灵雀云

全栈云原生开放平台v3.8.2

部署文档

北京凌云雀科技有限公司

目录

[1 前 言 1](#_Toc256000000)

[1.1 文档目的 1](#_Toc256000001)

[1.2 读者对象 1](#_Toc256000002)

[1.3 修订记录 1](#_Toc256000003)

[1.4 技术支持 1](#_Toc256000004)

[2 常规部署 2](#_Toc256000005)

[2.1 文档概述 2](#_Toc256000006)

[2.1.1 文档介绍 2](#_Toc256000007)

[2.1.2 名词解释 2](#_Toc256000008)

[2.1.3 部署架构 2](#_Toc256000009)

[2.1.4 规模/压力 3](#_Toc256000010)

[2.2 部署平台 4](#_Toc256000011)

[2.2.1 部署流程 4](#_Toc256000012)

[2.2.2 容量规划 5](#_Toc256000013)

[2.2.2.1 单点部署架构下，规模/压力1的容量规划 6](#_Toc256000014)

[2.2.2.2 标准部署架构下，规模/压力2 的容量规划 6](#_Toc256000015)

[2.2.2.3 数据中心部署架构下，规模/压力3的容量规划 7](#_Toc256000016)

[2.2.2.4 数据中心部署架构下，规模/压力4的容量规划 8](#_Toc256000017)

[2.2.2.5 日志容量规划 9](#_Toc256000018)

[2.2.3 硬件需求 10](#_Toc256000019)

[2.2.4 网络需求 10](#_Toc256000020)

[2.2.4.1 网络资源要求表 10](#_Toc256000021)

[2.2.4.2 网络配置要求表 11](#_Toc256000022)

[2.2.4.3 Global 平台四层转发规则 11](#_Toc256000023)

[2.2.4.4 业务服务集群转发规则 13](#_Toc256000024)

[2.2.5 软件需求 14](#_Toc256000025)

[2.2.5.1 快速配置命令 17](#_Toc256000026)

[2.2.6 安装介质 18](#_Toc256000027)

[2.2.6.1 下载介质 18](#_Toc256000028)

[2.2.6.2 验证介质 19](#_Toc256000029)

[2.2.7 开始部署 20](#_Toc256000030)

[2.2.7.1 部署过程 20](#_Toc256000031)

[2.2.7.2 验证平台 24](#_Toc256000032)

[2.2.7.3 访问平台 25](#_Toc256000033)

[2.2.7.4 删除安装器 25](#_Toc256000034)

[2.3 配置平台 25](#_Toc256000035)

[2.3.1 简介 25](#_Toc256000036)

[2.3.2 修改组件软件配置 25](#_Toc256000037)

[2.3.2.1 修改日志采集范围及日志、审计、事件等数据保存时间 26](#_Toc256000038)

[2.3.2.2 修改组件 limit 27](#_Toc256000039)

[2.3.2.3 修改apollo –es-enablealias 参数 27](#_Toc256000040)

[2.3.2.4 修改 Es分片数(ALAUDA\_ES\_SHARDING) 27](#_Toc256000041)

[2.3.2.5 修改 Es 节点数 27](#_Toc256000042)

[2.3.2.6 修改 Es index 的副本(replicas) 28](#_Toc256000043)

[2.3.3 容器平台部署镜像仓库 28](#_Toc256000044)

[2.3.3.1 部署平台的时候选择不安装 devops 28](#_Toc256000045)

[2.3.3.2 部署 docker-registry 29](#_Toc256000046)

[2.3.3.3 工具链集成 30](#_Toc256000047)

[2.4 清理环境 31](#_Toc256000048)

[2.4.1 说明 31](#_Toc256000049)

[3 轻量裸金属部署 32](#_Toc256000050)

[3.1 文档概述 32](#_Toc256000051)

[3.1.1 文档介绍 32](#_Toc256000052)

[3.1.2 名词解释 32](#_Toc256000053)

[3.1.3 部署架构 32](#_Toc256000054)

[3.2 部署平台 32](#_Toc256000055)

[3.2.1 部署流程 32](#_Toc256000056)

[3.2.2 容量规划 33](#_Toc256000057)

[3.2.2.1 部署架构一，轻量裸金属 规模/压力2 容量规划 33](#_Toc256000058)

[3.2.2.2 硬件需求 33](#_Toc256000059)

[3.2.3 网络需求 33](#_Toc256000060)

[3.2.4 软件需求 34](#_Toc256000061)

[3.2.5 安装介质 34](#_Toc256000062)

[3.2.5.1 下载介质 34](#_Toc256000063)

[3.2.5.2 合并介质 34](#_Toc256000064)

[3.2.6 开始部署 35](#_Toc256000065)

[3.2.6.1 部署平台 35](#_Toc256000066)

[3.2.6.2 部署插件 37](#_Toc256000067)

[3.3 常见故障 38](#_Toc256000068)

[3.3.1 DataVolumeError 38](#_Toc256000069)

# 前 言

## 文档目的

售前工程师依据本文档，规划项目方案。

项目经理依据本文档，申请硬件和网络资源，确认软件版本配置，保障交付环境满足项目需求。

实施工程师依据本文档，检查实施环境是否满足实施需求，依据文档部署环境。

平台管理员可以通过本文档，了解平台所需资源、部署方案。

## 读者对象

具备一定的linux、容器、Kubernetes及网络知识，想要安装和配置平台的实施工程师和平台管理员。

规划平台架构的售前工程师。

负责整个项目生命周期的项目经理。

## 修订记录

|  |  |  |
| --- | --- | --- |
| 日志节点仅支持选择 3 个文档版本 | 发布日期 | 修订内容 |
| v1.0 | 2022-08-10 | 创建文档 |
|  |  |  |

## 技术支持

可通过以下联系方式获得技术支持，或将你的意见反馈给我们。

|  |  |
| --- | --- |
| 技术支持 | 联系方式 |
| 公司地址 | 北京市海淀区知春路56号中航科技大厦二层 |

# 常规部署

## 文档概述

### 文档介绍

本文档介绍了global平台的部署架构及容量规划方法，给出了部署平台必须遵守的硬件资源需求、网络资源需求和软件资源需求。

本文档详细介绍了平台安装过程，还包括在基础平台部署成功后，基于平台的系列产品的安装过程。

### 名词解释

|  |  |
| --- | --- |
| 名词 | 解释 （部署时需要重点理解的部分名词） |
| global VIP | 这是核心资源，承载平台的 kubernetes 集群暴露的 apiserver 就是用这个IP地址。global 平台、业务服务集群和操作人员必须都能访问到这个地址。 |
| 平台访问地址(external IP) | 1、操作人员使用、维护平台的访问IP地址，也是业务集群与 global 交互的时候使用的地址。 2、有容灾需求，则此地址必须为域名。 |
| 内网或管理网 | 客户对某个网络分区的称呼，平台一般会位于这个网络里。 |
| 外网或业务网 | 客户对某个网络分区的称呼，业务服务集群一般位于这个网络里。 |
| 操作人员 | 操作人员使用位于某个网络分区内的电脑，通过浏览器访问平台的 UI，使用平台功能。 |
| 平台（global） | 也叫 global，指的是容器管理平台本身，包括但不限于 asm、devops等等产品，在配置 lb 转发及防火墙规则的章节，这个名词指承载 global 组件的 Kubernetes 集群的所有节点。 |
| global 集群 | 也称管理集群，是一个高可用的 kubernetes集群，用于承载global 平台的各个组件。 |
| 业务服务集群 | 用于承载客户业务服务的 Kubernetes 集群，在配置 lb 转发及防火墙规则的章节，这个名词也是指上述这个集群的所有节点。 |
| Kubernetes api server VIP | 高可用集群需要一个 VIP，配置到独立的 lb 上，作为集群内master 节点上的 kube-api 的负载均衡的入口，达到高可用的目的。 |

### 部署架构

平台有单点部署、标准部署、数据中心部署三种部署架构，不同部署架构的简要描述、优缺点和推荐适用场景请看下表：

|  |  |  |  |
| --- | --- | --- | --- |
|  | 单点部署 | 标准部署 | 数据中心部署 |
| 部署描述 | 管理集群单点部署，业务复用管理集群 | 平台和业务混合部署，平台管理组件部署在业务集群内 | 管理集群多节点部署，业务集群与管理集群隔离 |
| 部署优点 | 资源占用少，可快速部署 | 支持高可用 平台资源相对较少 | 支持高可用 平台和业务分离，业务稳定性高 支持容灾能力 |
| 部署缺点 | 仅基础能力 非高可用 无灾备方案 不支持扩展为标准部署和数据中心部署 | 无灾备方案 不支持扩展为数据中心部署 | 资源占用大 |
| 适用场景 | Demo演示 平台功能试用 | 中小规模生产环境 ISV底座、一体机、超融合 | 大规模业务场景 业务量会持续增长 有容灾需求 |
| 部署规模详见[容量规划](#scroll-bookmark-13) | Pod数：≦10 业务集群：0 | Pod数 ：≦300 业务集群：≦1 业务集群节点：≦10 | Pod数：≦10000 业务集群：≦30 业务集群节点：≦500 |

在规划云平台之前请先阅读本章，了解三种部署架构适应的场景，然后预估项目投产后的规模和压力，最后阅读本文档[容量规划](#scroll-bookmark-13)中给出的推荐部署方案，选择最接近项目规模的方案，并基于这个方案来规划部署容量和实施方案。

用于部署云平台的服务器的硬件配置必须大于等于本文档[硬件需求](#scroll-bookmark-14)中的要求，如果硬件配置型号低于本文档的要求或 cpu 架构是 ARM，服务器配置要酌情增加，比如8c16g 的配置增加到12c24g 或16c32g。

### 规模/压力

本章给出常见的几种规模和压力。只要提供符合 [容量规划](#scroll-bookmark-13) 章节要求的硬件资源，并且按照文档的指引配置软件，那么就可以在不超过本章给出的相应的资源数量前提下，承载本章给出的并发压力。比如以规模/压力2为例，能够同时支撑100个应用、100个ingress等等资源，并且支持10并发的压力（10并发的意思是平台的 api 同时处理10个请求，一般同时有100个用户在线使用平台，才能造成10并发的压力）。

