产品概览

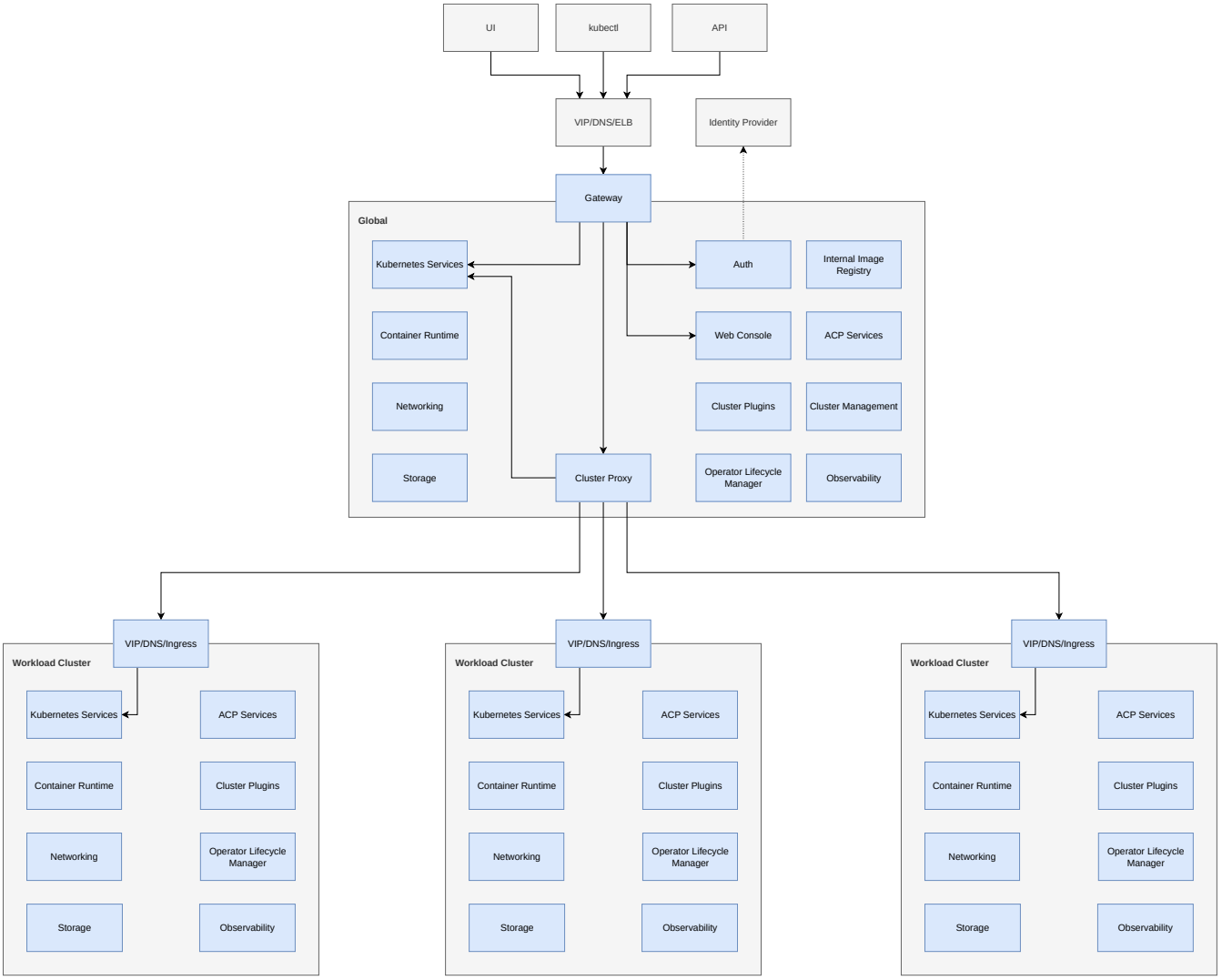
架构

目录

灵雀云容器平台 简介

灵雀云容器平台（ACP）提供了一个企业级的基于 Kubernetes 的平台，使组织能够在混合云和多云环境中一致地构建、部署和管理应用。ACP 集成了核心 Kubernetes 功能，并增强了管理、可观测性和安全服务，提供统一的控制平面和灵活的业务集群。

该架构采用**hub-and-spoke**模型，由一个 `global` 集群和多个业务集群组成。此设计实现了集中治理，同时允许独立的工作负载执行和弹性扩展。



核心架构组件

Global Cluster

`global` 集群作为 ACP 的集中管理和控制中心。它提供平台范围的服务，如认证、策略管理、集群生命周期操作和可观测性。同时，它也是多集群管理的核心枢纽，支持跨集群功能。

关键组件包括：

- **Gateway**
  作为平台的主要入口，管理来自 UI、CLI（kubectl）和自动化工具的 API 请求，并将其路由到相应的后端服务。
- 认证与授权（**Auth**）
  集成外部 Identity Providers（**IdPs**），提供单点登录（SSO）和基于 RBAC 的访问控制。

- **Web Console**
  提供基于 Web 的 ACP 界面，通过 Gateway 访问平台 API。
- 集群管理
  负责业务集群的注册、配置和生命周期管理。
- **ACP** 服务
- **Operator Lifecycle Manager (OLM)** 和集群插件
  管理 operator 和集群扩展的安装、更新及生命周期。
- 内部镜像仓库
  提供开箱即用的容器镜像仓库，支持基于角色的访问控制。
- 可观测性
  为 `global` 集群和业务集群提供集中式日志、指标和追踪。
- 集群代理
  实现 `global` 集群与业务集群之间的安全通信。

