DCE 5.0 商业版
安装部署手册

2025-03-09

DaoCloud 研发部门

本手册来源于 DCE 5.0 官方网站

请以官网信息为准，如有更改恕不另行通知

目录

安装简介

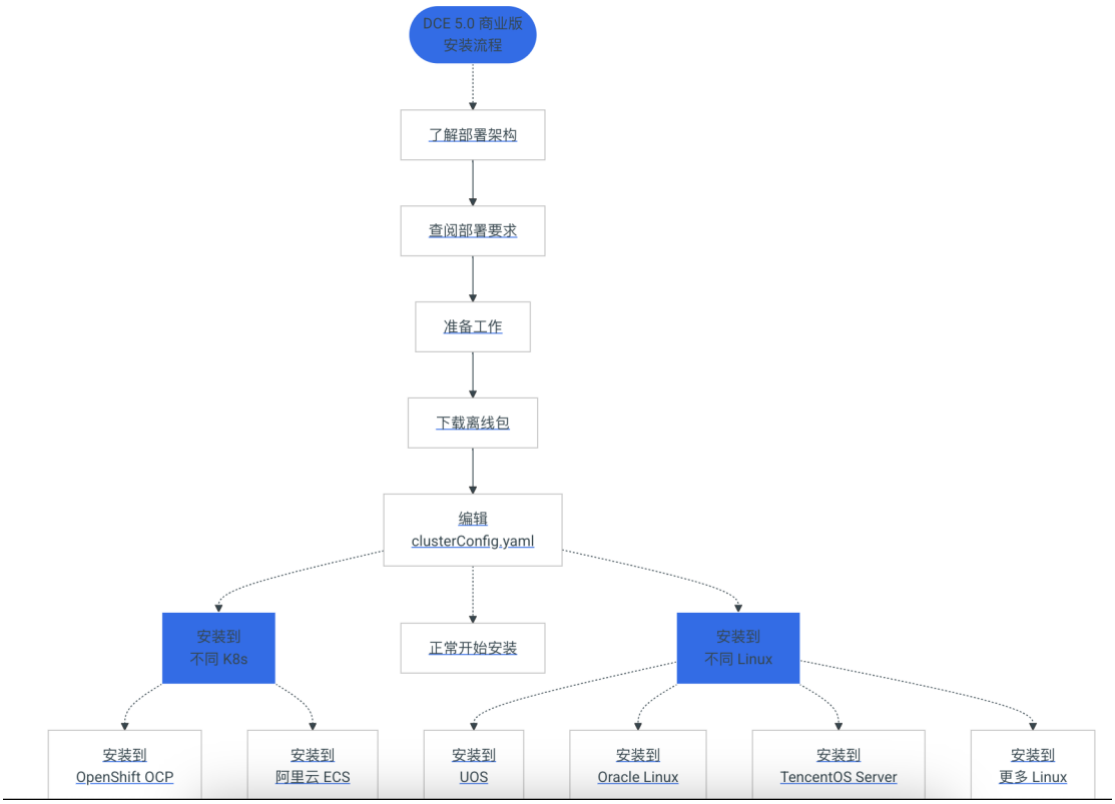
DCE 5.0 大体分为两个版本：社区版和商业版。

- 社区版包括容器管理、全局管理、可观测性三大模块，可永久免费使用。
- 商业版在社区版的基础上可按需购买服务网格、微服务引擎、多云编排、数据服务中间件、镜像仓库、云边协同、容器化虚拟机等高级模块， 功能更全面，更能适应各类生产环境需求。

版本	包含的模块	描述
社区版	♦ 全局管理 ♦ 容器管理 ♦ 可观测性	永久免费授权，3 个模块会保持持续更新，可随时 下载子模块的离线包
商业版	社区版上增加了： ♦ 应用工作台 ♦ 多云编排 ♦ 微服务引擎 ♦ 服务网格 ♦ 精选中间件 ♦ 镜像仓库 ♦ 云边协同 ♦ 容器化的虚拟机	联系我们授权：电邮 info@daocloud.io 或致电 400 002 6898，各个模块可按需自由组合，可随时 下载子模块的离线包

商业版安装流程

DCE 5.0 商业版的安装流程如下图：



联系我们

DCE 5.0 的安装流程可能会有变更。请收藏此页，关注更新动态，更多操作文档也在制作之中。

- 若有任何安装或使用问题，请提出反馈。
- 欢迎扫描二维码，与开发者畅快交流：

安装依赖项

安装 DCE 5.0 之前，需要先安装一些依赖项。

- 对于社区版，请在 K8s Master 节点上安装依赖项。
- 对于商业版，请在火种节点上安装依赖项。

安装的依赖项大致包括：

- podman
- helm
- skopeo
- kind
- kubectl
- yq
- minio client
- charts-syncer

安装过程中如果您的环境存在一些工具且版本低于我们定义的版本，会将对应工具进行强制更新替换。

在线安装依赖项

1. 下载脚本。
2. export VERSION=v0.20.0

3. curl -LO https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/install_prerequisite_${VERSION}.sh

   为 install_prerequisite_${VERSION}.sh 添加可执行权限：

   chmod +x install_prerequisite_${VERSION}.sh

4. 开始在线安装前置依赖。

   - 对于社区版

     bash install_prerequisite_${VERSION}.sh online community

   - 对于商业版

     bash install_prerequisite_${VERSION}.sh online full

离线安装依赖项

离线安装意味着目标主机的网络处于离线状态，无法下载所需依赖项，所以需先在一个在线环境中制作好离线包。

找一台能连通外网的机器，下载安装脚本。

export VERSION=v0.20.0
curl -LO https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/install_prerequisite_${VERSION}.sh

1. 下载前置依赖组件离线包。

export VERSION=v0.20.0
curl -LO https://qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/prerequisite_${VERSION}_amd64.tar.gz

   - 如果是 arm 架构，请使用下载地址： https://qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/prerequisite_${VERSION}_arm64.tar.gz
   - 确保离线包与脚本在同一个目录层级

2. 将下载的这些离线包上传到一台 K8s 集群控制平面节点，执行离线安装。

   - 对于社区版

     export BINARY_TAR=prerequisite_${VERSION}_amd64.tar.gz
     chmod +x install_prerequisite_${VERSION}.sh
     ./install_prerequisite_${VERSION}.sh offline community

   - 对于商业版

     export BINARY_TAR=prerequisite_${VERSION}_amd64.tar.gz

```
chmod +x install_prerequisite_${VERSION}.sh
./install_prerequisite_${VERSION}.sh offline full
```

接下来就可以安装 DCE 5.0 社区版或商业版了。

部署架构

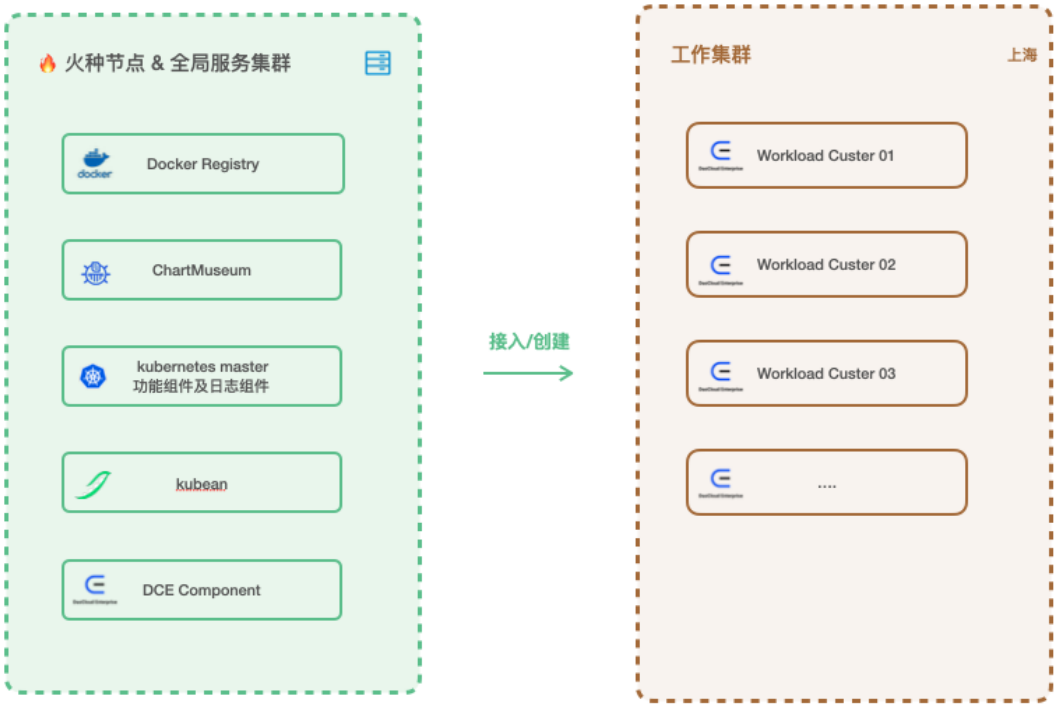
DCE 5.0 提供了三种部署架构：新手尝鲜模式、4 节点、7 节点。

下表介绍了架构中涉及到的术语：

名词	介绍
火种节点	火种节点也称为 bootstrap 节点，安装部署程序的执行，并运行平台所需的镜像仓库和 chart museum。
全局服务集群	用来部署 DCE 5.0 的所有组件，以及 Kubean，其中 Kubean 用来管理集群的生命周期。
工作集群	支撑业务应用和服务（成功安装 DCE 之后再部署工作集群）。
DCE 组件	DCE 5.0 的所有组件模块，包含全局管理、容器管理、可观测性、应用工作台、多云编排、镜像仓库、微服务引擎、服务网格、中间件等产品模块。

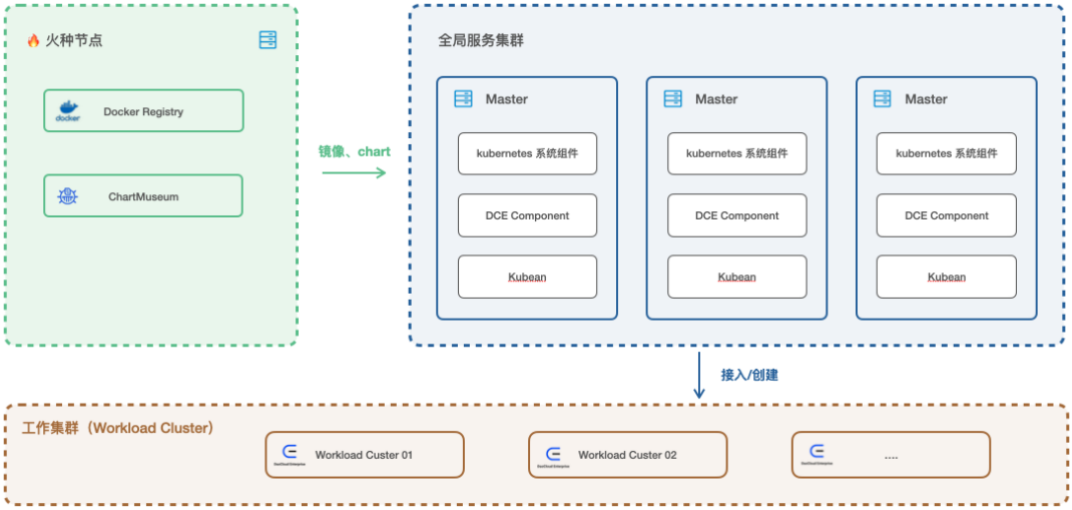
新手尝鲜模式

新手尝鲜模式仅需要一台主机（单节点），仅建议个人客户体验时采用此模式进行安装 DCE 5.0。 使用新手尝鲜模式时，建议使用最小化安装 DCE 5.0，即安装命令后添加 -z 参数。



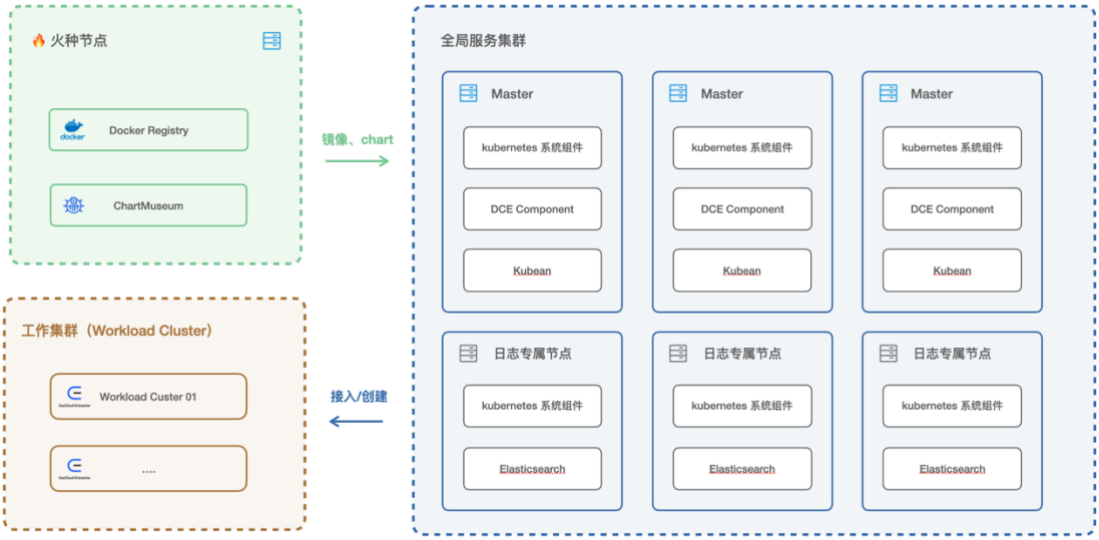
4 节点模式

4 节点模式由 1 个火种节点、3 个集群的主节点组成，仅建议 PoC 或测试环境时采用此模式进行安装 DCE 5.0。



7 节点模式（1 + 6）

7 节点模式由 1 个火种节点、3 个集群的主节点、3 个集群的工作节点组成，其中工作节点作为 Elasticsearch 组件的专属节点。 建议客户生产环境中采用此模式来安装 DCE 5.0。



集群配置文件 clusterConfig.yaml

此 YAML 文件包含了集群的各项配置字段，安装之前必须先配置此文件。 该文件将定义部署模式、集群节点信息等关键参数。默认位于 offline/sample/ 目录。

ClusterConfig 示例

以下是一个 ClusterConfig 文件示例。

clusterConfig.yaml

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
spec:
  clusterName: my-cluster

  # 火种节点的域名或 IP，默认解析为火种节点默认网关所在网卡的 IP；可手动填入 IP 或域名，若为域名，如果检测到无法解析，将自动建立此域名和火种节点默认 IP 的映射
  # bootstrapNode: auto

  # kind 火种集群的配置，以下为默认值
  # tinderKind:
  #   # kind 集群的容器名称
  #   instanceName: my-cluster-installer
  #   # kind 集群挂载的主机路径
  #   resourcesMountPath: /home/kind
  #   registryPort: 443
  #   minioServerPort: 9000
  #   minioConsolePort: 9001
  #   chartmuseumPort: 8081

  loadBalancer:

    # NodePort(default), metallb, cloudLB (Cloud Controller 暂不支持)
    type: metallb
    istioGatewayVip: xx.xx.xx.xx/32 # 当 loadBalancer.type 是 metallb 时必填，为 DCE 提供 UI 和 OpenAPI 访问权限
    insightVip: xx.xx.xx.xx/32 # 别丢弃 /32，当 loadBalancer.type 是 metallb 时必填，用作全局服务集群的 Insight 数据采集入口，子集群的 insight-agent 可以向这个 VIP 报告数据
    SourceIP: auto # 默认值 auto 表示开启审计日志获取源 IP 功能，设置为 false 则关闭审计日志获取源 IP 功能

  # 指定 ssh 私钥，定义后无需再定义节点的 ansibleUser、ansiblePass
  # privateKeyPath: /root/.ssh/id_rsa_sample

  masterNodes:
    - nodeName: "g-master1" # nodeName 将覆盖 hostName，应符合 RFC1123 标准
      ip: xx.xx.xx.xx
```

```
      ansibleUser: "root"
      ansiblePass: "dangerous"
      #ansibleSSHPort: "22"
      #ansibleExtraArgs: ""    # "ansible_shell_executable='/bin/sh'
ansible_python_interpreter='/usr/local/bin/python'" , format: "k='v'    k1='v1'    k2='v2' "
    - nodeName: "g-master2"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
      #ansibleSSHPort: "22"
      #ansibleExtraArgs: ""
    - nodeName: "g-master3"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
      #ansibleSSHPort: "22"
      #ansibleExtraArgs: ""
  workerNodes:
    - nodeName: "g-worker1"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
      #ansibleSSHPort: "22"
      #ansibleExtraArgs: ""
      nodeTaints:                          # 对于 7 节点模式：至少 3 个 worker 节点应打
污点（仅 ES 节点），如果使用外接 ES 则不需要添加该污点
        - "node.daocloud.io/es-only=true:NoSchedule"
      # nodeLabels:
      #     daocloud.io/hostname: g-worker1
    - nodeName: "g-worker2"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
      #ansibleSSHPort: "22"
      #ansibleExtraArgs: ""
      nodeTaints:
        - "node.daocloud.io/es-only=true:NoSchedule"
      # nodeLabels:
      #     daocloud.io/hostname: g-worker2
    - nodeName: "g-worker3"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
      #ansibleSSHPort: "22"
```

```
        #ansibleExtraArgs: ""
        nodeTaints:
          - "node.daocloud.io/es-only=true:NoSchedule"
        # nodeLabels:
        #     daocloud.io/hostname: g-worker3

    # ntpServer:
      # - 0.pool.ntp.org
      # - ntp1.aliyun.com
      # - ntp.ntsc.ac.cn

    fullPackagePath: "/root/offline" # 解压后的离线包的路径，离线模式下该字段必填

    osRepos: # 操作系统软件源

      # 支持 official-service(default), builtin
      type: builtin
      isoPath: "/root/CentOS-7-x86_64-DVD-2009.iso"
      osPackagePath: "/root/os-pkgs-centos7-v0.4.4.tar.gz"

      # skipValidateOSPackage: false # 跳过 ospackage 验证

      # type: external
      # Set the block below only if target is S3-compatible storage which need to upload files
automatically(e.g. minio).
      # isoPath: "/root/CentOS-7-x86_64-DVD-2009.iso"
      # osPackagePath: "/root/os-pkgs-centos7-v0.4.4.tar.gz"
      # externalRepoEndpoint: https://external-repo.daocloud.io
      # externalRepoUsername: rootuser
      # externalRepoPassword: rootpass123

      # type: external
      # Set the block below if target is other storage which cannot or does not need to upload
automatically(e.g. nginx).
      # That requires you to import the required packages(iso, os-pkgs) manually if not all the required
offline resources exist.
      # `centos` as CentOS, RedHat, kylin, AlmaLinux, Fedora or Openeuler
      # `debian` as Debian
      # `ubuntu` as Ubuntu
      # externalRepoType: centos
      # externalRepoURLs: ['https://extertal-
repo.daocloud.io/kubean/centos/\$releasever/os/\$basearch/']

    imagesAndCharts: # 镜像仓库和 Chart 仓库源
```

```
    # official-service(default), builtin or external

    type: builtin

    # type: external
    # IP or domain name
    # externalImageRepo: https://external-registry.daocloud.io
    # Set user and password. Optional
    # externalImageRepoUsername: admin
    # externalImageRepoPassword: Harbor12345
    # chartmuseum or harbor
    # externalChartRepoType: chartmuseum
    # IP or domain name
    # externalChartRepo: https://external-charts.daocloud.io:8081
    # Set user and password. Optional
    # externalChartRepoUsername: rootuser
    # externalChartRepoPassword: rootpass123

  addonPackage: # 应用商店 addon 离线包，定义后会对 addon 进行离线部署
    # path:
    #     - "/root/standard-addon-offline-package-v0.18.0-amd64.tar.gz"
    #     - "/root/gpu-addon-offline-package-v0.18.0-amd64.tar.gz"

  binaries: # 二进制可执行文件

    # official-service(default), builtin
    type: builtin

    # type: external
    # IP or domain name
    # externalRepository: https://external-binaries.daocloud.io:9000/kubean

#externalMiddlewares:
  #   database:
  #       kpanda:
  #           - dbDriverName: "mysql"
  #               # Please refer https://gorm.io/docs/connecting_to_the_database.html
  #               dataSourceName: "user:password@tcp(localhost:3306)/dbname"
  #               # readwrite(default) or readonly
  #               accessType: readwrite
  #               # The maximum number of open connections to the database.
  #               #maxOpenConnections: 100
  #               # The maximum number of connections in the idle connection pool.
```

```
  #             #maxIdleConnections: 10
  #             # The maximum amount of time a connection may be reused.
  #             #connectionMaxLifetimeSeconds: 3600
  #             # The maximum amount of time a connection may be idle.
  #             #connectionMaxIdleSeconds: 1800
  #     ghippoApiserver:
  #       - dbDriverName: "mysql"
  #         dataSourceName: "user:password@tcp(localhost:3306)/dbname"
  #     ghippoKeycloak:
  #       - dbDriverName: "mysql"
  #         dataSourceName: "user:password@tcp(localhost:3306)/dbname"
  #     ghippoAuditserver:
  #       - dbDriverName: "mysql"
  #         dataSourceName: "user:password@tcp(localhost:3306)/dbname"
  #   elasticsearch:
  #     insight:
  #       endpoint: "https://xx.xx.xx.xx:9200"
  #       insecure: false
  #       # basic auth
  #       username: "username"
  #       password: "password"
  #   kafka:
  #     brokers:
  #       - host1:9092
  #       - host2:9092
  #     # the username and password of kafka is not necessary
  #     username: "username"
  #     password: "password"
  #   S3Storage:
  #     default:
  #       endpoint: "xx.xx.xx.xx:9000"
  #       # Set if you dont want to verify the certificate.
  #       insecure: true
  #       bucket: "bucketname"
  #       accessKey: "YOUR-ACCESS-KEY-HERE"
  #       secretKey: "YOUR-SECRET-KEY-HERE"

  # Examples as below. More refer to kubespray options setting documentations.
  #kubeanConfig: |-
  #   this config will set the timezone of nodes , and it won't change timezone if this config is
commented out.
  #   ntp_timezone: Asia/Shanghai
  #   # Enable recommended node sysctl settings
  #   node_sysctl_tuning: true
```

```
  #    # Extra node sysctl settings while node_sysctl_tuning is enabled
  #    extra_sysctl: [{ name: net.ipv4.tcp_keepalive_time, value: 700 }]
  #    bin_dir: /usr/local/bin
  #    http_proxy: ""
  #    https_proxy: ""
  #    upstream_dns_servers:
  #      - 8.8.8.8
  #      - 8.8.4.4
  #    docker_mount_device: /dev/sdc
  #    docker_storage_options: "-s overlay2 --storage-opt overlay2.size=1G"

  # k8sVersion only take effect in online mode, don't set it in offline mode.
  # Unless to install a non-latest k8s version with offline pkg in place.
  #k8sVersion: v1.29.5
  #auditConfig:
  #    logPath: /var/log/audit/kube-apiserver-audit.log
  #    logHostPath: /var/log/kubernetes/audit
  #    #policyFile: /etc/kubernetes/audit-policy/apiserver-audit-policy.yaml
  #    #logMaxAge: 30
  #    #logMaxBackups: 10
  #    #logMaxSize: 100
  #    #policyCustomRules: >
  #    #    - level: None
  #    #      users: []
  #    #      verbs: []
  #    #      resources: []
  #network:
  #    cni: calico
  #    clusterCIDR: 10.233.64.0/18
  #    serviceCIDR: 10.233.0.0/18
  #cri:
  #    criProvider: containerd
  #    # criVersion only take effect in online mode, don't set it in offline mode
  #    #criVersion: 1.7.0
  #    # skip provision of CRI, default false. Currently only works with docker.
  #    #skipProvision: false

  #renewCerts:
  #    # there are only 2 modes of renew certs: `onetime` or `cyclical`, default value is `cyclical`.
  #    #mode: cyclical
  #    # 1. When mode is set to `cyclical`, certificate renewal will be performed on a timer in a cyclical
manner.
  #    #mode: cyclical
```

2. When mode is set to `onetime`, certificate renewal will be completed at once, and you can set the validity days of the certificate.

#mode: onetime

valid days can be set when in `onetime` mode, default valid days is 3650.

#oneTimeValidDays: 3650

关键字段

该 YAML 文件中的关键字段说明，请参阅下表。

字段	说明	默认值
clusterName	在 KuBean Cluster 里的全局服务集群命名	-
tinderKind	火种 kind 集群配置	-
tinderKind.instanceName	火种 kind 集群的容器名称	-
tinderKind.resourcesMountPath	kind 集群挂载的主机路径	/home/kind
tinderKind.registryPort	kind 集群中镜像仓库的端口	443
tinderKind.minioServerPort	kind 集群中 MinIO Server 的端口	9000
tinderKind.minioConsolePort	kind 集群中 MinIO Console 的端口	9001
tinderKind.chartmuseumPort	kind 集群中 ChartMuseum 的端口	8081
masterNodes	全局服务集群：Master 节点列表，包括 nodeName/ip/ansibleUser/ansiblePass 几个关键字段	-
masterNodes.nodeName	节点名称，将覆盖 hostName	-
masterNodes.ip	节点 IP	-
masterNodes.ansibleUser	节点账号	-
masterNodes.ansiblePass	节点密码	-

字段	说明	默认值
masterNodes.ansibleSSHPort	ssh 的端口，默认为 22	22
masterNodes.ansibleExtraArgs	指定 ansible 主机清单参数	-
workerNodes	全局服务集群：Worker 节点列表，包括 nodeName/ip/ansibleUser/ansiblePass 几个关键字段	-
privateKeyPath	kuBean 部署集群的 SSH 私钥文件路径，如果填写则不需要定义 ansibleUser、ansiblePass	-
k8sVersion	kuBean 安装集群的 K8s 版本必须跟 KuBean 和离线包相匹配	-
loadBalancer.insightVip	如果负载均衡模式是 metallb，则需要指定一个 VIP，供给全局服务集群的 insight 数据收集入口使用，子集群的 insight-agent 可上报数据到这个 VIP	-
loadBalancer.istioGatewayVip	如果负载均衡模式是 metallb，则需要指定一个 VIP，供给 DCE 的 UI 界面和 OpenAPI 访问入口	-
loadBalancer.type	所使用的 LoadBalancer 的模式，物理环境用 metallb，POC 用 NodePort，公有云和 SDN CNI 环境用 cloudLB（暂时还未未支持 cloudLB 模式）	NodePort (default) 、metallb 、cloudLB (Cloud Controller)
loadBalancer.SourceIP	审计日志获取源 IP，副作用：在节点层面无法进行负载均衡	auto
fullPackagePath	解压后的离线包的路径，离线模式下该字段必填	-
addonPackage.path	应用商店 addon 包本地文件系统路径	-
imagesAndCharts	镜像仓库和 Chart 仓库源	-
imagesAndCharts.externalChartRepo	外置 Chart 仓库的 IP 或域名	-

字段	说明	默认值
imagesAndCharts.externalChartRepoPassword	外置 Chart 仓库的密码，用于推送镜像	-
imagesAndCharts.externalChartRepoType	外置 Chart 仓库的类型，取值为 chartmuseum，harbor	-
imagesAndCharts.externalChartRepoUsername	外置 Chart 仓库的用户名，用于推送镜像	-
imagesAndCharts.externalImageRepo	指定 external 仓库的 IP 或者域名(需指定协议头)	-
imagesAndCharts.externalImageRepoPassword	外置镜像仓库的密码，用于推送镜像	-
imagesAndCharts.externalImageRepoUsername	外置镜像仓库的用户名,用于推送镜像	-
imagesAndCharts.type	镜像与 Chart 的访问模式，取值为 official-service(在线), buitin(火种内置 registry 和 chartmuseum), external(外置)	official-service
auditConfig	k8s api-server 的审计日志配置	默认关闭
binaries	二进制可执行文件	-
binaries.externalRepository	外置二进制可执行文件仓库的访问地址，URL 形式	-
binaries.type	二进制可执行文件的访问模式,取值为 official-service(在线), builtin(火种节点内置的 minio)	official-service
network.clusterCIDR	Cluster CIDR	-
network.cni	CNI 选择，比如 Calico、Cilium	calico
network.serviceCIDR	Service CIDR	-
ntpServer	可用的 NTP 服务器，供给新节点同步	-

字段	说明	默认值
	时间	
osRepos	操作系统软件源	-
osRepos.externalRepoType	外置软件源服务的操作系统类型，取值为 centos(所有红帽系列), debian, ubuntu	-
osRepos.externalRepoURLs	外置软件源的访问地址	-
osRepos.isoPath	操作系统 ISO 文件的路径，type 为 builtin 时不能为空	-
osRepos.osPackagePath	系统包文件的路径 ，type 为 builtin 时不能为空	-
osRepos.type	操作系统软件源的访问模式，取值为 official-service(在线), builtin(火种节点内置的 minio)	official-service
kubeanConfig.ntp_timezone	设置节点的时区，如果不配置该参数，默认按照节点中的时区	-
kubeanConfig.node_sysctl_tuning	开启后默认调整全局服务集群的 Systemctl 内核参数	false
kubeanConfig.extra_sysctl	设置额外的 Systemctl 内核参数	/usr/local/bin
externalMiddlewares	外置中间件	-
externalMiddlewares.database	外置数据库	-
externalMiddlewares.database.ghippoApiserver	ghippoApiserver 外置数据库的配置	-
externalMiddlewares.database.ghippoAuditserver	ghippoAuditserver 外置数据库的配置	-
externalMiddlewares.database.ghippoKeycloak	ghippoKeycloak 外置数据库的配置	-

字段	说明	默认值
externalMiddlewares.database.kpanda	kpanda 外置数据库的配置	-
externalMiddlewares.database.kpanda[0].accessType	kpanda 外置数据库的访问类型，取值：readwrite，readonly	readwrite
externalMiddlewares.database.kpanda[0].driver	kpanda 外置数据库的类型，取值：mysql	mysql
externalMiddlewares.database.kpanda[0].dataSourceName	kpanda 外置数据库的访数据源信息，用于连接数据库，可参考 Gorm 官网连接到数据库文档	-
externalMiddlewares.database.kpanda[0].maxOpenConnections	kpanda 外置数据库的最大连接数	10
externalMiddlewares.database.kpanda[0].maxIdleConnections	kpanda 外置数据库的最大空闲连接数	10
externalMiddlewares.database.kpanda[0].connectionMaxLifetimeSeconds	kpanda 外置数据库的最大连接生命周期	0
externalMiddlewares.database.kpanda[0].connectionMaxIdleTimeSeconds	kpanda 外置数据库的最大空闲连接生命周期	0
externalMiddleware.elasticsearch	外置 Elasticsearch	-
externalMiddleware.elasticsearch.insight	insight 所使用的外置 Elasticsearch 配置	-
externalMiddleware.elasticsearch.insight.endpoint	insight 所使用的外置 Elasticsearch 的访问地址	-
externalMiddleware.elasticsearch.insight.anonymous	insight 所使用的外置 Elasticsearch 的匿名访问，取值 true，false，配置为 true 时不应再填访问凭证	false
externalMiddleware.elasticsearch.insight.username	insight 所使用的外置 Elasticsearch 的访问用户名	-
externalMiddleware.elasticsearch.insight.password	insight 所使用的外置 Elasticsearch 的访问密码	-

字段	说明	默认值
externalMiddleware.kafka	外置 kafka	-
externalMiddleware.kafka.insight	insight 所使用的外置 kafka 配置	-
externalMiddleware.kafka.insight.brokers	brokers 地址	-
externalMiddleware.kafka.insight.username	insight 所使用的外置 kafka 的访问用户名	可选
externalMiddleware.kafka.insight.password	insight 所使用的外置 kafka 的访问密码	可选
renewCerts	集群证书续期	-
renewCerts.mode	证书续期的两种模式，支持 cyclical、onetime	-

精简配置说明

离线模式下采用 builtin 方式安装

builtin 模式意味着所需的第三方软件（如 chartMusem 、Minio、Docker registry）将由安装器进行部署并提供 DCE 5.0 平台使用。

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
  creationTimestamp: null
spec:
  clusterName: my-cluster
  masterNodes:
    - nodeName: "g-master1"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
  workerNodes:
  fullPackagePath: "/root/offline"
  osRepos:
    type: builtin
```

```
    isoPath: "/root/CentOS-7-x86_64-DVD-2009.iso"
    osPackagePath: "/root/os-pkgs-centos7-v0.4.4.tar.gz"
  imagesAndCharts:
    type: builtin
  addonPackage:
    #path:
    #   - "/root/standard-addon-offline-package-v0.18.0-amd64.tar.gz"
    #   - "/root/gpu-addon-offline-package-v0.18.0-amd64.tar.gz"
  binaries:
    type: builtin
```