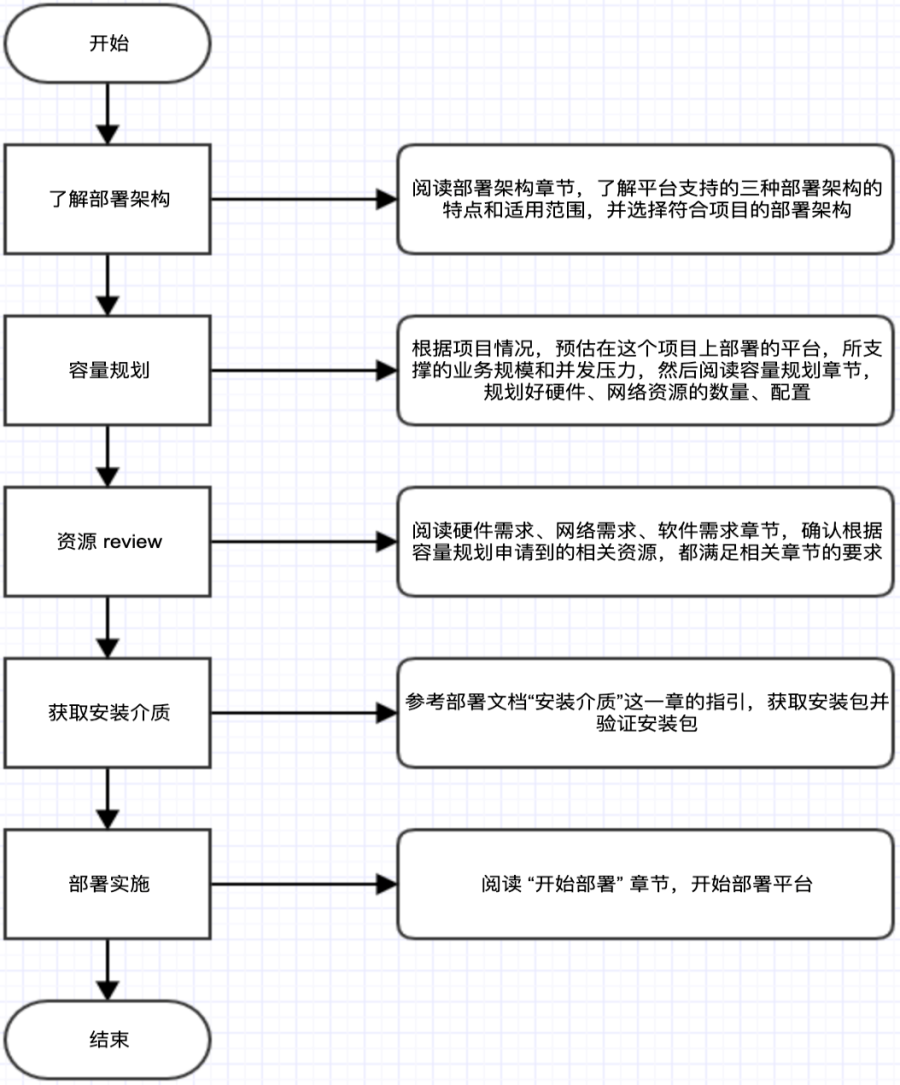
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 资源数量 | 规模/压力1 | 规模/压力2 | 规模/压力3 | 规模/压力4 | 备注 |
| 并发压力 | ≦1 | ≦10 | ≦100 | ≦500 |  |
| 集群 | 0 | ≦1 | ≦10 | ≦30 | 跑客户业务服务集群的数量 |
| 节点 | 1 | ≦10 | ≦50 | ≦500 | 数据中心部署，大集群500个节点，其他集群1个节点 |
| 应用 | ≦10 | ≦100 | ≦2000 | ≦1w | 一个应用一个 deployment，deploy副本是1，pod 内容器数量是1 |
| ingress | ≦10 | ≦100 | ≦200 | ≦1w |  |
| 内部路由 | ≦10 | ≦100 | ≦2000 | ≦1w |  |
| 项目 | ≦1 | ≦10 | ≦50 | ≦1000 |  |
| 命名空间 | ≦1 | ≦10 | ≦300 | ≦3000 |  |
| 用户 | ≦10 | ≦100 | ≦1000 | ≦5000 |  |
| 流水线 | ≦5 | ≦50 | ≦500 | ≦1500 |  |
| 镜像仓库 | ≦10 | ≦50 | ≦500 | ≦1500 |  |
| 代码仓库 | ≦10 | ≦50 | ≦100 | ≦200 |  |
| 日志/秒 | ≦20 | ≦100 | ≦2000 | ≦22000 | 如果日志规模大，请单独参考[日志容量规划](#scroll-bookmark-16) |

## 部署平台

平台也称之为 global，具体请看本文档[名词解释](#scroll-bookmark-10)。

### 部署流程

|  |
| --- |
| **注意**：一定要满足文档的需求才能部署成功，请按照以下顺序阅读了解具体需求。 |



### 容量规划

本章给出了满足三种场景的推荐的部署方案，包括硬件数量和配置要求，软件的配置要求。

如果满足硬件要求表中的独立分区要求，会存在因日志、模板或其他数据过多而耗尽系统资源，导致系统不可用的问题。

#### 单点部署架构下，规模/压力1的容量规划

##### 硬件配置

|  |  |
| --- | --- |
| **服务器角色** | **allinone** |
| **服务器数量** | 1 |
| **服务器用途** | 承载平台及业务服务的所有功能 |
| **是否可选** | 必须 |
| **CPU数量** | 32 |
| **内存容量** | 64G |
| **/分区可用空间** | 50G |
| **/cpaas 分区可用空间** | 400G |
| **/var/lib可用空间** | 50G |
| **/opt可用空间** | 30G |
| /var/lib/docker | 100G |
| **/var/lib/docker** | xfs |
| **单独的块设备（可选）** |  |

##### 软件配置

|  |
| --- |
| **注意**：软件配置方法，请参考 [修改组件软件配置](#scroll-bookmark-23) 章节 |

|  |  |  |
| --- | --- | --- |
| 组件 | 配置项 | 配置值 |
|  | 日志采集范围 | 不搜集产品日志和系统日志 |
|  | 日志存储策略 | 日志、审计、事件保存1天 |
| 业务服务使用的资源 | cpu 和内存 | 跑在环境上的客户的所有业务服务，所使用的资源之和不的超过2c4g，如果超过，请相应增加硬件配置 |
| prometheus | 数据保存时间 | 7d |

#### 标准部署架构下，规模/压力2 的容量规划

##### 硬件配置

|  |
| --- |
| **注意**：标准部署架构中，承载 global 平台的 k8s 集群也同时承载客户的业务服务，下面表格中的硬件配置要求仅仅能满足 global 平台的要求。售前工程师需要评估业务服务所需的资源，并增加到硬件配置表中。 |

|  |  |
| --- | --- |
| **服务器角色** | **master&slave&global&log&monitor** |
| **服务器数量** | 3 |
| **服务器用途** | 承载平台及业务服务的所有功能 |
| **是否可选** | 必须 |
| **CPU数量** | 16 |
| **内存容量** | 32G |
| **/分区可用空间** | 50G |
| **/cpaas 分区可用空间** | 300G |
| **/cpaas/data/** | 1800G 日志 |
| **/cpaas/monitoring/** | 100G 监控 |
| **/var/lib可用空间** | 50G |
| **/opt可用空间** | 30G |
| /var/lib/docker | 100G |
| **/var/lib/docker** | xfs |
| **单独的块设备（可选）** |  |

##### 软件配置

|  |
| --- |
| **注意：软件配置方法，请参考** [**修改组件软件配置**](#scroll-bookmark-23) **章节。** |

|  |  |  |
| --- | --- | --- |
| 组件 | 配置项 | 配置值 |
|  | 日志采集范围 | 不搜集产品日志和系统日志 |
|  | 日志存储策略 | 日志保留7天 |
| 所有平台组件 | cpu limit | 删掉所有组件的cpu的 limit |
| 日志和监控组件（nevermore、lanaya 等自研组件，和 es 、kafka 一样属于日志组件） | memory limit | 除日志和监控组件外，所有组件的内存的 limit 删掉，日志和监控组件的内存 limit 参考推荐最小配置方案的值 |
| prometheus | 数据保存时间 | 7d |

#### 数据中心部署架构下，规模/压力3的容量规划

##### 硬件配置

|  |  |  |  |
| --- | --- | --- | --- |
| **服务器角色** | **master&slave&global&log&monitor** | **业务集群master** | **业务集群slave** |
| **服务器数量** | 3 | 高可用最少3台 | 视业务而定 |
| **服务器用途** | 承载平台及业务服务的所有功能 | 业务服务集群的 master | 业务服务集群的计算节点 |
| **是否可选** | 必须 | 可选 | 可选 |
| **CPU数量** | 24 | 8 | 视业务而定 |
| **内存容量** | 48G | 16G | 视业务而定 |
| **/分区可用空间** | 50G | 50G | 50G |
| **/cpaas 分区可用空间** | 300G | 300G部署监控的节点需要 | 无 |
| **/cpaas/data/** | 2000G 日志 | 无 | 无 |
| **/cpaas/monitoring/** | 100G 监控 | 无 | 无 |
| **/var/lib可用空间** | 50G | 50G | 50G |
| **/opt可用空间** | 30G | 30G | 30G |
| /var/lib/docker | 100G | 100G | 100G |
| **/var/lib/docker** | xfs | xfs | xfs |
| **单独的块设备（可选）** |  |  |  |

##### 软件配置

|  |
| --- |
| **注意：软件配置方法，请参考** [**修改组件软件配置**](#scroll-bookmark-23) **章节。** |

|  |  |  |  |
| --- | --- | --- | --- |
| 集群 | 组件 | 配置项 | 配置值 |
| global 平台 | hermes | limit | 2C1G |
| devops-api | 实例数 | 3 |
| es | limit | 4C8G |
| prometheus | limit | 2C16G |
| 业务服务集群 | prometheus | limit | 4C60G |

#### 数据中心部署架构下，规模/压力4的容量规划

##### 硬件配置

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 服务器角色 | master | slave&global &monitor | slave&global&log | 业务集群master | 业务集群slave | 业务集群 monitor |
| 服务器数量 | 3 | 1或2(高可用) | 5 | 3\*业务集群数 | 最大500 | 1或2(高可用) |
| 服务器用途 | 平台集群的 master | 平台及监控组件 | 平台及日志组件 | 业务服务集群的 master | 客户服务的计算节点 | 运行监控组件 |
| 是否可选 | 必须 | 必须 | 必须 | 必须 | 必须 | 必须 |
| CPU数量 | 16 | 16 | 16 | 8 | 视业务而定 | 16 |
| 内存容量 | 32G | 32G | 32G | 32G | 视业务而定 | 64G |
| /分区 | 50G | 50G | 50G | 50G | 50G | 50G |
| /cpaas | 300G | 无 | 无 | 无 | 无 | 500G |
| /cpaas/data | 无 | 无 | 500G | 无 | 无 | 无 |
| /cpaas/monitoring/ | 无 | 500G | 无 | 无 | 无 | 无 |
| /var/lib | 50G | 50G | 50G | 50G | 50G | 50G |
| /opt | 30G | 30G | 30G | 30G | 30G | 30G |
| /var/lib/docker | 无 | 100G | 100G | 无 | 100G | 100G |
| /var/lib/dock文件系统 | xfs | xfs | xfs | xfs | xfs | xfs |
| 单独的块设备（可选） |  |  |  |  |  |  |

##### 软件配置

|  |
| --- |
| **注意：软件配置方法，请参考** [**修改组件软件配置**](#scroll-bookmark-23) **章节。** |

|  |  |  |  |
| --- | --- | --- | --- |
| 集群 | 组件 | 配置项 | 配置值 |
| global 平台 | 除监控和日志外所有组件 | 实例数 | 4 |
|  | cpu 和memory limit | 删掉limit |
| kafka zk | cpu limit | 删掉limit，pod 没有 cpu 的 limit |
|  | memory limit | 8G |
| es | 实例数 | 5 |
|  | cpu limit | 删掉limit，pod 没有 cpu 的 limit |
|  | memory limit | 8G |
| prometheus | limit | 去掉 cpu limit，memory24G |
| 业务服务集群 | metis | limit | 4C8G |
| 应用同步线程数 | 500 |
| prometheus | limit | 8c56g |
| 数据保存时间 | 12h |
| prometheus-node-exportor | limit | 1C2G |