业务集群

业务集群是由 `global` 集群管理的基于 Kubernetes 的环境。每个业务集群运行隔离的应用工作负载，并继承来自中央控制平面的治理和配置。

外部集成

- **Identity Provider (IdP)**
  支持通过标准协议（OIDC、SAML）实现联合认证，统一用户管理。
- **API** 和 **CLI** 访问
  用户可通过 RESTful API、Web Console 或命令行工具（如 `kubectl` 和 `ac`）与 ACP 交互。
- 负载均衡器（**VIP/DNS/SLB**）
  为 `global` 和业务集群的 Gateway 及入口端点提供高可用性和流量分发。

可扩展性与高可用性

ACP 设计支持水平扩展和高可用性：

- 各组件可冗余部署，消除单点故障。
- `global` 集群支持管理数十至数百个业务集群。
- 业务集群可根据工作负载需求独立扩展。
- 采用 VIP/DNS/Ingress 实现无缝路由和故障切换。

功能视角

灵雀云容器平台（ACP）的完整功能由 **ACP Core** 和基于两大技术栈的扩展组成：**Operator** 和 集群插件。

- **ACP Core**

  ACP 的最小交付单元，提供核心能力，如集群管理、容器编排、项目和用户管理。

  - 满足最高安全标准
  - 提供最大稳定性
  - 支持最长生命周期

- 扩展

  Operator 和集群插件技术栈中的扩展可分为：

  - **Aligned** – 生命周期策略包含多个维护分支，与 ACP 保持对齐。
  - **Agnostic** – 生命周期策略包含多个维护分支，独立于 ACP 发布。

  有关扩展的更多详情，请参见 Extend。

技术视角

平台组件运行时
所有平台组件作为容器运行在 Kubernetes 管理集群（即 `global` 集群）中。

高可用架构

- `global` 集群通常由至少三个控制平面节点和多个工作节点组成

- etcd 的高可用性是集群 HA 的核心；详情见*关键组件高可用机制*
- 负载均衡可由外部负载均衡器或集群内部自建 VIP 提供

请求路由

- 客户端请求首先经过负载均衡器或自建 VIP
- 请求转发至运行在指定入口节点（或配置为控制平面节点）的 **ALB**（平台默认 Kubernetes Ingress Gateway）
- ALB 根据配置规则将流量路由到目标组件 Pod

副本策略

- 核心组件至少运行两个副本
- 关键组件（如 registry、MinIO、ALB）运行三个副本

容错与自愈

- 通过 kubelet、kube-controller-manager、kube-scheduler、kube-proxy、ALB 等组件协作实现
- 包括健康检查、故障切换和流量重定向

数据存储与恢复

- 控制平面配置和平台状态以 Kubernetes 资源形式存储于 etcd
- 在灾难性故障时，可通过 etcd 快照进行恢复

主备灾备

- 两个独立的 `global` 集群：主集群 和 备集群
- 灾备机制基于主集群到备集群的 etcd 数据实时同步
- 主集群故障不可用时，服务可快速切换至备集群

关键组件高可用机制

etcd

- 部署在三个（或五个）控制平面节点上

- 使用 RAFT 协议进行领导选举和数据复制
- 三节点部署可容忍最多一个节点故障；五节点部署可容忍最多两个
- 支持本地和远程 S3 快照备份

监控组件

- **Prometheus**：多实例，结合 Thanos Query 去重，支持跨地域冗余
- **VictoriaMetrics**：集群模式，包含分布式 VMStorage、VMInsert 和 VMSelect 组件

日志组件

- **Nevermore** 收集日志和审计数据
- **Kafka / Elasticsearch / Razor / Lanaya** 以分布式和多副本模式部署

网络组件（**CNI**）

- **Kube-OVN / Calico / Flannel**：通过无状态 DaemonSet 或三副本控制平面组件实现高可用

ALB

- Operator 以三副本部署，启用领导选举
- 实例级健康检查与负载均衡

自建 **VIP**

- 基于 Keepalived 的高可用虚拟 IP
- 支持心跳检测和主备切换

Harbor

- 基于 ALB 的负载均衡
- PostgreSQL 采用 Patroni 实现高可用
- Redis 采用哨兵模式
- 无状态服务多副本部署

Registry 和 **MinIO**

- Registry 以三副本部署

- MinIO 采用分布式模式，支持纠删码、数据冗余和自动恢复

Kubernetes 支持矩阵

本文档提供了 ACP 的 Kubernetes 版本支持矩阵。该信息在创建集群、升级 ACP 以及管理第三方集群时至关重要。

目录

概述

ACP 支持多个 Kubernetes 版本，覆盖不同的 ACP 版本。了解支持的版本对于以下操作非常重要：

- 创建集群 – 确定在配置新集群时可使用的 Kubernetes 版本
- 升级 **ACP** – 确保所有工作负载集群在升级 global 集群之前满足兼容性要求
- 管理第三方集群 – 验证公有云或 CNCF 兼容的 Kubernetes 集群是否处于支持的版本范围内