离线模式下采用 external 方式安装

external 模式意味着所需的第三方软件（如 chartMusem 、Minio、Docker registry 等等）无需安装器安装，由使用者提供地址供 DCE 5.0 平台使用。

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
  creationTimestamp: null
spec:
  clusterName: my-cluster
  masterNodes:
    - nodeName: "g-master1"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
  workerNodes:

  fullPackagePath: "/root/offline"
  osRepos:
    type: external

    isoPath: "/root/CentOS-7-x86_64-DVD-2009.iso"

    osPackagePath: "/root/os-pkgs-centos7-v0.4.4.tar.gz"
    externalRepoType: centos
    externalRepoURLs: ["https://extertal-repo.daocloud.io/centos/\$releasever/os/\$basearch/"]
  imagesAndCharts:
    type: external
    externalImageRepo: https://external-registry.daocloud.io

    externalImageRepoUsername: admin
    externalImageRepoPassword: Harbor12345
    externalChartRepoType: chartmuseum
```

```
    externalChartRepo: https://external-charts.daocloud.io:8081
    externalChartUsername: rootuser
    externalChartMuseumPassword: rootpass123
  addonPackage:
    path: "/root/addon-offline-full-package-v0.4.8-amd64.tar.gz"
  binaries:
    type: external
    externalRepository: https://external-binaries.daocloud.io:9000/kubean
```

在线模式采用 official-service 方式安装

official-service 模式，当使用者采用在线安装 DCE 5.0 时，DCE 5.0 平台使用的资源将从 DaoCloud 的官方仓库进行获取。

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
  creationTimestamp: null
spec:
  clusterName: my-cluster
  masterNodes:
    - nodeName: "g-master1"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
  workerNodes:
```

通过命令行生成 clusterConfig 配置文件模板

全模式 1 节点模式

```
# 官方在线
./dce5-installer generate-config --install-mode=cluster-create --master=1 --access-type=official-service
# 官方在线简化版
./dce5-installer generate-config --master=1

# 内建离线
./dce5-installer generate-config --install-mode=cluster-create --master=1 --access-type=builtin
# 内建离线简化版
./dce5-installer generate-config --master=1 --access-type=builtin

# 扩展离线
./dce5-installer generate-config --install-mode=cluster-create --master=1 --access-type=external
```

```
# 扩展离线简化版
./dce5-installer generate-config --master=1 --access-type=external
```

全模式 4 节点模式

```
# 官方在线
./dce5-installer generate-config --install-mode=cluster-create --master=3 --access-type=official-service
# 官方在线简化版
./dce5-installer generate-config --master=3

# 内建离线
./dce5-installer generate-config --install-mode=cluster-create --master=3 --access-type=builtin
# 内建离线简化版
./dce5-installer generate-config --master=3 --access-type=builtin

# 扩展离线
./dce5-installer generate-config --install-mode=cluster-create --master=3 --access-type=external
# 扩展离线简化版
./dce5-installer generate-config --master=3 --access-type=external
```

全模式 7 节点模式

```
# 官方在线
./dce5-installer generate-config --install-mode=cluster-create --master=3 --worker=3 --access-type=official-service
# 官方在线简化版
./dce5-installer generate-config --master=3 --worker=3

# 内建离线
./dce5-installer generate-config --install-mode=cluster-create --master=3 --worker=3 --access-type=builtin
# 内建离线简化版
./dce5-installer generate-config --master=3 --worker=3 --access-type=builtin

# 扩展离线
./dce5-installer generate-config --install-mode=cluster-create --master=3 --worker=3 --access-type=external
# 扩展离线简化版
./dce5-installer generate-config --master=3 --worker=3 --access-type=external
```

社区版

```
# 官方在线
./dce5-installer generate-config --install-mode=install-app --access-type=official-service

# 官方在线简化版
./dce5-installer generate-config --install-mode=install-app

# 内建离线
./dce5-installer generate-config --install-mode=install-app --access-type=builtin

# 扩展离线
./dce5-installer generate-config --install-mode=install-app --access-type=external
```

产品清单文件 manifest.yaml

此 YAML 文件包含了 DCE 5.0 所有模块信息，主要分为基础设施模块、产品功能模块。

升级操作请参考文档 升级 DCE 5.0。

Manifest 示例

以下是一个 Manifest 文件示例。

manifest.yaml

```
apiVersion: manifest.daocloud.io/v1alpha1
kind: DCEManifest
metadata:
global:
  helmRepo: https://release.daocloud.io/chartrepo
  imageRepo: release.daocloud.io
infrastructures:
  hwameiStor:
    enable: true
    version: v0.10.4
    policy: drbd-disabled
  istio:
    version: 1.16.1
  metallb:
    version: 0.13.9
  contour:
    version: 10.2.2
    enable: false
```

```
    cert-manager:
      version: 1.11.0
      enable: false
    mysql:
      version: 8.0.29
      cpuLimit: 1
      memLimit: 1Gi
      enableAutoBackup: true
    redis:
      version: 6.2.6-debian-10-r120
      cpuLimit: 400m
      memLimit: 500Mi
components:
    kubean:
      enable: true
      helmVersion: v0.6.6
      helmRepo: https://kubean-io.github.io/kubean-helm-chart
      variables:
    ghippo:
      enable: true
      helmVersion: 0.18.0
      variables:
    kpanda:
      enable: true
      helmVersion: 0.19.0+rc4
      variables:
    kcoral:
      enable: true
      helmVersion: 0.4.0+rc1
      variables:
    kcollie:
      enable: true
      helmVersion: 0.4.0+rc7
      variables:
    insight:
      enable: true
      helmVersion: 0.18.0-rc5
      variables:
    insight-agent:
      enable: true
      helmVersion: 0.18.0-rc5
      features: tracing
    ipavo:
      enable: true
```

```
    helmVersion: 0.10.0
    variables:
kairship:
    enable: true
    helmVersion: 0.10.1
    variables:
amamba:
    enable: true
    helmVersion: 0.18.0+alpha.3
    features: argocd
jenkins:
    enable: true
    helmVersion: 0.1.12
    helmRepo: https://release.daocloud.io/chartrepo/amamba
skoala:
    enable: true
    helmVersion: 0.23.0
    variables:
mspider:
    enable: true
    helmVersion: v0.18.0-rc2
    variables:
mcamel-rabbitmq:
    enable: true
    helmVersion: 0.12.0-rc2
    variables:
mcamel-elasticsearch:
    enable: true
    helmVersion: 0.9.0-rc2
    variables:
mcamel-mysql:
    enable: true
    helmVersion: 0.10.0-rc2
    variables:
mcamel-redis:
    enable: true
    helmVersion: 0.9.0-rc2
    variables:
mcamel-kafka:
    enable: true
    helmVersion: 0.7.0-rc2
    variables:
mcamel-minio:
    enable: true
```

```
    helmVersion: 0.7.0-rc2
    variables:
  mcamel-postgresql:
    enable: true
    helmVersion: 0.3.0-rc2
    variables:
  mcamel-mongodb:
    enable: true
    helmVersion: 0.1.0-rc1
    variables:
  spidernet:
    enable: true
    helmVersion: 0.8.0
    variables:
  kangaroo:
    enable: true
    helmVersion: 0.9.0
    variables:
  gmagpie:
    enable: true
    helmVersion: 0.3.0
    variables:
  dowl:
    enable: true
    helmVersion: 0.3.0+rc1
```

关键字段

该 YAML 文件中的关键字段说明，请参阅下表。其中包含了基础设施所涉及的组件以及产品功能模块涉及的产品。

字段	说明
infrastructures	DCE 5.0 产品基础设施
infrastructures.xxx.enable	是否开启当前模块，默认为 true
infrastructures.xxx.helmVersion	当前模块的 chart 包版本
infrastructures.hwameiStor	HwameiStor 本地存储

字段	说明
infrastructures.istio	Istio 服务网格
infrastructures.metallb	MetalLB 负载均衡器
infrastructures.contour	Contour 入口控制器
infrastructures.cert-manager	Cert Manager 证书管理
infrastructures.mysql	Mysql 数据库
infrastructures.redis	Redis 数据库
components	DCE 5.0 产品功能模块
components.kubean	集群声明周期管理
components.ghippo	全局管理
components.kpanda	容器管理
components.kcoral	应用备份
components.kcollie	集群巡检
components.insight	可观测性
components.insight-agent	可观测性的数据采集组件
components.ipavo	仪表盘
components.kairship	多云编排
components.amamba	应用工作台
components.jenkins	应用工作台的流水线引擎组件
components.skoala	微服务引擎

字段	说明
components.mspider	服务网格
components.mcamel-*	中间件，包含了 ES、Kafka、MinIO 等
components.kangaroo	镜像仓库
components.gmagpie	报表
components.dowl	集群安全
components.kant	边缘计算
components.virtnest	虚拟机
components.kolm	olm 管理
components.baize	AI Lab

部署要求

部署 DCE 5.0 时需要先做好软件规划、硬件规划、网络规划。

操作系统要求

架构	操作系统	测试 OS、Kernel 信息	备注（安装指导文档）
AMD 64	CentOS 7.X	CentOS 7.9 3.10.0-1127.el7.x86_64 on an x86_64	离线安装 DCE 5.0 商业版 注意：CentOS 7 在 2024 年 6 月 30 日结束支持
	Redhat 8.X	Redhat 8.4 4.18.0-305.el8.x86_64	离线安装 DCE 5.0 商业版
	Redhat 7.X	Redhat 7.9 3.10.0-1160.e17.x86	离线安装 DCE 5.0 商业版
	Redhat 9.X	Redhat 9.2 5.14.0-284.11.1.e9_2.x86_64	离线安装 DCE 5.0 商业版

架构	操作系统	测试 OS、Kernel 信息	备注（安装指导文档）
	Ubuntu 20.04	5.10.104	离线安装 DCE 5.0 商业版
	Ubuntu 22.04	5.15.0-78-generic	离线安装 DCE 5.0 商业版
	Rocky Linux 9.2	5.14.0-284.11.1.el9_2.x86_64	离线安装 DCE 5.0 商业版
	统信 UOS V20（1020a）	5.4.0-125-generic	UOS V20 (1020a) 上部署 DCE 5.0 商业版
	openEuler 22.03	5.10.0-60.18.0.50.oe2203.x86_64	离线安装 DCE 5.0 商业版
	Oracle Linux R9/R8 U1	5.15.0-3.60.5.1.el9uek.x86_64	Oracle Linux R9 U1 上部署 DCE 5.0 商业版
	TencentOS Server 3.1	5.4.119-19.0009.14	TencentOS Server 3.1 上部署 DCE 5.0 商业版
ARM 64	银河麒麟 OS V10 SP2	4.19.90-24.4.v2101.ky10.aarch64	离线安装 DCE 5.0 商业版
	银河麒麟 OS V10 SP3	4.19.90-89.11.v2401.ky10.aarch64	离线安装 DCE 5.0 商业版

Note

- CentOS 7 将在 2024 年 6 月 30 日结束支持，企业需根据自身情况进行替换操作系统
- 上述的操作系统、内核均为测试人员使用的版本
- 若非上表中声明的操作系统，请参考文档其他 Linux 离线部署 DCE 5.0 商业版进行安装部署。

内核要求

由于一些组件或功能对操作系统的内核版本有要求，部署前请根据实际情况参照下表来选择内核版本：

组件/功能	内核版本要求
容器管理 GPU 能力	≥ 5.15
Cilium	≥ 5.12
HwameiStor DRBD 能力	DRBD 适配的内核版本
Kubevirt	> 4.11
Merbridge 要求	≥ 5.7

⚠ 内核注意事项

1. 内核版本小于 5.9 时，kube-proxy 使用 ipvs 模式会造成通过 Service 方式访问集群内部服务， 偶现 1 秒延时或者后端业务升级后访问 Service 失败的情况，详见 Kubernetes 社区 Issue #81775，可以采用以下方式解决办法：
    - 升级内核至 5.9 及以上
    - 切换 kube-proxy 模式为 iptables
    - 内核参数更新：
      net.ipv4.vs.conntrack=1 + net.ipv4.vs.conn_reuse_mode=0 + net.ipv4.vs.expire_nodest_conn=1
2. Ubuntu 内核自动更新升级，可能导致系统在不经意间被重启，若使用的软件依赖于特定版本的内核， 那么当系统自动更新到新的内核版本时，可能会出现兼容性问题

硬件要求

CPU、内存和硬盘

类型	具体要求
CPU	不得超售
内存	不得超售
硬盘	IOPS > 500，吞吐量 > 200 MB/s

新手尝鲜模式 CPU、内存、硬盘要求¶

参阅新手尝鲜模式说明。

数量	服务器角色	服务器用途	cpu	内存	系统硬盘	未分区的硬盘
1	all in one	镜像仓库、chart museum、全局服务集群本身	24 核	32 GB	300 GB	400 GB

4 节点模式 CPU、内存、硬盘要求

参阅 4 节点模式说明。

数量	服务器角色	服务器用途	cpu 数量	内存容量	系统硬
1	火种节点	1. 执行安装部署程序 2. 运行平台所需的镜像仓库和 chart museum	2 核	4 GB	200 GB
3	控制面	1. 运行 DCE 5.0 组件 2. 运行 kubernetes 系统组件	8 核	16 GB	100 GB

7 节点模式(生产环境推荐) CPU、内存、硬盘要求¶

参阅 7 节点模式说明。

数量	服务器角色	服务器用途	cpu 数量	内存容量	系统硬
1	火种节点	1. 执行安装部署程序 2. 运行平台所需的镜像仓库和 chart museum	2 核	4 GB	200 GB
3	master	1. 运行 DCE 5.0 组件 2. 运行 kubernetes 系统组件	8 核	16 GB	100 GB
3	worker	单独运行日志相关组件	8 核	16 GB	100 GB

etcd 磁盘建议

由于 etcd 将数据写入磁盘并在磁盘上持久化，所以其性能取决于磁盘性能。并且 etcd 对磁盘的写延迟非常敏感，所以一些延迟问题可能会导致 etcd 丢失心跳。

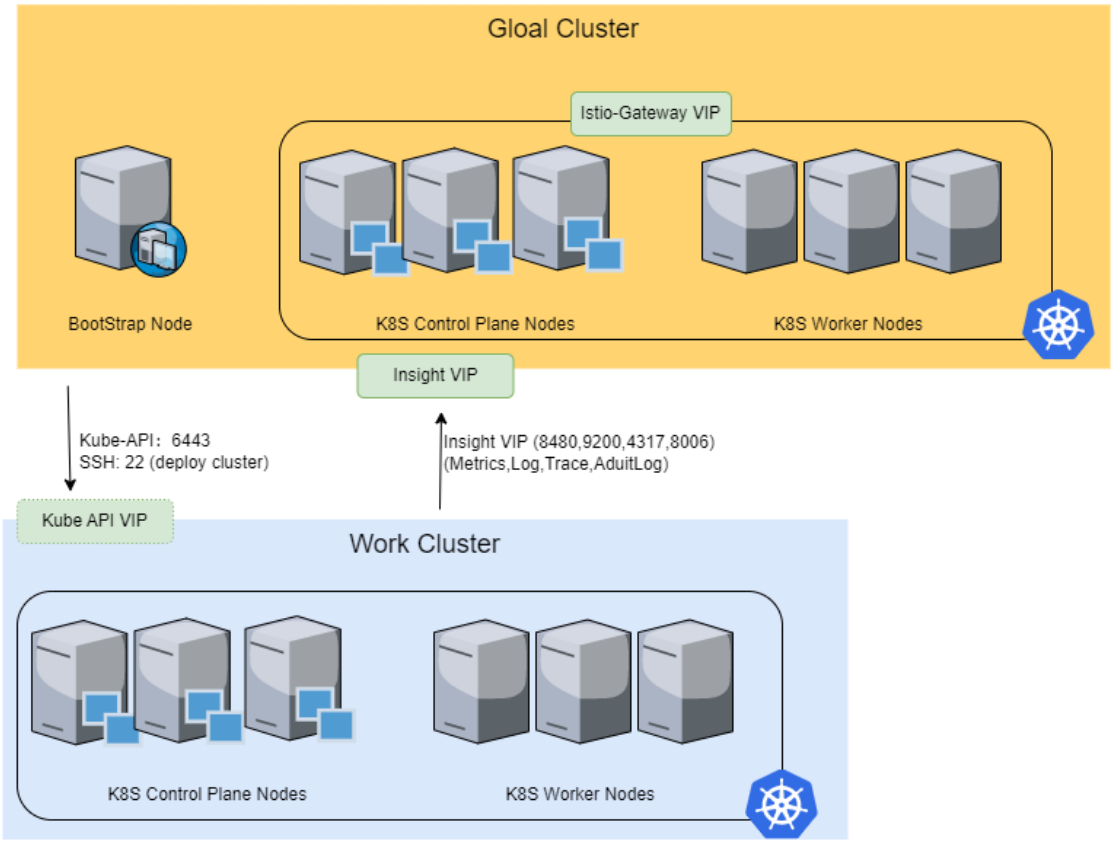
建议：

- 尽量在低延迟和高吞吐量的 SSD 或 NVMe 磁盘支持的机器上运行 etcd
- 使用固态硬盘作为最低选择。在生产环境中首选 NVMe 驱动器。

网络要求

网络拓扑

假设使用 VIP 作为全局服务集群的负载均衡方式：



网络要求

资源	要求	说明
istioGatewayVip	1 个	如果负载均衡模式是 metallb，则需要指定一个 VIP，供给 DCE 的 UI 界面和 OpenAPI 访问入口。
insightVip	1 个	如果负载均衡模式是 metallb，则需要指定一个 VIP，供给全局服务集群的 Insight 数据收集入口使用，子集群的 insight-agent 可上报数据到这个 VIP。
网络速率	1000 M/s	不低于千兆，建议万兆
协议	-	支持 IPv6
保留 IP 地址段	需保留两段	供 Pod（默认 10.233.64.0/18）和 Service （默认 10.233.0.0/18 使用）。如果已经在使用了，可以自定义其他网段避免 IP 地址冲突。

资源	要求	说明
路由	-	服务器有 default 或指向 0.0.0.0 这个地址的路由。
NTP 服务地址	1~4 个	确保您的数据中心有可以访问的 NTP 服务器 IP 地址。
DNS 服务地址	1~2 个	如果您的应用需要 DNS 服务，请准备可以访问 DNS 服务器 IP 地址。

客户端浏览器的要求

- Firefox ≥ v49
- Chrome ≥ v54

前置检查

本页说明了部署 DCE 5.0 需要进行的准备工作。

目前安装器的脚本中仅会针对火种机器进行前置检查，主要包含了是否已经安装前置依赖工具，及当前火种的 CPU > 10Core、Memory > 12G、disk > 100GB

机器检查

检查项	具体要求	说明
用户权限	root	必须使用 root 用户部署，各个服务器也必须允许 root 用户 ssh 登录
swap	关闭	如果不满足，系统会有一定几率出现 io 飙升，造成 容器运行时 卡死
防火墙	关闭（不强制）	-
selinux	关闭（不强制）	-
时间同步	所有集群节点要求时间必须同步	这是 Docker 和 Kubernetes 官方要求。否则 kube.conf 会报错 Unabl to connect to the server: x509: certificate has expired or is not yet
时区	所有服务器时区必须统一	建议设置为 Asia/Shanghai。 参考命令：timedatectl set-timezone Asia/Shanghai

检查项	具体要求	说明
Nameserver	/etc/resolv.conf 至少有一个 Nameserver	CoreDNS 要求，否则会有报错。该 nameserver 在纯离线环境下可以是一个不存在的 IP 地址。Centos8minial 默认没有 /etc/resolv 文件，需要手动创建
协议	支持 ipv6	火种节点使用 podman 时必须开启 ipv6

火种机器依赖组件检查

检查项	版本要求	说明
podman	v4.4.4	-
helm	≥ 3.11.1	-
skopeo	≥ 1.11.1	-
kind	v0.19.0	-
kubectl	≥ 1.25.6	-
yq	≥ 4.31.1	-
minio client	mc.RELEASE.2023-02-16T19-20-11Z	

如果不存在依赖组件，通过脚本进行安装依赖组件，安装前置依赖。

```
export VERSION=v0.16.0
# 下载脚本
curl -LO https://qiniu-download-
public.daocloud.io/DaoCloud_Enterprise/dce5/install_prerequisite_${VERSION}.sh

# 添加可执行权限
chmod +x install_prerequisite_${VERSION}.sh

# 开始安装
bash install_prerequisite_${VERSION}.sh online full
```

端口要求

为了正常运行，需要打开一些端口。如果网络配置了防火墙规则，则需要确保基础设施组件可以通过特定端口相互通信。 确保所需的以下端口在网络上处于打开状态，并配置为允许主机之间的访问。某些端口是根据配置和用途可选的。

火种节点

协议	端口	描述
TCP	443	Docker Registry
TCP	8081	Chart Museum
TCP	9000	Minio API
TCP	9001	Minio UI

Kube 集群（包括全局服务集群和工作集群）¶

全局服务集群和工作集群都是通过 Kubean 部署的，因此他们需要打开同样的端口。 除了标准的 K8s 端口，对于 CNI 和部分网络组件，也需要打开端口。

K8s 控制平面

协议	端口	描述
TCP	2379	etcd client port
TCP	2380	etcd peer port
TCP	2381	etcd metric port
TCP	6443	kubernetes api

协议	端口	描述
TCP	10250	kubelet api
TCP	10257	kube-scheduler
TCP	10259	kube-controller-manager

全部 K8s 节点

集群中的每一个节点都需要打开。

协议	端口	描述
TCP	22	ssh for ansible
TCP	9100	node exporter (Insight-Agent)
TCP	10250	kubelet api
TCP	30000-32767	kube nodePort range

参考 Kubernetes 端口和协议文档。

Calico（默认）

默认使用 Calico 作为 CNI， **全部 K8s 节点** 都需要打开。

协议	端口	描述
TCP	179	Calico networking (BGP)
UDP	4789	Calico CNI with VXLAN enabled
TCP	5473	Calico CNI with Typha enabled

协议	端口	描述
UDP	51820	Calico with IPv4 Wireguard enabled
UDP	51821	Calico with IPv6 Wireguard enabled
IPENCAP / IPIP	-	Calico CNI with IPIP enabled

参考 Calico 网络要求文档。

MetalLB（默认）

当启用 MetalLB 建 VIP 时， **全部 K8s 节点** 都需要打开。

协议	端口	描述
TCP/UDP	7472	metallb metrics ports
TCP/UDP	7946	metallb L2 operating mode

Cilium（可选）

如果使用 Cilium 作为 CNI， **全部 K8s 节点** 都需要打开。

协议	端口	描述
TCP	4240	Cilium Health checks (cilium-health)
TCP	4244	Hubble server
TCP	4245	Hubble Relay
UDP	8472	VXLAN overlay
TCP	9962	Cilium-agent Prometheus metrics

协议	端口	描述
TCP	9963	Cilium-operator Prometheus metrics
TCP	9964	Cilium-proxy Prometheus metrics
UDP	51871	WireGuard encryption tunnel endpoint
ICMP	-	health checks

参考 Cilium 系统要求文档。

SpiderPool（可选）

如果使用 SpiderPool 作为 CNI， **全部 K8s 节点** 都需要打开。

协议	端口	描述
TCP	5710	SpiderPool Agent HTTP Server
TCP	5711	SpiderPool Agent Metrics
TCP	5712	SpiderPool Agent gops enabled
TCP	5720	SpiderPool Controller HTTP Server
TCP	5721	SpiderPool Controller Metrics
TCP	5722	SpiderPool Controller Webhook Port
TCP	5723	Spiderpool-CLI HTTP server port.
TCP	5724	SpiderPool Controller gops enabled

参考文档：spiderpool-controller 及 spiderpool-agent。

KubeVIP（可选）

当启用 KubeVIP 建 Kube API VIP 时，**全部 Control Plane 节点** 都需要打开。

协议	端口	描述
TCP	2112	kube-vip metrics ports

全局服务集群其他需要开放的端口

Istio-Gateway VIP

协议	端口	描述	使用方
TCP	80	Istio-Gateway HTTP	Web Browser or API Client
TCP	443	Istio-Gateway HTTPS	Web Browser or API Client

Insight VIP

协议	端口	描述	使用方
TCP	8480	Insight VIP for Metrics	所有节点
TCP	9200	Insight VIP for Log	所有节点
TCP	4317	Insight VIP for Trace	所有节点
TCP	8006	Insight VIP for AduitLog	所有节点

工作集群其他需要开放的端口

工作集群需要开放 K8s API 的 6443 端口，给到全局服务集群访问。如果要是用部署功能，还要开放 22 端口，给全局服务集群访问。

协议	端口	描述	使用方
TCP	22	各节点 SSH (for ansible)	全局服务集群
TCP	6443	K8s API 访问入口(如 VIP)	全局服务集群