#### 日志容量规划

|  |
| --- |
| **注意：软件配置方法，请参考** [**修改组件软件配置**](#scroll-bookmark-23) **章节。** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **≦12000条/s** | **≦22000条/s** | **≦44000条/s** |
| 硬件配置 | 跑日志组件的服务器数量 | 3 | 5 | 11 |
| cpu | 16C | 16C | 16C |
| 内存 | 32G | 32G | 32G |
| /cpaas/data 数据保留7天 | 2500G | 1800G | 3500G |
| 软件配置 | es limit （删掉 cpu limit） | 8G | 8G | 8G |
| Es的index的副本(replicas) | 默认 | 默认 | 1 |
| Es分片数(ALAUDA\_ES\_SHARDING) | 默认 | 默认 | 11 |
| kafka limit （删掉 cpu limit） | 8G | 8G | 8G |
| apollo --es-enablealias 参数 | false | false | false |
| 业务集群 nevermore limit | 默认 | 4C4G | 4C4G |
| lanaya 副本数量 | 默认 | 默认 | 5 |
| lanaya limit （删掉 cpu limit） | 默认 | 默认 | 8G |

### 硬件需求

|  |  |  |
| --- | --- | --- |
| 硬件 | 具体要求 | 最低型号或配置 |
| CPU | 主频不小于 2.5GHz，在 iaas 层不得超售。若不满足，cpu 数量需酌情增加。如果是 arm 的 cpu，数量增加1.5 倍，建议增加2倍。 | Intel 8255c |
| 内存 | 在 iaas 层不得超售。 | 六通道DDR4 |
| 硬盘 | 单个块设备的iops > 2000 ；吞吐量 > 200M/s 。 | ssd |
| GPU | 仅对 418.87.00 CUDA Version: 10.1 驱动的 GPU进行了充分测试。 | Nvidia |

### 网络需求

|  |
| --- |
| **注意**： 本文档要求所有资源，除 LB 外都是部署平台必须提供的，不能提供就无法部署环境。  如果现场无法提供LB ，会有如下隐患：   * 实施在现场通过软件方式（ haproxy+keeplived ）部署的 LB 性能不如硬件 LB （比如 F5），这是个瓶颈。 * 软件方式部署的 LB ，受限现场同事对 keeplived 的熟悉程度和值班同事反应速度，常常造成环境因为 LB 的问题造成几个小时甚至更长时间不可用，修复时间长，可靠性和可用性无法保证， SLA 指标会降低。 |

#### 网络资源要求表

|  |  |  |  |
| --- | --- | --- | --- |
| 资源 | 可选 | 数量 | 说明 |
| 证书 | 可选 | 1 | 如果不提供证书，部署脚本会自动生成一个证书，但是浏览器访问平台 UI 会提示安全警告，因为证书不是认证机构签发的。 |
| 平台访问地址(external IP) | 必须 | 1 | 域名或 ip 地址，详细介绍请参考名词解释中"平台访问地址"。 |
| global VIP | 必须 | 1 | 详细介绍请参考名词解释中"global VIP"的介绍。 |
| Kubernetes api server VIP | 必须 | 多个 | 生产环境必须，给高可用的 Kubernetes 集群的 kube-api 使用，每一个高可用的 Kubernetes 集群都需要一个 vip。 |
| ALB VIP | 必须 | 多个 | 如果客户使用 alb 有高可用需求，这是必须的资源。每个客户业务服务集群的负载均衡器需要一个 VIP（注意，是每个负载均衡器需要一个 vip，不是每个 alb 实例需要一个 vip）。 |
| 内网 LB | 必须 | 1 | 生产环境必须，否则无法达到高可用要求。类似 F5 的负载均衡设备，Kubernetes api server vip 配置到这个负载均衡设备上，global vip 也配置到这个负载均衡上。 |
| 外网 LB | 必须 | 1 | 生产环境必须，否则无法达到高可用要求。如果客户没有内外网区别，可以和内网 lb 复用。external address 配置到这个负载均衡设备上。 |
| 更多的访问地址 | 可选 | 多个 | 如果想通过external address 之外的更多的 ip 或域名访问 global 平台，请准备好域名和 ip，部署平台的时候在安装页面的高级设置中添加。 |

#### 网络配置要求表

|  |  |
| --- | --- |
| 类型 | 要求说明 |
| 网络速率 | 不低于千兆，建议万兆。如果global 平台和业务服务集群在不同的数据中心内或是混合云，global 与业务服务集群之间网络速率不低于百兆，建议用千兆。如果没有搜集业务服务集群上的服务日志、审计等数据的需求，速度还可酌情降低。 |
| 网络时延 | 不大于2ms。如果global 平台和业务服务集群在不同的数据中心内或是混合云， global 与业务服务集群之间网络的延迟请保证在30ms内，最大不要超过100ms。 |
| 安全及防火墙 | Global平台的服务器之间无防火墙限制。 业务服务集群的服务器之间没有防火墙限制。 业务服务集群和平台之间建议无防火墙，如果有，请参考本章节转发规则，将端口在防火墙上放开。 calico使用 ipinip 协议，如果业务集群使用 calico 插件，不得限制 ipinip 协议。 |
| ip 地址范围 | 部署平台的服务器，不得使用 172.16-32 网段的 ip，如果已经使用，无法更改，就需要修改每一台服务器上的 docker 的配置，加上 bip 参数，躲过这个 ip 段，Kubernetes 集群使用10.96.0.0/12 作为 cluster IP 范围段，这段地址客户不能使用。10.199.0.0/16 网段是 global 集群的 cidr，如果有冲突，请在部署的时候增加 –network-cidr 参数指定其他网段 |
| 协议 | 如果客户打算使用双栈网络（alpha 功能），那么就要求支持 ipv6。 |
| 路由 | 服务器有 default 或指向 0.0.0.0 这个地址的路由。 |

#### Global 平台四层转发规则

|  |
| --- |
| **注意**：   * 只支持四层转发。 * lb要加上健康检查。 * 下面表格只是 global 组件的转发要求，承载 global 组件的 k8s 集群也有转发要求，请参考业务服务集群转发规则。 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 目的IP | 目的端口 | 资源池 | 协议 | 源 IP | 源端口 | 备注 |
| 第一台 master 的 ip | 8080 | 第一台 master | tcp | 操作人员的电脑 | any | 实施通过这个地址访问安装器，用于部署平台 |
| external IP | 80 | 承载平台的 Kubernetes 集群的 master 节点 | tcp | 平台和操作人员的电脑 | any | 平台 http 服务，也就是集群的 ingress |
| external IP | 443 | 承载平台平台的 Kubernetes 集群的 master 节点 | tcp | 平台和操作人员的电脑 | any | 平台出口，也就是集群的 ingress |
| global VIP | 2379 | 承载平台的 Kubernetes 集群的 master 节点 | tcp | 容灾 global 平台 | any | 容灾环境的 global 平台之间同步 etcd 数据 |
| global VIP | 80 | 承载平台的 Kubernetes 集群的 master 节点 | tcp | 平台、Kubernetes 集群和调用 api 的设备 | any | 平台 http 服务，也就是集群的 ingress |
| global VIP | 443 | 承载平台的 Kubernetes 集群的 master 节点 | tcp | 平台、Kubernetes 集群和调用 api 的设备 | any | 平台出口，也就是集群的 ingress |
| external IP | 60080 | 承载平台的 Kubernetes 集群的 master 节点 | tcp | 平台、Kubernetes 集群 | any | 平台的镜像仓库和 chart 仓库，lb 要加上会话保持 |
| external IP | 30000 | 承载平台的 Kubernetes 集群上，添加了 global=true 标签的节点 | tcp | 平台和操作人员的电脑 | any | AMP（API 管理平台） 的 kong，如果不部署 AMP，就不需要 |
| external IP | 30443 | 承载平台的 Kubernetes 集群上，添加了 global=true 标签的节点 | tcp | 平台和操作人员的电脑 | any | AMP 的 kong，如果不部署 AMP，就不需要 |
| external IP | 32305 | 承载平台的 Kubernetes 集群上，添加了 global=true 标签的节点 | tcp | 平台和操作人员的电脑 | any | AMP 访问地址 用来标识 AMP 入口开关，如果不部署 AMP，就不需要 |
| external IP | 31311 | 承载平台的 Kubernetes 集群的 master 节点 | tcp | 平台和操作人员的电脑 | any | AMP 使用的 minio 的端口，如果不部署 AMP ，则不需要。 |
| external IP | 32080 | 承载平台的 Kubernetes 集群的 master 节点 | tcp | 平台和操作人员的电脑 | any | 如果不部署 AMP ，则不需要。 |

#### 业务服务集群转发规则

|  |
| --- |
| **注意**：   * 只支持四层转发 * lb 要加上健康检查 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 目的IP | 目的端口 | 资源池 | 协议 | 源 IP | 源端口 | 备注 |
| alb VIP（部署在业务集群上的平台集群组件需要暴露部分端口，可以单独使用一个 alb，也可以和承载客户自己的业务服务的 alb 复用） | 11780 | 客户业务服务集群的 alb 的所有节点 | tcp | 平台 | any | prometheus 的端口 |
| 15012 | 客户业务服务集群的 alb 的所有节点 | tcp | 平台和业务集群 | any | istiod，多集群asm，从集群的 sidecar 要连接主集群的 istiod |
| Kubernetes api的vip | 6443 | Kubernetes 集群的 master 节点 | tcp | 平台、Kubernetes集群 | any | Kubernetes 集群 api |
| Kubernetes api 的vip | 30665 | 客户业务服务 Kubernetes 集群的 master 节点 | tcp | 访问istio ingressgateway的客户端 | any | istio 的网关，只有安装了微服务治理平台的集群才需要。 |
| Kubernetes api 的vip | 30666 | 客户业务服务 Kubernetes 集群的 master 节点 | tcp | 访问istio ingressgateway的客户端 | any | istio 的网关，只有安装了微服务治理平台的集群才需要。 |
| 业务集群所有节点的 ip | 22或其他 ssh | 客户业务服务 Kubernetes 集群的所有节点 | tcp | Global 集群的所有节点 | any | 集群管理组件会持续的 ssh 到各个 node 上同步信息 |
| 业务集群第一台 master 节点 | 6443 | 业务服务集群第一台 master 节点 | tcp | Global 集群所有节点 | any | 在部署业务集群的时候，global 会尝试链接 kube-api 判断第一台 master 是否成功部署， global 不会链接 vip ，会尝试直连 master 节点，所以需要在部署集群期间放开，部署集群完毕之后就不需要了。 |
| Kubernetes api 的vip | 11781 | 客户的高可用的业务服务集群的 master 节点 | tcp | 平台、操作人员的电脑和业务集群 | any | 日志下沉后，部署在业务集群上的 es 的访问端口 |