版本支持矩阵

下表显示了每个 ACP 版本支持的 Kubernetes 版本。

INFO

表中列出了 ACP 的小版本号，不区分补丁版本。补丁版本仅包含错误修复和安全更新，因此同一小版本内所有补丁版本的 Kubernetes 小版本保持一致。

从 **ACP 4.1** 开始，每个 ACP 版本仅支持 一个 **Kubernetes** 版本 用于集群创建。 这确保了一致性并简化了新集群的升级路径。

ACP 版本	支持用于集群创建的版本	兼容版本
ACP 4.2	1.33	1.33, 1.32, 1.31, 1.30
ACP 4.1	1.32	1.32, 1.31, 1.30, 1.29
ACP 4.0	1.31, 1.30, 1.29, 1.28	1.31, 1.30, 1.29, 1.28

升级要求

对于 ACP 4.2 及更早版本，所有工作负载集群必须在升级 ACP global 集群之前，升级到兼容版本列表中的 最新 Kubernetes 版本。

在未来版本中，工作负载集群只需处于兼容版本范围内即可升级 ACP global 集群。

发版日志

目录

4.0.8

修复的问题

- 长期未登录而被系统自动禁用的用户，在管理员手动激活后会被再次自动禁用，导致激活操作无法生效。此问题已经解决。

- 修复手动删除 ovn-db 文件后 ovn-central 无法自动恢复的问题
- 修复由于 ovn-central 竞态问题引发的同时存在两个 leader 的问题。

已知问题

- 在某些情况下，用户会发现平台自动在 ResourcePatch 资源中记录的操作与用户实际对组件做的修改不一致，导致 ResourcePatch 控制器 apply 时，引发组件的非预期更改。
  临时解决方案： 用户需手动修改 ResourcePatch 资源，使其与预期变更保持一致。
- 之前当集群中存在 Display Name 为空的节点时，用户在节点详情页面打开面包屑的节点下拉筛选框，会无法通过输入内容来筛选节点。此问题已在 ACP 4.2.0 修复。
- 日志归档结束后未删除临时文件，导致磁盘无法得到释放，此问题已修复
- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。
- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。
- 偶发情况下，Pod 可能卡在 Terminating 状态且无法被 containerd 删除。尽管 containerd 执行了删除操作，但容器仍处于伪运行状态。containerd 日志显示 "OCI runtime exec failed: exec failed: cannot exec in a stopped container: unknown"，而容器状态显示为 Running。此问题在 containerd 1.7.23 版本中极少发生（仅观察到一次），触发时只影响单个 Pod。如遇此情况，可通过重启 containerd 作为临时解决方法。这是 containerd 社区已知问题，详见 https://github.com/containerd/containerd/issues/6080。
- 当单容器的日志量过大时(标准输出或者文件日志)，会出现一个日志文件达到了 rotate 的阈值，触发了 rotate，但是其中的日志内容还没有采集完毕，从而导致新老日志文件同时采集，日志顺序混乱。
- Statefulset 的 POD 停止后再启动，平台会以当天 POD 的最早的运行时间为开始时间，以最晚的运行时间为结束时间，而忽略中间没有运行的时间。最终导致该 POD 的计量数据比实际时长多。
- 将集群升级到 Kubernetes 1.31 时，集群中的所有 Pod 将会重启。这是由于 Kubernetes 1.31 中对 Pod Spec 字段的变更所导致，无法避免。更多详情，请参阅 Kubernetes 问题报告：https://github.com/kubernetes/kubernetes/issues/129385
- 通过 YAML 创建应用时使用 defaultMode 字段导致应用创建失败。
  操作路径：容器平台 → 应用管理 → 应用列表 → 通过 YAML 创建（Create from YAML），当提交的 YAML 文件中包含 defaultMode 字段（通常用于 ConfigMap/Secret 的卷挂载权限

配置）时，应用创建会失败并返回校验错误。
解决方案：创建应用前手动移除 YAML 中所有 defaultMode 声明。

- ceph-mgr 创建的默认存储池 .mgr 会使用默认的 Crush Rule，在延伸集群中不能正常选出 osd，所以必须使用 CephBlockPool 创建名为 .mgr 的存储池，但是因为时序上的不确定导致 mgr 可能先于 Rook Operator 去创建 .mgr 存储池导致出现该问题。
  遇到该问题后可以尝试重启 rook-ceph-mgr 的 pod，如果不能恢复需要清理后重新部署。
- 当 Helm Chart 中设置了 pre-delete post-delete hook。
  执行删除模板应用，卸载 Chart 时，遇到某些原因导致 hook 执行失败，进而导致应用无法删除。需要排查原因，并优先解决 hook 执行失败的问题。