其他各产品需要开放的端口

镜像仓库

协议	端口	描述	使用方
TCP	443	访问入口(如 VIP) 的端口	所有节点

使用外接服务存储 Binaries 资源

本文描述如何使用第三方存储服务存储 Binaries 资源并且在安装器安装时进行指定。 支持两种类型：S3 兼容服务（如 minio）、非 S3 兼容服务（如 nginx）。

操作步骤

使用 S3 兼容服务

S3 兼容的服务只需要在 集群配置文件 clusterConfig.yaml 中简单配置即可，无需其他操作。

1. 配置 clusterConfig.yaml，设置 binaries 相关的参数，如下：

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
spec:
  ..........
  binaries:
    type: external
    externalRepoEndpoint: https://external-repo.daocloud.io
```

```
  externalRepoUsername: rootuser
  externalRepoPassword: rootpass123
.....
```

给定的用户名需要具有 bucket 的读写权限

使用非 S3 兼容服务

非 S3 兼容的服务需要先手动将下载好的镜像离线包 目录下的 offline/kubespray-binary/offline-files.tar.gz binaries 离线包导入， 然后在集群配置文件 clusterConfig.yaml 中配置相关参数。

以下内容以 CentOS 7.9 x86_64 作为集群节点，使用 nignx 作为 http server，理论上其他通用 http server 也能支持，需要注意 URL 访问路径和文件路径的映射关系。

1. 确保有一个可用的 nginx 服务，及服务所在节点的登录和文件写入权限；
2. 将 binaries 离线包从火种节点（<解压后离线包路径>/offline/kubespray-binary/offline-files.tar.gz, <解压后离线包路径>/offline/component-tools.tar.gz）拷贝至 nginx 服务所在节点；

   Note

   binaries 离线包路径为 ./offline/kubespray-binary/offline-files.tar.gz

3. 确定需要导入的路径;
   a. 通过 nginx.conf 检测 nginx 服务所在节点的文件路径和 URL 路径的映射关系，下方示例供参考：

```
http {
    server {
        listen          8080;
        server_name     _;
        location / {
            root    /usr/share/nginx/html;
            index   index.html index.htm;
        }
    }
}
```

   上方配置说明 nginx http 服务的访问根路径映射本地目录 /usr/share/nginx/html

如果是普通方式部署的 nginx 服务，则选定导入路径为 /usr/share/nginx/html

如果是容器部署的 nginx 服务，需要挂载宿主机路径至容器，且挂载的宿主机路径对应着映射了 http 服务的容器本地路径， 即存在这样的关系： http-path -> container-path -> host-path。则导入路径应为 host-path。host-path 需要手动按照附录确认。

4. 执行如下命令导入离线 binaries 离线包：

```
cat > import.sh << "EOF"
[ ! -d "${MAPPING_PATH}" ] && echo "mapping path ${MAPPING_PATH} not found"
&& exit 1
[ ! -f "${BINARIRES_PKG_PATH}" ] && echo "binaries package path
${BINARIRES_PKG_PATH} not found" && exit 1
[ ! -f "${COMPONENT_TOOLS_PATH}" ] && echo "comonent-tools package path
${COMPONENT_TOOLS_PATH} not found" && exit 1
tar -xzvf ${BINARIRES_PKG_PATH} --strip-components=1 -C ${MAPPING_PATH}
tar -xzvf ${COMPONENT_TOOLS_PATH} --strip-components=1 -C
${MAPPING_PATH}
EOF
export MAPPING_PATH="/usr/share/nginx/html"
export BINARIRES_PKG_PATH="./offline-files.tar.gz"
export COMPONENT_TOOLS_PATH="./component-tools.tar.gz"
bash ./import.sh
```

其中环境变量 MAPPING_PATH 代表步骤 3 中提及的导入路径。

5. 在 集群配置文件 clusterConfig.yaml 中，配置 binaries 相关的参数。

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
spec:
  ..........
  binaries:
    type: external
    externalRepoEndpoint: http://10.0.1.1:8080
  ..........
```

6. 完成上述配置后，可以继续执行部署 DCE 5.0 商业版。

附录

查看容器卷挂载列表：

CLI tool	Command
docker	docker inspect ${CONTAINER_ID} -f '{{range .Mounts}}{{printf "hostPath: %s containerPath: %s\n" .Source .Destination}}{{end}}'
nerdctl	nerdctl inspect ${CONTAINER_ID} -f '{{range .Mounts}}{{printf "hostPath: %s containerPath: %s\n" .Source .Destination}}{{end}}'
podman	podman inspect ${CONTAINER_ID} -f '{{range .Mounts}}{{printf "hostPath: %s containerPath: %s\n" .Source .Destination}}{{end}}'
crictl	crictl inspect -o go-template --template '{{range .status.mounts}}{{printf "hostPath: %s containerPath: %s\n" .hostPath .containerPath }}{{end}}' ${CONTAINER_ID}
ctr	ctr c info ${CONTAINER_ID} --spec 检查 mounts 字段
kubectl	kubectl -n ${NAMESPACE} get pod ${POD_NAME} -oyaml 检查 volumes 和 volumeMounts 字段

使用外接镜像仓库与 Chart 仓库存储镜像与 Chart 包

以下说明如何使用第三方镜像仓库及 Chart 仓库。

1. 在集群配置文件 clusterConfig.yaml 中，配置 imagesAndCharts 参数：

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
spec:
  ..........
  imagesAndCharts:
    type: external

    externalImageRepo: https://external-registry.daocloud.io
    externalImageRepoUsername: admin
    externalImageRepoPassword: Harbor12345

    externalChartRepoType: chartmuseum
    externalChartRepo: https://external-charts.daocloud.io:8081
    externalChartRepoPassword: rootpass123
  ..........
```

- 镜像仓库基本支持社区中开源的所有镜像仓库类型
- Chart 仓库仅支持 chartmuseum、harbor、jfrog 三种

2. 完成上述配置后，可以继续执行部署 DCE 5.0 商业版。

使用外接中间件服务

本文描述如何使用第三方中间服务，包含：mysql、redis、elasticsearch、S3Storage。

使用外接数据库

前置说明

- DCE 5.0 产品模块中内置了 MySQL 数据库来存储数据，但是安装器支持了使用外接 MySQL、Kingbase、PostgreSQL 三种数据库外接的方式
- 下述示例脚本仅用于演示目的，实际应用中应该根据具体的需求进行修改，例如数据库名称、用户名、密码等， 并且可以将以下语句拆分至不同的 DBMS 执行

外接 MySql 操作步骤

1. 准备一个 MySQL 数据库，并且具有创建数据库、创建用户、授予权限的权限。
2. 连接到 MySQL 数据库，执行如下 SQL，完成 database、用户的创建并授予对应权限。

```
# ghippo apiserver
CREATE DATABASE ghippo CHARACTER SET utf8 COLLATE utf8_general_ci;
CREATE USER 'ghippo' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON ghippo.* TO 'ghippo';

# ghippo keycloak
CREATE DATABASE keycloak CHARACTER SET utf8 COLLATE utf8_general_ci;
CREATE USER 'keycloak' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON keycloak.* TO 'keycloak';

# ghippo audit
CREATE DATABASE audit CHARACTER SET utf8 COLLATE utf8_general_ci;
CREATE USER 'audit' IDENTIFIED BY 'password';
```

```
GRANT ALL PRIVILEGES ON audit.* TO 'audit';

# kpanda
CREATE DATABASE kpanda CHARACTER SET utf8 COLLATE utf8_general_ci;
CREATE USER 'kpanda' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON kpanda.* TO 'kpanda';

# set sort_buffer_size (used for clusterpedia)
SET GLOBAL sort_buffer_size=8*1024*1024;
SET SESSION sort_buffer_size=8*1024*1024;

# skoala
CREATE DATABASE skoala CHARACTER SET utf8 COLLATE utf8_general_ci;
CREATE USER 'skoala' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON skoala.* TO 'skoala';

# amamba
CREATE DATABASE amamba CHARACTER SET utf8 COLLATE utf8_general_ci;
CREATE USER 'amamba' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON amamba.* TO 'amamba';

# insight
CREATE DATABASE insight CHARACTER SET utf8 COLLATE utf8_general_ci;
CREATE USER 'insight' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON insight.* TO 'insight';

# ipavo
CREATE DATABASE ipavo CHARACTER SET utf8 COLLATE utf8_general_ci;
CREATE USER 'ipavo' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON ipavo.* TO 'ipavo';

# kcollie
CREATE DATABASE kcollie CHARACTER SET utf8 COLLATE utf8_general_ci;
CREATE USER 'kcollie' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON kcollie.* TO 'kcollie';

# gmagpie
CREATE DATABASE gmagpie CHARACTER SET utf8 COLLATE utf8_general_ci;
CREATE USER 'gmagpie' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON gmagpie.* TO 'gmagpie';

# dowl
CREATE DATABASE dowl CHARACTER SET utf8 COLLATE utf8_general_ci;
CREATE USER 'dowl' IDENTIFIED BY 'password';
```

GRANT ALL PRIVILEGES ON dowl.* TO 'dowl';

在 集群配置文件 clusterConfig.yaml 中，配置 externalMiddlewares.database 参数， 假设数据库访问地址为 localhost:3306；不同的数据库类型有不同的 dataSourceName 配置格式， 参阅 GORM 文档连接到数据库。

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
  creationTimestamp: null
spec:
  ..............
  externalMiddlewares:
    database:
      kpanda:
        - dbDriverName: "mysql"
          # Please refer https://gorm.io/docs/connecting_to_the_database.html
          dataSourceName: "kpanda:password@tcp(localhost:3306)/dbname"
          # readwrite(default) or readonly
          accessType: readwrite
          # The maximum number of open connections to the database.
          # maxOpenConnections: 100
          # The maximum number of connections in the idle connection pool.
          # maxIdleConnections: 10
          # The maximum amount of time a connection may be reused.
          # connectionMaxLifetimeSeconds: 3600
          # The maximum amount of time a connection may be idle.
          # connectionMaxIdleSeconds: 1800
      ghippoApiserver:
        - dbDriverName: "mysql"
          dataSourceName: "ghippo:password@tcp(localhost:3306)/ghippo"
      ghippoKeycloak:
        - dbDriverName: "mysql"
          dataSourceName: "keycloak:password@tcp(localhost:3306)/keycloak"
      ghippoAuditserver:
        - dbDriverName: "mysql"
          dataSourceName: "audit:password@tcp(localhost:3306)/audit"
      skoala:
        - dbDriverName: "mysql"
          dataSourceName: "skoala:password@tcp(172.30.41.0:3308)/skoala"
      amamba:
        - dbDriverName: "mysql"
          dataSourceName: "amamba:password@tcp(172.30.41.0:3308)/amamba"
      insight:
```

```
    - dbDriverName: "mysql"
      dataSourceName: "insight:password@tcp(172.30.41.0:3308)/insight"
  ipavo:
    - dbDriverName: "mysql"
      dataSourceName: "ipavo:password@tcp(172.30.41.0:3308)/ipavo"
  kcollie:
    - dbDriverName: "mysql"
      dataSourceName: "kcollie:password@tcp(172.30.41.0:3308)/kcollie"
  gmagpie:
    - dbDriverName: "mysql"
      dataSourceName: "gmagpie:password@tcp(172.30.41.0:3308)/gmagpie"
  dowl:
    - dbDriverName: "mysql"
      dataSourceName: "dowl:password@tcp(172.30.41.0:3308)/dowl"
```

3. 完成上述配置后，可以继续执行部署 DCE 5.0 商业版。

外接 Kingbase 数据库操作步骤

1. 准备一个 Kingbase 数据库，并且具有创建数据库、创建用户、授予权限的权限。
2. 连接到 Kingbase 数据库，执行如下 SQL，完成 database、用户的创建并授予对应权限。

```
CREATE DATABASE ghippo;
CREATE USER ghippo WITH encrypted password 'password';
GRANT ALL PRIVILEGES ON DATABASE ghippo TO ghippo;

CREATE DATABASE keycloak;
CREATE USER keycloak WITH encrypted password 'password';
GRANT ALL PRIVILEGES ON DATABASE keycloak TO keycloak;

CREATE DATABASE audit;
CREATE USER audit WITH encrypted password 'password';
GRANT ALL PRIVILEGES ON DATABASE audit TO audit;

CREATE DATABASE kpanda;
CREATE USER kpanda WITH encrypted password 'password';
GRANT ALL PRIVILEGES ON DATABASE kpanda TO kpanda;

CREATE DATABASE skoala;
CREATE USER skoala WITH encrypted password 'password';
GRANT ALL PRIVILEGES ON DATABASE skoala TO skoala;
```

```
CREATE DATABASE amamba;
CREATE USER amamba WITH encrypted password 'password';
GRANT ALL PRIVILEGES ON DATABASE amamba TO amamba;

CREATE DATABASE insight;
CREATE USER insight WITH encrypted password 'password';
GRANT ALL PRIVILEGES ON DATABASE insight TO insight;

CREATE DATABASE ipavo;
CREATE USER ipavo WITH encrypted password 'password';
GRANT ALL PRIVILEGES ON DATABASE ipavo TO ipavo;

CREATE DATABASE kcollie;
CREATE USER kcollie WITH encrypted password 'password';
GRANT ALL PRIVILEGES ON DATABASE kcollie TO kcollie;

CREATE DATABASE gmagpie;
CREATE USER gmagpie WITH encrypted password 'password';
GRANT ALL PRIVILEGES ON DATABASE gmagpie TO gmagpie;

CREATE DATABASE dowl;
CREATE USER dowl WITH encrypted password 'password';
GRANT ALL PRIVILEGES ON DATABASE dowl TO dowl;
```

在 集群配置文件 clusterConfig.yaml 中，配置 externalMiddlewares.database 参数， 假设 Kingbase 数据库访问地址为 172.30.41.2:54321；不同的数据库类型有不同的 dataSourceName 配置格式， 详见文档 https://gorm.io/docs/connecting_to_the_database.html

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
  creationTimestamp: null
spec:
  ..............
  externalMiddlewares:
    database:
      kpanda:
        - dbDriverName: "kingbase"
          # Please refer https://gorm.io/docs/connecting_to_the_database.html
          dataSourceName: "host=172.30.41.2 user=kpanda password=password
dbname=kpanda port=54321"
          # readwrite(default) or readonly
          accessType: readwrite
          # The maximum number of open connections to the database.
```

```
            # maxOpenConnections: 100
            # The maximum number of connections in the idle connection pool.
            # maxIdleConnections: 10
            # The maximum amount of time a connection may be reused.
            #connectionMaxLifetimeSeconds: 3600
            # The maximum amount of time a connection may be idle.
            # connectionMaxIdleSeconds: 1800
        ghippoApiserver:
          - dbDriverName: "kingbase"
            dataSourceName: "host=172.30.41.2 user=ghippo password=password
dbname=ghippo port=54321"
        ghippoKeycloak:
          - dbDriverName: "kingbase"
            dataSourceName: "host=172.30.41.2 user=keycloak password=password
dbname=keycloak port=54321"
        ghippoAuditserver:
          - dbDriverName: "kingbase"
            dataSourceName: "host=172.30.41.2 user=audit password=password
dbname=audit port=54321"
        skoala:
          - dbDriverName: "kingbase"
            dataSourceName: "host=172.30.41.2 user=skoala password=password
dbname=skoala port=54321"
        amamba:
          - dbDriverName: "kingbase"
            dataSourceName: "host=172.30.41.2 user=amamba password=password
dbname=amamba port=54321"
        insight:
          - dbDriverName: "kingbase"
            dataSourceName: "host=172.30.41.2 user=insight password=password
dbname=insight port=54321"
        ipavo:
          - dbDriverName: "kingbase"
            dataSourceName: "host=172.30.41.2 user=ipavo password=password
dbname=ipavo port=54321"
        kcollie:
          - dbDriverName: "kingbase"
            dataSourceName: "host=172.30.41.2 user=kcollie password=password
dbname=kcollie port=54321"
        gmagpie:
          - dbDriverName: "kingbase"
            dataSourceName: "host=172.30.41.2 user=gmagpie password=password
dbname=gmagpie port=54321"
        dowl:
```

```
    - dbDriverName: "kingbase"
      dataSourceName: "host=172.30.41.2 user=dowl password=password
dbname=dowl port=54321"
```

3. 完成上述配置后，可以继续执行部署 DCE 5.0 商业版。

外接 PostgreSQL 数据库操作步骤

1. 准备一个 PostgreSQL 数据库，并且具有创建数据库、创建用户、授予权限的权限。
2. 连接到 PostgreSQL 数据库，执行如下 SQL，完成 database、用户的创建并授予对应权限。

```
CREATE USER ghippo WITH encrypted password 'password';
CREATE DATABASE ghippo OWNER ghippo;
GRANT ALL PRIVILEGES ON DATABASE ghippo TO ghippo;

CREATE USER keycloak WITH encrypted password 'password';
CREATE DATABASE keycloak OWNER keycloak;
GRANT ALL PRIVILEGES ON DATABASE keycloak TO keycloak;

CREATE USER audit WITH encrypted password 'password';
CREATE DATABASE audit OWNER audit;
GRANT ALL PRIVILEGES ON DATABASE audit TO audit;

CREATE USER kpanda WITH encrypted password 'password';
CREATE DATABASE kpanda OWNER kpanda;
GRANT ALL PRIVILEGES ON DATABASE kpanda TO kpanda;

CREATE USER skoala WITH encrypted password 'password';
CREATE DATABASE skoala OWNER skoala;
GRANT ALL PRIVILEGES ON DATABASE skoala TO skoala;

CREATE USER amamba WITH encrypted password 'password';
CREATE DATABASE amamba OWNER amamba;
GRANT ALL PRIVILEGES ON DATABASE amamba TO amamba;

CREATE USER insight WITH encrypted password 'password';
CREATE DATABASE insight OWNER insight;
GRANT ALL PRIVILEGES ON DATABASE insight TO insight;

CREATE USER ipavo WITH encrypted password 'password';
CREATE DATABASE ipavo OWNER ipavo;
GRANT ALL PRIVILEGES ON DATABASE ipavo TO ipavo;
```

```
CREATE USER kcollie WITH encrypted password 'password';
CREATE DATABASE kcollie OWNER kcollie;
GRANT ALL PRIVILEGES ON DATABASE kcollie TO kcollie;

CREATE USER gmagpie WITH encrypted password 'password';
CREATE DATABASE gmagpie OWNER gmagpie;
GRANT ALL PRIVILEGES ON DATABASE gmagpie TO gmagpie;

CREATE USER dowl WITH encrypted password 'password';
CREATE DATABASE dowl OWNER dowl;
GRANT ALL PRIVILEGES ON DATABASE dowl TO dowl;
```

1. 修改 clusterConfig.yaml 关于数据库的配置，假设数据库访问地址为 172.30.41.2:5432；不同的数据库类型有不同的 dataSourceName 配置格式，详见文档 https://gorm.io/docs/connecting_to_the_DATABASE.html

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
  creationTimestamp: null
spec:
  ..............
  externalMiddlewares:
    database:
      kpanda:
        - dbDriverName: "postgres"
          # Please refer https://gorm.io/docs/connecting_to_the_database.html
          dataSourceName: "host=172.30.41.2 user=kpanda password=password
dbname=kpanda port=5432"
          # readwrite(default) or readonly
          accessType: readwrite
          # The maximum number of open connections to the database.
          # maxOpenConnections: 100
          # The maximum number of connections in the idle connection pool.
          # maxIdleConnections: 10
          # The maximum amount of time a connection may be reused.
          # connectionMaxLifetimeSeconds: 3600
          # The maximum amount of time a connection may be idle.
          # connectionMaxIdleSeconds: 1800
      ghippoApiserver:
        - dbDriverName: "postgres"
          dataSourceName: "host=172.30.41.2 user=ghippo password=password
dbname=ghippo port=5432"
```

```
    ghippoKeycloak:
      - dbDriverName: "postgres"
        dataSourceName: "host=172.30.41.2 user=keycloak password=password
dbname=keycloak port=5432"
    ghippoAuditserver:
      - dbDriverName: "postgres"
        dataSourceName: "host=172.30.41.2 user=audit password=password
dbname=audit port=5432"
    skoala:
      - dbDriverName: "postgres"
        dataSourceName: "host=172.30.41.2 user=skoala password=password
dbname=skoala port=5432"
    amamba:
      - dbDriverName: "postgres"
        dataSourceName: "host=172.30.41.2 user=amamba password=password
dbname=amamba port=5432"
    insight:
      - dbDriverName: "postgres"
        dataSourceName: "host=172.30.41.2 user=insight password=password
dbname=insight port=5432"
    ipavo:
      - dbDriverName: "postgres"
        dataSourceName: "host=172.30.41.2 user=ipavo password=password
dbname=ipavo port=5432"
    kcollie:
      - dbDriverName: "postgres"
        dataSourceName: "host=172.30.41.2 user=kcollie password=password
dbname=kcollie port=5432"
    gmagpie:
      - dbDriverName: "postgres"
        dataSourceName: "host=172.30.41.2 user=gmagpie password=password
dbname=gmagpie port=5432"
    dowl:
      - dbDriverName: "postgres"
        dataSourceName: "host=172.30.41.2 user=dowl password=password
dbname=dowl port=5432"
```

2. 完成上述配置后，可以继续执行部署 DCE 5.0 商业版。

使用外接 Redis

操作步骤如下：

1. 在 集群配置文件 clusterConfig.yaml 中，配置 externalMiddlewares.redis 参数：

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
spec:
  ..........
  externalMiddlewares:
    redis:
      kpanda: "redis://:password@localhost:6379"
  ..........
```

   - 支持 Redis Standalone、Redis Sentinel、Redis Cluster 三种模式
   - Standalone URL 格式为：redis://[[user]:password@]host[:port][/db-number][?option=value]
   - Sentinel URL 格式为：redis+sentinel://[[user]:password@]host1[:port1][,host2[:port2]]/master-name[/db-number][?option=value]
   - Cluster URL 格式为：redis://[[user]:password@]host1[:port1]?addr=host2[:port2][&addr=host3:[port3][&option=value]] 或 rediss://[[user]:password@]host1[:port1]?addr=host2[:port2][&addr=host3:[port3][&option=value]]
   - 目前仅有容器管理产品模块使用到了 Redis 组件

2. 完成上述配置后，可以继续执行部署 DCE 5.0 商业版。

使用外接 Elasticsearch

使用外接 Elasticsearch 时需要注意：若外接 Elasticsearch 未开启 TLS， 则需要在 Insight 的 Helm 参数中 logging:output 中将 TLS 设置为 off。

操作步骤如下：

1. 在 集群配置文件 clusterConfig.yaml 中，配置 externalMiddlewares.elasticsearch 参数：

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

```
externalMiddlewares:
  elasticsearch:
    insight:
      endpoint: "https://xx.xx.xx.xx:9200"
      # basic auth
      username: "username"
      password: "password"
.....
```

目前仅有可观测产品模块使用到了 Elasticsearch 组件。 如果使用外接中间件后，不建议使用 7 节点模式下的 worker 节点，不然占用资源。

2. 完成上述配置后，可以继续执行部署 DCE 5.0 商业版。

使用外接 S3Storage

操作步骤如下：

1. 在 集群配置文件 clusterConfig.yaml 中，配置 externalMiddlewares.S3Storage 参数：

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
spec:
  .....
  externalMiddlewares:
    S3Storage:
      default:
        endpoint: "https://xx.xx.xx.xx:9200"
        # Set if you dont want to verify the certificate.
        insecure: true
        bucket: "bucketname"
        accessKey: "YOUR-ACCESS-KEY-HERE"
        secretKey: "YOUR-SECRET-KEY-HERE"
  .....
```

2. 完成上述配置后，可以继续执行部署 DCE 5.0 商业版。

使用外接 kafka

操作步骤如下：

1. 在 集群配置文件 clusterConfig.yaml 中，配置 externalMiddlewares.elasticsearch 参数：

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
spec:
  .........
  externalMiddlewares:
    kafka:
      brokers:
          - host1:9092
          - host2:9092
        # the username and password of kafka is not necessary
        username: "username"
        password: "password"
  .........
```

   目前仅有可观测产品模块使用到了 Kafka 组件。

2. 完成上述配置后，可以继续执行部署 DCE 5.0 商业版。

使用外接服务存储 OS Repo 资源

本文描述如何使用第三方存储服务的 OS Repo 资源并且在安装器安装时进行指定。支持两种类型：S3 兼容服务(如 minio)，非 S3 兼容服务(如 nginx)

前提条件

- 根据要部署的环境下载 ISO 操作系统镜像文件
- 根据要部署的环境下载 osPackage 离线包

操作步骤

使用 S3 兼容服务

S3 兼容的服务只需要在 集群配置文件 clusterConfig.yaml 中简单配置即可，无需其他操作。

1. 在 集群配置文件 clusterConfig.yaml 中，配置 osRepo 相关的参数：

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

```
kind: ClusterConfig
metadata:
spec:
.....
osRepos:
    type: external
    isoPath: "/root/CentOS-7-x86_64-DVD-2009.iso"
    osPackagePath: "/root/os-pkgs-centos7-v0.4.4.tar.gz"
    externalRepoEndpoint: https://external-repo.daocloud.io
    externalRepoUsername: rootuser
    externalRepoPassword: rootpass123
.....
```

给定的用户名需要具有 bucket 的读写权限。

使用非 S3 兼容服务

非 S3 兼容的服务需要先手动将下载好的 ISO 操作系统镜像文件、osPackage 离线包导入， 然后在集群配置文件 clusterConfig.yaml 中配置相关参数。

以下内容以 CentOS 7.9 x86_64 作为集群节点，使用 nignx 作为 http server，理论上其他通用 http server 也能支持，需要注意 URL 访问路径和文件路径的映射关系。

1. 确保有一个可用的 nginx 服务，及服务所在节点的登录和文件写入权限；
2. 下载/拷贝 ISO 操作系统镜像文件、osPackage 离线包至 nginx 服务所在节点，并将 ISO 导入脚本从火种节点拷贝至 nginx 服务所在节点；

   ISO 导入脚本在离线包镜像包中，路径为 ./offline/offline-iso/import_iso.sh

3. 确定需要导入的路径；

   通过 nginx.conf (nginx -t 命令查看改文件路径) 检测 nginx 服务所在节点的文件路径和 URL 路径的映射关系，下方示例供参考：

   ```
   http {
       server {
           listen          8080;
           server_name     _;
           location / {
               root    /usr/share/nginx/html;
   ```

```
            index    index.html index.htm;
        }
    }
}
```

上方配置说明 nginx http 服务的访问根路径映射本地目录 /usr/share/nginx/html。

- 如果是普通方式部署的 nginx 服务，则选定导入路径为 /usr/share/nginx/html。
- 如果是容器部署的 nginx 服务，需要挂载宿主机路径至容器，且挂载的宿主机路径对应着映射了 http 服务的容器本地路径，即存在这样的关系： http-path -> container-path -> host-path。则导入路径应为 host-path，host-path 需要手动按照附录 2 确认。

4. 执行如下命令导入 ISO 操作系统镜像文件、osPackage 离线包：

```
cat > import.sh << "EOF"
[ ! -d "${MAPPING_PATH}" ] && echo "mapping path ${MAPPING_PATH} not found"
&& exit 1
[ ! -x "${ISO_IMPORT_SH_PATH}" ] && echo "iso import script
${ISO_IMPORT_SH_PATH} not found or not executable" && exit 1
[ ! -f "${OS_PKGS_PATH}" ] && echo "os pkgs ${OS_PKGS_PATH} not found" && exit
1
[ ! -f "${ISO_PATH}" ] && echo "iso ${ISO_PATH} not found" && exit 1
tar -xzvf ${OS_PKGS_PATH} && for arch in amd64 arm64; do tar --strip-components=1 -
xzvf os-pkgs/os-pkgs-${arch}.tar.gz -C ${MAPPING_PATH}; done && rm -rf os-pkgs
bash ${ISO_IMPORT_SH_PATH} ${MAPPING_PATH} ${ISO_PATH}
EOF
export MAPPING_PATH="/usr/share/nginx/html"
export ISO_IMPORT_SH_PATH="./import_iso.sh"
export OS_PKGS_PATH="./os-pkgs-centos7-v0.4.5-rc3.tar.gz"
export ISO_PATH="./CentOS-7-x86_64-DVD-2009.iso"
bash ./import.sh
```

其中环境变量 MAPPING_PATH 代表步骤 3 中提及的导入路径

5. 验证是否导入成功

登录一台全局服务集群节点，假设 nignx 访问地址为 http://10.0.1.1:8080，参考附录 1 进行配置，执行如下命令：