### 软件需求

|  |  |  |  |
| --- | --- | --- | --- |
| 需求项 | 具体要求 | 说明 |  |
| 操作系统 | X86：   * 推荐Tencent Linux release 2.4 (Final) * RedHat 7.8 * centos7.6、7.7、7.8 * EulerOS release 2.0 (SP9)  ARM： * EulerOS v2.0 SP8 * Kylin Linux Advanced Server release V10 (Sword) * UnionTech OS Server 20 Enterprise | 最小安装，只需要最基础的软件包。 UOS 需要手动修改配置文件，开始部署后，修改/cpaas/conf/check\_list.json  文件，找到"type": "os" 这一行，在其上增加 "enable": false, 如下图：  _scroll_external/attachments/worddavf8cf57e9a3a9a1cc153166fef1208dc6-4c1602c012956c619a7a4c61b1b0d0f556dcf88ccfeb9975038bffb3820594ed.png |  |
| kernel 版本 | * Centos: >=3.10.0-1127.el7 <=3.11 * Redhat: 3.10.0-1127 * Tlinux: 4.14.105-19.0016 * 其他操作系统:默认安装的版本即可 * 注1:如果内核版本不满足右侧表格内的要求，虽然可以部署，但是会导致无法部署成功或部署后平台不稳定 | 1、ovn 网络要求https://github.com/Tencent/TencentOS-kernel/pull/31[_scroll_external/remote/linkext7-7250d18c7ae84cfe9df28c93bc0d5cbee1bc354bda0c183632a6991ea315e4cb.gif](https://github.com/Tencent/TencentOS-kernel/pull/31)  2、xfs 碎片 3、kmem 问题链接：https://access.redhat.com/solutions/532663[_scroll_external/remote/linkext7-7250d18c7ae84cfe9df28c93bc0d5cbee1bc354bda0c183632a6991ea315e4cb.gif](https://access.redhat.com/solutions/532663)  https://github.com/opencontainers/runc/issues/1725[_scroll_external/remote/linkext7-7250d18c7ae84cfe9df28c93bc0d5cbee1bc354bda0c183632a6991ea315e4cb.gif](https://github.com/opencontainers/runc/issues/1725)  https://github.com/kubernetes/kubernetes/issues/61937[_scroll_external/remote/linkext7-7250d18c7ae84cfe9df28c93bc0d5cbee1bc354bda0c183632a6991ea315e4cb.gif](https://github.com/kubernetes/kubernetes/issues/61937)  https://github.com/kubernetes/kubernetes/issues/61937#issuecomment-567042968[_scroll_external/remote/linkext7-7250d18c7ae84cfe9df28c93bc0d5cbee1bc354bda0c183632a6991ea315e4cb.gif](https://github.com/kubernetes/kubernetes/issues/61937#issuecomment-567042968)  https://github.com/alauda/kube-ovn/wiki/%E5%87%86%E5%A4%87%E5%B7%A5%E4%BD%9C[_scroll_external/remote/linkext7-7250d18c7ae84cfe9df28c93bc0d5cbee1bc354bda0c183632a6991ea315e4cb.gif](https://github.com/alauda/kube-ovn/wiki/%E5%87%86%E5%A4%87%E5%B7%A5%E4%BD%9C) |  |
| grub 启动参数1，解决 kmem | 编辑/etc/default/grub （centos\redhat\tlinux） 或/boot/efi/EFI/kylin/grub.cfg（麒麟）文件，在GRUB\_CMDLINE\_LINUX= 这一行，在 crashkernel 后增加 cgroup.memory=nokmem 参数并执行grub2-mkconfig -o /boot/grub2/grub.cfg命令并重启后，能在 /proc/cmdline 中找到增加的，即代表更改成功 | https://github.com/opencontainers/runc/issues/1725https://github.com/kubernetes/kubernetes/issues/61937[_scroll_external/remote/linkext7-7250d18c7ae84cfe9df28c93bc0d5cbee1bc354bda0c183632a6991ea315e4cb.gif](https://github.com/kubernetes/kubernetes/issues/61937)  https://github.com/kubernetes/kubernetes/issues/61937#issuecomment-567042968[_scroll_external/remote/linkext7-7250d18c7ae84cfe9df28c93bc0d5cbee1bc354bda0c183632a6991ea315e4cb.gif](https://github.com/kubernetes/kubernetes/issues/61937#issuecomment-567042968) |  |
| grub 启动参数2，关闭大页 | 编辑/etc/default/grub （centos\redhat\tlinux） 或/boot/efi/EFI/kylin/grub.cfg（麒麟）文件，在GRUB\_CMDLINE\_LINUX加入选项 transparent\_hugepage=never，并执行grub2-mkconfig -o /boot/grub2/grub.cfg 然后重启服务器。按说明这一列中的图片所述方式检查 | 不关闭会严重影响性能  _scroll_external/attachments/worddav2eb7a1d60ddf5fbf7d02da747573fd53-96cfe7df033c24d27b2c8b172005449edb8e1cd827193645e0cd2916a1e94d47.png |  |
| 内核模块 | 需要加载 iptable\_nat 模块 | 以 centos 7 为例，以 root 权限执行: cat <<EOF > /etc/modules-load.d/cpaas.conf iptable\_nat EOF 然后重启服务器，执行lsmod | grep iptable\_nat 命令发现有 iptables\_nat 模块即待办配置成功 |
| 用户权限 | root | 可以接受通过非 root 用户 ssh 登录，再 su - 成 root 用户 |  |
| sshd 配置 | * 各个服务器必须允许 global 集群的各个节点通过 ssh 远程登录 * UseDNS 设置为 no | 如果不是 root 用户，需要配置 /etc/sudoers 文件，做到这个用户执行 sudo 命令，不需要输入密码 如果 dns 没有设置反向解析，有概率因此造成超时失败 |  |
| swap | 关闭 | 如果不满足，系统会有一定几率出现 io 飙升，造成 docker 卡死 |  |
| 防火墙 | 关闭 | Kubernetes 官方要求 |  |
| selinux | 关闭 | Kubernetes 官方要求 |  |
| 时间同步 | 所有服务器要求时间必须同步，误差不得超过 2 秒 | docker 和 Kubernetes 官方要求 |  |
| 时区 | 所有服务器时区必须统一 | 设置为 Asia/Shanghai |  |
| /etc/sysctl.conf 内核参数 | * vm.max\_map\_count=262144 * net.ipv4.ip\_forward=1 * vm.drop\_caches=3 * net.ipv4.tcp\_tw\_recycle=0 * net.ipv4.tcp\_mtu\_probing=1 * ipv4.conf.all.rp\_filter=0 * ipv4.conf.eth0.rp\_filter=0 * net.ipv4.conf.default.rp\_filter=0 | vm.max\_map\_count是 es 运行的服务器的要求 net.ipv4.ip\_forward是Kubernetes 要求 关闭 file cachehttps://cloud.tencent.com/developer/article/1637682[_scroll_external/remote/linkext7-7250d18c7ae84cfe9df28c93bc0d5cbee1bc354bda0c183632a6991ea315e4cb.gif](https://cloud.tencent.com/developer/article/1637682)https://www.cnblogs.com/wx170119/p/11995533.html[_scroll_external/remote/linkext7-7250d18c7ae84cfe9df28c93bc0d5cbee1bc354bda0c183632a6991ea315e4cb.gif](https://www.cnblogs.com/wx170119/p/11995533.html)https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=4396e46187ca5070219b81773c4e65088dac50cc[_scroll_external/remote/linkext7-7250d18c7ae84cfe9df28c93bc0d5cbee1bc354bda0c183632a6991ea315e4cb.gif](https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=4396e46187ca5070219b81773c4e65088dac50cc)https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt[_scroll_external/remote/linkext7-7250d18c7ae84cfe9df28c93bc0d5cbee1bc354bda0c183632a6991ea315e4cb.gif](https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt)  rp\_filter 相关配置，是两个不同模式的calico子网内的组件互相通信需要 |  |
| hostname 格式 | * 不能超过253个字符 * 只能包含小写字母、数字，以及"- "和"." * 须以字母数字开头和结尾 | https://kubernetes.io/docs/concepts/overview/working-with-objects/names/ |  |
| /etc/hosts | 所有服务器可以通过 hostname 解析成 ip，可以将 localhost 解析成 127.0.0.1 注意：hosts 文件内，不能有重复的 hostname |  |  |
| 数据库 | 只在 postgresql 9.6.13 上进行了充分测试，建议提供这个版本，并且创建 ampfiledb 、 kongswagger 和 kong 这三个库 | amp 需求，如果不部署 amp， 可不提供 |  |
| core 文件 | 关闭core文件的生成，执行ulimit -c 0 关闭，并且在/etc/profile 文件内增加'ulimit -S -c 0' 这一行 | 某些情况下，pod 内的进程重启，会在 pod 内创建 core 文件，大量占用磁盘空间，最终 pod 挂掉，甚至拖累宿主机 |  |
| /etc/resolv.conf 的要求 | 如果有search 域 ，可能会造成解析 svc 错误，需要删掉这个文件中 search 字段 |  |  |
| 使用 nfs 存储类 | 管理员在ACP管理视图上创建NFS存储类时，依赖每个节点上安装nfs客户端才能正常使用 | 在需要使用 nfs 存储类的集群的每个节点上执行yum -y install nfs-utils rpcbind |  |
| 使用 topolvm 和 rook | 用户部署 topolvm 和 rook 的时，依赖每个节点上安装 lvm2 软件包才能正常安装 | 在每个节点上执行yum install -y lvm2 |  |
| DefaultTasks | 执行systemctl show --property=DefaultTasksMax 命令，如果返回的值不是infinity 或18446744073709551615 这样的很大的数字，就需要更改 | 修改/etc/systemd/system.conf 文件，将DefaultTasksMax 改成DefaultTasksMax=infinity  影响范围：在单点或标准部署架构下，global 平台也当业务集群使用，这个配置会影响客户业务服务的数量，会在客户起了较多业务服务的时候，部分 pod 异常。 |  |

#### 快速配置命令

用 root 用户，将下面的命令复制、粘贴到每个节点上执行即可快速配置操作系统，满足部署要求。（下面的命令只适合环境没修改相关参数的，具备 epel 源的部分centos 环境，因环境千差万别，所以不推荐使用快速配置命令来配置软件环境）。

|  |
| --- |
| #关闭swap  swapoff -a  sed -i -r '/swap/s/^/#/' /etc/fstab  free -m  #关闭防火墙  systemctl stop firewalld.service  systemctl disable firewalld.service  #关闭selinux  setenforce 0  sed -ri 's/SELINUX=enforcing/SELINUX=disabled/' /etc/selinux/config  cat /etc/selinux/config | grep -w "SELINUX"  #时间同步  systemctl enable chronyd.service  systemctl start chronyd.service  sed -i -e '/^server/s/^/#/' -e '1a server ntp.aliyun.com iburst' /etc/chrony.conf  systemctl restart chronyd.service  #设置时区  timedatectl set-timezone Asia/Shanghai  #修改内核参数  echo 'vm.max\_map\_count = 262144' >> /etc/sysctl.conf  echo 'net.ipv4.ip\_forward = 1' >> /etc/sysctl.conf  echo 'vm.drop\_caches = 3' >> /etc/sysctl.conf  echo 'net.ipv4.tcp\_tw\_recycle = 0' >> /etc/sysctl.conf  echo ' net.ipv4.tcp\_mtu\_probing = 1' >> /etc/sysctl.conf  sysctl -p  ulimit -c 0 && echo 'ulimit -S -c 0' >>/etc/profile  modprobe iptable\_nat && echo iptable\_nat >> /etc/modules-load.d/cpaas.conf  sed -i -e 's/^UseDNS/#UseDNS/g' -e '$a UseDNS no' /etc/ssh/sshd\_config  systemctl restart sshd  yum install -y lvm2 |

|  |
| --- |
| **注意：**   * 安装完依赖需要重启服务器。 * 快速配置命令不包括配置/etc/resolv.conf 、配置core 文件修改/etc/hosts、修改 sshd 配置、修改 grub 启动参数、升级内核版本。 * 如果是内网环境，请手动配置 ntp 服务。 |

