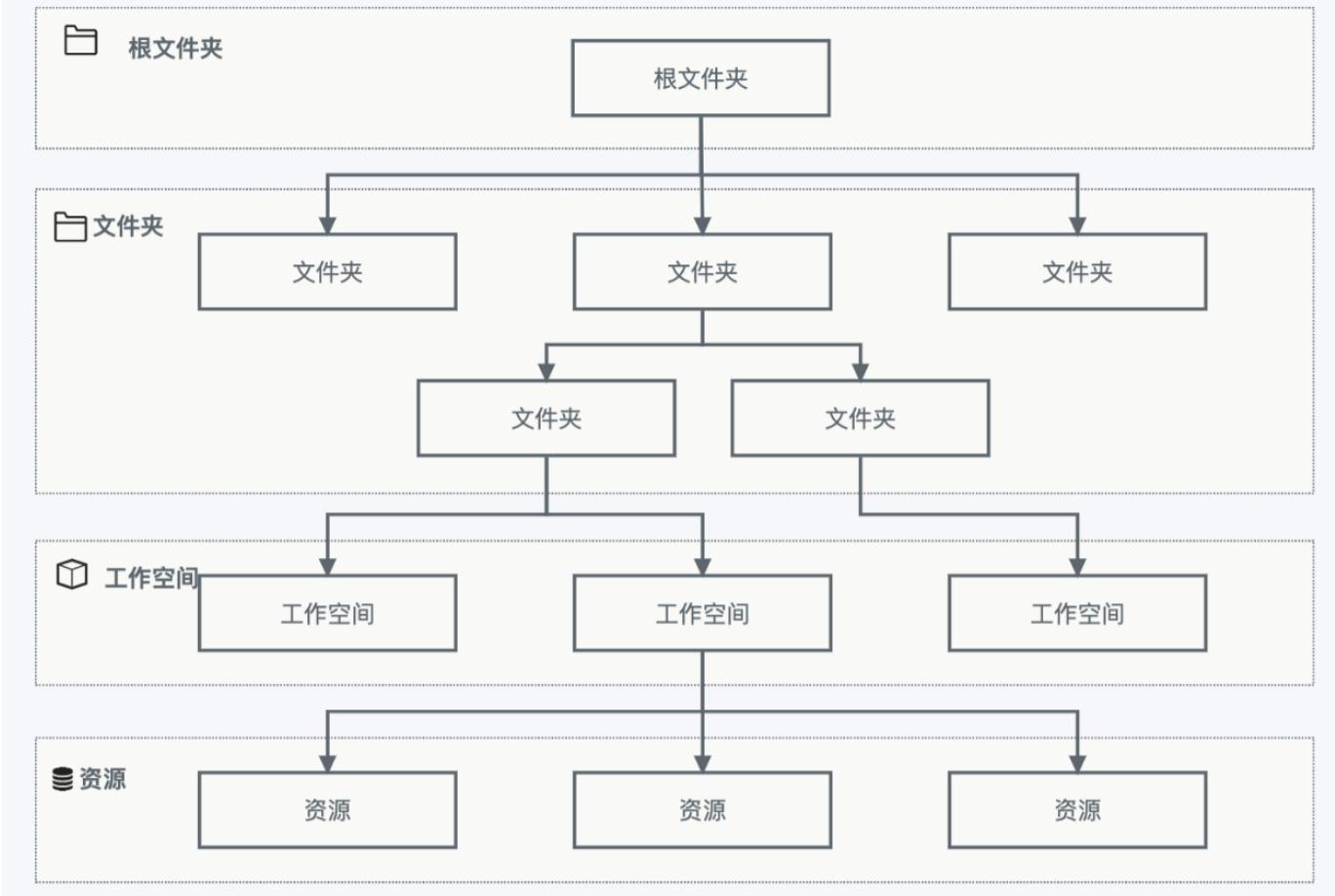
资源管理模块是一个具有层级的资源隔离和资源分组模块，主要解决资源统一授权、资源分组以及资源限额问题。资源管理模块有两个概念：层级和工作空间。在资源管理模块中您可以通过层级构建企业业务层级关系，通过工作空间中的资源组和共享资源管理资源，使资源管理模块的用户（用户组）之间能够共享工作空间中的资源。