```
cat > /etc/yum.repos.d/test.repo << "EOF"
[test0]
baseurl = http://10.1.1.1:8080/centos/$releasever/os/$basearch
gpgcheck = 0
name = test0

[test1]
```

```
baseurl = http://10.1.1.1:8080/kubean/centos-iso/$releasever/os/$basearch
gpgcheck = 0
name = test0
EOF
yum clean all && yum makecache --disablerepo=* --enablerepo=test0,test1
```

其他操作系统也是类似操作，因为具体的操作系统的包管理器的软件源配置有一些差异

6. 在 集群配置文件 clusterConfig.yaml 中，配置 osRepo 相关的参数，externalRepoURLs 参考附录 1。

```
apiVersion: provision.daocloud.io/v1alpha3
kind: ClusterConfig
metadata:
spec:
  ..........
  osRepos:
    type: external
    # `centos` as CentOS, RedHat,kylin AlmaLinux or Fedora,Openeuler
    # `debian` as Debian
    # `ubuntu` as Ubuntu
    externalRepoType: centos
    externalRepoURLs:
      - 'http://10.0.1.1:8080/centos/\$releasever/os/\$basearch/'
      - 'http://10.0.1.1:8080/centos-iso/\$releasever/os/\$basearch/'
  ..........
```

7. 完成上述配置后，可以继续执行部署 DCE 5.0 商业版。

附录

1. 操作系统与对应的 RepoURLs

${address_prefix} 替换为 HTTP 服务的外部访问地址，如 http://10.0.1.1:8080

OS	RepoURLs
CentOS	['${address_prefix}/centos/\$releasever/os/\$basearch','${address_prefix}/centos-iso/\$releasever/os/\$basearch']

OS	RepoURLs
RedHat	['${address_prefix}/redhat/\$releasever/os/\$basearch','${address_prefix}/redhat-iso/\$releasever/os/\$basearch/BaseOS','${address_prefix}/redhat-iso/\$releasever/os/\$basearch/AppStream']
Kylin V10	['${address_prefix}/kubean/kylin/\$releasever/os/\$basearch','${address_prefix}/kubean/kylin-iso/\$releasever/os/\$basearch']
UOS V20	['${address_prefix}/kubean/uos/\$releasever/os/\$basearch','${address_prefix}/kubean/uos-iso/\$releasever/os/\$basearch/AppStream','${address_prefix}/kubean/uos-iso/\$releasever/os/\$basearch/BaseOS']
Oracle 9	['${address_prefix}/kubean/oracle/\$releasever/os/\$basearch','${address_prefix}/kubean/oracle-iso/\$releasever/os/\$basearch/AppStream','${address_prefix}/kubean/oracle-iso/\$releasever/os/\$basearch/BaseOS']
OpenEuler 20.03	['${address_prefix}/kubean/openeuler/22.03/os/\$basearch','${address_prefix}/kubean/openeuler-iso/22.03/os/\$basearch']
Ubuntu bionic	['deb [trusted=yes] ${address_prefix}/kubean/ubuntu/amd64 bionic/','deb [trusted=yes] ${address_prefix}/kubean/ubuntu-iso bionic main restricted']
Ubuntu focal	['deb [trusted=yes] ${address_prefix}/kubean/ubuntu/amd64 focal/','deb [trusted=yes] ${address_prefix}/kubean/ubuntu-iso focal main restricted']

2. 查看容器卷挂载列表

CLI tool	Command
docker	docker inspect ${CONTAINER_ID} -f '{{range .Mounts}}{{printf "hostPath: %s containerPath: %s\n" .Source .Destination}}{{end}}'
nerdctl	nerdctl inspect ${CONTAINER_ID} -f '{{range .Mounts}}{{printf "hostPath: %s containerPath: %s\n" .Source .Destination}}{{end}}'

CLI tool	Command
podman	podman inspect ${CONTAINER_ID} -f '{{range .Mounts}}{{printf "hostPath: %s containerPath: %s\n" .Source .Destination}}{{end}}'
crictl	crictl inspect -o go-template --template '{{range .status.mounts}}{{printf "hostPath: %s containerPath: %s\n" .hostPath .containerPath }}{{end}}' ${CONTAINER_ID}
ctr	ctr c info ${CONTAINER_ID} --spec 检查 mounts 字段
kubectl	kubectl -n ${NAMESPACE} get pod ${POD_NAME} -oyaml 检查 volumes 和 volumeMounts 字段

离线安装 DCE 5.0 商业版

请在安装之前阅读并了解部署要求、部署架构、准备工作。

查阅安装器 Release Notes，避免所安装版本的已知问题，还可以从中查阅新增的功能特性。

第 1 步：下载离线包

请根据业务环境下载对应版本的离线包。

离线镜像包（必需）

离线镜像包包含安装 DCE 5.0 各个产品模块所需的配置文件、镜像资源以及 Chart 包。 可以在下载中心下载最新版本。

CPU 架构	版本	点击下载
AMD64	v0.26.0	offline-v0.26.0-amd64.tar
ARM64	v0.26.0	offline-v0.26.0-arm64.tar

下载完毕后解压离线包。以 amd64 架构离线包为例：

tar -xvf offline-v0.26.0-amd64.tar

ISO 操作系统镜像文件（必需）

对于 ISO 格式的操作系统镜像文件，在安装过程中请根据不同操作系统来下载对应的 ISO 文件。

ISO 操作系统镜像文件需要在集群配置文件 clusterConfig.yaml 中进行配置。

CPU 架构	操作系统版本	点击下载
AMD64	CentOS 7	CentOS-7-x86_64-DVD-2009.iso
	Redhat 7、8、9	assembly-field-downloads-page-content-61451 注意：Redhat 操作系统需要 Redhat 的账号才可以下载
	Ubuntu 20.04.6 LTS	ubuntu-20.04.6-live-server-amd64.iso
	Ubuntu 22.04.5 LTS	ubuntu-22.04.5-live-server-amd64.iso
	统信 UOS V20（1020a）	uniontechos-server-20-1020a-amd64.iso
	openEuler 22.03	openEuler-22.03-LTS-SP1-x86_64-dvd.iso
	Oracle Linux R9 U1	OracleLinux-R9-U1-x86_64-dvd.iso
	Oracle Linux R8 U7	OracleLinux-R8-U7-x86_64-dvd.iso
	Rocky Linux 9.2	Rocky-9.2-x86_64-dvd.iso
	Rocky Linux 8.10	Rocky-8.10-x86_64-dvd1.iso
ARM64	Kylin Linux Advanced Server release V10 (Sword) SP2	查看申请地址

CPU 架构	操作系统版本	点击下载
	Kylin Linux Advanced Server release V10 (Halberd) SP3	查看申请地址

- 建议所有操作系统以 Server 模式安装
- 麒麟操作系统需要提供个人信息才能下载使用，下载时请选择 V10 (Sword) SP2

osPackage 离线包（必需）

osPackage 离线包是 Kubean 这个开源项目为 Linux 操作系统离线软件源做的补充内容，例如 openEuler 22.03 中缺少了 selinux-policy-35.5-15.oe2203.noarch.rpm。

安装器从 v0.5.0 版本，需要提供操作系统的 osPackage 离线包，并在集群配置文件 clusterConfig.yaml 中定义 osPackagePath。

其中 Kubean 提供了不同操作系统的 osPackage 离线包， 可以前往 https://github.com/kubean-io/kubean/releases 查看。

目前安装器版本要求 osPackage 离线包的版本与之匹配，请根据对应版本下载 osPackage 离线包：

操作系统版本	点击下载
CentOS 7	os-pkgs-centos7-v0.21.1.tar.gz
Redhat 8	os-pkgs-redhat8-v0.21.1.tar.gz
Redhat 7	os-pkgs-redhat7-v0.21.1.tar.gz
Redhat 9	os-pkgs-redhat9-v0.21.1.tar.gz
Ubuntu 20.04	os-pkgs-ubuntu2004-v0.21.1.tar.gz

操作系统版本	点击下载
Ubuntu 22.04	os-pkgs-ubuntu2204-v0.21.1.tar.gz
openEuler 22.03	os-pkgs-openeuler22.03-v0.21.1.tar.gz
Oracle Linux R9 U1	os-pkgs-oracle9-v0.21.1.tar.gz
Oracle Linux R8 U7	os-pkgs-oracle8-v0.21.1.tar.gz
Rocky Linux 9.2	os-pkgs-rocky9-v0.21.1.tar.gz
Rocky Linux 8.10	os-pkgs-rocky8-v0.21.1.tar.gz
Kylin Linux Advanced Server release V10 (Sword) SP2	os-pkgs-kylinv10-v0.21.1.tar.gz
Kylin Linux Advanced Server release V10 (Halberd) SP3	os-pkgs-kylinv10sp3-v0.21.1.tar.gz

统信 UOS V20（1020a）osPackage 部署请参考 UOS V20 (1020a) 操作系统上部署 DCE 5.0。

Addon 离线包（可选）

Addon 离线包包含一些常用组件的 Helm Chart 离线包，具体清单请参考 Addon。

安装器从 v0.5.0 版本，支持了 Addon 的离线包导入能力，如果需要支持 Addon 中所有的 Helm Chart 离线化。 可以在下载中心下载最新版本。

首先需要事先下载好离线包，并在集群配置文件 clusterConfig.yaml 中定义 addonOfflinePackagePath。

一键下载所需离线包

我们提供了脚本来一键下载安装 DCE 5.0 所需的离线包。

以下是包含的离线包：

- 前置依赖工具离线包
- osPackage 离线包
- 安装器离线包

由于不同的 ISO 操作系统下载方式不一致，所以一键下载的离线包并不包含 ISO 文件。

第 2 步：配置 clusterConfig.yaml

这是集群配置文件，位于离线镜像包 offline/sample 目录下，具体的参数介绍请参考 clusterConfig.yaml。

目前离线镜像包中提供了标准的 7 节点模式模板。 使用 Redhat 9.2 操作系统部署时，需要开启内核调优参数 node_sysctl_tuning: true。

第 3 步：安装

1. 执行以下命令开始安装 DCE 5.0，安装器二进制文件位于 offline/dce5-installer。
2. ./offline/dce5-installer cluster-create -c ./offline/sample/clusterConfig.yaml -m ./offline/sample/manifest.yaml

   安装器脚本命令说明：

   - -c 来指定集群配置文件，必选
   - -m 参数指定 manifest 文件
   - -z 最小化安装
   - -d 开启 debug 模式
   - --use-original-repo 从源站下载可执行文件、拉取镜像等
   - 更多参数请使用 --help 查询
3. 安装完成后，命令行会提示安装成功。恭喜您！现在可以通过屏幕提示的 URL 使用默认的账号和密码（admin/changeme）探索全新的 DCE 5.0 啦！

请记录好提示的 URL，方便下次访问。

4. 成功安装 DCE 5.0 商业版之后，请联系我们授权：电邮 info@daocloud.io 或致电 400 002 6898。

在 OpenShift OCP 上安装 DCE 5.0 商业版

本文将介绍如何在 OCP 上安装 DCE 5.0。

前提条件

- DCE 5.0 默认支持的 Kubernetes 版本为 v1.22.x、v1.23.x、v1.24.x、v1.25.x、v1.26.x
- 已经拥有一个 OCP 环境，并且版本不低于 v1.22.x
- 准备一个私有镜像仓库，并且保证集群可以访问到
- 确保资源充足，建议集群至少还有 12 核 24 GB 的可用资源

离线安装

1. 通过堡垒机登录到 Control plane 节点。
2. 下载全模式离线包，可以在下载中心下载最新版本。

CPU 架构	版本	下载地址
AMD64	v0.26.0	https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.26.0-amd64.tar
ARM64	v0.26.0	https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.26.0-arm64.tar

3. 下载完毕后解压离线包：

```
# 以 amd64 架构离线包为例
tar -xvf offline-v0.26.0-amd64.tar
```

4. 设置集群配置文件 clusterConfig.yaml，可以在离线包 offline/sample 下获取该文件并按需修改。

   参考配置为：

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
  creationTimestamp: null
spec:
  loadBalancer:
    type: metallb # 建议 metallb
    istioGatewayVip: 10.5.14.XXX/32
    insightVip: 10.5.14.XXX/32
  fullPackagePath: /home/offline # 离线包目录
  imagesAndCharts:
    type: external
    externalImageRepo: http://10.5.14.XXX:XXXX # 私有镜像仓库地址
    externalImageRepoUsername: admin
    externalImageRepoPassword: Harbor12345
```

5. 配置 manifest 文件（可选），可以在离线包 offline/sample 下获取该文件并按需修改。

   如果要使用 hwameiStor 作为 StorageClass，请确保当前集群中有没有默认的 StorageClass。 如果有，需要去除。如果不去除，默认 StorageClass 需要关闭 hwameiStor，即更改 enable 的值为 fasle。

6. 开始安装 DCE 5.0。

```
./offline/dce5-installer install-app -m ./offline/sample/manifest.yaml -
c ./offline/sample/clusterConfig.yaml --platform openshift -z
```

   部分参数介绍，更多参数可以通过 ./dce5-installer --help 来查看：

- -z 最小化安装
- -c 指定集群配置文件，使用 NodePort 暴露控制台时不需要指定 -c
- -d 开启 debug 模式
- --platform 用来声明在哪个 Kubernetes 发行版上部署 DCE 5.0，目前仅支持 openshift
- --serial 指定后所有安装任务串行执行

7. 安装完成后，命令行会提示安装成功。恭喜您！ 现在可以通过屏幕提示的 URL 使用默认的账户和密码（admin/changeme）探索全新的 DCE 5.0 啦！

请记录好提示的 URL，方便下次访问。

8. 成功安装 DCE 5.0 商业版之后，请联系我们授权：电邮 info@daocloud.io 或致电 400 002 6898。

在阿里云 ECS 上安装 DCE 5.0 商业版

本文将介绍如何阿里云 ECS 上安装 DCE 5.0。

前提条件

- 准备好阿里云 ECS 虚拟机，本示例创建 3 台 ubuntu22.10 server 64bit 的云服务器，每台配置为 8 核 16 GB
- 请按照准备工作完成相关准备

开始部署

在阿里云 ECS 部署 DCE 5.0 时主要是负载均衡的能力需要特殊处理，由于虚拟机中不会安装 CloudProvider，LoadBalancer 类型的 svc 无法被识别，所以要么是手动安装相关 CloudProvider，要么使用用 NodePort 的方式，所以目前提供了以下三种方案：

- 方案 1：NodePort + 阿里云 SLB
- 方案 2：cloudLB + 部署 CCM 组件
- 方案 3：NodePort + 部署 CCM 组件

方案 1 ：NodePort + 阿里云 SLB

1. 登录一台机器，下载 dce5-installer 二进制文件。

   假定 VERSION 为 v0.19.0

   ```
   export VERSION=v0.19.0
   curl -Lo ./dce5-installer https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/dce5-installer-$VERSION
   chmod +x ./dce5-installer
   ```

2. 配置集群配置文件 clusterConfig.yaml。

   参考以下配置，注意设置 loadBalancer.type = NodePort，并填写主机的私网 IP 地址：

   clusterConfig.yaml

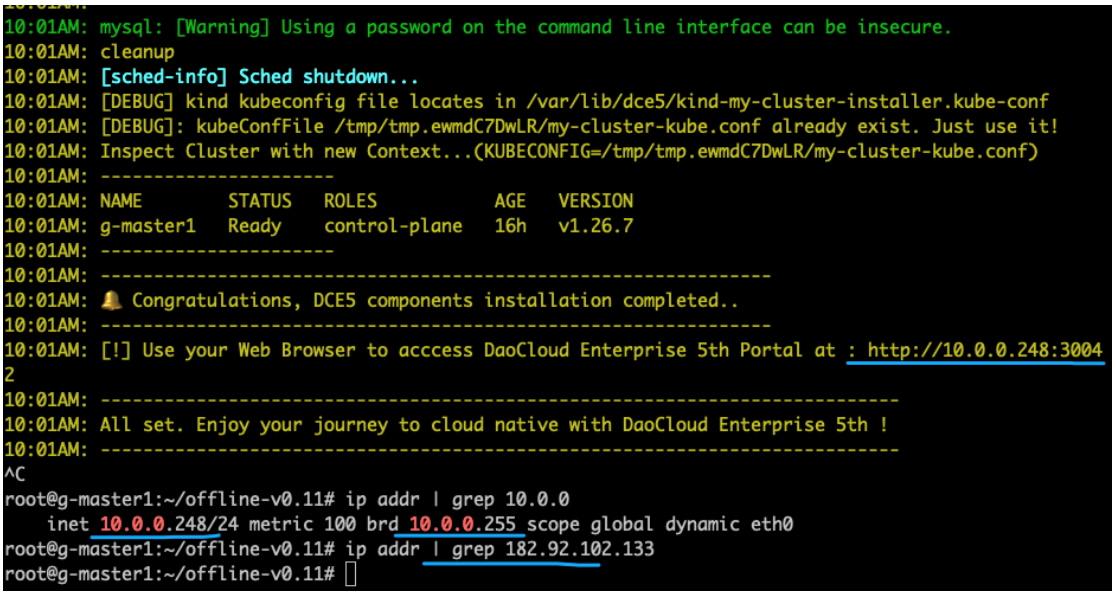
   ```
   apiVersion: provision.daocloud.io/v1alpha4
   kind: ClusterConfig
   spec:
     loadBalancer:
       type: NodePort
     masterNodes:
       - nodeName: "g-master1"
         ip: xx.xx.xx.xx
         ansibleUser: "root"
         ansiblePass: "dangerous"
       - nodeName: "g-master2"
         ip: xx.xx.xx.xx
   ```

```
    ansibleUser: "root"
    ansiblePass: "dangerous"
  - nodeName: "g-master3"
    ip: xx.xx.xx.xx
    ansibleUser: "root"
    ansiblePass: "dangerous"
```

3. 开始安装

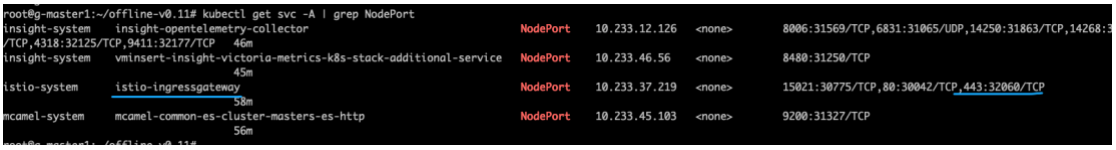
./dce5-installer cluster-create -c sample/clusterConfig.yaml

4. 安装成功

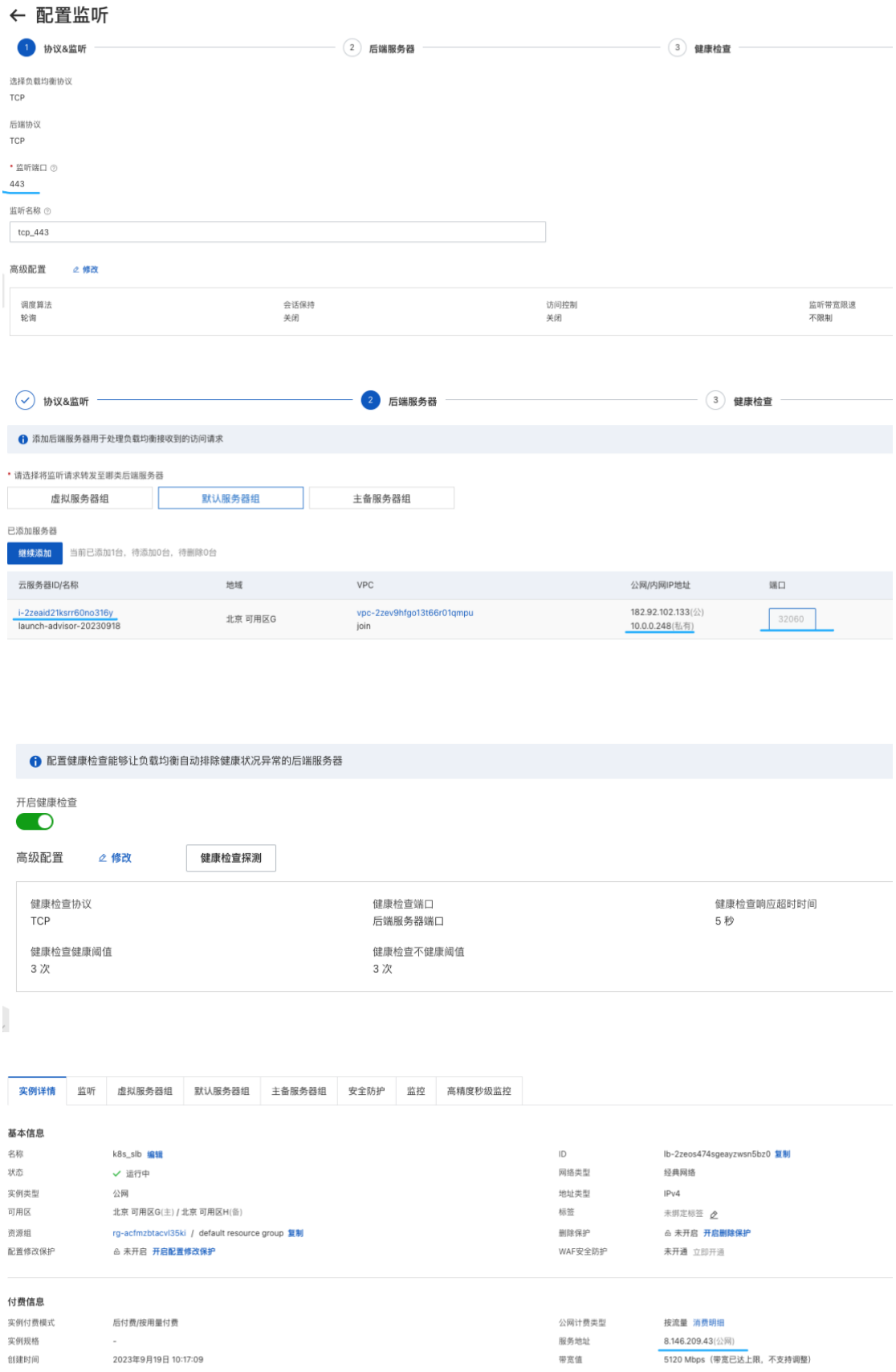


5. 查询 svc istio-ingressgateway 的 https 服务暴露在 NodePort 端口，本示例是 32060

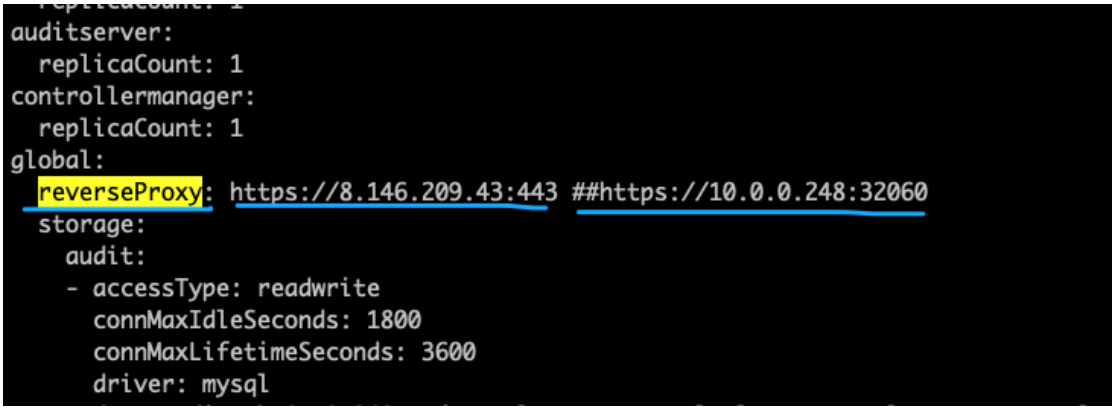
kubectl get svc -A | grep NodePort



6. 创建阿里 SLB ，将 SLB 的公网 TCP 流量指向 ECS 主机的 32060 端口，3 台主机均需要添加。


← 配置监听
1 协议&监听
2 后端服务器
3 健康检查
选择负载均衡协议
TCP
后端协议
TCP
* 监听端口
443
监听名称
tcp_443
高级配置
修改
调度算法
轮询
会话保持
关闭
访问控制
关闭
监听带宽限速
不限制
协议&监听
2 后端服务器
3 健康检查
添加后端服务器用于处理负载均衡接收到的访问请求
* 请选择将监听请求转发至哪类后端服务器
虚拟服务器组
默认服务器组
主备服务器组
已添加服务器
继续添加
当前已添加1台，待添加0台，待删除0台
云服务器ID/名称
地域
VPC
公网/内网IP地址
端口
i-2zeaid21ksrr60no316y
launch-advisor-20230918
北京 可用区G
vpc-2zev9hfgo13t66r01qmpu
join
182.92.102.133(公)
10.0.0.248(私有)
32060
配置健康检查能够让负载均衡自动排除健康状况异常的后端服务器
开启健康检查
高级配置
修改
健康检查探测
健康检查协议
TCP
健康检查端口
后端服务器端口
健康检查响应超时时间
5 秒
健康检查健康阈值
3 次
健康检查不健康阈值
3 次
实例详情
监听
虚拟服务器组
默认服务器组
主备服务器组
安全防护
监控
高精度秒级监控
基本信息
名称
k8s_slb 编辑
ID
lb-2zeos474sgeayzwsn5bz0 复制
状态
运行中
网络类型
经典网络
实例类型
公网
地址类型
IPv4
可用区
北京 可用区G(主) / 北京 可用区H(备)
标签
未绑定标签
资源组
rg-acfmzbtacvl35ki / default resource group 复制
删除保护
未开启 开启删除保护
配置修改保护
未开启 开启配置修改保护
WAF安全防护
未开通 立即开通
付费信息
实例付费模式
后付费/按用量付费
公网计费类型
按流量 消费明细
实例规格
-
服务地址
8.146.209.43(公网)
创建时间
2023年9月19日 10:17:09
带宽值
5120 Mbps（带宽已达上限，不支持调整）

7. 修改 ghippo 反向代理配置，参考文档 https://docs.daocloud.io/ghippo/install/reverse-proxy/#_1，修改后直接通过 SLB 的公网 IP +Port 访问 DCE 5.0。如下图：



方案 2 ： cloudLB + 部署 CCM 组件¶

1. 登录一台机器，下载 dce5-installer 二进制文件。

   假定 VERSION 为 v0.19.0

   ```
   export VERSION=v0.19.0
   curl -Lo ./dce5-installer https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/dce5-installer-$VERSION
   chmod +x ./dce5-installer
   ```

2. 配置集群配置文件 clusterConfig.yaml。

参考以下配置，注意设置 loadBalancer.type = cloudLB，并填写主机的私网 IP 地址：

clusterConfig.yaml

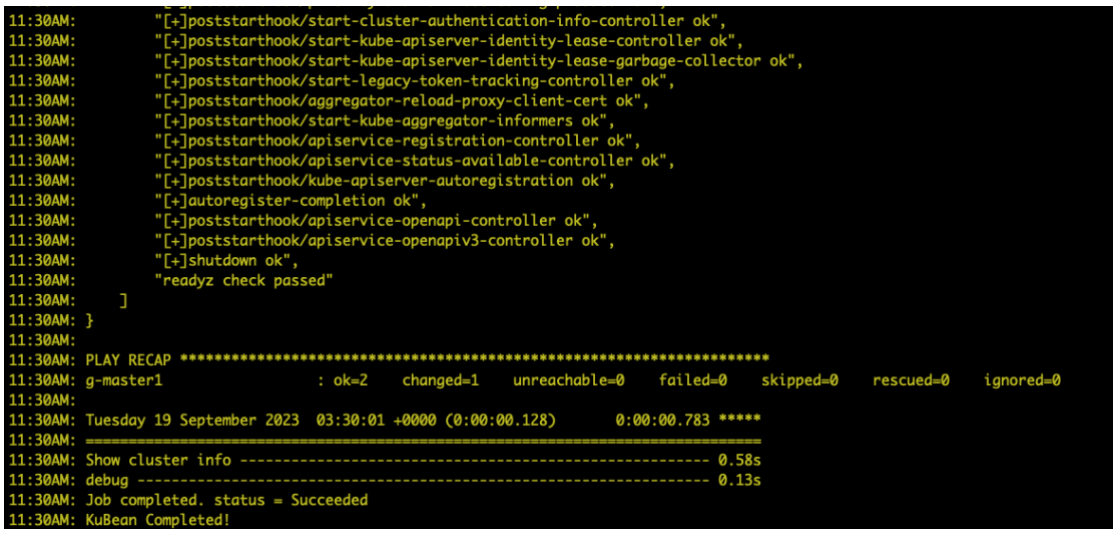
```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
spec:
  loadBalancer:
    type: cloudLB
  masterNodes:
    - nodeName: "g-master1"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
    - nodeName: "g-master2"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
    - nodeName: "g-master3"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
```