### 安装介质

#### 下载介质

|  |
| --- |
| * 请使用租户账号访问平台下载安装包和相关文档，或联系服务经理 * 访问地址： https://cloud.alauda.cn[_scroll_external/remote/linkext7-7250d18c7ae84cfe9df28c93bc0d5cbee1bc354bda0c183632a6991ea315e4cb.gif](https://cloud.alauda.cn) |

#### 验证介质

|  |
| --- |
| 说明：可选步骤，可以依据本章内容校验安装包是否安全可信，未被篡改。 |

##### 背景介绍

要了解什么是GPG，就要先了解[PGP](http://en.wikipedia.org/wiki/Pretty_Good_Privacy)。1991年，程序员[Phil Zimmermann](http://en.wikipedia.org/wiki/Phil_Zimmermann)为了避开政府监视，开发了加密软件PGP。这个软件非常好用，迅速流传开来，成了许多程序员的必备工具。但是，它是商业软件，不能自由使用。所以，自由软件基金会决定，开发一个PGP的替代品，取名为GnuPG。这就是GPG的由来。作为PGP的替代，如今已经有一个开放源代码的类似产品可供使用。GPG（Gnu Privacy Guard），它不包含专利算法，能够无限制的用于商业应用。官方 HOWTO <https://www.gnupg.org/howtos/zh/index.html>。

##### 安装 GPG

* GPG有两种安装方式。可以[下载源码](http://www.gnupg.org/download/index.en.html)，自己编译安装。

|  |
| --- |
| ./configure  make  make install |

* 也可以安装编译好的二进制包。

|  |
| --- |
| Ubuntu：  sudo apt-get install gnupg  Centos：  yum install gnupg -y  Mac：  brew install gpg |

##### 导入公钥

|  |
| --- |
| **注意**：   * 公钥是验证安装介质没有被篡改的可信方法，请从[www.alauda.cn](http://www.alauda.cn) 的公网下载 * 公钥 MD5 是2eaddfab97d2951a8915f327acb53562 ，请下载后验证，确保不被篡改 * 导入公钥后，执行gpg --list-keys 查看公钥 ID 是不是BB097AE6 ，确保不被篡改 |

|  |
| --- |
| curl https://www.alauda.cn/download/verify-key.pub | gpg --import  # 执行上面的命令，会有如下输出  gpg: key BB097AE6: public key "cpaas (Special for packing) <wht@126.com>" imported  gpg: Total number processed: 1  gpg: imported: 1 (RSA: 1) |

##### 查看公钥

|  |
| --- |
| gpg --list-keys  # 执行上面的命令，会输出公钥信息  /root/.gnupg/pubring.gpg  ------------------------  pub 4096R/BB097AE6 2020-08-11  uid cpaas (Special for packing) <wht@126.com>  sub 4096R/3750351A 2020-08-11 |

##### 核对公钥签名

|  |
| --- |
| gpg --fingerprint BB097AE6  # 执行上面的命令，会输出公钥信息  pub 4096R/BB097AE6 2020-08-11  Key fingerprint = 09EE E7B9 A30C F4B3 5E31 A91B 2704 1C16 BB09 7AE6  uid cpaas (Special for packing) <wht@126.com>  sub 4096R/3750351A 2020-08-11 |

##### 签名验证

下载签名文件，参考[安装介质](#scroll-bookmark-42)，获取签名文件。

|  |
| --- |
| gpg --verify <签名文件> <安装包>  # 如下为校验正常的输出结果  gpg: Signature made Thu 03 Sep 2020 03:51:35 PM CST using RSA key ID BB097AE6  gpg: Good signature from "cpaas (Special for packing) <wht@126.com>"  gpg --verify finger/cpaas-devops-2.14.0-20200901.tgz.sig 42.50s user 3.08s system 68% cpu 1:06.76 total  # 如果出现如下 warning 提示，请核对公钥的，如果公钥确实与下载链接中给出的一致，即可忽略。  gpg: WARNING: This key is not certified with a trusted signature! |

### 开始部署

#### 部署过程

1. 将主安装包解压缩到global 平台的第一台master节点的/root/cpaas-install（也可以解压缩到其他目录，要求存储解压缩文件的目录最少有100G 的空间，部署完毕后可以删除该目录）。
2. 在第一台master上执行如下命令，解压并进入安装目录：

|  |
| --- |
| tar -xvf <安装包文件地址，例如：installer-v3.6.0.tar> -C /root/cpaas-install #解压安装包  cd /root/cpaas-install /installer |

1. 选择global集群网络，执行相应安装命令。
2. 使用Kube-OVN Overlay网络部署global集群。

|  |
| --- |
| bash setup.sh |

1. 使用Calico网络部署global集群。

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

1. 根据命令行中的回显，使用浏览器输入访问地址，访问平台部署页面。

##### 部署单节点平台

|  |  |  |  |
| --- | --- | --- | --- |
| 配置 | 参数 | 说明 | 备注 |
| 账号设置 | 用户名 | 系统超级管理员账号。 |  |
| 密码 | 系统超级管理员密码。 |  |
| 确认密码 | 确认输入密码。 |  |
| 集群设置 | Kubernetes版本 | 选择集群的Kubernetes组件的版本及该版本支持的容器运行时组件类型和版本。 | IPv4/v6双栈、服务网格依赖 Kubernetes 新版本。 |
| 网卡名称 | 集群网络插件所使用的主机网卡，若不填写，系统将自动获取节点默认路由所对应的网卡。 |  |
| SSH 地址 | 部署global的节点地址。 |  |
| SSH 端口 | 节点SSH服务端口。 |  |
| 节点名称 | 可选节点IP作为节点名称或主机名称作为节点名称。 | 选择主机名称作为节点名称时，您需确保主机名称在当前集群中唯一。 |
| SSH 认证方式 | 根据选择的密码或秘钥方式填写认证信息。 |  |
| 容器网络 | Cluster CIDR | 表示默认子网网段。 | 平台global默认使用 Kube-OVN Overlay网络，您需确保容器网络和宿主机网络属于不同网段，否则系统部署可能会出现异常。  **注意：若您global集群使用Calico网络，则**Join CIDR**参数填写无效。** |
| Service CIDR | 供类型为ClusterIP的Kubernetes Service 使用的IP地址段，不可与默认子网的网段重叠。 |
| Join CIDR | Kube-OVN  Overlay传输方式下供节点与容器组间通信使用的IP地址段。不可与默认子网、Service 网段重叠。 |
|  | 高级设置 | IPv4/IPv6双栈可以有效弥补IPv4网络地址资源有限的问题。  打开开关后，集群下节点可在双栈模式下运行，即节点可以同时拥有 IPv4 地址和 IPv6 地址，使用这两个IP地址都可以进行内网/公网访问。  注意：启用了双栈的集群，不支持还原为 IPv4 单栈。 | 当您的业务应用涉及到以下场景时，建议您启用双栈：   * 您的应用需要为使用 IPv6 终端的用户提供访问服务。 * 您需要对使用 IPv6 终端访问应用提供的服务的访问来源进行数据分析处理。 * 如果您的应用系统与其他系统（例如：数据库系统）、应用系统之间需要使用 IPv6 进行内网访问。 |
| 控制台设置 | 平台访问地址 | 平台global集群的访问地址。   * 若填写IP地址，在单节点部署下一般为单节点IP地址； * 若填写域名，请确保已将域名解析至单节点IP地址。 | 默认使用HTTPS协议部署平台。  提示：若确认使用不安全的HTTP协议部署平台，可在高级设置中打开开关。 |
| 证书 | 支持自签证书和导入已有证书。 | 若平台访问地址和平台其他访问地址同时为HTTPS访问，则该证书将同时供两个地址使用。 |
| 平台其他访问地址 | 可输入多个IP或域名。   * 若填写IP地址，您须确保该 IP 地址可以转发至单节点IP地址； * 若填写域名，请确保已将域名解析至单节点IP地址。 |  |
| GPU 类型 | 若该节点需要使用GPU，请选择相应GPU类型。 | 请确保该节点中已安装GPU驱动。 |
| Pod 数上限 | 每个节Pod数最大值，上限为255。 |  |