4.0.7

修复的问题

- 在此更新之前，Tekton Pipeline 组件存在 Kubernetes STIG 安全漏洞，其中密钥通过 tekton-hub-api 部署中的环境变量暴露，违反了安全最佳实践。通过此次更新，环境变量中的密钥挂载逻辑已被完全移除，确保 tekton-hub-api 部署不再暴露任何凭据，符合 Kubernetes STIG 安全要求。
- 在此更新之前，Tekton Results 中的 tekton-results-retention-policy-agent 容器在环境变量中包含敏感信息，在容器操作和日志记录场景中存在以明文形式暴露凭据的安全风险。通过此次更新，敏感信息已得到适当保护并从环境变量中移除，以防止凭据泄露，确保 retention-policy-agent 容器在其配置中不再包含明文密码或令牌，从而增强了 Tekton Results 系统的整体安全态势。
- 在此更新之前，tekton-results-postgres-0 中的 PostgreSQL 容器包含带有敏感信息的环境变量，如 PASSWORD、password、TOKEN 和 token，当这些凭据以明文形式暴露时存在安全风险。通过此次更新，敏感环境变量已得到适当保护，不再包含明文密码或令牌，确保敏感凭据得到安全处理，不会在容器环境变量中暴露。
- 在此更新之前，tekton-results-api 容器的环境变量包含敏感信息，当这些凭据以明文形式暴露时存在安全风险。通过此次更新，敏感环境变量已得到适当保护，密码和令牌信息不再以明文形式暴露，增强了 tekton-results-api 组件的安全性。
- 在此更新之前，流水线界面存在多个显示问题，包括文本显示问题、变量完成多行功能用户体验不佳，以及更新触发器时的不稳定行为，其中参数和工作空间有时会出现有时会消失，需要用户重新选择流水线才能使它们出现（包括流水线列表）。通过此次更新，这些显示问

题已得到解决。流水线和流水线运行页面现在正确显示，具有改进的文本渲染、增强的变量完成多行功能以提供更好的用户体验，以及稳定的触发器更新行为，其中参数和工作空间一致出现，无需重新选择流水线。

- 之前 upmachinepool 资源的 status 字段会保存对应的 machine 资源，但未进行排序，导致在每次 reconcile 时都被判定为需要更新，从而造成审计数据过大。该问题现已修复。
- 修复了在项目导入命名空间过程中，修改 Pod 安全策略配置不生效的问题。
- 修复了在控制台中更新 Deployment 时，特定操作顺序会导致容器 lifecycle 配置意外丢失的问题。

已知问题

- 之前当集群中存在 Display Name 为空的节点时，用户在节点详情页面打开面包屑的节点下拉筛选框，会无法通过输入内容来筛选节点。此问题已在 ACP 4.2.0 修复。
- 日志归档结束后未删除临时文件，导致磁盘无法得到释放，此问题已修复
- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。
- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。
- 偶发情况下，Pod 可能卡在 Terminating 状态且无法被 containerd 删除。尽管 containerd 执行了删除操作，但容器仍处于伪运行状态。containerd 日志显示 "OCI runtime exec failed: exec failed: cannot exec in a stopped container: unknown"，而容器状态显示为 Running。此问题在 containerd 1.7.23 版本中极少发生（仅观察到一次），触发时只影响单个 Pod。如遇此情况，可通过重启 containerd 作为临时解决方法。这是 containerd 社区已知问题，详见 https://github.com/containerd/containerd/issues/6080。
- 当单容器的日志量过大时(标准输出或者文件日志)，会出现一个日志文件达到了 rotate 的阈值，触发了 rotate，但是其中的日志内容还没有采集完毕，从而导致新老日志文件同时采集，日志顺序混乱。
- Statefulset 的 POD 停止后再启动，平台会以当天 POD 的最早的运行时间为开始时间，以最晚的运行时间为结束时间，而忽略中间没有运行的时间。最终导致该 POD 的计量数据比实际时长多。
- 将集群升级到 Kubernetes 1.31 时，集群中的所有 Pod 将会重启。这是由于 Kubernetes 1.31 中对 Pod Spec 字段的变更所导致，无法避免。更多详情，请参阅 Kubernetes 问题报告：https://github.com/kubernetes/kubernetes/issues/129385

- 修复手动删除 ovn-db 文件后 ovn-central 无法自动恢复的问题
- 修复由于 ovn-central 竞态问题引发的同时存在两个 leader 的问题。
- 通过 YAML 创建应用时使用 defaultMode 字段导致应用创建失败。
  操作路径：容器平台 → 应用管理 → 应用列表 → 通过 YAML 创建（Create from YAML），当提交的 YAML 文件中包含 defaultMode 字段（通常用于 ConfigMap/Secret 的卷挂载权限配置）时，应用创建会失败并返回校验错误。
  解决方案：创建应用前手动移除 YAML 中所有 defaultMode 声明。
- ceph-mgr 创建的默认存储池 .mgr 会使用默认的 Crush Rule，在延伸集群中不能正常选出 osd，所以必须使用 CephBlockPool 创建名为 .mgr 的存储池，但是因为时序上的不确定导致 mgr 可能先于 Rook Operator 去创建 .mgr 存储池导致出现该问题。
  遇到该问题后可以尝试重启 rook-ceph-mgr 的 pod，如果不能恢复需要清理后重新部署。
- 当 Helm Chart 中设置了 pre-delete post-delete hook。
  执行删除模板应用，卸载 Chart 时，遇到某些原因导致 hook 执行失败，进而导致应用无法删除。需要排查原因，并优先解决 hook 执行失败的问题。

4.0.6

修复的问题

- 修复了在平台从 v3.x 升级至 v4.x 后，若业务集群未升级，则其在新自定义监控面板中创建的监控指标无法用于 HPA 的问题。

已知问题

- 之前 upmachinepool 资源的 status 字段会保存对应的 machine 资源，但未进行排序，导致在每次 reconcile 时都被判定为需要更新，从而造成审计数据过大。该问题现已修复。
- 之前当集群中存在 Display Name 为空的节点时，用户在节点详情页面打开面包屑的节点下拉筛选框，会无法通过输入内容来筛选节点。此问题已在 ACP 4.2.0 修复。
- 日志归档结束后未删除临时文件，导致磁盘无法得到释放，此问题已修复
- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。

- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。
- 偶发情况下，Pod 可能卡在 Terminating 状态且无法被 containerd 删除。尽管 containerd 执行了删除操作，但容器仍处于伪运行状态。containerd 日志显示 "OCI runtime exec failed: exec failed: cannot exec in a stopped container: unknown"，而容器状态显示为 Running。此问题在 containerd 1.7.23 版本中极少发生（仅观察到一次），触发时只影响单个 Pod。如遇此情况，可通过重启 containerd 作为临时解决方法。这是 containerd 社区已知问题，详见 https://github.com/containerd/containerd/issues/6080。
- 当单容器的日志量过大时(标准输出或者文件日志)，会出现一个日志文件达到了 rotate 的阈值，触发了 rotate，但是其中的日志内容还没有采集完毕，从而导致新老日志文件同时采集，日志顺序混乱。
- Statefulset 的 POD 停止后再启动，平台会以当天 POD 的最早的运行时间为开始时间，以最晚的运行时间为结束时间，而忽略中间没有运行的时间。最终导致该 POD 的计量数据比实际时长多。
- 将集群升级到 Kubernetes 1.31 时，集群中的所有 Pod 将会重启。这是由于 Kubernetes 1.31 中对 Pod Spec 字段的变更所导致，无法避免。更多详情，请参阅 Kubernetes 问题报告：https://github.com/kubernetes/kubernetes/issues/129385
- 修复手动删除 ovn-db 文件后 ovn-central 无法自动恢复的问题
- 修复由于 ovn-central 竞态问题引发的同时存在两个 leader 的问题。
- 通过 YAML 创建应用时使用 defaultMode 字段导致应用创建失败。
  操作路径：容器平台 → 应用管理 → 应用列表 → 通过 YAML 创建（Create from YAML），当提交的 YAML 文件中包含 defaultMode 字段（通常用于 ConfigMap/Secret 的卷挂载权限配置）时，应用创建会失败并返回校验错误。
  解决方案：创建应用前手动移除 YAML 中所有 defaultMode 声明。
- 修复了在控制台中更新 Deployment 时，特定操作顺序会导致容器 lifecycle 配置意外丢失的问题。
- ceph-mgr 创建的默认存储池 .mgr 会使用默认的 Crush Rule，在延伸集群中不能正常选出 osd，所以必须使用 CephBlockPool 创建名为 .mgr 的存储池，但是因为时序上的不确定导致 mgr 可能先于 Rook Operator 去创建 .mgr 存储池导致出现该问题。
  遇到该问题后可以尝试重启 rook-ceph-mgr 的 pod，如果不能恢复需要清理后重新部署。
- 当 Helm Chart 中设置了 pre-delete post-delete hook。
  执行删除模板应用，卸载 Chart 时，遇到某些原因导致 hook 执行失败，进而导致应用无法删除。需要排查原因，并优先解决 hook 执行失败的问题。

4.0.5

修复的问题

- 之前从集群卸载 Operator 后，其状态会错误地显示为 Absent，尽管实际状态仍为 Ready。用户需要通过 violet upload 手动重新上传才能恢复。此问题现已修复，Operator 在卸载后将正确显示为 Ready。
- 使用 violet upload 上传 Operator 新版本后，偶尔会出现无法安装新版本的情况。该问题已被修复。
- 当 Operator 或 Cluster Plugin 中包含多个前端扩展时，前端扩展的左侧导航可能会出现点击无反应的问题。此前的临时解决方法是为扩展的 ConfigMap 添加注解 cpaas.io/auto-sync: "false"。该问题现已在代码层面正式修复。

已知问题

- Redis 哨兵实例从 v5 升级到 v7 时，偶现脑裂的情况，会导致数据丢失。
  解决方案：在跨版本升级前，对 Redis 实例进行数据备份。
- 当集群网络异常， PostgreSQL 实例的主节点 Lable 更新失败时，会导致 PostgreSQL 状态异常，部分新建连接可能会失败。
  临时解决方案：网络恢复正常后，PostgreSQL 实例会自动恢复至正常状态。
- 在此更新之前，流水线界面存在多个显示问题，包括文本显示问题、变量完成多行功能用户体验不佳，以及更新触发器时的不稳定行为，其中参数和工作空间有时会出现有时会消失，需要用户重新选择流水线才能使它们出现（包括流水线列表）。通过此次更新，这些显示问题已得到解决。流水线和流水线运行页面现在正确显示，具有改进的文本渲染、增强的变量完成多行功能以提供更好的用户体验，以及稳定的触发器更新行为，其中参数和工作空间一致出现，无需重新选择流水线。
- 之前 upmachinepool 资源的 status 字段会保存对应的 machine 资源，但未进行排序，导致在每次 reconcile 时都被判定为需要更新，从而造成审计数据过大。该问题现已修复。
- 之前当集群中存在 Display Name 为空的节点时，用户在节点详情页面打开面包屑的节点下拉筛选框，会无法通过输入内容来筛选节点。此问题已在 ACP 4.2.0 修复。
- 日志归档结束后未删除临时文件，导致磁盘无法得到释放，此问题已修复
- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。

- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。
- 偶发情况下，Pod 可能卡在 Terminating 状态且无法被 containerd 删除。尽管 containerd 执行了删除操作，但容器仍处于伪运行状态。containerd 日志显示 "OCI runtime exec failed: exec failed: cannot exec in a stopped container: unknown"，而容器状态显示为 Running。此问题在 containerd 1.7.23 版本中极少发生（仅观察到一次），触发时只影响单个 Pod。如遇此情况，可通过重启 containerd 作为临时解决方法。这是 containerd 社区已知问题，详见 https://github.com/containerd/containerd/issues/6080。
- 当单容器的日志量过大时(标准输出或者文件日志)，会出现一个日志文件达到了 rotate 的阈值，触发了 rotate，但是其中的日志内容还没有采集完毕，从而导致新老日志文件同时采集，日志顺序混乱。
- Statefulset 的 POD 停止后再启动，平台会以当天 POD 的最早的运行时间为开始时间，以最晚的运行时间为结束时间，而忽略中间没有运行的时间。最终导致该 POD 的计量数据比实际时长多。
- 将集群升级到 Kubernetes 1.31 时，集群中的所有 Pod 将会重启。这是由于 Kubernetes 1.31 中对 Pod Spec 字段的变更所导致，无法避免。更多详情，请参阅 Kubernetes 问题报告：https://github.com/kubernetes/kubernetes/issues/129385
- 修复手动删除 ovn-db 文件后 ovn-central 无法自动恢复的问题
- 修复由于 ovn-central 竞态问题引发的同时存在两个 leader 的问题。
- 修复了在平台从 v3.x 升级至 v4.x 后，若业务集群未升级，则其在新自定义监控面板中创建的监控指标无法用于 HPA 的问题。
- 通过 YAML 创建应用时使用 defaultMode 字段导致应用创建失败。
  操作路径：容器平台 → 应用管理 → 应用列表 → 通过 YAML 创建（Create from YAML），当提交的 YAML 文件中包含 defaultMode 字段（通常用于 ConfigMap/Secret 的卷挂载权限配置）时，应用创建会失败并返回校验错误。
  解决方案：创建应用前手动移除 YAML 中所有 defaultMode 声明。
- 修复了在控制台中更新 Deployment 时，特定操作顺序会导致容器 lifecycle 配置意外丢失的问题。
- ceph-mgr 创建的默认存储池 .mgr 会使用默认的 Crush Rule，在延伸集群中不能正常选出 osd，所以必须使用 CephBlockPool 创建名为 .mgr 的存储池，但是因为时序上的不确定导致 mgr 可能先于 Rook Operator 去创建 .mgr 存储池导致出现该问题。
  遇到该问题后可以尝试重启 rook-ceph-mgr 的 pod，如果不能恢复需要清理后重新部署。
- 当 Helm Chart 中设置了 pre-delete post-delete hook。
  执行删除模板应用，卸载 Chart 时，遇到某些原因导致 hook 执行失败，进而导致应用无法

删除。需要排查原因，并优先解决 hook 执行失败的问题。

4.0.4

修复的问题

- 以前在升级集群时，会遗留 CRI（Container Runtime Interface）Pod，从而阻塞集群继续升级到 4.1。该问题已在 4.0.4 中修复。

已知问题

此次发版无相关问题。

4.0.3

修复的问题

- 修复了无法删除使用 Calico 的高可用集群 master 节点的问题。

已知问题

- 以前在升级集群时，会遗留 CRI（Container Runtime Interface）Pod，从而阻塞集群继续升级到 4.1。该问题已在 4.0.4 中修复。
- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。
- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。
- 偶发情况下，Pod 可能卡在 Terminating 状态且无法被 containerd 删除。尽管 containerd 执行了删除操作，但容器仍处于伪运行状态。containerd 日志显示 "OCI runtime exec failed: exec failed: cannot exec in a stopped container: unknown"，而容器状态显示为 Running。此问题在 containerd 1.7.23 版本中极少发生（仅观察到一次），触发时只影响单个 Pod。如

遇此情况，可通过重启 containerd 作为临时解决方法。这是 containerd 社区已知问题，详见 https://github.com/containerd/containerd/issues/6080。

- 将集群升级到 Kubernetes 1.31 时，集群中的所有 Pod 将会重启。这是由于 Kubernetes 1.31 中对 Pod Spec 字段的变更所导致，无法避免。更多详情，请参阅 Kubernetes 问题报告：https://github.com/kubernetes/kubernetes/issues/129385

4.0.2

修复的问题

- 修复了在托管到平台的公有云 Kubernetes 集群（如 ACK）上执行节点 Drain 操作失败并返回 404 的问题。

已知问题

- 修复了无法删除使用 Calico 的高可用集群 master 节点的问题。
- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。
- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。
- 偶发情况下，Pod 可能卡在 Terminating 状态且无法被 containerd 删除。尽管 containerd 执行了删除操作，但容器仍处于伪运行状态。containerd 日志显示 "OCI runtime exec failed: exec failed: cannot exec in a stopped container: unknown"，而容器状态显示为 Running。此问题在 containerd 1.7.23 版本中极少发生（仅观察到一次），触发时只影响单个 Pod。如遇此情况，可通过重启 containerd 作为临时解决方法。这是 containerd 社区已知问题，详见 https://github.com/containerd/containerd/issues/6080。
- 将集群升级到 Kubernetes 1.31 时，集群中的所有 Pod 将会重启。这是由于 Kubernetes 1.31 中对 Pod Spec 字段的变更所导致，无法避免。更多详情，请参阅 Kubernetes 问题报告：https://github.com/kubernetes/kubernetes/issues/129385