3. 开始安装，先部署集群

   通过 -j 参数指定 1,2,3,4,5,6 步骤，完成 k8s 集群的部署。

   ./dce5-installer cluster-create -c sample/clusterConfig.yaml -j 1,2,3,4,5,6

   成功后输出结果如下图：



4. 安装阿里云 CCM

   参考阿里云文档进行部署。

文件 ccm.yaml 中的 nodeSelector 参数需要改动为 node-role.kubernetes.io/control-plane: ""

安装成功后如下图：

```
root@g-master1:~# kubectl get ds -A
NAMESPACE       NAME                      DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR                          AGE
kruise-system   kruise-daemon             1         1         1       1            1           <none>                                 4h25m
kube-system     calico-node               1         1         1       1            1           kubernetes.io/os=linux                 4h32m
kube-system     cloud-controller-manager  1         1         1       1            1           node-role.kubernetes.io/control-plane=  113m
kube-system     kube-proxy                1         1         1       1            1           kubernetes.io/os=linux                 4h33m
root@g-master1:~#
```


```
root@g-master1:~# kubectl get ds -A
NAMESPACE       NAME                      DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR                          AGE
kruise-system   kruise-daemon             1         1         1       1            1           <none>                                 4h25m
kube-system     calico-node               1         1         1       1            1           kubernetes.io/os=linux                 4h32m
kube-system     cloud-controller-manager  1         1         1       1            1           node-role.kubernetes.io/control-plane=  113m
kube-system     kube-proxy                1         1         1       1            1           kubernetes.io/os=linux                 4h33m
root@g-master1:~#
```


5. 继续安装 DCE 5.0，将所有产品组件安装

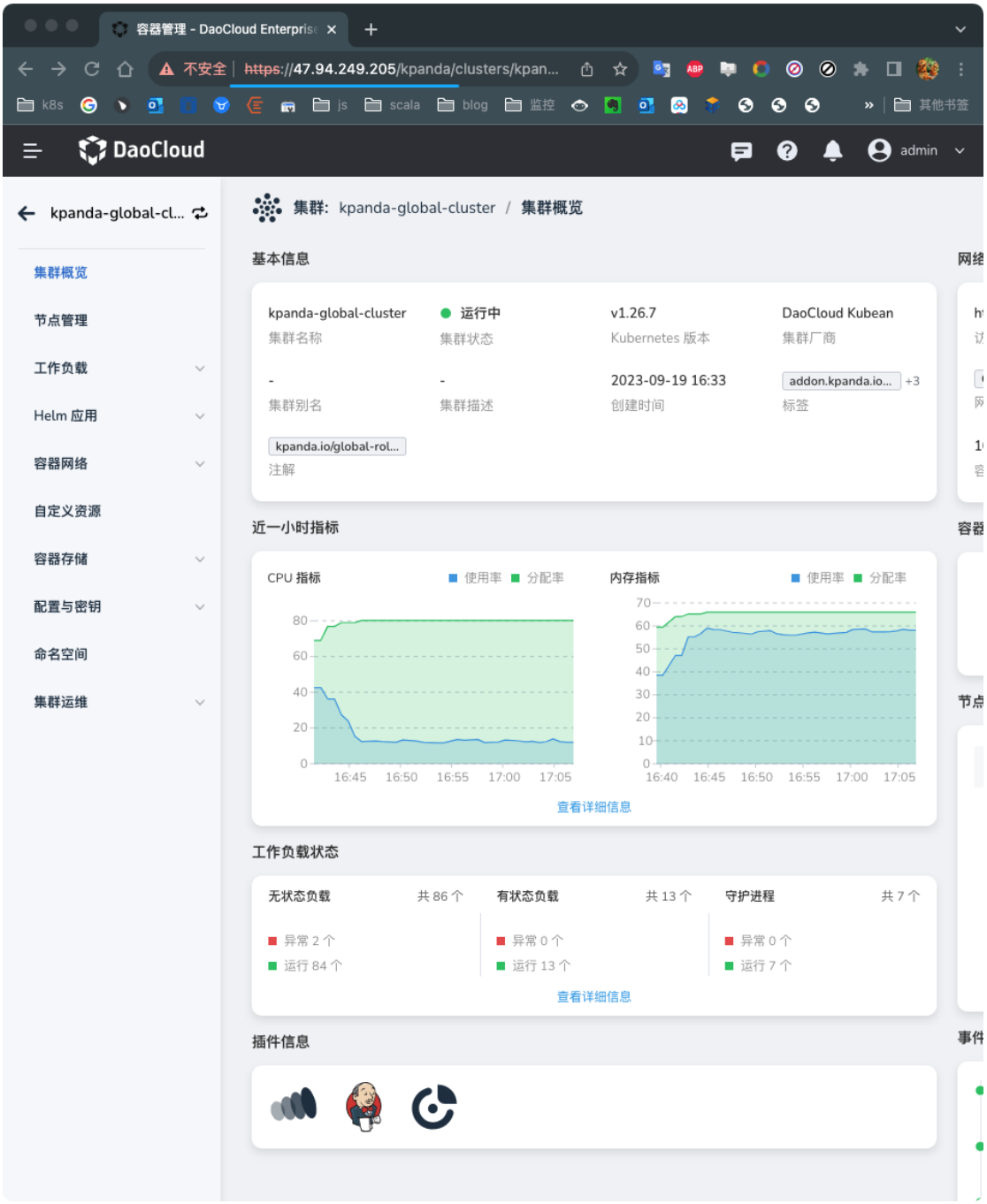
   通过 -j 参数指定 7+ 完成剩余步骤的执行。

   ./dce5-installer cluster-create -c sample/clusterConfig.yaml -j 7+

6. 安装成功后，会默认创建公网的 LB 实例，并且可基于分配的 IP 来访问 DCE 5.0。

```
4:44PM: Update Complete. ⎈Happy Helming!⎈
4:44PM: [helm repo update mcamel] successfully.
4:44PM:  -- Install Middleware mcamel-redis 0.11.0
4:44PM: helm successful.
4:44PM: [INFO]: time elasped : 1323 (s)
4:44PM:
4:44PM: mysql: [Warning] Using a password on the command line interface can be insecure.
4:44PM: cleanup
4:44PM: [sched-info] Sched shutdown...
4:44PM: [DEBUG] kind kubeconfig file locates in /var/lib/dce5/kind-my-cluster-installer.kube-conf
4:44PM: [DEBUG]: kubeConfFile /tmp/tmp.RJUOIEtsI3/my-cluster-kube.conf already exist. Just use it!
4:44PM: Inspect Cluster with new Context...(KUBECONFIG=/tmp/tmp.RJUOIEtsI3/my-cluster-kube.conf)
4:44PM: ---------------------
4:44PM: NAME        STATUS   ROLES           AGE     VERSION
4:44PM: g-master1   Ready    control-plane   5h16m   v1.26.7
4:44PM: ---------------------
4:44PM: ------------------------------------------------------------------
4:44PM:  Congratulations, DCE5 components installation completed..
4:44PM: ------------------------------------------------------------------
4:44PM: [!] Use your Web Browser to acccess DaoCloud Enterprise 5th Portal at : https://47.94.249.205
4:44PM: ------------------------------------------------------------------------------
4:44PM: All set. Enjoy your journey to cloud native with DaoCloud Enterprise 5th !
4:44PM: ------------------------------------------------------------------------------

gproductlicense.ghippo.io/kpanda created
gproductlicense.ghippo.io/kairship created
gproductlicense.ghippo.io/mcamel created
gproductlicense.ghippo.io/mspider created
gproductlicense.ghippo.io/skoala created
gproductlicense.ghippo.io/amamba created
gproductlicense.ghippo.io/ghippo created
gproductlicense.ghippo.io/kangaroo created
gproductlicense.ghippo.io/kant created
root@g-master1:~/offline-v0.11#
root@g-master1:~/offline-v0.11#
root@g-master1:~/offline-v0.11# kubectl get svc -A | grep -i load
default          nginx                                        LoadBalancer   10.233.15.219   47.94.243.224   80:31041/TCP
insight-system   lb-insight-opentelemetry-collector           LoadBalancer   10.233.43.120   39.106.24.108   4317:31008/TCP,8006:31495/TCP
insight-system   lb-vminsert-insight-victoria-metrics-k8s-stack LoadBalancer   10.233.43.231   182.92.167.164  8480:30579/TCP
istio-system     istio-ingressgateway                         LoadBalancer   10.233.27.109   47.94.249.205   15021:31366/TCP,80:32078/TCP,443:30685/TCP
root@g-master1:~/offline-v0.11#
```



方案 3：NodePort + 部署 CCM 组件

1. 登录一台机器，下载 dce5-installer 二进制文件。

   假定 VERSION 为 v0.19.0

   ```
   export VERSION=v0.19.0
   curl -Lo ./dce5-installer https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/dce5-installer-$VERSION
   chmod +x ./dce5-installer
   ```

2. 配置集群配置文件 clusterConfig.yaml。

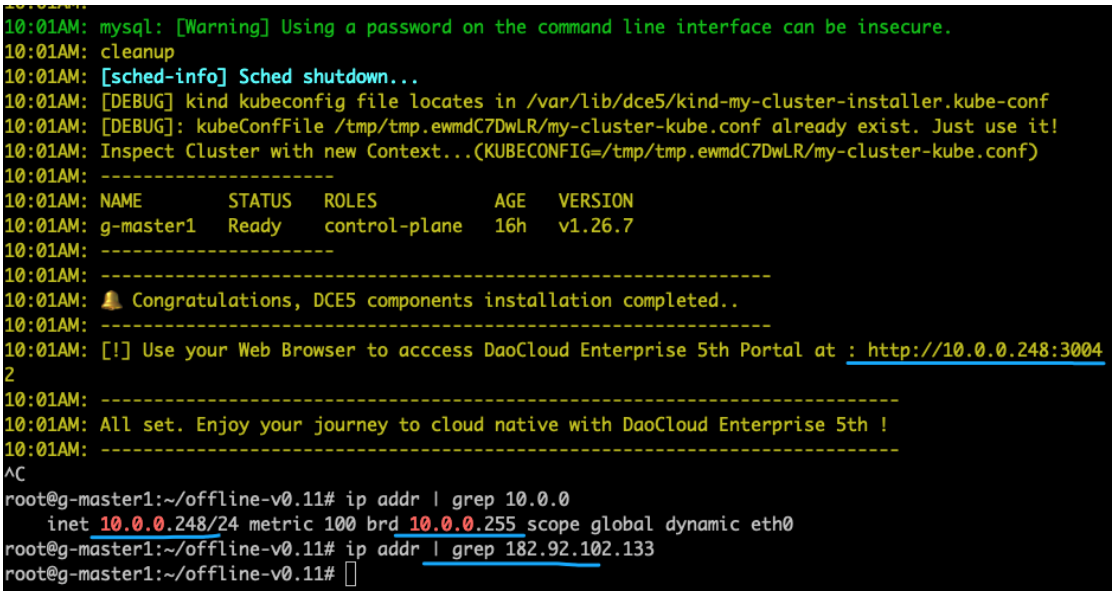
参考以下配置，注意设置 loadBalancer.type = NodePort，并填写主机的私网 IP 地址：

```yaml title="clusterConfig.yaml"
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
spec:
  loadBalancer:
    type: NodePort
  masterNodes:
    - nodeName: "g-master1"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
    - nodeName: "g-master2"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
    - nodeName: "g-master3"
      ip: xx.xx.xx.xx
      ansibleUser: "root"
      ansiblePass: "dangerous"
```

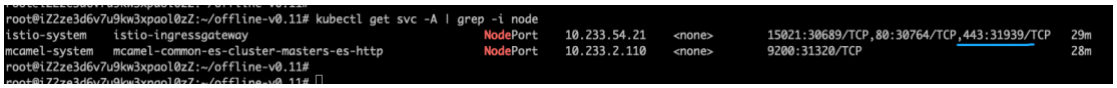
1. 开始安装

   ./dce5-installer cluster-create -c sample/clusterConfig.yaml

2. 安装成功

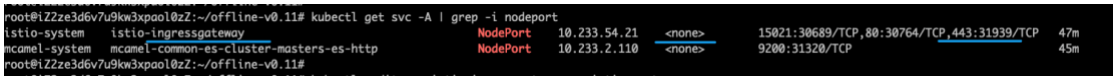


3. 安装后的 svc istio-ingressgateway 如下图所示：

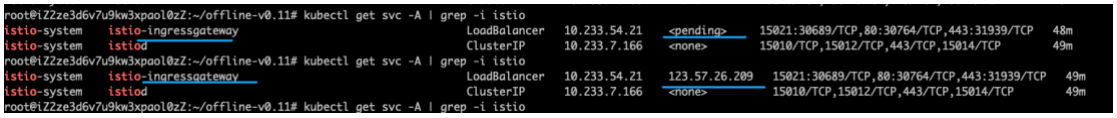


4. 安装 CCM，参考方案 2 中的步骤
5. 修改 svc istio-ingressgateway 的 type 为 LoadBalancer

   修改前：

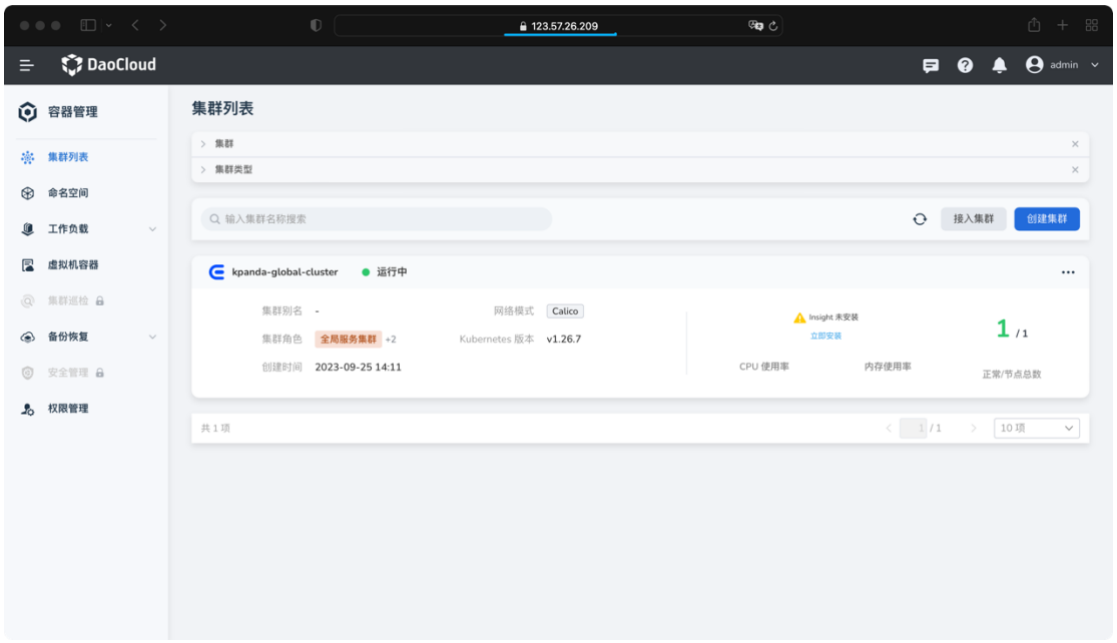


   修改后：



6. 修改 ghippo 反向代理配置

   参考文档 自定义反向代理服务器地址，其中代理地址为上一步中 istio-ingressgateway 的 type 为 LoadBalancer 时分配的 IP 地址。修改成功后即可通过该 IP 地址进行访问。



UOS V20 (1020a) 操作系统上部署 DCE 5.0 商业版

本文将介绍如何在 UOS V20(1020a) 操作系统上部署 DCE 5.0。 安装器 v0.6.0 及更高版本支持这种部署方式。

前提条件

- 请提前阅读部署架构，确认本次部署模式。
- 请提前阅读部署要求，确认网络、硬件、端口等是否符合需求。
- 请提前阅读准备工作，确认机器资源及前置检查。

离线安装

1. 由于安装器依赖 python，所以需要在火种机器中先安装 python3.6。

```
# 执行以下命令下载依赖
dnf install -y --downloadonly --downloaddir=rpm/python36

## 执行以下命令开始安装
rpm -ivh python3-pip-9.0.3-18.uelc20.01.noarch.rpm python3-setuptools-39.2.0-7.uelc20.2.noarch.rpm python36-3.6.8-2.module+uelc20+36+6174170c.x86_64.rpm
```

2. 下载全模式离线包，可以在下载中心下载最新版本。

CPU 架构	版本	下载地址
AMD64	v0.26.0	https://qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.26.0-amd64.tar

3. 下载完毕后解压离线包：

```
curl -LO https://qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.26.0-amd64.tar
tar -xvf offline-v0.26.0-amd64.tar
```

4. 下载 UnionTech Server V20 1020a ISO 镜像文件。

```
curl -LO https://cdimage-download.chinauos.com/uniontechos-server-20-1020a-amd64.iso
```

5. 制作 **os-pkgs-uos-20.tar.gz** 文件。

   下载制作脚本

   ```
   curl -Lo ./build.sh https://raw.githubusercontent.com/kubean-io/kubean/main/build/os-packages/others/uos_v20/build.sh
   chmod +x build.sh
   ```

   执行脚本生成 **os-pkgs-uos-20.tar.gz** 文件：

   ```
   ./build.sh
   ```

6. 下载 addon 离线包，可以在下载中心下载最新版本（可选）

7. 设置集群配置文件 clusterConfig.yaml，可以在离线包 offline/sample 下获取该文件并按需修改。参考配置为：

```
apiVersion: provision.daocloud.io/v1alpha3
kind: ClusterConfig
metadata:
spec:
  clusterName: my-cluster
  loadBalancer:
    type: metallb
    istioGatewayVip: 172.30.41.XXX/32
    insightVip: 172.30.41.XXX/32
  masterNodes:
    - nodeName: "g-master1"
      ip: 10.5.14.XXX
      ansibleUser: "root"
      ansiblePass: "******"
  fullPackagePath: "/home/offline"
  osRepos:
    type: builtin
    isoPath: "/home/uniontechos-server-20-1020a-amd64.iso" ## ISO 的目录
    osPackagePath: "/home/os-pkgs-uos-20.tar.gz" ## os-pkgs 的目录
  imagesAndCharts:
    type: builtin
  addonPackage:
    path: "/home/addon-offline-full-package-v0.5.3-alpha2-amd64.tar.gz" ## addon 的目录
  binaries:
    type: builtin
```

8. 开始安装 DCE 5.0。

```
./dce5-installer cluster-create -m ./sample/manifest.yaml -c ./sample/clusterConfig.yaml
```

部分参数介绍，更多参数可以通过 ./dce5-installer --help 来查看：

- -z 最小化安装
- -c 指定集群配置文件，使用 NodePort 暴露控制台时不需要指定 -c
- -d 开启 debug 模式
- --serial 指定后所有安装任务串行执行

9. 安装完成后，命令行会提示安装成功。恭喜您！ 现在可以通过屏幕提示的 URL 使用默认的账户和密码（admin/changeme）探索全新的 DCE 5.0 啦！

请记录好提示的 URL，方便下次访问。

10. 成功安装 DCE 5.0 商业版之后，请联系我们授权：电邮 info@daocloud.io 或致电 400 002 6898。

在 Oracle Linux R9/R8 U1 上部署 DCE 5.0 商业版

本文将介绍如何在 Oracle Linux R9/R8 U1 操作系统上部署 DCE 5.0。 安装器 v0.8.0 及更高版本支持这种部署方式。

前提条件

- 请提前阅读部署架构，确认本次部署模式。
- 请提前阅读部署要求，确认网络、硬件、端口等是否符合需求。
- 请提前阅读准备工作，确认机器资源及前置检查。

离线安装

1. 下载全模式离线包，可以在下载中心下载最新版本。

CPU 架构	版本	下载地址
AMD64	v0.26.0	https://qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.26.0-amd64.tar

2. 下载完毕后解压离线包：

```
curl -LO https://qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.26.0-amd64.tar
tar -xvf offline-v0.26.0-amd64.tar
```

3. 下载 Oracle Linux R9/R8 U1 镜像文件。

```
# Oracle Linux R9 U1
curl -LO https://yum.oracle.com/ISOS/OracleLinux/OL9/u1/x86_64/OracleLinux-R9-U1-x86_64-dvd.iso

# Oracle Linux R8 U1
curl -LO https://yum.oracle.com/ISOS/OracleLinux/OL8/u7/x86_64/OracleLinux-R8-U7-x86_64-dvd.iso
```

4. 下载 Oracle Linux R9/R8 U1 osPackage 离线包。

```
# Oracle Linux R9 U1
curl -LO https://files.m.daocloud.io/github.com/kubean-io/kubean/releases/download/v0.13.9/os-pkgs-oracle9-v0.13.9.tar.gz

# Oracle Linux R8 U1
curl -LO https://files.m.daocloud.io/github.com/kubean-io/kubean/releases/download/v0.13.9/os-pkgs-oracle8-v0.13.9.tar.gz
```

5. 下载 addon 离线包，可以在下载中心下载最新版本（可选）

6. 设置集群配置文件 clusterConfig.yaml，可以在离线包 offline/sample 下获取该文件并按需修改。 参考配置为：

```
apiVersion: provision.daocloud.io/v1alpha3
kind: ClusterConfig
metadata:
spec:
  clusterName: oracle-cluster
  loadBalancer:
    type: metallb
    istioGatewayVip: 172.30.41.XXX/32
    insightVip: 172.30.41.XXX/32
  masterNodes:
    - nodeName: "g-master"
```

```
      ip: 172.30.41.XXX
      ansibleUser: "root"
      ansiblePass: "******"
  fullPackagePath: "/root/workspace/offline"
  osRepos:
    type: builtin
    isoPath: "/root/workspace/iso/OracleLinux-R9-U1-x86_64-dvd.iso"
    osPackagePath: "/root/workspace/os-pkgs/os-pkgs-oracle9-v0.5.4.tar.gz"
  imagesAndCharts:
    type: builtin
  binaries:
    type: builtin
  kubeanConfig: |-
    # 打开 sysctl 推荐配置，避免出现`too many open files`问题
    node_sysctl_tuning: true
    # 禁止 kubespray 为 oracel linux 安装 public repo
    use_oracle_public_repo: false
```

由于安装过程中 kpanda-controller-manager 组件报错 failed to create fsnotify watcher: too many open files.， 所以需要在 clusterConfig.yaml 文件中设置 node_sysctl_tuning: true。

7. 开始安装 DCE 5.0。

```
./dce5-installer cluster-create -m ./sample/manifest.yaml -c ./sample/clusterConfig.yaml
```

部分参数介绍，更多参数可以通过 ./dce5-installer --help 来查看：

- -z 最小化安装
- -c 指定集群配置文件，使用 NodePort 暴露控制台时不需要指定 -c
- -d 开启 debug 模式
- --serial 指定后所有安装任务串行执行

8. 安装完成后，命令行会提示安装成功。恭喜您！ 现在可以通过屏幕提示的 URL 使用默认的账户和密码（admin/changeme）探索全新的 DCE 5.0 啦！

请记录好提示的 URL，方便下次访问。

9. 成功安装 DCE 5.0 商业版之后，请联系我们授权：电邮 info@daocloud.io 或致电 400 002 6898。

在 TencentOS Server 3.1 上部署 DCE 5.0 商业版

本文将介绍如何在 TencentOS Server 3.1 上部署 DCE 5.0。 安装器 v0.9.0 及更高版本支持这种部署方式。

前提条件

- 请提前阅读部署架构，确认本次部署模式。
- 请提前阅读部署要求，确认网络、硬件、端口等是否符合需求。
- 请提前阅读准备工作，确认机器资源及前置检查。

离线安装

1. 下载全模式离线包，可以在下载中心下载最新版本。

CPU 架构	版本	下载地址
AMD64	v0.26.0	https://qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.26.0-amd64.tar

2. 下载完毕后解压离线包：

```
curl -LO https://qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.26.0-amd64.tar
tar -xvf offline-v0.26.0-amd64.tar
```

3. 下载 TencentOS Server 3.1 镜像文件。

```
curl -LO http://mirrors.tencent.com/tlinux/3.1/iso/x86_64/TencentOS-Server-3.1-TK4-x86_64-minimal-2209.3.iso
```

4. 下载 TencentOS Server 3.1 osPackage 离线包。

```
bash curl -LO https://files.m.daocloud.io/github.com/kubean-io/kubean/releases/download/v0.6.6/os-pkgs-tencent31-v0.6.6.tar.gz
```

5. 下载 addon 离线包，可以在下载中心下载最新版本（可选）
6. 设置集群配置文件 clusterConfig.yaml，可以在离线包 offline/sample 下获取该文件并按需修改。 参考配置为：

```
apiVersion: provision.daocloud.io/v1alpha3
kind: ClusterConfig
metadata:
spec:
  clusterName: tencent-cluster
  loadBalancer:
    type: metallb
    istioGatewayVip: 172.30.41.XXX/32
    insightVip: 172.30.41.XXX/32
  masterNodes:
    - nodeName: "g-master"
      ip: 172.30.41.XXX
      ansibleUser: "root"
      ansiblePass: "******"
  fullPackagePath: "/root/workspace/offline"
  osRepos:
    type: builtin
    isoPath: "/root/workspace/iso/TencentOS-Server-3.1-TK4-x86_64-minimal-2209.3.iso"
    osPackagePath: "/root/workspace/os-pkgs/os-pkgs-tencent31-v0.6.1.tar.gz"
  imagesAndCharts:
    type: builtin
  binaries:
```

```
    type: builtin
  kubeanConfig: |-
    allow_unsupported_distribution_setup: true
    redhat_os_family_extensions:
      - "TencentOS"
```

- TencentOS Server 3.1 操作系统属于 Redhat 系统族，所以需要在 kubeanConfig 中定义 redhat_os_family_extensions 参数。
- 执行如下命令查看 TencentOS Server 3.1 操作系统的 OS Family 标识，输出结果为 TencentOS。

```
export USER=root
export PASS=dangerous
export ADDR=172.30.41.166
export ANSIBLE_HOST_KEY_CHECKING=False
ansible -m setup -a 'filter=ansible_os_family' -e "ansible_user=${USER}
ansible_password=${PASS}" -i ${ADDR}, all
```

7. 开始安装 DCE 5.0。

```
./dce5-installer cluster-create -m ./sample/manifest.yaml -c ./sample/clusterConfig.yaml
```

部分参数介绍，更多参数可以通过 ./dce5-installer --help 来查看：

- -z 最小化安装
- -c 指定集群配置文件，使用 NodePort 暴露控制台时不需要指定 -c
- -d 开启 debug 模式
- --serial 指定后所有安装任务串行执行

8. 安装完成后，命令行会提示安装成功。恭喜您！ 现在可以通过屏幕提示的 URL 使用默认的账户和密码（admin/changeme）探索全新的 DCE 5.0 啦！

请记录好提示的 URL，方便下次访问。

9. 成功安装 DCE 5.0 商业版之后，请联系我们授权：电邮 info@daocloud.io 或致电 400 002 6898。

在 NeoKylin Linux Advanced Server V7Update6 上部署 DCE 5.0 商业版

本文将介绍如何在 NeoKylin Linux Advanced Server V7Update6 操作系统上部署 DCE 5.0。

前提条件

- 请提前阅读部署架构，确认本次部署模式。
- 请提前阅读部署要求，确认网络、硬件、端口等是否符合需求。
- 请提前阅读准备工作，确认机器资源及前置检查。
- 需提前在节点上安装好 iptables、iproute。

```
# 安装 iptables
wget https://rpmfind.net/linux/centos/7.9.2009/os/x86_64/Packages/iptables-1.4.21-35.el7.x86_64.rpm
rpm -ivh iptables-1.4.21-35.el7.x86_64.rpm
iptables --version
```

```
# 安装 iproute
wget https://rpmfind.net/linux/centos/7.9.2009/os/x86_64/Packages/iproute-4.11.0-30.el7.x86_64.rpm
rpm -ivh iproute-4.11.0-30.el7.x86_64.rpm
```

离线安装

1. 下载全模式离线包，可以在下载中心下载最新版本。

CPU 架构	版本	下载地址
AMD64	v0.26.0	https://qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.26.0-amd64.tar

2. 下载完毕后解压离线包：

```
curl -LO https://qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.26.0-amd64.tar
tar -xvf offline-v0.26.0-amd64.tar
```

3. 下载 NeoKylin Linux Advanced Server V7Update6 镜像文件。

```
# Oracle Linux R9 U1
curl -LO https://yum.oracle.com/ISOS/OracleLinux/OL9/u1/x86_64/OracleLinux-R9-U1-x86_64-dvd.iso
```