##### 部署高可用平台

|  |  |  |  |
| --- | --- | --- | --- |
| 配置 | 参数 | 说明 | 备注 |
| 账号设置 | 用户名 | 系统超级管理员账号。 |  |
| 密码 | 系统超级管理员密码。 |  |
| 确认密码 | 确认输入密码。 |  |
| 平台设置 | Kubernetes版本 | 选择集群的Kubernetes组件的版本及该版本支持的容器运行时组件类型和版本。 | IPv4/v6双栈、服务网格依赖 Kubernetes 新版本。 |
| 集群地址 | global所在高可用Kubernetes集群暴露的apiserver地址。 | 由于高可用的集群控制节点个数为3，您可以在以下两种方式中任选其一，为集群地址提供高可用：   * 在下一页的**可选配置**中打开**自建 VIP开关，按要求配置相关参数。** * IP地址/域名也可以为提前准备的集群外的负载均衡（例如：F5 设备、IaaS 层的负载均衡器或 HAProxy 软件等）的访问地址或域名。 |
| 平台访问地址 | 平台global集群的访问地址，默认与集群地址相同。   * 若填写IP地址，您须确保该 IP 地址可以转发到集群地址； * 若填写域名，请确保已将域名解析至集群地址。 | 默认使用HTTPS协议部署平台。  提示：若确认使用不安全的HTTP协议部署平台，可在高级设置中打开开关。 |
| 镜像仓库 | 镜像仓库 | 拉取平台组件镜像的仓库。   * 平台默认：部署global时默认给您配置的内置镜像仓库。 * 外部：提前搭建的外部镜像仓库。需输入访问镜像仓库的私有镜像仓库地址、端口、用户名、密码。 | * 选择平台默认：则填写平台访问的IP或域名即可，如果是容灾环境，则必须是域名。 * 选择外部：在容灾环境必须保证仓库地址在所有容灾节点都可以访问。 |
| 容器网络 | Cluster CIDR | 表示默认子网网段。 | 平台global默认使用Kube-OVN Overlay网络，您需确保容器网络和宿主机网络属于不同网段，否则系统部署可能会出现异常。  **注意：若您global集群使用Calico网络，则该参数填写无效。** |
| Service CIDR | 供类型为ClusterIP的Kubernetes Service使用的IP地址段，不可与默认子网的网段重叠。 |
| Join CIDR | Kube-OVN Overlay传输方式下，供节点与容器组间通信使用的IP地址段。不可与默认子网、Service网段重叠。 |
| 高级设置 | IPv4/IPv6双栈可以有效弥补IPv4网络地址资源有限的问题。  打开开关后，集群下节点可在双栈模式下运行，即节点可以同时拥有 IPv4 地址和 IPv6 地址，使用这两个IP地址都可以进行内网/公网访问。  注意：启用了双栈的集群，不支持还原为 IPv4 单栈。 | 当您的业务应用涉及到以下场景时，建议您启用双栈：   * 您的应用需要为使用 IPv6 终端的用户提供访问服务。 * 您需要对使用 IPv6 终端访问应用提供的服务的访问来源进行数据分析处理。 * 如果您的应用系统与其他系统（例如：数据库系统）、应用系统之间需要使用 IPv6 进行内网访问。 |
| 节点设置 | 网卡名称 | 集群网络插件所使用的主机网卡，若不填写系统将自动获取节点默认路由所对应的网卡。 | 选择Containerd为容器运行时，您须指定网卡名称。 |
| 节点名称 | 可选节点IP作为节点名称或主机名称作为节点名称。 | 选择主机名称作为节点名称时，您需确保主机名称在当前集群中唯一。 |
| 添加节点 | * 网卡名称：请填写集群网络插件所使用的主机网卡，填写后平台对于该节点将优先使用此项配置，若不填写，系统将按如下顺序自动配置：集群配置的网卡名称，节点默认路由所对应的网卡。 * 设置作为计算节点：勾选后，即当前节点允许Pod调度。 | 请先参考容量规划章节进行部署架构选择。 |
| 日志监控设置 | 监控组件类型 | 推荐您选择Prometheus作为平台监控组件，选择节点时建议您选择非控制节点搭建监控服务。  选择VictoriaMetrics监控组件时，您须配置部署VictoriaMetrics代理实例数，即VMAgent的数量。推荐添加一个，最多支持添加三个。 | 为确保监控节点的配置，您需要符合本文档硬件需求章节内的要求。可靠性要求高的场景可以部署高可用的监控组件。 后续需要调整时，可在部署成功后的平台管理中关闭监控组件，选择重新部署，再选择节点部署高可用监控组件。 |
| 日志存储节点 | 在该节点上部署Elasticsearch等组件，建议选择非控制节点进行日志服务的搭建；默认配置的日志服务支持规模有限，如果日志规模较大，请联系相关人员。 | 日志节点仅支持选择 3 个，如果日志组件有更多节点的需求，请参考[修改ES节点数](#scroll-bookmark-56)小节。 |
| 控制台设置 | 平台其他访问地址 | 可输入多个IP或域名。 | 默认与集群地址相同。   * 若填写IP地址，您须确保该 IP 地址可以转发到集群地址； * 若填写域名，请确保已将域名解析至集群地址。 |
| 可选配置 | GPU 类型 | 若该节点需要使用GPU，请选择相应GPU类型。 | 请确保该节点中已安装GPU驱动。 |
| Pod 数上限 | 每个节点Pod数最大值，上限为255。 |  |
| 自建 VIP | 需输入已申请的 VRID（虚拟路由器 ID）并设置 IP 地址/域名 为已申请的 VIP（端口 必须为 6443）。  提示：集群使用自建 VIP 并创建成功后，集群的访问地址为 <VIP>:6443。  如需使用**自建 VIP**作为集群地址，使集群地址高可用，请提前联系平台的网络管理员或运维人员申请 VIP（Virtual IP）及 VRID（虚拟路由器 ID）。 | VIP 需为集群节点 IP 所在子网中的空闲 IP（IPv4 地址）且不可与节点 IP 相同。 |
| 产品 | 选择需要部署的产品名称。 |  |

上述参数输入完毕后，点击右下角的 **开始部署** 按钮即开始部署进程。

#### 验证平台

检查 global 平台是否部署成功，在运行 global 组件的 Kubernetes 集群的 master 节点上执行以下命令：

|  |
| --- |
| # 检查 sentry 部署的 chart 是否成功：执行如下命令查找部署失败的 chart  kubectl get apprelease --all-namespaces  #检查 所有 pod 是否正常：执行如下命令查找失败的 pod  kubectl get pod --all-namespaces | awk '{if ($4 != "Running" && $4 != "Completed")print}' | awk -F'[/ ]+' '{if ($3 != $4)print}' |

#### 访问平台

|  |
| --- |
| **注意**：因浏览器的兼容性不同，使用不同的浏览器访问平台时，可能存在平台界面显示错误或功能无法正常使用的情况。兼容浏览器的版本说明如下：   * Google Chrome浏览器支持**93 及以上版本；** * Firefox浏览器支持**92 及以上版本。** |

1. 访问地址，通过浏览器访问安装器，部署安装完毕后，在浏览器上会有平台访问按钮，点击即可跳转过去。
2. 产品路径：部署成功之后，访问平台，登录之后点击左上角的产品下拉菜单（九宫格图标）选择访问某个产品也可以在登录之后，平台地址后加路径直接访问各个产品，例如：

* acp： <平台访问地址>/console-acp
* devops： <平台访问地址>/console-devops
* asm： <平台访问地址>/console-asm
* 平台成熟度设置（alpha 功能开关设置）： <平台访问地址>/console-platform/home/feature-gate

#### 删除安装器

在解压缩安装包，执行 setup.sh 的那台机器上执行docker rm -f cpaas-installer，删掉安装器。

|  |
| --- |
| **注意**：也可选择不手动删除，等2个小时后系统自动删除。 |

## 配置平台

### 简介

|  |
| --- |
| **说明**：部署完毕之后，平台某些功能或配置可能会不满足项目要求，就需要依据本章内容，对平台进行配置。 |

### 修改组件软件配置

|  |
| --- |
| **说明**：在 [容量规划](#scroll-bookmark-13) 这一章节，为满足知道规模/压力下的业务，不仅需要保证硬件配置符合相应章节的要求，还需要依据本章，修改组件的软件配置。 |

#### 修改日志采集范围及日志、审计、事件等数据保存时间

##### 日志采集范围

平台已经支持在 UI 管理页面上修改，请参考用户手册进行修改。

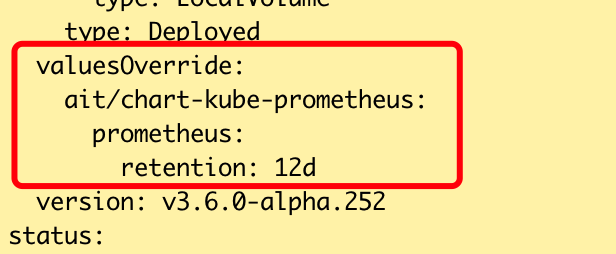
##### 日志保存时间

平台已经支持在 UI 管理页面上修改，请参考用户手册进行修改。

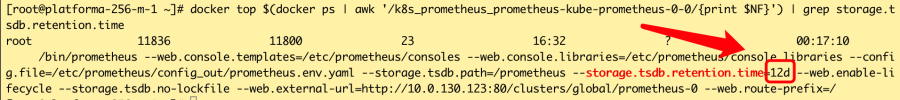
##### 监控数据保存时间

在 global 集群的第一台 master 上执行如下命令：

|  |
| --- |
| # 执行如下命令搜索对应集群的moduleinfo资源：  kubectl get moduleinfo | grep prometheus | grep <calico1> #比如找calico1 这个集群，如下图：    # 然后执行如下命令修改：  kubectl edit moduleinfo < calico1-ff2e17be871fa31b17dca13d0d59b0ab > # calico1-ff2e17be871fa31b17dca13d0d59b0ab 就是上图中，我们搜索到的 calico1 集群的 moduleinfo 资源的名  # 在 .spec 下找valuesOverride ，如果没有就增加这个key ，然后增加如下内容：    valuesOverride:  ait/chart-kube-prometheus:  prometheus:  retention: 12d # 要保存监控数据的时间，12d 代表保存12 天，如下图 |



|  |
| --- |
| # 修改完毕后保存，等5分钟后，在 Prometheus 节点执行如下命令检查：  docker top $(docker ps | awk '/k8s\_prometheus\_prometheus-kube-prometheus-0-0/{print $NF}') | grep storage.tsdb.retention.time  # 如果出现如下图内容，即代表修改成功 |



#### 修改组件 limit

|  |
| --- |
| # 在第一台 master 节点执行如下命令搜索要修改的组件：  kubectl get deploy,sts,ds -A | grep apoll # 这条命令会找到 apollo 的资源名，然后执行如下命令修改：  kubectl edit -n cpaas-system deployment.apps/apollo |

#### 修改apollo –es-enablealias 参数

|  |
| --- |
| # 在第一台 master 节点执行如下命令：  kubectl edit prdb base  # 在 .spec 下找valuesOverride ，如果没有就增加这个key ，然后增加如下内容：  valuesOverride:  ait/chart-alauda-base:  logging:  esEnableAliases: false # 这是布尔型参数，只有 true 和 false 两个值 |

#### 修改 Es分片数(ALAUDA\_ES\_SHARDING)

请参考用户手册中，平台中心 > 平台管理 > 集群管理 > 集群 > 插件管理 > 部署日志存储组件 的相关内容。

#### 修改 Es 节点数

平台已经支持在 UI 管理页面上修改，请以管理员身份登录平台，然后进入 平台管理 > 产品管理 ，单击 **操作** ，在下拉菜单中选择 "更新日志存储组件" 进行修改。

|  |
| --- |
| **说明**：目前仅支持1节点和3节点的Es。无法修改为更多节点。如果期望使用多于3个节点的Es，目前需要手动修改部分参数。   1. 在命令行中执行 moduleinfo(kubectl get moduleinfo |grep logcenter | grep <cluster-name>) 并获取 spec.config.components.elasticsearch.nodes 部分。 2. 将需要添加的节点名称加到下面即可。（获取节点名称可以使用命令 kubectl get nodes） |

#### 修改 Es index 的副本(replicas)

请参考用户手册中，平台中心 > 平台管理 > 集群管理 > 集群 > 插件管理 > 部署日志存储组件 的相关内容。

### 容器平台部署镜像仓库

容器平台需要在不依赖DevOps(页面上不依赖)集成自己的镜像仓库，并实现镜像的展示与管理等功能。  
**背景说明** **：**

* 某些产品交付时不想要 DevOps、Service Mesh 产品，希望环境干净且无多余业务组件，用于保证平台本身占用资源足够小。
* DevOps 已经提供私有化交付的 Harbor，可以通过工具链进行绑定，并给业务使用。