4.0.1

修复的问题

- 在集群 api-server 压力过大的场景下，kyverno-report-controller 中的 aggregate worker 有概率无法启动，导致合规报告无法正常创建。这会阻止 PolicyReport 资源的创建，使 Web Console 无法展示违规资源信息或只能显示部分报告数据。排查方法是检查 kyverno-report-controller pod 日志中是否存在 "starting worker aggregate-report-controller/worker" 信息以验证其是否正常运行。如未正常运行，临时解决方案是手动重启 kyverno-report-controller。

已知问题

- 修复了无法删除使用 Calico 的高可用集群 master 节点的问题。
- 修复了在托管到平台的公有云 Kubernetes 集群（如 ACK）上执行节点 Drain 操作失败并返回 404 的问题。
- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。
- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。
- 偶发情况下，Pod 可能卡在 Terminating 状态且无法被 containerd 删除。尽管 containerd 执行了删除操作，但容器仍处于伪运行状态。containerd 日志显示 "OCI runtime exec failed: exec failed: cannot exec in a stopped container: unknown"，而容器状态显示为 Running。此问题在 containerd 1.7.23 版本中极少发生（仅观察到一次），触发时只影响单个 Pod。如遇此情况，可通过重启 containerd 作为临时解决方法。这是 containerd 社区已知问题，详见 https://github.com/containerd/containerd/issues/6080。
- 将集群升级到 Kubernetes 1.31 时，集群中的所有 Pod 将会重启。这是由于 Kubernetes 1.31 中对 Pod Spec 字段的变更所导致，无法避免。更多详情，请参阅 Kubernetes 问题报告：https://github.com/kubernetes/kubernetes/issues/129385

4.0.0

新功能与增强

安装与升级：模块化架构

我们彻底重新设计了平台架构，提供前所未有的灵活性、更快的更新速度以及更低的运维负担。

简化安装流程
平台现通过一个精简的核心包部署，仅包含必要组件。基础搭建完成后，客户可以根据需求选择所需的 Operators 或集群插件——无论是 DevOps、Service Mesh 还是其他专业功能——并单独下载、上传和安装。

针对性补丁

- 补丁版本仅包含实际需要修复的组件。
- 未修复的组件保持不变，确保平台其他部分不受影响。
- 客户通过平台内置的标准化升级机制应用补丁，而非手动更新单个组件，使维护和追踪更为简便。

智能升级

- 升级过程中仅替换并重启有新代码的组件。
- 未修改的组件保持现有版本和运行状态。
- 最大限度减少停机时间，缩短维护窗口，提升升级体验。

独立组件版本管理

- 大多数 Operators 遵循独立的发布计划，与核心平台分开。
- 新功能和修复一经准备即可上线，无需等待整个平台更新。
- 此举加快交付速度，让客户更快享受改进成果。

集群：基于 **Cluster API** 的声明式集群生命周期管理

本地集群现利用 Kubernetes Cluster API 实现全声明式操作，包括：

- 集群创建
- 节点扩缩容与加入

该无缝的 Cluster API 集成直接融入您的 IaC 流水线，实现集群生命周期的端到端编程式控制。

Operator 与扩展：全面能力可见性

完整的 **Operator** 目录

OperatorHub 现展示所有支持的 Operators，无论其包是否已上传至平台。此改进：

- 即使在隔离环境中也能全面了解平台能力
- 消除用户对可用功能的认知盲区
- 降低探索平台能力时的发现障碍

版本灵活性

用户安装时可选择特定 Operator 版本，而非仅限最新版本，增强组件兼容性和升级路径的控制。

Web Console 扩展

Operators 现支持基于锚点的 Web Console 扩展，允许将功能特定的前端镜像包含在 Operators 中，并无缝集成到平台 Web Console。

集群插件增强

所有关于 Operator 可见性、版本选择和 Web Console 扩展的改进同样适用于集群插件，确保所有平台扩展的一致用户体验。

日志查询逻辑优化

日志查询页面进行了优化，解决用户使用日志查询功能时遇到的体验和性能问题：

- 原有单选框替换为高级搜索组件，使用日志搜索如同使用 GIT 搜索一般便捷。
- 日志内容查询条件独立设置。
- 时间查询条件位置调整，调整时间范围时不会重置日志筛选条件。
- 优化日志查询 API，提升整体查询性能。

ElasticSearch 升级至 **8.17**

我们将 ElasticSearch 版本升级至 8.17，以跟进社区的新功能和改进。

ALB 认证

ALB 现支持多种认证机制，允许用户在 Ingress 级别处理认证，而无需在每个后端应用中实现。

ALB 支持 ingress-nginx 注解

本版本新增对 ALB 中常见 ingress-nginx 注解的支持，包括 keepalive 设置、超时配置和 HTTP 重定向，增强与社区 ingress-nginx 的兼容性。

Kubevirt 实时迁移优化

实时迁移过程中，网络中断时间缩短至不足 0.5 秒，且现有 TCP 连接不会断开。
此优化显著提升虚拟机迁移在生产环境中的稳定性和可靠性。

LDAP/OIDC 集成优化

LDAP/OIDC 集成表单字段进行了调整，主要包括删除不必要或重复字段以及优化字段描述。
LDAP/OIDC 集成现支持通过 YAML 配置，允许在 YAML 文件中进行用户属性映射。