4. 制作 NeoKylin Linux Advanced Server V7Update6 OS Package 离线包。

   前提条件：

   - 检查 libselinux-python 是否存在，如不存在，可参考下面的安装依赖包方式进行安装。
   - rpm -q libselinux-python
   - 安装依赖包 device-mapper-libs、conntrack、sshpass， 查看依赖包的下载地址。安装命令如下：
   - rpm -ivh <package name>
   - 手动修改 os-release 文件中的 VERSION_ID="7"
   - vi /etc/os-release

   参考制作 OS Package 离线包，开始制作离线包。

5. 下载 addon 离线包，可以在下载中心下载最新版本（可选）
6. 设置集群配置文件 clusterConfig.yaml，可以在离线包 offline/sample 下获取该文件并按需修改。 参考配置为：

```
apiVersion: provision.daocloud.io/v1alpha3
kind: ClusterConfig
metadata:
  creationTimestamp: null
spec:
  clusterName: my-cluster
  masterNodes:
    - nodeName: "master1"
      ip: 10.6.135.199
      ansibleUser: "root"
      ansiblePass: "dangerous@2024"
  workerNodes: []
  osRepos:
    type: none
  imagesAndCharts:
    type: builtin
  binaries:
    type: builtin
  loadBalancer:
    type: NodePort
  fullPackagePath: /home/offline
```

7. 开始安装 DCE 5.0。

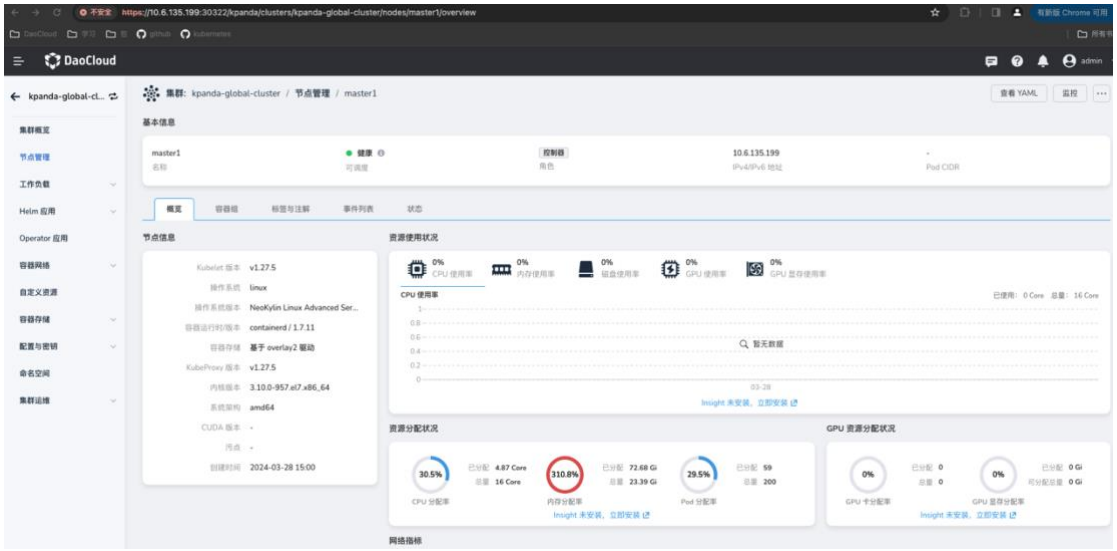
```
./dce5-installer cluster-create -m ./sample/manifest.yaml -c ./sample/clusterConfig.yaml
```

部分参数介绍，更多参数可以通过 ./dce5-installer --help 来查看：

- -z 最小化安装
- -c 指定集群配置文件，使用 NodePort 暴露控制台时不需要指定 -c
- -d 开启 debug 模式
- --serial 指定后所有安装任务串行执行

8. 安装完成后，命令行会提示安装成功。恭喜您！ 现在可以通过屏幕提示的 URL 使用默认的账户和密码（admin/changeme）探索全新的 DCE 5.0 啦！

请记录好提示的 URL，方便下次访问。



9. 成功安装 DCE 5.0 商业版之后，请联系我们授权：电邮 info@daocloud.io 或致电 400 002 6898。

在其他 Linux 上离线部署 DCE 5.0 商业版

本文将介绍如何在其他 Linux 上部署 DCE 5.0。安装器 v0.7.0 及更高版本支持这种部署方式。

其他 Linux 本质上是由于 DCE 5.0 对某些 Linux 没有提供安装系统离线包（OS package），需要您自己去制作。

已验证的操作系统

架构	操作系统	所属系统族	推荐内核
AMD64	统信 UOS V20 (1050d)	Debian	4.19.0-server-amd64
AMD64	AnolisOS 8.8 GA	Redhat	5.10.134-13.an8.x86_64
AMD64	Ubuntu 22.04.3	Debian	5.15.0-78-generic

没有验证的操作系统，可以尝试参考本文的教程来部署。

前提条件

- 请提前阅读部署架构，确认本次部署模式。
- 请提前阅读部署要求，确认网络、硬件、端口等是否符合需求。
- 请提前阅读准备工作，确认机器资源及前置检查。

制作操作系统离线包（OS package）

制作及安装

1. 下载制作工具。

```
cd /home
curl -Lo ./other_os_pkgs.sh https://raw.githubusercontent.com/kubean-io/kubean/main/build/os-packages/others/other_os_pkgs.sh && chmod +x other_os_pkgs.sh
```

2. 构建操作系统离线包

```
# 执行系统离线包构建命令
./other_os_pkgs.sh build
```

3. 安装操作系统离线包

```
# 指定 os pkgs 离线包的路径
export PKGS_TAR_PATH=/home/os-pkgs-${DISTRO}-${VERSION}.tar.gz
# 指定集群 master/worker 节点 IP（多节点 IP 地址以空格分割）
export HOST_IPS='192.168.10.11 192.168.10.12'
```

```
# 指定安装的目标节点接入信息（多节点用户名密码需保持一致）
export SSH_USER=root
export SSH_CRED=dangerous
# 执行安装命令，并输出日志
./other_os_pkgs.sh install >>log.txt
```

4. 安装成功后，会输出如下日志：

```
[root@master test]# cat log.txt |egrep 'INFO|WARN'

[WARN]    skip install yq ...
[INFO]    succeed to install package 'python-apt'
[INFO]    succeed to install package 'python3-apt'
[INFO]    succeed to install package 'aufs-tools'
[INFO]    succeed to install package 'apt-transport-https'
[INFO]    succeed to install package 'software-properties-common'
[INFO]    succeed to install package 'conntrack'
[INFO]    succeed to install package 'apparmor'
[WARN]    the package 'libseccomp2' has been installed
[INFO]    succeed to install package 'ntp'
[WARN]    the package 'openssl' has been installed
[INFO]    succeed to install package 'curl'
[INFO]    succeed to install package 'rsync'
[INFO]    succeed to install package 'socat'
[WARN]    the package 'unzip' has been installed
[WARN]    the package 'e2fsprogs' has been installed
[WARN]    the package 'xfsprogs' has been installed
[INFO]    succeed to install package 'ebtables'
[WARN]    the package 'bash-completion' has been installed
[WARN]    the package 'tar' has been installed
[INFO]    succeed to install package 'ipvsadm'
[INFO]    succeed to install package 'ipset'
[INFO]    All packages for Node (192.168.10.11) have been installed.
```

- 你可以通过 cat log.txt |egrep 'INFO|WARN' 检查安装情况：

  如果出现 failed to install package 关键字，则说明未安装成功，并且最终失败时， 会输出 the packages that failed to install are: ipset ipvsadm xfsprogs。

- 相同系统族（os family）的不同版本（major version）所对应的包名存在差异:

系统族	版本	包名
Debian	< 11	python-apt
	>= 11	python3-apt

系统族	版本	包名
Redhat Major Version	< 8	libselinux-python
	>= 8	python3-libselinux

开始离线安装

1. 下载全模式离线包，可以在下载中心下载最新版本。

CPU 架构	版本	下载地址
AMD64	v0.26.0	https://qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.26.0-amd64.tar

2. 下载完毕后解压离线包：

```
curl -LO https://qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.26.0-amd64.tar
tar -xvf offline-v0.26.0-amd64.tar
```

3. 参考制作操作系统离线包（OS package）。
4. 下载 addon 离线包，可以在下载中心下载最新版本（可选）。
5. 设置集群配置文件 clusterConfig.yaml， 可以在离线包 offline/sample 下获取该文件并按需修改。

   UnionTech OS Server 20 1050dAnolisOS 8.8 GAUbuntu 22.04.3

   clusterConfig.yaml

```
apiVersion: provision.daocloud.io/v1alpha3
kind: ClusterConfig
metadata:
spec:
  clusterName: test-cluster
  loadBalancer:
    type: metallb
    istioGatewayVip: 172.30.41.XXX/32
    insightVip: 172.30.41.XXX/32
  masterNodes:
    - nodeName: "g-master1"
      ip: 172.30.41.xxx
      ansibleUser: "root"
```

```
      ansiblePass: "******"
  fullPackagePath: "/root/offline"
  osRepos:
    type: none
  imagesAndCharts:
    type: builtin
  binaries:
    type: builtin
  kubeanConfig: |-
    allow_unsupported_distribution_setup: true
    debian_os_family_extensions:
      - "UnionTech OS Server 20\" "
```

配置参数说明：

参数	说明	是否必填
spec.kubeanConfig.allow_unsupported_distribution_setup	是否跳过已支持发行版检测	必填
spec.kubeanConfig.debian_os_family_extensions	可通过查看 ansible_os_family 来填写	若为 Debian 系统族则需填
spec.kubeanConfig.redhat_os_family_extensions	可通过查看 ansible_os_family 来填写	若为 Redhat 系统族则需填

查看当前发行版环境的系统族标识：

```
export USER=root
export PASS=xxxx
export ADDR=192.168.10.xxx
export ANSIBLE_HOST_KEY_CHECKING=False
ansible -m setup -a 'filter=ansible_os_family' -e "ansible_user=${USER}
ansible_password=${PASS}" -i ${ADDR}, all
```

执行成功后将输出以下信息：

```
192.168.10.xxx | SUCCESS => {
    "ansible_facts": {
        "ansible_os_family": "UnionTech OS Server 20\" ",
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false
}
```

6. 开始安装 DCE 5.0。

```
./dce5-installer cluster-create -m ./sample/manifest.yaml -c ./sample/clusterConfig.yaml
```

部分参数介绍，更多参数可以通过 ./dce5-installer --help 来查看：

- -z 最小化安装
- -c 指定集群配置文件，使用 NodePort 暴露控制台时不需要指定 -c
- -d 开启 debug 模式
- --serial 指定后所有安装任务串行执行

7. 安装完成后，命令行会提示安装成功。恭喜您！ 现在可以通过屏幕提示的 URL 使用默认的账户和密码（admin/changeme）探索全新的 DCE 5.0 啦！

请记录好提示的 URL，方便下次访问。

8. 成功安装 DCE 5.0 商业版之后，请联系我们授权：电邮 info@daocloud.io 或致电 400 002 6898。

轻量化部署裁剪方案验证

考虑到在某些场景下，对于云端软件部署资源要求较苛刻，特别是在军工场景，故本文介绍基于 DCE 5.0 社区版部署轻量化裁剪方案和验证流程，旨在满足客户场景部署轻量化需求。

本次裁剪方案分为四个阶段逐步缩容验证，详情如下：

验证环境

- 操作系统：Kylin Linux Advanced Server V10 (Halberd) for ARM
- CPU: 10C
- 内存: 17Gi

安装组件及资源统计

DCE 5.0 安装组件及裁剪方案及分阶段裁剪思路如下：

内存占用高的组件：

组件	内存占用	总占用
elasticsearch	2460.38MiB (2.40GiB)	7.46 GiB
kind-cluster	1005.62 MiB (0.98GiB)	
insight (server+agent)	2655.17MiB (2.59GiB)	
mysql instance	1250.13MiB (1.22 GiB)	

Phase1~Phase2 版本已做的优化措施：

1. 裁剪 infrastructure & components 非必须组件
2. mysql 由双实例整合为单实例
3. 拆件 insight 部分非必须组件
4. 移除所有 pod 的 istio sidecar
5. 关停火种 kind 集群

Phase3~Phase4 版本预计的优化措施：

1. 解耦并禁用 es 分析组件
2. 禁用 insight 监控组件

裁剪方案的四个阶段：

	组件	phase1	phase2	phase3	phase4
kind-cluster	registry/minio/chartmuseum 等	✔	✖	✖	✖
infrastructure	minio	✔	✔	✔	✔
	istio	✔	✔	✔	✔
	mysql	✔	整合单实例优化	整合单实例优化	整合单实例优化
	redis	✔	✔	✔	✔
	elasticsearch	✔	✔	✖	✖
components	ghippo	✔	✔	✔	✔
	ipavo	✔	✔	✔	✔
	kpanda	✔	✔	✔	✔
	kairship	✔	✔	✔	✔
	insight	✔	部分组件裁剪优化	部分组件裁剪优化	✖
	insight-agent	✔			✖
	kangaroo	✔	✔	✔	✔
内存资源占用	/	13.57 GiB	11.16 GiB	预计 8.76 GiB	预计 6.17 GiB

组件轻量化裁剪 Phase1 全貌：

infrastructures	minio 对象存储	metrics-server 资源监控	hwameiStor 本地存储			
	mysql 关系数据库	kafka 消息队列				
	redis 内存数据库	cert-manager 证书管理				
	istio 服务网格	metallb 负载均衡				
	elasticsearch 搜索引擎	contour 入口控制器				
components	ipavo 仪表盘	insight-agent 可观测性	kcoral 应用备份	skoala 微服务引擎	mcamel-mysql 关系数据库	mcamel-postgresql 高级数据库
	ghippo 全局管理	baize 智算模块	amamba 应用工作台	mspider 服务网格	mcamel-redis 内存数据库	mcamel-mongodb 文档数据库
	kpanda 容器管理	baize-agent 智算模块	jenkins 持续集成	spidernet 网络	mcamel-kafka 消息队列	mcamel-rocketmq 消息中间件
	kairship 多云编排	kubean 集群管理	argo-cd gitops 部署	kolm operator 管理	mcamel-minio 对象存储	mcamel-rabbitmq 消息队列
	kangaroo 镜像仓库	gmagpie 运营管理	dowl 安全管理	kant 云边协调	virtnest 虚机管理	mcamel-elasticsearch 数据检索引擎
	insight 可观测性	kcollie 集群巡检				

保留
裁剪

优化措施

1. Insight 在保证监控能力正常的前提下，可以停止以下 Pod:

Pod name	Mem Size
insight-agent-fluent-bit-5x2rn	99.62 MiB
insight-agent-otel-kubernetes-collector-69f67cc745-xt5hj	74.94 MiB
insight-agent-tailing-sidecar-operator-6f85f7bb75-67xc8	46.81 MiB
insight-elastic-alert-64bbb468dc-l4mk5	30.38 MiB
insight-jaeger-collector-5cd5b94dcc-mwgcl	32.50 MiB
insight-jaeger-query-5495c59bbd-fk287	28.88 MiB
insight-opentelemetry-collector-5d47dd6c6b-nk54t	62.12 MiB
可优化内存	375.25 MiB

2. 通过脚本 clean_istio_proxy.sh 移除 Istio 边车
3. 关闭火种 kind-cluster 以及 elasticsearch 组件
    a. 安装器部署之前，manifest.yaml disable elasticSearch 组件
    [不可行] insight-server 强依赖 es
    b. 安装器部署完后，关闭 kind-cluster 容器
    [可行] 镜像拉取策略存在隐患，需调整为 **IfNotPresent**
4. Global 中部署 registry，通过 kangaroo 纳管 [可行]
5. 部署 mysql 单实例，容器管理外接 mysql 实例 [可行]

阶段优化

Phase1 优化

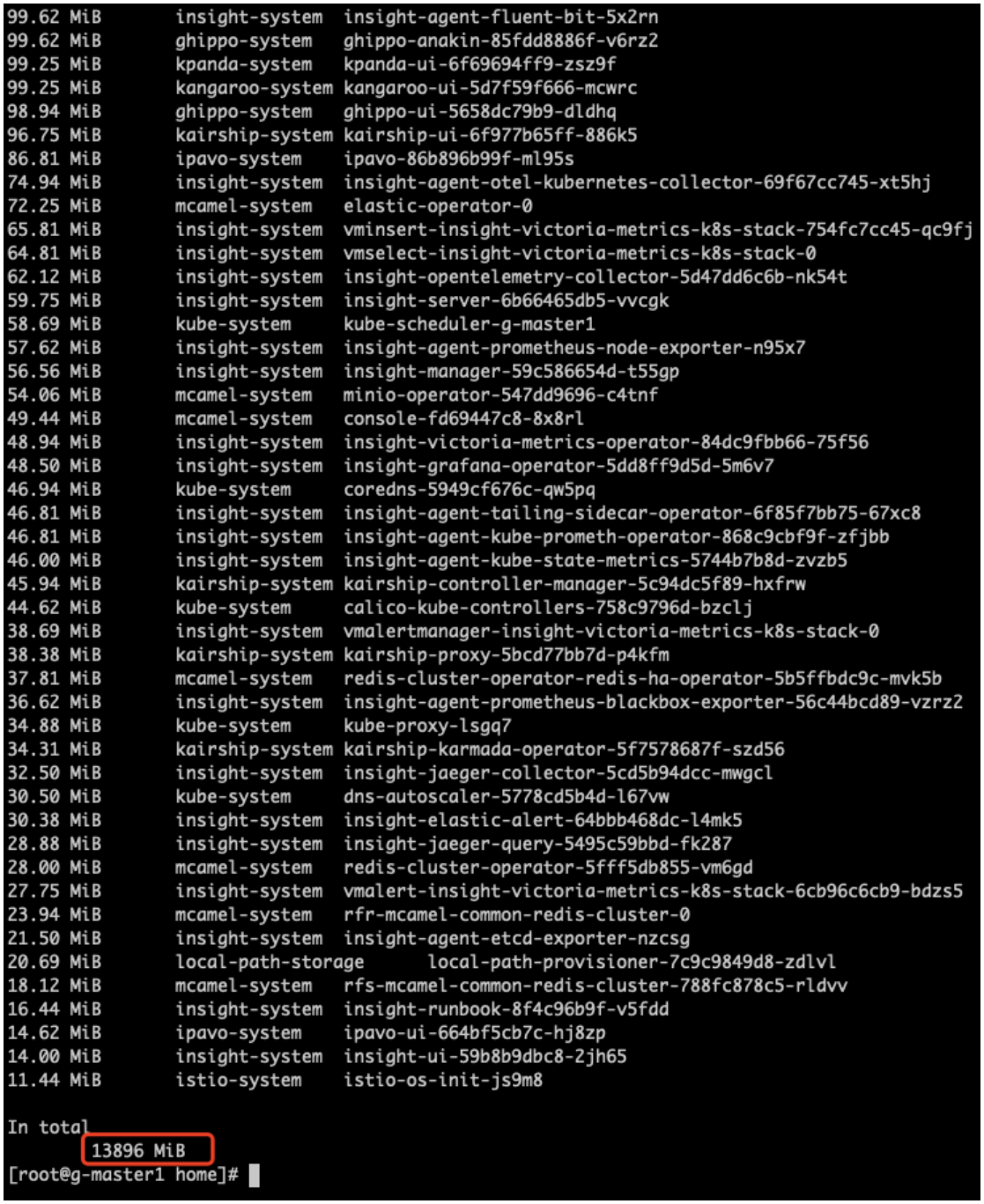
Phase1 裁剪 infrastructure & components 非必须组件。

1. 火种内资源占用

```
281.69 MiB      kube-system     kube-apiserver-my-cluster-installer-control-plane
117.06 MiB      minio-system    minio-55fd89fc84-hxhzj
64.94 MiB       kube-system     etcd-my-cluster-installer-control-plane
64.75 MiB       kube-system     kube-controller-manager-my-cluster-installer-control-plane
34.44 MiB       kube-system     kube-scheduler-my-cluster-installer-control-plane
31.69 MiB       kubean-system   kubean-75c9c47ffd-tk8lm
27.25 MiB       kube-system     coredns-787d4945fb-rzzll
24.69 MiB       museum-system   chartmuseum-5db46b89f4-vnmkn
23.81 MiB       kube-system     coredns-787d4945fb-6pskx
23.31 MiB       kube-system     kube-proxy-pjgqr
18.62 MiB       kubean-system   kubean-admission-584659965b-xc7r7
18.50 MiB       registry-system registry-docker-registry-66d888d7c8-sr4nk
17.94 MiB       kube-system     kindnet-c9n8q
16.88 MiB       local-path-storage      local-path-provisioner-84f55fc489-2kzrt
```

2. Global + kind 火种资源占用

```
2460.38 MiB     mcamel-system   mcamel-common-es-cluster-masters-es-data-0
1005.62 MiB     kind-cluster    kind-cluster
877.12 MiB      kube-system     kube-apiserver-g-master1
828.94 MiB      insight-system  prometheus-insight-agent-kube-prometh-prometheus-0
668.62 MiB      insight-system  vmstorage-insight-victoria-metrics-k8s-stack-0
635.88 MiB      ghippo-system   ghippo-keycloakx-0
630.69 MiB      mcamel-system   mcamel-common-kpanda-mysql-cluster-mysql-0
620.62 MiB      mcamel-system   mcamel-common-mysql-cluster-mysql-0
311.12 MiB      kube-system     calico-node-l9v27
243.00 MiB      kpanda-system   kpanda-clusterpedia-apiserver-5b6f78cdbc-rpz9t
232.25 MiB      kpanda-system   kpanda-controller-manager-7945b8b99b-khtbc
227.62 MiB      kpanda-system   kpanda-clusterpedia-clustersynchro-manager-77bf6768cc-q7mx9
195.25 MiB      kpanda-system   kpanda-apiserver-dff49458c-rpvkg
172.50 MiB      mcamel-system   mcamel-common-minio-cluster-pool-0-0
166.25 MiB      kpanda-system   kpanda-egress-549c6ffcb-c4g56
143.69 MiB      ghippo-system   ghippo-controller-manager-6569686d98-l579p
142.81 MiB      kpanda-system   kpanda-ingress-cd8d8d449-tvsrd
142.19 MiB      kpanda-system   kpanda-bindings-syncer-7f5d696d55-hc9x4
139.56 MiB      ghippo-system   ghippo-apiserver-7758667755-7f8md
139.50 MiB      mcamel-system   mysql-mgr-operator-56c9cd6b79-8pfwr
139.44 MiB      kube-system     kube-controller-manager-g-master1
139.06 MiB      kairship-system kairship-apiserver-56777c5f7b-hfqg2
138.19 MiB      kangaroo-system kangaroo-controller-manager-7444b7cb4b-qt46j
136.94 MiB      kangaroo-system kangaroo-apiserver-664cc79d5b-qr7vs
135.44 MiB      kube-system     etcd-g-master1
133.69 MiB      ghippo-system   ghippo-auditserver-64c89dbd96-6t22c
132.56 MiB      insight-system  grafana-deployment-767bb9c6f8-wln9n
126.62 MiB      istio-system    istiod-5766cdd849-h5zjc
118.06 MiB      kangaroo-system kangaroo-upload-7bb65cf7df-8d4jq
118.00 MiB      kpanda-system   kpanda-clusterpedia-controller-manager-578d4ccf4c-lz8h4
115.62 MiB      kpanda-system   kpanda-cloudtty-controller-manager-6c9b4796b6-txxft
108.50 MiB      istio-system    istio-ingressgateway-6fc44f5765-5hdcp
100.00 MiB      mcamel-system   mysql-operator-0
```



按脚本统计，内存总消耗：13.6 GiB

可裁剪组件理论内存消耗统计：

统计计算结果与实际部署消耗会存在差异，比如系统运行期间产生的动态缓存，都会影响内存实际占用。

组件	占用
elasticsearch	2460.38 MiB (2.40 GiB)
kind-cluster	1005.62 MiB (0.98 GiB)
insight	2655.17 MiB (2.59 GiB)
kangaroo	277.25 MiB (0.27 GiB)
总计	6.24 GiB

若不装此四类组件，内存消耗预计 7.36 GiB，但前期做过社区版裁剪，即不带此四类组件，8G 内存勉强运行，但不稳定。

Phase2 优化

Phase2 mysql 由双实例整合为单实例，同时裁剪优化 insight 部分组件。

```
2457.19 MiB     mcamel-system   mcamel-common-es-cluster-masters-es-data-0
1524.19 MiB     kube-system     kube-apiserver-g-master1
1071.56 MiB     insight-system  prometheus-insight-agent-kube-prometh-prometheus-0
845.88 MiB      mcamel-system   mcamel-common-mysql-cluster-mysql-0
743.25 MiB      insight-system  vmstorage-insight-victoria-metrics-k8s-stack-0
471.00 MiB      ghippo-system   ghippo-keycloakx-0
202.62 MiB      kube-system     calico-node-c6gph
183.69 MiB      kube-system     etcd-g-master1
174.06 MiB      kpanda-system   kpanda-controller-manager-b9c7b94c-4s7qc
164.31 MiB      mcamel-system   mcamel-common-minio-cluster-pool-0-0
163.50 MiB      kube-system     kube-controller-manager-g-master1
143.38 MiB      istio-system    istiod-5766cdd849-z9xsv
132.94 MiB      mcamel-system   mysql-mgr-operator-56c9cd6b79-ldc4v
130.19 MiB      insight-system  grafana-deployment-767bb9c6f8-cptdc
127.69 MiB      kpanda-system   kpanda-clusterpedia-clustersynchro-manager-659d84f6bb-7wl6v
109.38 MiB      kpanda-system   kpanda-apiserver-6f84c4ddbb-6xp6t
107.19 MiB      istio-system    istio-ingressgateway-54cbfcf8f5-g2ztp
101.62 MiB      kpanda-system   kpanda-clusterpedia-apiserver-84865f9c94-4jsww
93.56 MiB       kpanda-system   kpanda-egress-7cfd449445-cc8km
89.00 MiB       mcamel-system   mysql-operator-0
88.00 MiB       ipavo-system    ipavo-86b896b99f-qrrs5
87.69 MiB       insight-system  vmselect-insight-victoria-metrics-k8s-stack-0
81.56 MiB       kpanda-system   kpanda-ingress-f48855c68-cz6rb
78.06 MiB       mcamel-system   elastic-operator-0
76.75 MiB       kpanda-system   kpanda-bindings-syncer-6d7b974746-gr9tm
68.38 MiB       insight-system  insight-server-6d75485b88-pnlv7
67.88 MiB       ghippo-system   ghippo-apiserver-69585c76cb-c2jv8
67.38 MiB       ghippo-system   ghippo-controller-manager-7f5475db45-kfdfc
66.50 MiB       kube-system     kube-scheduler-g-master1
66.44 MiB       kangaroo-system kangaroo-controller-manager-9f968dbd5-w2j4b
66.12 MiB       kangaroo-system kangaroo-apiserver-6c6cd7fc59-bvwcl
60.75 MiB       kairship-system kairship-apiserver-68dcd8df8c-zlm6l
58.38 MiB       insight-system  insight-manager-5458bccb98-9tght
57.06 MiB       insight-system  insight-grafana-operator-5dd8ff9d5d-5x4q7
56.69 MiB       kube-system     calico-kube-controllers-758c9796d-bgvrs
55.31 MiB       insight-system  insight-victoria-metrics-operator-84dc9fbb66-bhfj2
55.00 MiB       mcamel-system   minio-operator-547dd9696-z5scf
54.12 MiB       ghippo-system   ghippo-auditserver-5fc74bc9bc-b86kn
52.38 MiB       insight-system  insight-agent-kube-prometh-operator-868c9cbf9f-rs2wq
50.50 MiB       insight-system  insight-agent-prometheus-node-exporter-9wsj7
49.62 MiB       insight-system  vminsert-insight-victoria-metrics-k8s-stack-754fc7cc45-7tgz4
48.88 MiB       insight-system  insight-agent-kube-state-metrics-5744b7b8d-kw2hd
47.75 MiB       kpanda-system   kpanda-cloudtty-controller-manager-774456884c-77v54
47.25 MiB       kairship-system kairship-controller-manager-5c94dc5f89-7wnw5
41.94 MiB       kube-system     coredns-5949cf676c-qw5pq
41.56 MiB       mcamel-system   redis-cluster-operator-redis-ha-operator-5b5ffbdc9c-hbjgc
40.56 MiB       kangaroo-system kangaroo-upload-88c748c76-tgj6m
40.06 MiB       kairship-system kairship-proxy-5bcd77bb7d-t2hmv
39.69 MiB       kube-system     kube-proxy-spk7f
```

```
39.19 MiB       kpanda-system   kpanda-clusterpedia-controller-manager-7b4bff86dd-4g64d
36.19 MiB       mcamel-system   redis-cluster-operator-5fff5db855-78lnr
36.12 MiB       insight-system  vmalertmanager-insight-victoria-metrics-k8s-stack-0
34.00 MiB       kairship-system kairship-karmada-operator-5f7578687f-p6lps
32.50 MiB       insight-system  insight-agent-prometheus-blackbox-exporter-56c44bcd89-ms9rx
32.31 MiB       mcamel-system   console-fd69447c8-522hx
31.94 MiB       registry-system registry-docker-registry-59fc99c798-hcvp4
31.62 MiB       insight-system  vmalert-insight-victoria-metrics-k8s-stack-6cb96c6cb9-z2bf4
29.06 MiB       kube-system     dns-autoscaler-5778cd5b4d-l67vw
27.50 MiB       local-path-storage      local-path-provisioner-7c9c9849d8-nv6xr
21.94 MiB       mcamel-system   rfr-mcamel-common-redis-cluster-0
18.81 MiB       ghippo-system   ghippo-anakin-8ddbfbdfb-kphrc
18.69 MiB       kangaroo-system kangaroo-ui-6474b87545-w4rcj
18.50 MiB       mcamel-system   rfs-mcamel-common-redis-cluster-788fc878c5-ck7ch
17.19 MiB       kairship-system kairship-ui-8695c96b6d-d9nqx
16.31 MiB       kpanda-system   kpanda-ui-6f59cf7c4b-xdtwv
16.25 MiB       insight-system  insight-agent-etcd-exporter-kv56t
15.62 MiB       ipavo-system    ipavo-ui-664bf5cb7c-9hkws
15.06 MiB       insight-system  insight-ui-59b8b9dbc8-5jm4j
15.06 MiB       insight-system  insight-runbook-8f4c96b9f-fvdx5
14.75 MiB       ghippo-system   ghippo-ui-5858f4c49d-kc2rx
9.25 MiB        istio-system    istio-os-init-6cv42

In total
       11276.3 MiB
```

```
[root@g-master1 home]# free -h
               total        used        free      shared  buff/cache   available
Mem:            17Gi        11Gi       1.3Gi       494Mi       4.4Gi       2.7Gi
Swap:             0B          0B          0B
[root@g-master1 home]#
```