**实现方案：**

* 去除平台页面上的 devops 相关内容。
* 平台附带一个轻量级的镜像仓库，能够管理 ISV 产品的镜像。

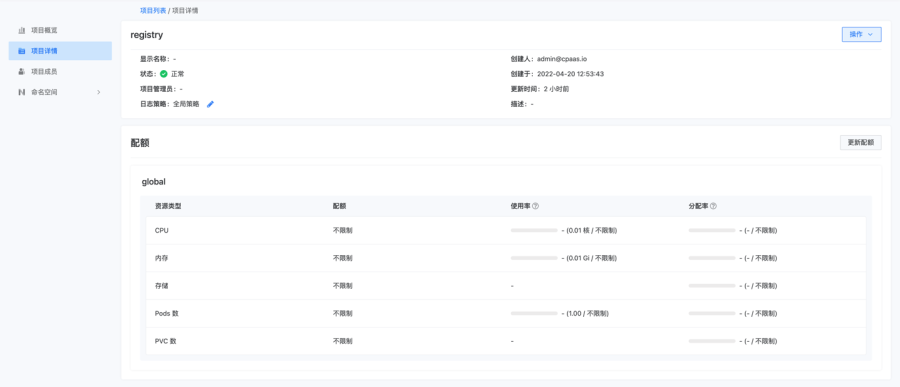
#### 部署平台的时候选择不安装 devops



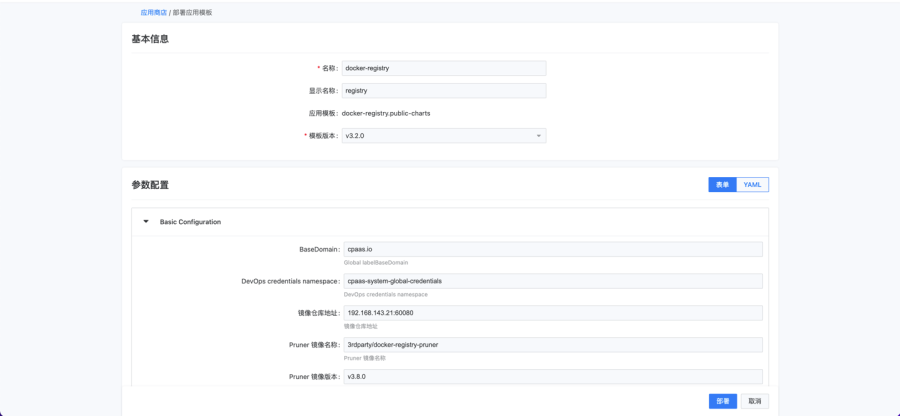
在可选配置中去掉DevOps组件，只独立安装Contaioner Platform。

#### 部署 docker-registry

1. 在 global 集群中切换到项目管理，创建项目 registry。



1. 切换到应用商店，部署应用目录 docker-registry。



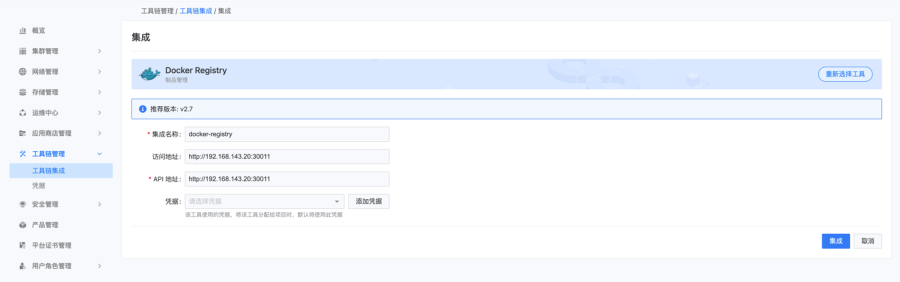
参数说明

|  |
| --- |
| type=NodePort  nodePort=30011 (自定,一般在30000-32000之间)  host=192.168.25.127 (自定，一般选一个master的地址即可)  enabled=true (开启basic认证，默认false不开启，设置为true时，需要指定 basicAuth.htpasswd.username和basicAuth.htpasswd.password，开启后docker pull/push操作需 要先docker login)  hostpath.path=/cpaas/docker-registry (默认使用节点hostpath类型作为后端存储docker镜像数据，而且默认指定Pod调度到有registry: true标签的节点上，因此需要提前给某个节点打registry: true的label)  storageClass: (如果指定了storageClass，将不再使用hostpath类型，也不再要求提前给节点打registry: true的label，storageClass值有三种场景："-"表示不使用任何storageClass，这将启用pv动态存储供应；null表示将会自动使用系统默认的storageclass；“mysc”表示将固定使用名称为mysc的storageClass)  tlsSecretName: (指定一个已存在的secret名称，该secret是tls类型的，这将意味着registry以tls方式启动，启动时自动挂载该secret中的crt和key)  其他：需要docker push的机器应该将上面的地址加入到 insure-registry列表里。 |

1. 部署完成后如下图：

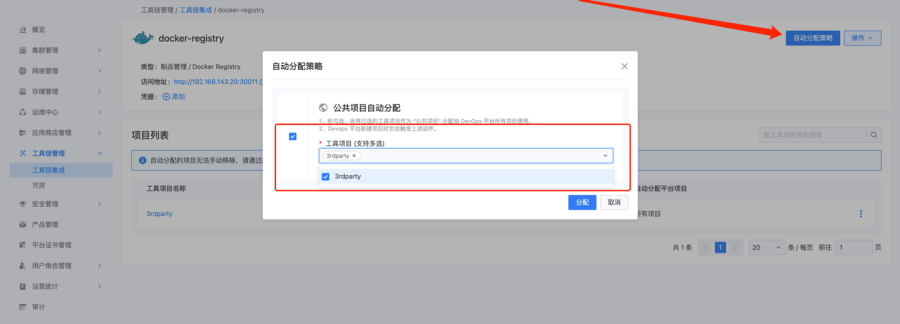


#### 工具链集成

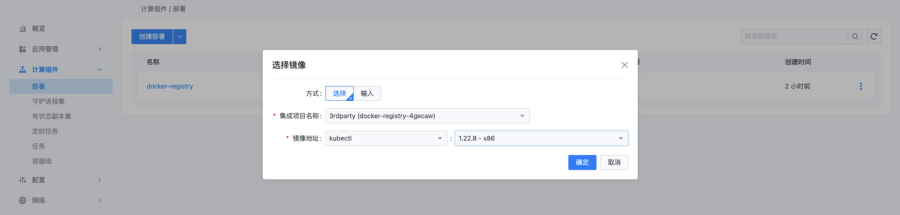
1. 在global集群中切换到 平台管理界面下找到工具链管理 > 工具链集成 > Docker Registry。  
   

**部署高可用平台**

1. 给镜像仓库分配项目。注意：如果在第二步"部署decker-registry"时，设置了开启basic认证，则在本步骤前后还需要额外操作：
2. 手动在global集群上创建 cpass-system-global-credentials 这个 namespace
3. 根据第二步"部署decker-registry"时指定的用户名和密码，通过点击上图的"添加凭据"按钮，生成凭据。并在上图页面指定使用该凭据



1. 验证，在global集群中切换到 Container Platform界面下 > 计算组件 > 部署 > 创建部署，如果能看到如下图一样的，正常选择集成项目名称和镜像地址及Tag说明操作成功。



1. 若客户使用外部镜像仓库或个人镜像仓库，可以直接从第三步工具链集成开始操作。

## 清理环境

### 说明

推荐重新创建新的服务器或将服务器重装操作系统的方式来清空，如果上述方法不能采用，可以将安装包解压缩，在安装目录下，有 cleanup.sh 这个脚本。将这个脚本复制到每一台服务器上运行，就可以清空这台服务器。建议运行这个脚本两遍之后，重启服务器，然后再运行一遍，防止某些资源因为被进程占用无法删除。

# 轻量裸金属部署

## 文档概述

### 文档介绍

本文档介绍了 轻量裸金属部署方案架构及容量规划方法，给出了部署必须遵守的硬件资源需求、网络资源需求和软件资源需求。

本文档详细介绍了 轻量裸金属 安装过程，global 平台是 轻量裸金属 的一部分，部署 global 平台所需了解名词、部署架构、软硬件和网络要求、容量规划等等信息，请参考本文档中，前半部分中的 global 平台的部署。

### 名词解释

|  |  |
| --- | --- |
| 名词 | 解释  （部署时需要重点理解的部分名词） |
| 轻量裸金属 | 轻量裸金属部署，本文档所要部署的产品，基于 k8s 提供丰富的 IAAS 管理功能 |

其他名词解释请参考本请参考本文档中，前半部分中的 global 平台的部署。

### 部署架构

客户业务和 Global 部署到不同 kubernetes 集群，业务集群运行在物理机上，Global 运行于 kubevirt 创建的虚拟机里。

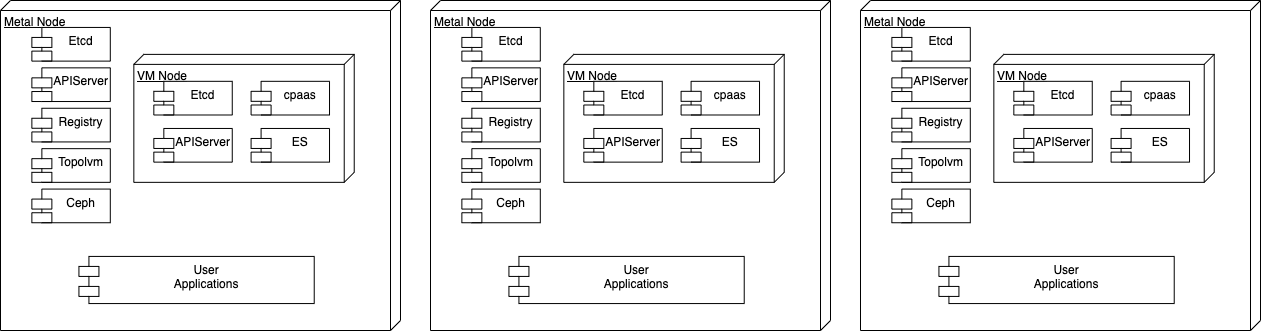


Figure 1 sddc-Page-2.png

## 部署平台

### 部署流程

1. 了解部署架构
2. 容量规划
3. 资源 review
4. 获取 global 平台的安装介质
5. 获取 轻量裸金属 的安装介质
6. 部署实施

### 容量规划

#### 部署架构一，轻量裸金属 规模/压力2 容量规划

##### 硬件数量及配置

|  |  |
| --- | --- |
| **服务器角色** | **轻量裸金属 master** |
| **服务器数量** | 3 |
| **服务器用途** | 承载纯软数据中心，及纯软数据中心的 global 平台、和业务服务集群 |
| **是否可选** | 必须 |
| **CPU数量** | 2 |
| **内存容量** | 256G |
| **系统盘** | 1T |
| **数据盘** | 8T |
| **网卡** | 10Gb \*2 |

#### 硬件需求

|  |  |  |
| --- | --- | --- |
| 硬件 | 具体要求 | 推荐最低型号或配置 |
| CPU | 核心数不低于16C，频率不低于 2.10 GHz | 2 \* 英特尔® 至强® 银牌 4216 处理器（22M 高速缓存，2.10 GHz） |
| 内存 | 32GB RDIMM | 8 \* 3200MT/s RDIMMs |
| 硬盘 | * 系统盘：480G SSD \*2 * 数据盘：3.84T NVMe SSD \* 2 | nvme ssd |

### 网络需求

规划并配置如下资源：

* 物理机的 keepalived vip，是一个给三台物理机使用，用作高可用负载均衡的 ip 地址，建议和物理机 ip 在同一网段
* 物理机 keepalived vrid，如果同一个二层的网络内，也有其他 keepalived，必须不能重复，在1-225 间随机选一个
* 物理机 topolvm 设备，要求必须是独立的块设备
* 物理机集群、global 集群的 cluster 和 service cidr，建议每个 cidr 是一个 b 类地址
* kube-ovn underlay vland id，提供给物理集群 ovn 的vland id，需要物理环境的网络管理员分配一个
* kube-ovn underlay 网关 ip 和 cidr，需要物理环境的网络管理员分配
* L3 以太网交换机，10Gb