源代码构建（S2I）支持

- 新增 **Alauda Container Platform Builds** operator，实现从源代码自动构建镜像。
- 支持 Java/Go/Node.js/Python 语言栈。
- 通过源代码仓库简化应用部署流程。

本地 Registry 解决方案

- **ACP Registry** 提供轻量级 Docker Registry，具备企业级功能。
- 开箱即用的镜像管理能力。
- 简化应用交付。

GitOps 模块重构

- 将 **ACP GitOps** 解耦为独立的集群插件架构。
- 升级 Argo CD 至 v2.14.x 版本。
- 增强基于 GitOps 的应用生命周期管理。

命名空间级监控

- 引入命名空间级动态监控仪表盘。
- 提供 Applications/Workloads/Pods 指标可视化。

Crossplane 集成

- 发布 **Alauda Build of Crossplane** 发行版。
- 通过 XRD 组合实现以应用为中心的资源配置。

虚拟化更新

- 升级至 KubeVirt 1.4，增强虚拟化能力。
- 优化镜像处理，加快虚拟机启动速度。
- 优化虚拟机实时迁移，支持从 UI 直接发起并显示迁移状态。
- 改进绑定网络，支持双栈（IPv4/IPv6）。
- 新增 vTPM 支持，提升虚拟机安全性。

Ceph 存储更新

- Metro-DR 通过拉伸集群实现跨可用区实时数据同步。
- Regional-DR 采用基于池的镜像增强数据保护。

TopoLVM 更新

- 新增多路径设备部署支持，提升灵活性和稳定性。

修复的问题

- 以前，Operator 上架新版本后，用户需等待 10 分钟才能安装新版本。现在等待时间已缩短至 2 分钟，使用户能更快地安装 Operator 的新版本。
- 在单节点多张卡的gpu节点上，gpu-manager 偶尔会存在，针对使用 vgpu 的应用调度不成功问题。
- 使用pgpu插件时，需要将gpu节点上的默认 runtimeclass设置为nvidia。如果不设置，可能会导致应用无法正常请求gpu资源。
- 在单张 GPU 卡上，使用 gpu-manager 无法同时创建基于 vllm、mlserver 的多个推理服务在 AI 平台上，使用 gpu-manager 创建多个推理服务时，会出现该问题；在容器平台上，使

用 gpu-manager 创建多个智能应用时，不会出现该问题。

- 使用mps时，当节点资源不足时，pod会无限重启。

已知问题

- 修复了无法删除使用 Calico 的高可用集群 master 节点的问题。
- 在从 3.18.0 升级到 4.0.1 时，如果 global 集群使用内置镜像仓库且开启了 protect-secret-files 开关，执行升级脚本可能会超时报错。此问题已在 ACP 4.1.0 修复。
- 偶发情况下，Pod 可能卡在 Terminating 状态且无法被 containerd 删除。尽管 containerd 执行了删除操作，但容器仍处于伪运行状态。containerd 日志显示 "OCI runtime exec failed: exec failed: cannot exec in a stopped container: unknown"，而容器状态显示为 Running。此问题在 containerd 1.7.23 版本中极少发生（仅观察到一次），触发时只影响单个 Pod。如遇此情况，可通过重启 containerd 作为临时解决方法。这是 containerd 社区已知问题，详见 https://github.com/containerd/containerd/issues/6080。
- 将集群升级到 Kubernetes 1.31 时，集群中的所有 Pod 将会重启。这是由于 Kubernetes 1.31 中对 Pod Spec 字段的变更所导致，无法避免。更多详情，请参阅 Kubernetes 问题报告：https://github.com/kubernetes/kubernetes/issues/129385
- 在集群 api-server 压力过大的场景下，kyverno-report-controller 中的 aggregate worker 有概率无法启动，导致合规报告无法正常创建。这会阻止 PolicyReport 资源的创建，使 Web Console 无法展示违规资源信息或只能显示部分报告数据。排查方法是检查 kyverno-report-controller pod 日志中是否存在 "starting worker aggregate-report-controller/worker" 信息以验证其是否正常运行。如未正常运行，临时解决方案是手动重启 kyverno-report-controller。
- 修复了通过 UI 界面创建的 Secret 仅包含 Username 和 Password，而缺少 auth 认证信息，与 kubectl create 行为不一致的问题。该问题曾导致依赖完整认证信息的构建工具（如 buildah）出现认证失败。
- ceph-mgr 创建的默认存储池 .mgr 会使用默认的 Crush Rule，在延伸集群中不能正常选出 osd，所以必须使用 CephBlockPool 创建名为 .mgr 的存储池，但是因为时序上的不确定导致 mgr 可能先于 Rook Operator 去创建 .mgr 存储池导致出现该问题。
  遇到该问题后可以尝试重启 rook-ceph-mgr 的 pod，如果不能恢复需要清理后重新部署。

此次发版无相关问题。

- 日志归档结束后未删除临时文件，导致磁盘无法得到释放，此问题已修复
- 当单容器的日志量过大时(标准输出或者文件日志)，会出现一个日志文件达到了 rotate 的阈值，触发了 rotate，但是其中的日志内容还没有采集完毕，从而导致新老日志文件同时采

集，日志顺序混乱。