Phase3 优化

优化项：移除 ES（副本 = 0），依据 issue 2268

```
1761.56 MiB     kube-system     kube-apiserver-g-master1
1093.06 MiB     insight-system  prometheus-insight-agent-kube-prometh-prometheus-0
919.44 MiB      mcamel-system   mcamel-common-mysql-cluster-mysql-0
689.62 MiB      insight-system  vmstorage-insight-victoria-metrics-k8s-stack-0
498.00 MiB      ghippo-system   ghippo-keycloakx-0
192.38 MiB      kube-system     etcd-g-master1
191.62 MiB      kube-system     calico-node-c6gph
189.12 MiB      insight-system  grafana-deployment-6756c876bf-kq2wq
163.94 MiB      kpanda-system   kpanda-clusterpedia-apiserver-84865f9c94-4jsww
148.62 MiB      kpanda-system   kpanda-controller-manager-b9c7b94c-4s7qc
147.50 MiB      mcamel-system   mcamel-common-minio-cluster-pool-0-0
143.88 MiB      kube-system     kube-controller-manager-g-master1
132.44 MiB      kpanda-system   kpanda-clusterpedia-clustersynchro-manager-659d84f6bb-7wl6v
129.12 MiB      kpanda-system   kpanda-apiserver-6f84c4ddbb-6xp6t
128.06 MiB      mcamel-system   mysql-mgr-operator-56c9cd6b79-ldc4v
124.56 MiB      istio-system    istiod-5766cdd849-z9xsv
108.00 MiB      istio-system    istio-ingressgateway-54cbfcf8f5-g2ztp
86.06 MiB       mcamel-system   mysql-operator-0
79.56 MiB       kpanda-system   kpanda-egress-7cfd449445-cc8km
77.38 MiB       ipavo-system    ipavo-86b896b99f-qrrs5
69.44 MiB       kube-system     calico-kube-controllers-758c9796d-bgvrs
68.56 MiB       insight-system  insight-server-5f4c9b8ccd-qrvrf
67.88 MiB       insight-system  insight-grafana-operator-749bc4c6cc-g55kq
60.94 MiB       insight-system  insight-manager-9d798b4f9-7kqlv
59.12 MiB       kpanda-system   kpanda-ingress-f48855c68-cz6rb
59.12 MiB       insight-system  insight-victoria-metrics-operator-6c6777cbdc-2f9gk
56.06 MiB       insight-system  vmselect-insight-victoria-metrics-k8s-stack-0
55.88 MiB       kairship-system kairship-apiserver-68dcd8df8c-zlm6l
53.81 MiB       insight-system  insight-agent-kube-prometh-operator-868c9cbf9f-rs2wq
53.75 MiB       ghippo-system   ghippo-controller-manager-7f5475db45-kfdfc
53.50 MiB       kube-system     kube-scheduler-g-master1
52.50 MiB       kpanda-system   kpanda-bindings-syncer-6d7b974746-gr9tm
52.12 MiB       insight-system  insight-agent-prometheus-node-exporter-9wsj7
50.12 MiB       mcamel-system   minio-operator-547dd9696-z5scf
49.31 MiB       insight-system  vmalertmanager-insight-victoria-metrics-k8s-stack-0
49.31 MiB       ghippo-system   ghippo-apiserver-69585c76cb-c2jv8
47.50 MiB       kangaroo-system kangaroo-apiserver-6c6cd7fc59-bvwcl
46.75 MiB       insight-system  insight-agent-kube-state-metrics-5744b7b8d-kw2hd
44.19 MiB       kangaroo-system kangaroo-controller-manager-9f968dbd5-w2j4b
39.38 MiB       insight-system  vminsert-insight-victoria-metrics-k8s-stack-8859d7668-6m4t5
37.94 MiB       registry-system registry-docker-registry-556df79d9b-4hwqk
37.50 MiB       kairship-system kairship-controller-manager-5c94dc5f89-7wnw5
35.56 MiB       kube-system     coredns-5949cf676c-qw5pq
34.94 MiB       kpanda-system   kpanda-cloudtty-controller-manager-774456884c-77v54
```

```
34.31 MiB       insight-system  vmalert-insight-victoria-metrics-k8s-stack-c4cbdc94f-b42s6
34.25 MiB       kube-system     kube-proxy-spk7f
32.94 MiB       mcamel-system   redis-cluster-operator-redis-ha-operator-5b5ffbdc9c-hbjgc
30.88 MiB       ghippo-system   ghippo-auditserver-5fc74bc9bc-b86kn
30.69 MiB       mcamel-system   console-fd69447c8-522hx
28.38 MiB       kairship-system kairship-karmada-operator-5f7578687f-p6lps
27.94 MiB       kube-system     dns-autoscaler-5778cd5b4d-l67vw
27.62 MiB       kpanda-system   kpanda-clusterpedia-controller-manager-7b4bff86dd-4g64d
27.19 MiB       mcamel-system   redis-cluster-operator-5fff5db855-78lnr
26.62 MiB       insight-system  insight-agent-prometheus-blackbox-exporter-56c44bcd89-ms9rx
26.31 MiB       kairship-system kairship-proxy-5bcd77bb7d-t2hmv
21.38 MiB       mcamel-system   rfr-mcamel-common-redis-cluster-0
20.38 MiB       insight-system  insight-runbook-74dcd5d9c4-95r64
19.81 MiB       kangaroo-system kangaroo-upload-88c748c76-tgj6m
18.94 MiB       local-path-storage      local-path-provisioner-7c9c9849d8-nv6xr
18.00 MiB       mcamel-system   rfs-mcamel-common-redis-cluster-788fc878c5-ck7ch
17.06 MiB       insight-system  insight-ui-767686d5d5-ws4hf
14.94 MiB       insight-system  insight-agent-etcd-exporter-kv56t
13.19 MiB       kairship-system kairship-ui-8695c96b6d-d9nqx
12.81 MiB       ipavo-system    ipavo-ui-664bf5cb7c-9hkws
10.94 MiB       kangaroo-system kangaroo-ui-6474b87545-w4rcj
9.44 MiB        ghippo-system   ghippo-ui-5858f4c49d-kc2rx
9.31 MiB        kpanda-system   kpanda-ui-6f59cf7c4b-xdtwv
8.94 MiB        ghippo-system   ghippo-anakin-8ddbfbdfb-kphrc
8.00 MiB        istio-system    istio-os-init-6cv42

In total
        8769.37 MiB
```

Phase4 优化

```
1673.62 MiB     kube-system     kube-apiserver-g-master1
921.12 MiB      mcamel-system   mcamel-common-mysql-cluster-mysql-0
498.44 MiB      ghippo-system   ghippo-keycloakx-0
191.62 MiB      kube-system     etcd-g-master1
188.00 MiB      kube-system     calico-node-c6gph
152.69 MiB      mcamel-system   mcamel-common-minio-cluster-pool-0-0
144.81 MiB      kpanda-system   kpanda-controller-manager-b9c7b94c-4s7qc
141.56 MiB      kube-system     kube-controller-manager-g-master1
128.50 MiB      mcamel-system   mysql-mgr-operator-56c9cd6b79-ldc4v
127.19 MiB      kpanda-system   kpanda-clusterpedia-clustersynchro-manager-659d84f6bb-7wl6v
127.06 MiB      istio-system    istiod-5766cdd849-z9xsv
109.19 MiB      kpanda-system   kpanda-apiserver-6f84c4ddbb-6xp6t
107.75 MiB      istio-system    istio-ingressgateway-54cbfcf8f5-g2ztp
98.88 MiB       kpanda-system   kpanda-clusterpedia-apiserver-84865f9c94-4jsww
84.50 MiB       mcamel-system   mysql-operator-0
78.88 MiB       kpanda-system   kpanda-egress-7cfd449445-cc8km
77.00 MiB       ipavo-system    ipavo-86b896b99f-qrrs5
70.06 MiB       kube-system     calico-kube-controllers-758c9796d-bgvrs
58.12 MiB       ghippo-system   ghippo-controller-manager-7f5475db45-kfdfc
58.00 MiB       kairship-system kairship-apiserver-68dcd8df8c-zlm6l
57.75 MiB       kpanda-system   kpanda-ingress-f48855c68-cz6rb
55.38 MiB       kube-system     kube-scheduler-g-master1
54.12 MiB       ghippo-system   ghippo-apiserver-69585c76cb-c2jv8
50.50 MiB       mcamel-system   minio-operator-547dd9696-z5scf
50.19 MiB       kpanda-system   kpanda-bindings-syncer-6d7b974746-gr9tm
47.44 MiB       kangaroo-system kangaroo-apiserver-6c6cd7fc59-bvwcl
43.88 MiB       kairship-system kairship-controller-manager-5c94dc5f89-7wnw5
42.50 MiB       kangaroo-system kangaroo-controller-manager-9f968dbd5-w2j4b
41.31 MiB       kube-system     coredns-5949cf676c-qw5pq
36.81 MiB       registry-system registry-docker-registry-556df79d9b-4hwqk
36.25 MiB       kube-system     kube-proxy-spk7f
35.12 MiB       kpanda-system   kpanda-cloudtty-controller-manager-774456884c-77v54
34.38 MiB       ghippo-system   ghippo-auditserver-5fc74bc9bc-b86kn
32.88 MiB       mcamel-system   redis-cluster-operator-redis-ha-operator-5b5ffbdc9c-hbjgc
30.75 MiB       mcamel-system   console-fd69447c8-522hx
30.62 MiB       kpanda-system   kpanda-clusterpedia-controller-manager-7b4bff86dd-4g64d
30.25 MiB       kairship-system kairship-karmada-operator-5f7578687f-p6lps
29.50 MiB       mcamel-system   redis-cluster-operator-5fff5db855-78lnr
27.75 MiB       kube-system     dns-autoscaler-5778cd5b4d-l67vw
27.00 MiB       kairship-system kairship-proxy-5bcd77bb7d-t2hmv
21.44 MiB       mcamel-system   rfr-mcamel-common-redis-cluster-0
21.00 MiB       kangaroo-system kangaroo-upload-88c748c76-tgj6m
19.19 MiB       local-path-storage      local-path-provisioner-7c9c9849d8-nv6xr
```

```
17.44 MiB       mcamel-system   rfs-mcamel-common-redis-cluster-788fc878c5-ck7ch
13.19 MiB       kairship-system kairship-ui-8695c96b6d-d9nqx
12.75 MiB       ipavo-system    ipavo-ui-664bf5cb7c-9hkws
11.00 MiB       kangaroo-system kangaroo-ui-6474b87545-w4rcj
9.62 MiB        kpanda-system   kpanda-ui-6f59cf7c4b-xdtwv
9.12 MiB        ghippo-system   ghippo-ui-5858f4c49d-kc2rx
8.94 MiB        ghippo-system   ghippo-anakin-8ddbfbdfb-kphrc
7.94 MiB        istio-system    istio-os-init-6cv42

In total
        5983 MiB
```

结论

此轻量化方案是通过在充足内存下的完整 K8s + dce5 社区版，不断裁剪缩容而成的静置状态，从统计内存数值来看，内存刚好能满足。

在实际真实场景下，基于 8Gi 内存环境，按序安装（自适应裁剪），但因尾部涉及手动执行脚本， 会导致触发 deploy 滚动更新，临时内存需求激增，另操作系统本身也会占用一部分动态内存资源；

故综合情况下，实际上 8G 内存环境中运行 DCE5 轻量化裁剪环境，资源依然不足；

即 **安装状态所需内存 != 静置状态内存**

以 Phase 4 为最终裁剪目标来看，至少需要 10G+ 内存。 以 Phase 3 为最终裁剪目标来看（包含可观测性组件），至少需要 12G+ 内存。

如何通过安装器实现部署轻量化，参见安装器轻量化部署方案。

安装器实现轻量化部署

本文档介绍如何通过安装器实现 DCE5.0 轻量化部署。如果您想了解轻量化具体做了哪些裁剪优化， 请参考轻量化部署裁剪方案验证。

准备事项

配置准备

1. 组件配置文件 manifest.yaml 示例

   在具体组件版本应与离线包中版本一致。

   manifest.yaml

```
apiVersion: manifest.daocloud.io/v1alpha1
kind: DCEManifest
metadata:
  creationTimestamp: null
global:
  helmRepo: https://release.daocloud.io/chartrepo
  imageRepo: release.daocloud.io
infrastructures:
  istio:
    enable: true
    version: 1.16.1
  mysql:
    kpandaMode: master-slave # mgr|master-slave
    kpandaModeEnable: true
    commonMode: master-slave # mgr|master-slave
```

```
      commonModeEnable: true
      version: 8.0.29
      cpuLimit: 1
      memLimit: 2Gi
      enableAutoBackup: true
      pvcSize: 15Gi
      pvcSizeClusterPedia: 25Gi
    redis:
      version: 6.2.6-debian-10-r120
      cpuLimit: 400m
      memLimit: 500Mi
components:
      kubean:
        enable: true
        helmVersion: v0.18.4
        helmRepo: https://kubean-io.github.io/kubean-helm-chart
        variables:
      ghippo:
        enable: true
        helmVersion: 0.29.0
        variables:
      kpanda:
        enable: true
        helmVersion: 0.31.0
        variables:
      kangaroo:
        enable: true
        helmVersion: 0.20.0
        variables:
      kairship:
        enable: true
        helmVersion: 0.22.0
        variables:
      mcamel-mysql:
        enable: false
        helmVersion: 0.21.0
        variables:
      mcamel-redis:
        enable: false
        helmVersion: 0.21.0
        variables:
      mcamel-minio:
        enable: false
        helmVersion: 0.17.0
```

```
    variables:
```

2. 集群配置文件 ClusterConfig.yaml 示例

ClusterConfig.yaml

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
  creationTimestamp: null
spec:
  clusterName: my-cluster

loadBalancer:
  type: NodePort

masterNodes:
  - nodeName: "g-master1"
    ip: 10.7.10.7
    ansibleUser: "root"
    ansiblePass: "dangerous@2022"

ntpServer:
  - 0.pool.ntp.org
  - ntp1.aliyun.com
  - ntp.ntsc.ac.cn

fullPackagePath: "/home/offline"
osRepos:
  type: builtin
  isoPath: "/home/iso/Kylin-Server-V10-SP3-2403-Release-20240426-arm64.iso"
  osPackagePath: "/home/ospkgs/os-pkgs-kylin-v10sp3-v0.17.5.tar.gz"

imagesAndCharts:
  type: builtin

binaries:
  type: builtin
```

离线包准备

安装包下载需要在线网络环境，请在到达用户现场之前，提前下载好。

1. 下载安装器的离线包

安装包版本需高于（包含）v0.21.0 版本，推荐下载官网最新版本。

名称	版本	下载	更新日期
offline-v0.21.0-arm64.tar	v0.21.0	下载	2024-11-04
offline-v0.21.0-amd64.tar	v0.21.0	下载	2024-11-04

2. 准备好 jq 工具

名称	版本	下载	更新日期
jq-linux-arm64	V1.7.1	下载	2024-11-04
jq-linux-arm64	V1.7.1	下载	2024-11-04

脚本准备

请在到达用户现场之前，提前下载好。

1. 清理 Istio 边车脚本： clean_istio_proxy.sh
2. 部署 register 脚本： mv-registry.sh

安装步骤

安装依赖项

开始安装前，需先安装好前置依赖项。

```
export VERSION=v0.20.0
curl -LO https://proxy-qiniu-download-
public.daocloud.io/DaoCloud_Enterprise/dce5/install_prerequisite_${VERSION}.sh
curl -LO https://qiniu-download-
public.daocloud.io/DaoCloud_Enterprise/dce5/prerequisite_${VERSION}_amd64.tar.gz

export BINARY_TAR=prerequisite_${VERSION}_amd64.tar.gz
```

```
chmod +x install_prerequisite_${VERSION}.sh
./install_prerequisite_${VERSION}.sh offline full
```

开始安装

详细安装流程参考安装教程

```
export INSTALLER_CUSTOM_CPU_THRESHOLD=8
export INSTALLER_CUSTOM_MEM_THRESHOLD=8
export INSTALLER_COMMON_DB_SHARED=1
./dce5-installer cluster-create -d -c clusterConfig.yaml -m manifest.yml -z -j1,2,3,4,5,6,8,10,11,12
```

精简版 Insight 安装（可选）

追加 Insight 组件需要至少 12 G 的内存环境， 若资源受限，则建议不要安装，否则影响整体功能稳定性。

增加 Insight 精简组件的操作步骤如下：

1. 在 manifest.yaml 底部新增如下内容

   manifest.yaml

   ```
   insight:
     enable: true
     helmVersion: 0.30.0-rc2
     variables:
   insight-agent:
     enable: true
     helmVersion: 0.30.0-rc2
   ```

   Note

   具体组件版本应与离线包中版本一致。

2. 再次执行安装器的第 11、12 步

   ```
   ./dce5-installer cluster-create -c clusterConfig.yaml -m manifest.yml -z -j11,12
   ```

3. 保存下述文件为 insight-server-patch.yaml

   insight-server-patch.yaml

   ```
   global:
     kafka:
       enabled: false
   ```

```
    tracing:
      enable: false
      aggregator:
        enabled: false
      kafkaReceiver:
        enabled: false
    applyDependenciesCrds: true
    ghippo:
      applyCR: true
    elasticAlert:
      enable: false
  vector:
    enabled: false
```

4. 保存下述文件为 insight-agent-patch.yaml

insight-agent-patch.yaml

```
global:
  exporters:
    trace:
      enable: false
tailing-sidecar-operator:
  enabled: false
opentelemetry-kubernetes-collector:
  enabled: false
fluent-bit:
  enabled: false
```

5. 更新 insight-server 和 insight-agent

```
helm upgrade -n insight-system insight offline/dce5/insight-0.30.0-rc2.tgz --version 0.30.0-rc2 --reuse-values -f insight-server-patch.yml
helm upgrade -n insight-system insight-agent offline/dce5/insight-agent-0.30.0-rc2.tgz --version 0.30.0-rc2 --reuse-values -f insight-agent-patch.yml
```

安装后的裁剪

1. 清理 istio sidecar

```
# 需要提前安装 jq 工具
chmod +x jq-linux-amd64
mv jq-linux-amd64 /usr/local/bin/jq

# 执行 sidecar 清理脚本
bash clean_istio_proxy.sh
```

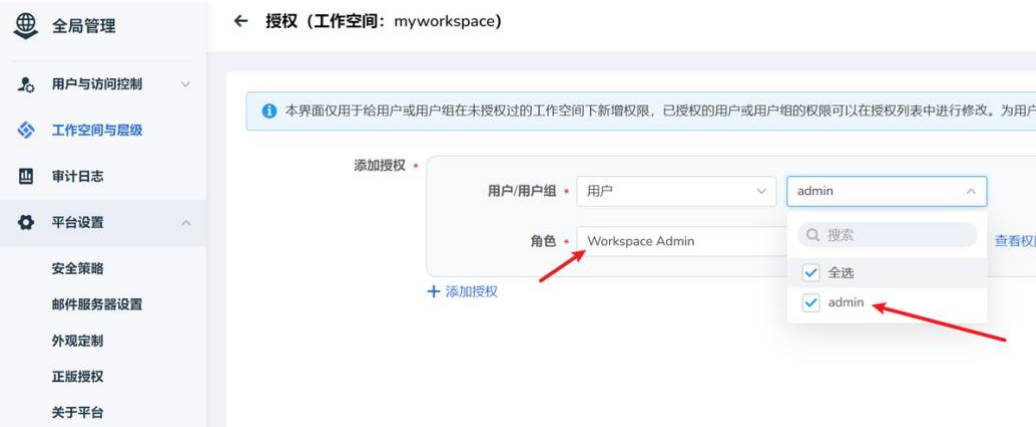
2. 部署 register，并手动纳管

export REGISTRY_NODEPORT=30143 # 设置新 registry nodeport
bash mv-registry.sh clusterConfig.yaml

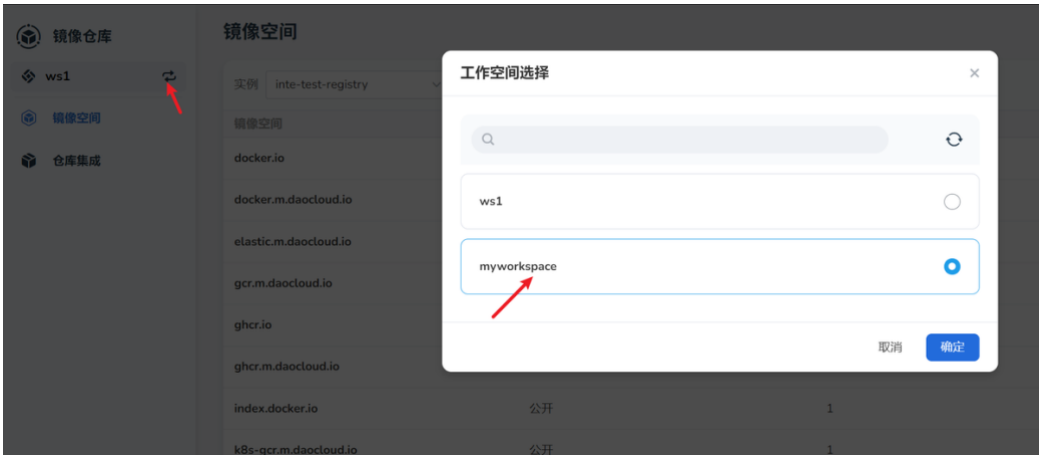
纳管操作步骤如下：

a. **全局管理 -> 工作空间与层级 -> 创建工作空间**

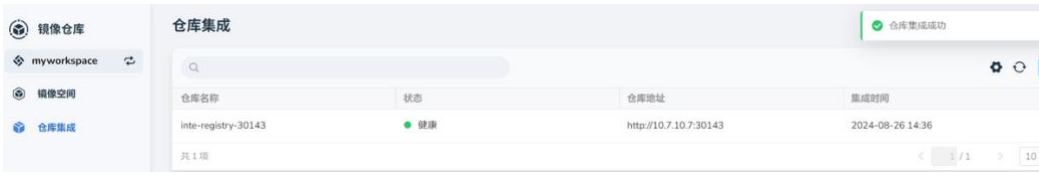
b. 添加授权，设置当前登录用户的角色为工作空间管理员

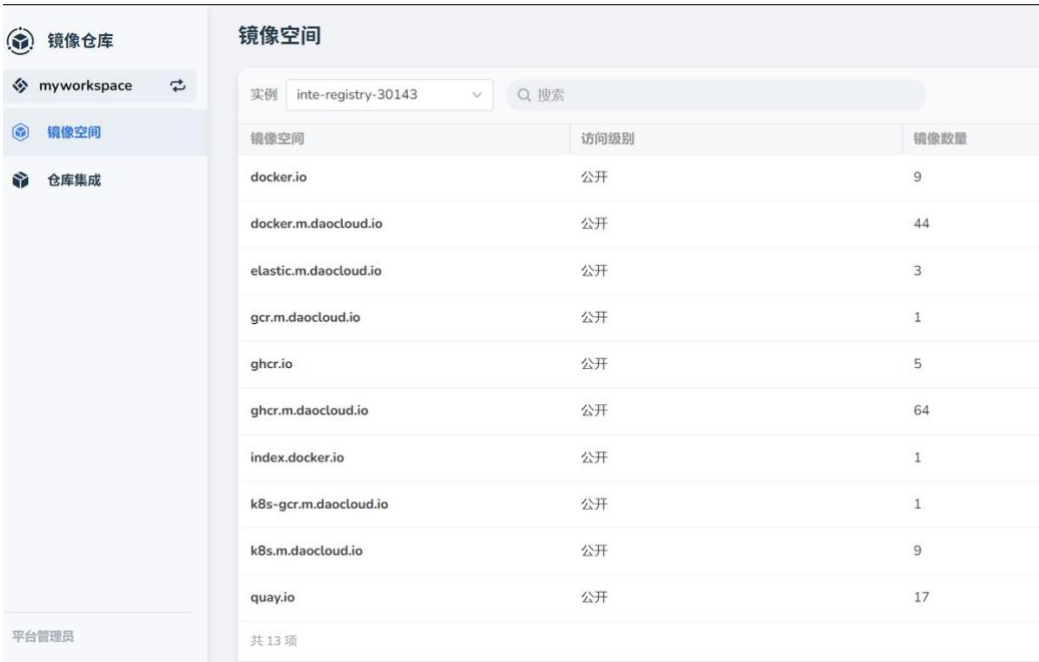


c. 镜像仓库选择工作空间



d. 仓库集成





3. 更新多云管理的配置

需要将多云管理组件中的火种镜像地址更新到新的 registry 当中。

```
# 假设当前节点 IP 为 10.5.14.160
# 且由之前的 REGISTRY_NODEPORT 环境变量得知端口为 30143
# 则生成关于 10.5.14.160:30143 的 patch 配置
# * 实际操作请以具体 IP 和端口为准 *
export REGISTRY_ADDR="10.5.14.160:30143"
cat << EOF >> kairship-patch.yml
```

```
global:
  imageRegistry: ${REGISTRY_ADDR}/release.daocloud.io
  kairship:
    instanceConfig:
      karmadaRegistry: ${REGISTRY_ADDR}/release.daocloud.io/karmada
      kubeRegistry: ${REGISTRY_ADDR}/release.daocloud.io/karmada
EOF

# 这里命令默认:
# 安装器离线包的解压路径为 /home/offline/,
# 且 kairship 的版本 ji 为 0.22.0-rc1
# * 实际操作请以具体路径和版本为准 *
export KAIRSHIP_VERSION="0.22.0-rc1"
export OFFLINE_RESOURCES_PATH="/home/offline"
helm upgrade -n kairship-system kairship \
    ${OFFLINE_RESOURCES_PATH}/dce5/kairship-${KAIRSHIP_VERSION}.tgz
\
    --reuse-values -f kairship-patch.yml \
    --version ${KAIRSHIP_VERSION}
```

4. 关停 kind 火种集群

   火种集群的关停应该在所有安装操作都已正常完成之后执行。

   ```
   tinder_name=$(yq '.spec.tinderKind.instanceName' clusterConfig.yaml)
   [ "$tinder_name" == null ] && tinder_name=my-cluster-installer
   kind delete cluster -n $tinder_name
   ```

以上，完成轻量化版本安装部署。

升级 DCE 5.0 组件

DCE 5.0 组件的升级包含升级 DCE 5.0 产品功能模块、升级 DCE 5.0 基础设施模块。

- DCE 5.0 产品功能模块由容器管理、全局管理、可观测性等十几个子模块构成，主要指 manifest.yaml 文件中的 components 部分。
- DCE 5.0 基础设施模块的组件特指 manifest.yaml 文件中的 infrastructures 部分。

Warning

- 由于 DCE 5.0 包含较多产品模块，所以使用安装器升级 DCE 5.0 组件时，建议逐版本升级，请勿跨多个版本进行升级！

- 升级 DCE 5.0 组件可能会覆盖您的业务数据，请先备份好数据，重要!!!

前提条件

- 您需要有一个 DCE 5.0 的集群环境，参阅离线化部署商业版
- 请确保您的火种机器还存活
- 请确认您想要升级的版本，参阅版本发布说明

离线升级操作步骤

本次操作步骤演示如何从 v0.20.0 升级到 v0.21.0。目

第 1 步：下载 DCE 5.0 离线包

可以在下载中心下载最新版本。 本文以 v0.21.0 为例。

CPU 架构	版本	下载地址
AMD64	v0.21.0	https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.21.0-amd64.tar
ARM64	v0.21.0	https://proxy-qiniu-download-public.daocloud.io/DaoCloud_Enterprise/dce5/offline-v0.21.0-arm64.tar

下载完毕后解压离线包，以 AMD64 架构离线包为例：