其他网络需求请参考请参考本文档中，前半部分中的 global 平台的部署部分。

### 软件需求

请参考文档中，前半部分中的 global 平台的部署部分。

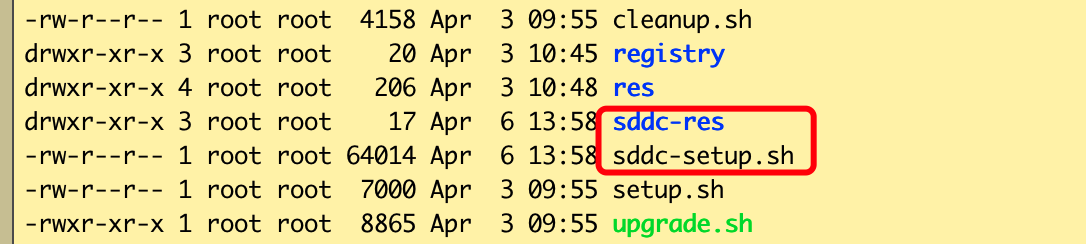
### 安装介质

#### 下载介质

请参考[安装介质](#scroll-bookmark-42)章节。

#### 合并介质

* 解压缩 3.8.2 安装包
* 解压缩 轻量裸金属 安装包
* 拷贝 轻量裸金属部署 安装包内的所有内容，到3.8.2 安装包解压缩后的第一层目录，如下图：



将轻量裸金属部署 安装包内解压出来的 sddc-res 文件夹和 sddc-setup.sh 脚本复制到和 res 目录、setup.sh 脚本同样的目录里。比如安装包解压到 /cpaas/installer 下，那么 sddc-res 复制到 /cpaas/installer/sddc-res  setup.sh 复制到 /cpaas/installer/setup.sh

### 开始部署

#### 部署平台

##### 执行 sddc-setup.sh 脚本部署

例子如下：

|  |
| --- |
| cd /cpaas/installer ## 假定安装包解压到 /cpaas/installer  bash sddc-setup.sh \  --servers 192.168.254.72,192.168.254.73,192.168.254.74 \  --keepalived-vip 192.168.254.201 \  --keepalived-vrid 73 \  --topolvm-device /dev/sdc \  --physical-cluster-cidr 192.168.171.0/24 \  --global-masters 192.168.171.3,192.168.171.4,192.168.171.5 \  --global-vip 192.168.171.2 \  --global-vrid 4 \  --acp-admin-password '07Apples@' \  --underlay-vlan-id 0 \  --underlay-network-interface em1 \  --underlay-gateway 192.168.171.1 \ |

##### 部署脚本参数说明

|  |
| --- |
| sddc-setup.sh <options>  options:  --servers 要部署的三台物理服务器, 逗号分割, 例如: 192.168.1.1,192.168.1.2,192.168.1.3  --keepalived-vip 三台物理服务器的 keepalived vip, 与 servers 在同一子网, 不能被占用.  --keepalived-vrid keepalived 的 VRID, 取值 1-255, 同一子网下不能重复, 默认值: 207  --ssh-port 物理机的 ssh 访问端口, 默认值: 22  --ssh-user 物理机的 ssh 用户名, 默认值: root  --ssh-key-file 物理机的ssk 登录key文件的位置, 要求能登录 servers, 可以使用 ssh-keygen 生成, 使用 ssh-copy-id 配置 登录授权, 默认值: /root/.ssh/id\_rsa  --topolvm-device 物理机集群的topolvm 使用的磁盘, 例如: /dev/sdc  --physical-cluster-name 物理集群的名字, 默认值: physical  --physical-service-cidr 物理集群的 kubernetes service CIDR, 默认值: 10.96.0.0/16  --physical-cluster-cidr 物理集群的 kubernetes cluster CIDR, 即 kube-ovn underlay CIDR  --global-service-cidr global 集群的 kubernetes service CIDR, 默认值: 10.97.0.0/16  --global-cluster-cidr global 集群的 kubernetes cluster CIDR, 默认值: 10.199.0.0/16  --global-masters global 集群的 master 虚拟机的 ip，从物理机的underlay网段中取 ip，部署过程脚本会创建虚机并分配这个 ip  --global-vip global 集群的 master 虚拟机的 keepalived vip ，从物理机的underlay网段中取ip，部署过程脚本会标记这个 ip 为已用，并分配给 global 使用  --global-vrid global 集群的 master 虚拟机的 keepalived VRID, 取值 1-255, 同一子网下不能重复, 默认值: 217  --acp-admin ACP 管理员用户名, 默认值: admin@cpaas.io  --acp-admin-password ACP 管理员密码, 不能为空  --underlay-vlan-id kube-ovn underlay 网络的 VLAN id  --underlay-network-interface kube-ovn underlay 网络的网卡名，物理机 underlay 网络的网卡名。要用单独的网卡，不能是server ip所在的网卡。  --underlay-gateway kube-ovn underlay 网络的网关 ip  --physical-kube-ovn-join-cidr 物理集群的 join CIDR, 默认值: 169.254.16.0/20  --global-kube-ovn-join-cidr global 集群的 join CIDR, 默认值: 169.254.32.0/20  --docker-bip docker bip网段, 默认值 169.254.0.1/24 |

#### 部署插件

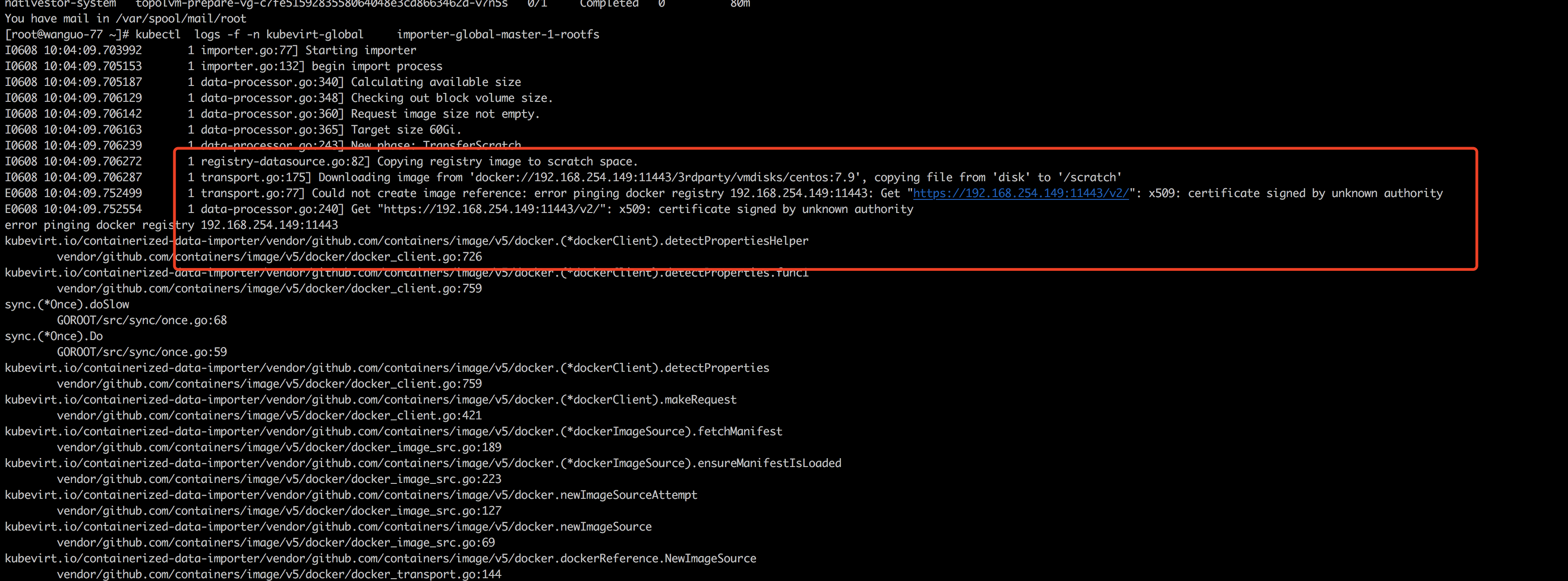
##### 日志存储组件和监控

请参考用户手册

## 常见故障

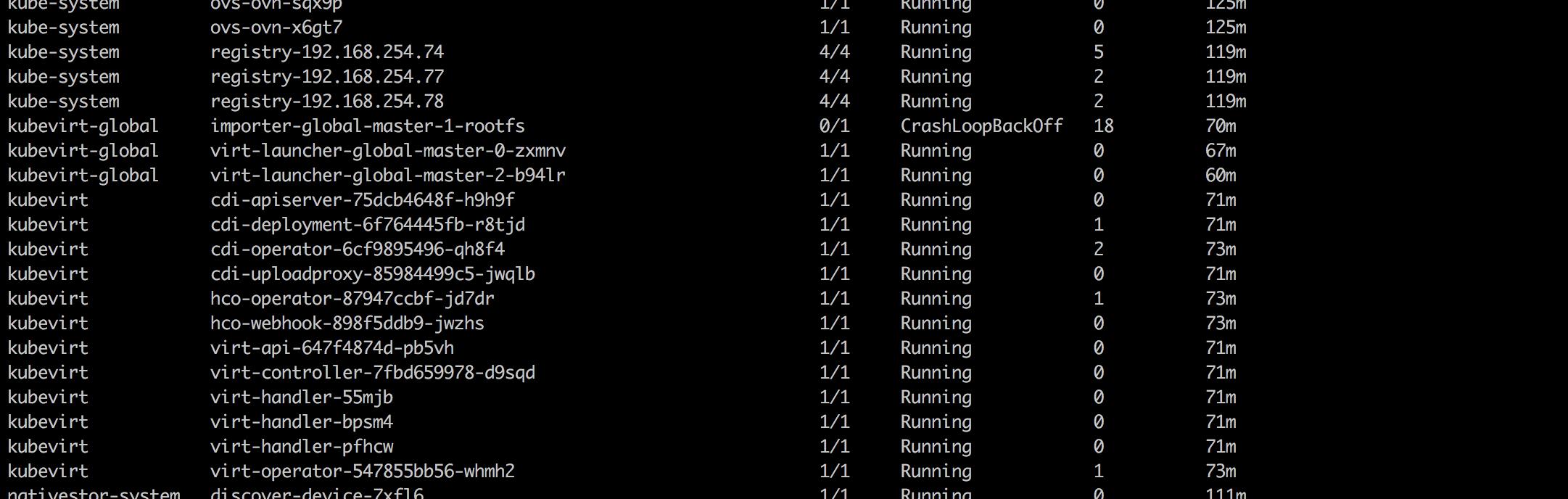
### DataVolumeError

如果报错vm global-master-0' status is Running, vm global-master-1' status is DataVolumeError, vm global-master-2' status is Running, waiting，这说明有pod 状态 crash 了，这是偶发的故障，一般是 importer 操作系统镜像出错了，如下图：



处理方法：

* 找到 crash 的 pod ，如下图



* 重启这个 pod 即可