资源处于资源管理模块层级结构的最低层级，资源包括 Cluster、Namespace、Pipeline、网关等。所有这些较低层级的资源的父级只能是工作空间，而工作空间作为资源容器是一种资源分组单位。工作空间通常代指一个项目或环境，每个工作空间的资源相对于其他工作空间中的资源时逻辑隔离的。您可以通过工作空间中的授权，授予用户（用户组）同一组资源的不同访问权限。

从层次结构的底层算起，工作空间位于第一层，且包含资源。除共享资源外，所有资源有且仅有一个父项。所有工作空间也有且仅有一个父级层级。

层级是在工作空间基础之上的进一步分组机制，具有层级结构。一个层级可以包含工作空间、其他层级或两者的组合，能够形成树状的组织关系。借助层级您可以映射企业业务层级关系，按照部门对工作空间进行分组。层级不直接与资源挂钩，而是通过工作空间间接实现资源分组。层级有且仅有一个父级层级，根层级是层次结构的最高层级，因此没有父级，层级和工作空间均挂靠到根层级下。

同时，用户（用户组）在层级中能够通过层级结构继承来自于父项的权限。用户在层次结构中每个节点权限来自于直接在该节点获取的权限和继承其父项权限的组合结果，权限之间是加合关系不存在互斥。



资源通过工作空间进行分组，而工作空间中存在两种分组模式，分别是资源组和共享资源。一个资源只能加入一个资源组，资源组与工作空间一一对应，资源被加入到资源组后，工作空间的所有者将获得资源的管理权限，相当于该资源的所有者。而对于共享资源来说，多个工作空间可以共享同一个或者多个资源。资源的所有者，可以选择将自己拥有的资源共享给工作空间使用，一般共享时资源所有者会限制被共享工作空间能够使用的资源额度。资源被共享后，工作空间的所有者仅具有资源限额下的资源使用权限，无法管理资源或者调整工作空间能够使用的资源量。同时共享资源对于资源本身也具有一定要求，仅具有层级的资源能够被共享，比如 Cluster-Namespace，Cluster 资源所有者能够将 Cluster 资源分享给不同的工作空间使用，并且限制工作空间在此 Cluster 上的使用额度，工作空间所有者在资源限额内能够创建多个Namespace，但是 Namespace 的资源额度总和不能超过 Cluster 在该工作空间的资源限额。对于 Kubernetes 资源，当前能够分享的资源类型仅有 Cluster。

[申请社区免费体验](#)

## 24.2 资源管理要素

安装 DCE 5.0 后，在初始化阶段平台会自动为用户创建根层级，根层级位于资源管理模块资源层级结构的顶层，没有父级。资源关系依据根层级向下分布，最多支持 5 级层级。

## 层级

平台管理员根据企业现有的层级结构在根层级下创建新的层级。 层级可以映射为企业的项目、环境、供应商等概念，层级具有层级结构和权限继承，是资源管理模块资源层级结构的节点。一个层级可以包含工作空间、其他层级或两者的组合。新创建的层级必须具有父级，且下级层级和工作空间会继承父级文件的权限，因此用户在层级或工作空间的实际权限是来自于继承父级层级的权限和在该节点获得的权限之和。

## 工作空间

工作空间是资源管理模块中的资源隔离单元，具有资源组和共享资源两种模式，与资源直接关联。工作空间有且只有一个父级层级，工作空间及其下的资源会继承父级层级的权限。

在 DCE 5.0 中，由于资源存在不同的授权方式，因此衍生出了强依赖于工作空间授权的资源，如网关、Pipeline 等；以及可以选择性绑定到工作空间的资源，如 Cluster、Namespace。

应用工作台、微服务引擎、服务网格的资源由于以工作空间为顶层概念，资源完全依赖于工作空间进行授权，因此资源会被自动绑定到某个工作空间，工作空间的所有者即是资源的所有者。此类资源无需手动绑定，资源也无需在工作空间的资源组进行显示，用户（用户组）使用资源的前提是被授予工作空间的相应权限。而对于 Cluster、Namespace 资源，由于容器管理有独立的授权模块，资源所有者有选择性绑定的权利。资源被手动绑定到工作空间的资源组，表示允许工作空间所有者能够管理使用该资源，此时工作空间所有者相当于资源所有者，被绑定的资源将在工作空间的资源组进行显示，并可以随时解绑。资源被绑定到工作空间的共享资源，表示允许资源所有者在资源限额内使用资源，当前能够被共享的资源类型仅有 Cluster。