```
tar -xvf offline-v0.21.0-amd64.tar
```

第 2 步：配置 clusterConfig.yaml

- 需要确保集群配置文件 clusterConfig.yaml 与安装时使用的参数一致
- 目前仅对 imagesAndCharts 的 builtin 方式进行了测试

文件在解压后的离线包 offline/sample 目录下，参考配置文件如下：

clusterConfig.yaml

```
apiVersion: provision.daocloud.io/v1alpha4
kind: ClusterConfig
metadata:
spec:
  clusterName: my-cluster
  loadBalancer:
    type: metallb
    istioGatewayVip: 172.30.**.**/32
    insightVip: 172.30.**.**/32
  masterNodes:
    - nodeName: "g-master1"
      ip: 172.30.**.**
      ansibleUser: "root"
      ansiblePass: "*****"
  workerNodes:
    - nodeName: "g-worker1"
      ip: 172.30.**.**
      ansibleUser: "root"
      ansiblePass: "*****"
    - nodeName: "g-worker2"
      ip: 172.30.**.**
      ansibleUser: "root"
      ansiblePass: "*****"

  fullPackagePath: "/home/installer/offline"
  osRepos:
    type: builtin
    isoPath: "/home/installer/CentOS-7-x86_64-DVD-2207-02.iso"
    osPackagePath: "/home/installer/os-pkgs-centos7-v0.4.4.tar.gz"
  imagesAndCharts:
    type: builtin

  addonPackage:
  binaries:
    type: builtin    #
```

第 3 步：配置 manifest.yaml（可选）¶

文件在解压后的离线包 offline/sample 目录下。

配置 DCE 5.0 产品功能模块

DCE 5.0 产品功能模块的组件特指 manifest.yaml 文件中的 components 部分。 如果有些产品组件不需要升级，可以在对应组件下选择关闭。如果采用以下配置，更新时将不会对 Kpanda（容器管理）进行升级：

manifest.yaml

```
components:
  kpanda:
    enable: false
    helmVersion: 0.17.0
    variables:
```

配置 DCE 5.0 基础设施模块

DCE 5.0 基础设施模块的组件特指 manifest.yaml 文件中的 infrastructures 部分，如下配置就是基础设施中的 hwameiStor 组件：

manifest.yaml

```
infrastructures:
  hwameiStor:
    enable: true
    version: v0.10.4
    policy: drbd-disabled
```

Note

目前仅支持对当前环境中已经安装的产品组件进行升级，不存在的组件将会跳过升级步骤。

第 4 步：开始升级

升级 DCE 5.0 产品功能模块

执行升级命令：

```
./offline/dce5-installer cluster-create -c ./offline/sample/clusterConfig.yaml -
m ./offline/sample/manifest.yaml --upgrade gproduct
```

升级 DCE 5.0 基础设施模块

执行升级命令：

./offline/dce5-installer cluster-create -c ./offline/sample/clusterConfig.yaml -m ./offline/sample/manifest.yaml --upgrade infrastructure

升级 DCE 5.0

执行升级命令：

./offline/dce5-installer cluster-create --help

provision DaoCloud 5.0 clusters and install software stacks

Usage:
  dce5-installer cluster-create [flags]

Flags:
  -c, --clusterConfig string The cluster config file
  -y, --dry-run Dump installer scripts only
  -h, --help help for cluster-create
  -m, --manifest string manifest BOM file
      --max-tasks int Controls the maximum number of concurrent tasks. Must be positive number. (default 4)
      --multi-arch Whether to use the multi-arch image import mode.
      --serial Disable concurrent run
  -u, --upgrade string Choose the component which you want to upgrade, for example tinder,cluster,infrastructure,hwameistor,middleware,gproduct,addon .

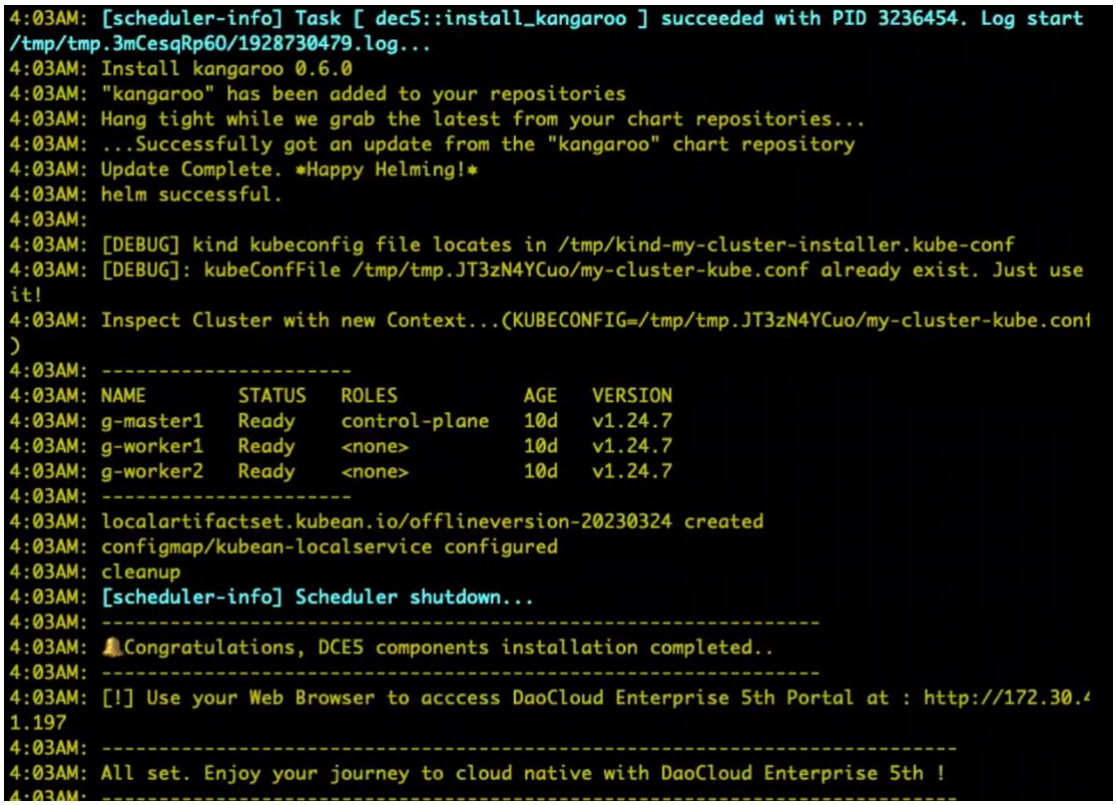
Global Flags:
  -s, --customized-script string (Optional)Your override script path
  -d, --debug Enable debug output
  -l, --logfile string The installation log to be dump (default "/var/log/dce5.log")
  -z, --minimized-replicas Whether to minimized all components replicas as small as possible.
  -j, --steps string (Optional)Debug Only, to specific a range of steps to be executed(format, 2+; 1,2,4; 3) (default "1+")
  -t, --tinder-host-ip string (Optional)The desired host IP on tinder node if it is not on default route.

./offline/dce5-installer cluster-create -c ./offline/sample/clusterConfig.yaml -m ./offline/sample/manifest.yaml --upgrade infrastructure,gproduct

升级参数说明：

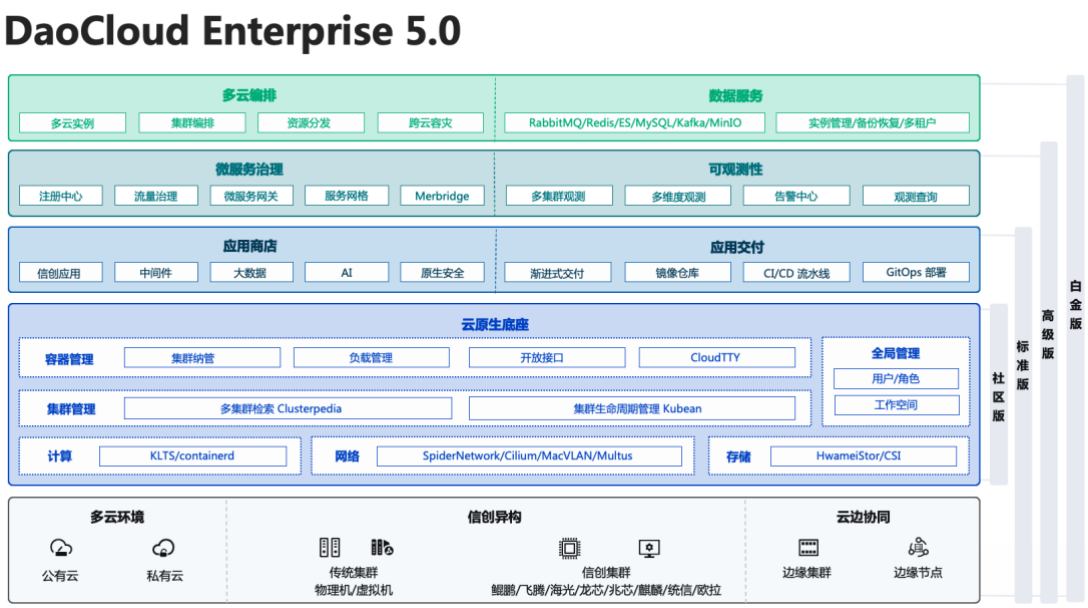
- install-app 或 cluster-create，代表安装 DCE 5.0 的安装模式类型。如果最初的环境是通过 cluster-create 来安装的，则升级时也采用这个命令
- --upgrade 可以简写为 -u，目前支持升级：
    - DCE 5.0 产品功能模块（gproduct）
    - 基础设施模块（infrastructure）
    - 本地存储模块（hwameistor）
- 如果需要一起升级产品功能模块和基础设施模块，则可以指定参数 --upgrade infrastructure,gproduct
- 安装器自 v0.12.0 支持了 --multi-arch 参数，主要是用户在当前环境存在多架构镜像时，进行升级过程中添加该参数可以避免覆盖原有的多架构镜像。

第 5 步：升级成功提示



标准版升级到白金版

DCE 5.0 支持多种版本，除社区版以外的其他版本均为商业版，本文演示如何从标准版升级到白金版。



前提条件

- 您需要有一个 DCE 5.0 的集群环境，参阅离线化部署商业版
- 请确保您的火种机器还存活，并且当前环境对应的离线包还在，否则需要重新下载

操作步骤

第 1 步：配置 manifest.yaml

将 manifest.yaml 文件中所有 enable: false 改为 enable: true。

manifest.yaml

```
skoala:
    enable: false
    helmVersion: 0.26.1
    variables:
 mspider:
    enable: false
    helmVersion: v0.18.0-rc2
    variables:
 mcamel-rabbitmq:
```

```
    enable: false
    helmVersion: 0.12.2
    variables:
  mcamel-elasticsearch:
    enable: false
    helmVersion: 0.9.2
    variables:
  ...
  mcamel-mysql:
    enable: false
    helmVersion: 0.10.2
    variables:
  mcamel-redis:
    enable: false
    helmVersion: 0.10.0-rc2
    variables:
  mcamel-kafka:
    enable: false
    helmVersion: 0.7.2
    variables:
  mcamel-minio:
    enable: false
    helmVersion: 0.7.2
    variables:
  mcamel-postgresql:
    enable: false
    helmVersion: 0.4.0-rc2
    variables:
  mcamel-mongodb:
    enable: false
    helmVersion: 0.2.0-rc2
    variables:
```

第 2 步：执行命令

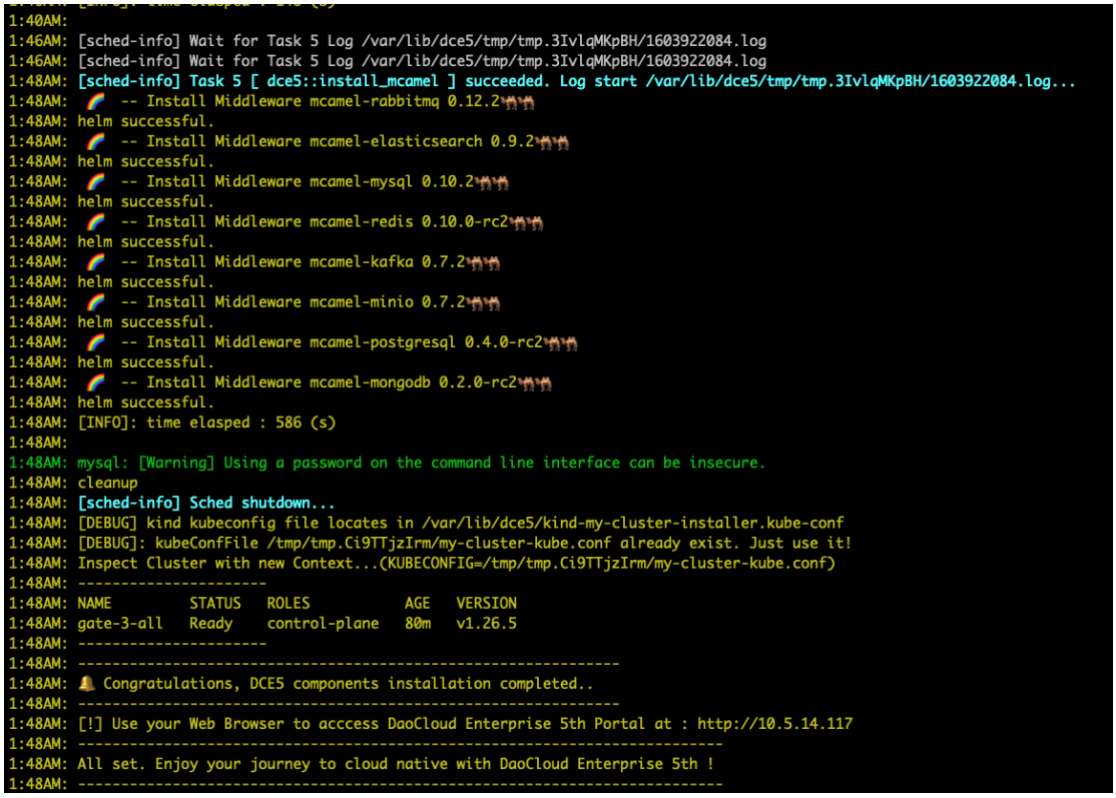
执行升级命令：

```
./offline/dce5-installer cluster-create -c ./sample/clusterConfig.yaml -m ./sample/manifest.yaml -j 11,12
```

clusterConfig.yaml 文件需要与安装时使用的参数保持一致。

-j 11,12 是目前执行标准版升级到白金版的必须参数。

第 3 步：升级成功提示



卸载

卸载社区版

依次执行以下命令从集群卸载 DCE 5.0，卸载过程中不会进行任何备份，请谨慎操作。

kubectl -n mcamel-system delete mysql mcamel-common-mysql-cluster
kubectl -n mcamel-system delete mysql mcamel-common-kpanda-mysql-cluster
kubectl -n mcamel-system delete elasticsearches mcamel-common-es-cluster-masters
kubectl -n mcamel-system delete redisfailover mcamel-common-redis-cluster
kubectl -n mcamel-system delete tenant mcamel-common-minio-cluster
kubectl -n ghippo-system delete gateway ghippo-gateway
kubectl -n istio-system delete requestauthentications ghippo
helm -n mcamel-system uninstall eck-operator mysql-operator redis-operator minio-operator
helm -n ghippo-system uninstall ghippo
helm -n insight-system uninstall insight-agent insight
helm -n ipavo-system uninstall ipavo

```
helm -n kpanda-system uninstall kpanda
helm -n istio-system uninstall istio-base istio-ingressgateway istiod
kubectl delete namespace mcamel-system ghippo-system insight-system ipavo-system
kpanda-system istio-system
```

离线资源导入

安装器 dce5-installer（仅支持 v0.9.0 版本及以上）支持 import-artifact 命令来导入离线资源。目前可被导入的离线资源包括：

- *.iso 操作系统 ISO 镜像文件
- os-pkgs-${disto}-${kubean_version}.tar.gz Kubean 提供的 osPackage 离线包
- offline-${install_version}-${arch}.tar 安装器全模式离线镜像包：
    - K8s 二进制和镜像
    - DCE 5.0 各个模块的镜像文件和 Chart 包

使用场景

场景一

当全局服务集群的操作系统与创建的工作集群操作系统不一致时，需要导入对应工作集群的操作系统的 ISO 镜像文件 和 osPackage 离线包。

例如在 CentOS 的全局服务集群上创建 Ubuntu 操作系统的工作集群。

场景二

在混合架构的离线部署场景中，需要在原有的 amd64 离线资源（K8s 二进制）和（K8s 镜像 + 各个模块的镜像文件和 Chart 包）基础上，导入并整合 arm64 的对应资源。

例如如何为工作集群添加异构节点。

导入命令介绍

需要提前下载好 dce5-installer 二进制文件。

导入操作系统 ISO 镜像文件

以导入 TencentOS-Server-3.1-TK4-x86_64-minimal-2209.3.iso 为例：

```
# 采用 build-in 内建模式部署火种集群时，可以不用指定 clusterConfig.yml 配置文件
dce5-installer import-artifact --iso-path=/home/iso/TencentOS-Server-3.1-TK4-x86_64-minimal-2209.3.iso

# 采用 external 外接模式部署火种集群时，需要指定 clusterConfig.yml 配置文件
dce5-installer import-artifact -c clusterConfig.yml --iso-path=/home/iso/TencentOS-Server-3.1-TK4-x86_64-minimal-2209.3.iso
```

导入 Kubean 提供的 osPackage 离线包

以导入 os-pkgs-tencent31-v0.6.2.tar.gz 为例：

```
# 采用 build-in 内建模式部署火种集群时，可以不用指定 clusterConfig.yml 配置文件
dce5-installer import-artifact --os-pkgs-path=/home/os-pkgs/os-pkgs-tencent31-v0.6.2.tar.gz

# 采用 external 外接模式部署火种集群时，需要指定 clusterConfig.yml 配置文件
dce5-installer import-artifact -c clusterConfig.yml --os-pkgs-path=/home/os-pkgs/os-pkgs-tencent31-v0.6.2.tar.gz
```

导入安装器离线镜像包 Offline 目录内容

```
# 采用 build-in 内建模式部署火种集群时，可以不用指定 clusterConfig.yml 配置文件
dce5-installer import-artifact --offline-path=/home/offline/

# 采用 external 外接模式部署火种集群时，需要指定 clusterConfig.yml 配置文件
dce5-installer import-artifact -c clusterConfig.yml --offline-path=/home/offline/
```

同时指定导入多种离线资源

```
# 采用 build-in 内建模式部署火种集群时，可以不用指定 clusterConfig.yml 配置文件
dce5-installer import-artifact \
    --offline-path=/home/offline/ \
    --os-pkgs-path=/home/os-pkgs/os-pkgs-tencent31-v0.6.2.tar.gz \
    --iso-path=/home/iso/TencentOS-Server-3.1-TK4-x86_64-minimal-2209.3.iso

# 采用 external 外接模式部署火种集群时，需要指定 clusterConfig.yml 配置文件
dce5-installer import-artifact -c clusterConfig.yml \
```

```
--offline-path=/home/offline/ \
--os-pkgs-path=/home/os-pkgs/os-pkgs-tencent31-v0.6.2.tar.gz \
--iso-path=/home/iso/TencentOS-Server-3.1-TK4-x86_64-minimal-2209.3.iso
```

安装排障

本页汇总了常见的安装问题及排障方案，便于用户快速解决安装及运行过程中遇到的问题。

UI 访问问题

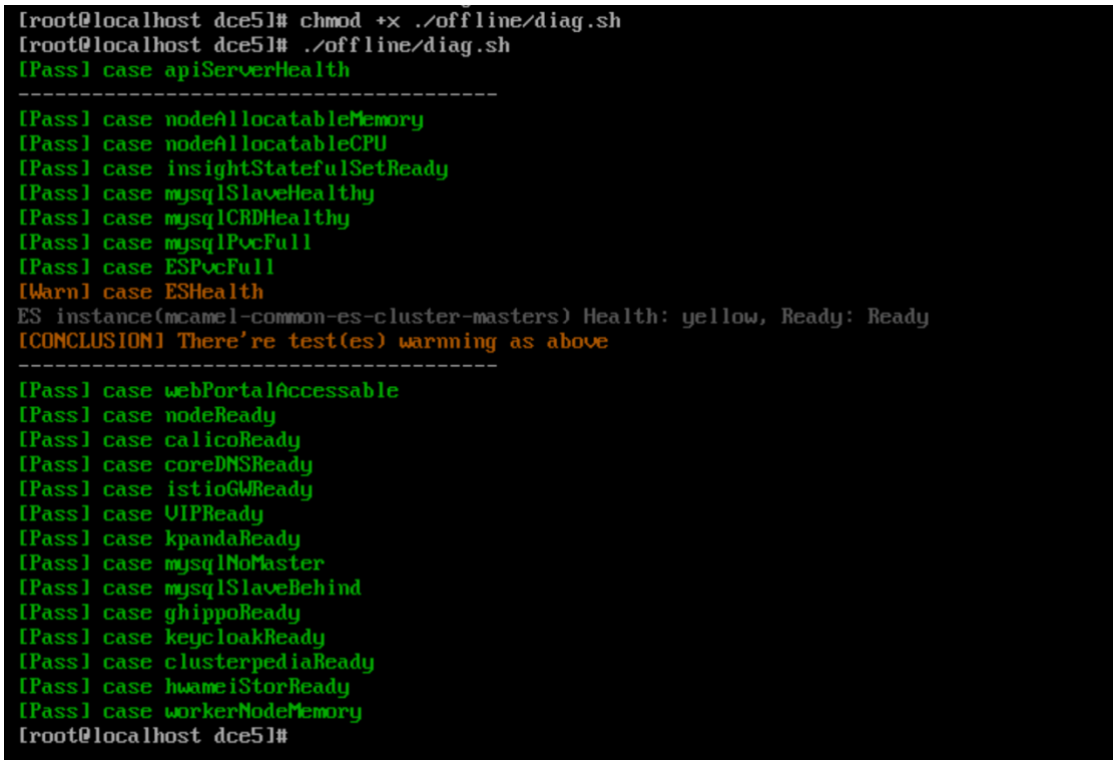
DCE 5.0 界面打不开时，执行 diag.sh 脚本快速排障

安装器自 v0.12.0 版本之后新增了 diag.sh 脚本，方便用户在 DCE 5.0 界面打不开时快速排障。

执行命令：

```
./offline/diag.sh
```

执行结果示例：



使用 Metallb 时 VIP 访问不通导致 DCE 登录界面无法打开

1. 排查 VIP 的地址是否和主机在同一个网段，Metallb L2 模式下需要确保在同一个网段
2. 如果是在全局服务集群中的控制节点新增了网卡导致访问不通，需要手动配置 L2Advertisement。

   请参考 Metallb 这个问题的文档。

火种节点问题

火种节点关机重启后，kind 集群无法正常重启

火种节点关机重启后，由于部署时在 openEuler 22.03 LTS SP2 操作系统上未设置 kind 集群开机自启动，会导致 kind 集群无法正常开启。

需要执行如下命令开启：

```
podman restart $(podman ps | grep installer-control-plane | awk '{print $1}')
```

如果其他环境中发生了上述场景，也可以执行该命令进行重启。

Ubuntu 20.04 作为火种机器部署时缺失 ip6tables

Ubuntu 20.04 作为火种机器部署，由于缺失 ip6tables 会导致部署过程中报错。

请参阅 Podman 已知问题。

临时解决方案：手动安装 iptables，参考 Install and Use iptables on Ubuntu 22.04。

禁用 IPv6 后安装时，火种节点 Podman 无法创建容器

报错信息如下：

```
ERROR: failed to create cluster: command "podman run --name kind-control-plane...
```

解决方案：重新启用 IPv6 或者更新火种节点底座为 Docker。

参阅 Podman 相关 Issue： podman 4.0 hangs indefinitely if ipv6 is disabled on system

火种节点 kind 容器重启后，kubelet 服务无法启动

kind 容器重启后，kubelet 服务无法启动，并报以下错误：

failed to initialize top level QOS containers: root container [kubelet kubepods] doesn't exist

解决方案：

- 方案 1：重启，执行命令 podman restart [kind] --time 120，执行过程中不能通过 Ctrl+C 中断该任务
- 方案 2：运行 podman exec 进入 kind 容器，执行以下命令：

```
for i in $(systemctl list-unit-files --no-legend --no-pager -l | grep --color=never -o .*.slice | grep kubepod);
do systemctl stop $i;
done
```

如何卸载火种节点的数据

商业版部署后，如果进行卸载，除了本身的集群节点外，还需要对火种节点进行重置，重置步骤如下：

需要使用 sudo rm -rf 命令删除这三个目录：

- /tmp
- /var/lib/dce5/
- /home/kind/etcd

证书问题

全局服务集群的 kubeconfig 在火种的副本需要更新

v0.20.0 之前的版本中，火种机上存储的全局服务集群的 kubeconfig 不会自动更新，v0.20.0 版本支持了自动更新，每个月执行一次。

之前的版本需要将 dce5-installer 更新到 v0.20.0 然后执行：

dce5-installer cluster-create -c clusterconfig.yaml -m mainfest.yaml --update-global-kubeconf

火种节点的 kind 集群本身的证书更新以及 kubeconfig

v0.20.0 之前的版本中，火种机上存储的 kind 集群的 kubeconfig 不会自动更新，v0.20.0 版本支持了自动更新，每个月执行一次。

之前的版本需要将 dce5-installer 更新到 v0.20.0 然后执行：

dce5-installer cluster-create -c clusterconfig.yaml -m mainfest.yaml --update-kind-certs

Contour 安装后，证书默认有效期仅一年，且不会自动 renew，过期后导致 contour-envoy 组件不断重启

v0.21.0 之前的版本，支持启用安装 Contour 组件，后续版本将不再支持，对于之前版本并且安装了 Contour 的客户，需要执行 helm upgrade 命令来更新证书有效期：

helm upgrade -n contour-system contour --reuse-values --set contour.contour.certgen.certificateLifetime=36500

操作系统相关问题

在 CentOS 7.6 安装时报错

在安装全局服务集群的每个节点上执行 modprobe br_netfilter，将 br_netfilter 加载之后就好了。

CentOS 环境准备问题

运行 yum install docker 时报错：

Failed to set locale, defaulting to C.UTF-8
CentOS Linux 8 - AppStream
93 B/s | 38 B 00:00
Error: Failed to download metadata for repo 'appstream': Cannot prepare internal mirrorlist: No URLs in mirrorlist

可以尝试下述方法来解决：

- 安装 glibc-langpack-en
- sudo yum install -y glibc-langpack-en
- 如果问题依然存在，尝试：
- sed -i 's/mirrorlist/#mirrorlist/g' /etc/yum.repos.d/CentOS-*
- sed -i 's|#baseurl=http://mirror.centos.org|baseurl=http://vault.centos.org|g' /etc/yum.repos.d/CentOS-*
- sudo yum update -y

社区版问题

kind 集群重装 DCE 5.0 时 Redis 卡住

问题：Redis Pod 出现了 0/4 running 很久的情况，提示：primary ClusterIP can not unset

1. 在 mcamel-system 命名空间下删除 rfs-mcamel-common-redis
2. kubectl delete svc rfs-mcamel-common-redis -n mcamel-system
3. 然后重新执行安装命令

社区版 fluent-bit 安装失败

报错：

DaemonSet is not ready: insight-system/insight-agent-fluent-bit. 0 out of 2 expected pods are ready

排查 Pod 日志是否出现下述关键信息：

[warn] [net] getaddrinfo(host='mcamel-common-es-cluster-masters-es-http.mcamel-system.svc.cluster.local',errrt11):Could not contact DNS servers

出现上述问题是一个 fluent-bit 的 bug，可以参考 aws/aws-for-fluent-bit 的一个 Issue： Seeing Timeout while contacting DNS servers with latest v2.19.1