## 资源限额

共享资源不意味着被共享者可以无限的使用被共享的资源，一般资源所有者会限制工作空间在该资源上能够使用的额度上限。比如，当 Cluster 被共享到工作空间后，工作空间所有者就获得了在该 Cluster 上创建 Namespace 的权利。若 Cluster 所有者设置了工作空间在该 Cluster 上的资源额度上限为 Quota=100，那么 Namespace 的 Quota 总和不能超过 100。同样的，当多个 Cluster 被共享到该工作空间时，那么每个 Cluster 所有者都可以设置工作空间在各个 Cluster 上的额度使用上限。另外，除了 Quota 外，资源所有者还可以通过 CPU、内存、Secret 等多个维度对资源额度上限进行限制。（资源所有者能够共享资源的前提：同时是 Cluster 和工作空间的所有者）。

## 移动层级和工作空间

部门关系或项目关系的调整随时可能使资源的授权关系发生变化，为了能够随机应变，移动层级和移动工作空间功能应运而生。以移动层级为例，由于资源、工作空间、层级的权限具有继承关系，因此移动层级后，原父级层级将失去对该层级以及其子层级、工作空间、空间中资源的管理权限。新的父级获得对该层级以及其子层级、工作空间、空间中资源的管理权限。对于层级本身以及其下的子层级授权关系没有发生任何变化。工作空间同理。

## 24.3 谁需要资源管理

需要对 K8S、微服务引擎、服务网格等资源进行管理，且资源具有一定规模的企业用户

## 解决什么问题

1. 企业人员众多同时存在多层级结构，如多级供应商，多级部门等，而资源往往聚焦在少部分运维人员手中，资源下发与资源管理耗费大量的人力物力
2. 企业层级结构多变，无法灵活应对部门调整带来的人员权限和资源归属变化
3. 以项目为驱动力的企业，对于同一批人，同样的资源需要进行多次授权
4. 对于同一种资源，多人拥有相同的权限存在安全风险

## 适用场景

1. 探索更为便捷的资源下发与资源管理方式，减少人力物力消耗，缩减成本
2. 资源管理方式映射企业层级结构，按照层级进行分权自制，更为灵活的应对企业变化，同时接入更多供应商，扩大业务范围
3. 更为精细化的资源权限控制，同一种资源设置管理、使用、只读等多种权限，为不同的成员下发不同的权限，将安全风险降到最低，进一步降低事故率

## 24.4 FAQs

### 1. 什么场景下需要使用层级？

答：层级可以映射为企业的项目、环境、供应商等概念，具有层级结构和权限继承能力。可以有效应对企业遇到的多级供应商的资源分配问题。

### 2. 什么场景下需要使用工作空间？

答：资源与工作空间直接关联，同时资源会继承工作空间的权限。当资源数量很大时，通过工作空间对资源进行统一授权能够有效减少资源运维的工作量。另外在 DCE 5.0 中，应用工作台、微服务引擎、服务网格中的资源依赖于工作空间授权，因此创建此类资源的前提是拥有工作空间的相应权限。

### 3. 层级、工作空间及资源的关系？

答：资源与工作空间直接关联，资源权限继承工作空间的权限；而工作空间一定归属于某个层级，且工作空间权限继承层级权限，所以拥有层级权限的用户相当于对其下工作空间中的资源有相应权限，但是资源不与层级直接绑定。

### 4. 移动层级或者工作空间，资源和权限是否会发生变化？

答：资源绑定在工作空间下，会随着层级或者工作空间的移动而移动。因为资源继承工作空间的权限，工作空间继承层级的权限，所以移动工作空间或者层级后会改变原有的权限继承关系，原来层级或者工作空间的上级失去对层级或者工作空间的管理权限，目标层级及其上级开始拥有对该层级或者工作空间的管理权限。

### 5. 谁能够创建 Folder？

答：Admin、Workspace Owner 和上一级 Folder Admin。

### 6. 谁能够创建 Workspace？

答：Admin、Workspace Owner、Folder Admin。

### 7. 想要获得 微服务治理、应用工作台、服务网格权限找谁授权？

答：微服务治理、应用工作台、服务网格下的资源均自动绑定 Workspace，且权限强依赖于 Workspace，需要通过获得 Workspace 权限继而获取相关的资源权限。能够进行 Workspace 授权的有 Admin、Workspace Owner、Folder Admin、Workspace Admin。

[了解服务网格](#) { .md-button }

[下载 DCE 5.0](#) { .md-button .md-button--primary } [安装 DCE 5.0](#) { .md-button .md-button--primary } [申请社区免费体验](#) { .md-button .md-button--primary }