Overview

Architecture

Release Notes

Architecture

The Alauda Container Platform (ACP) architecture is designed to provide a robust, scalable, and flexible cloud-native infrastructure solution. This document outlines the core architectural principles, cluster types, and design considerations that form the foundation of our platform.

TOC

Architectural Concepts

`global` Cluster

The `global` cluster is the central nervous system of the Alauda Container Platform. It serves as the primary management and control plane for the entire platform, responsible for:

- Platform-wide cluster management

- Tenant management and access control
- Global scheduling and orchestration
- Platform-level configuration and policy enforcement
- Monitoring and observability of the ecosystem

Key characteristics of the `global` cluster include:

- Dedicated physical resources
- Isolation from workload processing
- Centralized management capabilities
- High availability and security configuration

Workload Clusters

Workload clusters are the execution environments designed to host and run actual applications and services. These clusters are:

- Dedicated to running application workloads
- Capable of horizontal scaling
- Flexible across different infrastructure types
- Managed and orchestrated by the `global` cluster

Architectural Design Principles

1. Control Plane Isolation

The architecture mandates a strict separation between the `global` cluster and workload clusters:

- The `global` cluster uses dedicated physical resources
- Prevents resource contention and performance degradation
- Ensures stable and secure platform management

2. Operational Flexibility

ACP supports multiple operational scenarios:

- Fully separated `global` and workload clusters (Recommended for production)
- Co-located configurations for development and resource-constrained environments
- Hybrid cloud and multi-cloud architectures

3. Cluster Access and Management

Two primary methods for workload cluster integration:

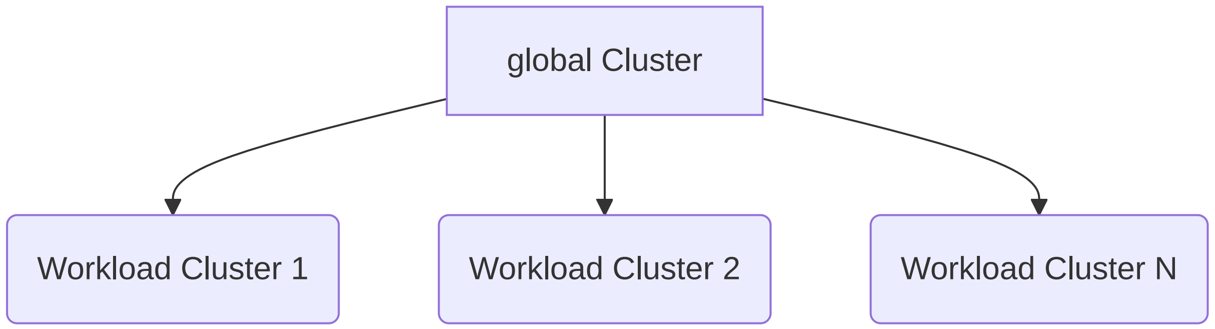
a) **Create New Workload Clusters**

- Recommended approach
- Fully managed by the `global` cluster
- Consistent configuration and compliance
- Simplified lifecycle management

b) **Attach Existing Clusters**

- Support for connecting pre-existing Kubernetes clusters
- Version and compatibility checks required
- Standardized access and management protocols

Architecture Visualization



Key Benefits of This Architecture

- **Enhanced Security**: Isolated control and execution planes
- **Scalability**: Easy addition of new workload clusters
- **Flexibility**: Support for diverse infrastructure environments
- **Efficiency**: Centralized management with distributed processing
- **Compliance**: Consistent policy enforcement across clusters

Architectural Considerations

When planning ACP architecture, consider:

- Infrastructure diversity
- Scalability requirements
- Compliance and security needs
- Application workload characteristics

Release Notes

TOC

4.0.3

Fixed Issues

- Fixed an issue where master nodes in HA clusters using Calico could not be deleted.

Known Issues

- When upgrading from 3.18.0 to 4.0.1, running the upgrade script may fail with a timeout if the global cluster uses the built-in image registry with the protect-secret-files feature enabled. There is currently no available workaround.
- Occasionally, a pod may become stuck in the Terminating state and cannot be deleted by containerd. Although containerd attempts the deletion operation, the container remains in a pseudo-running state. The containerd logs show OCI "runtime exec failed: exec failed: cannot exec in a stopped container: unknown" while the container status appears as Running. This issue occurs very rarely in containerd 1.7.23 (observed only once) and affects only individual pods when triggered. If encountered, restart containerd as a temporary workaround. This is a known issue in the containerd community, tracked at https://github.com/containerd/containerd/issues/6080.
- When upgrading clusters to Kubernetes 1.31, all pods in the cluster will restart. This behavior is caused by changes to the Pod spec fields in Kubernetes 1.31 and cannot be avoided. For more details, please refer to the Kubernetes issue: https://github.com/kubernetes/kubernetes/issues/129385

4.0.2

Fixed Issues

- Fixed an issue where performing a node drain on a public cloud Kubernetes cluster (such as ACK) managed by the platform failed with a 404 error.

Known Issues

- Fixed an issue where master nodes in HA clusters using Calico could not be deleted.
- When upgrading from 3.18.0 to 4.0.1, running the upgrade script may fail with a timeout if the global cluster uses the built-in image registry with the protect-secret-files feature enabled. There is currently no available workaround.
- Occasionally, a pod may become stuck in the Terminating state and cannot be deleted by containerd. Although containerd attempts the deletion operation, the container remains in a pseudo-running state. The containerd logs show OCI "runtime exec failed: exec failed: cannot exec in a stopped container: unknown" while the container status appears as Running. This issue occurs very rarely in containerd 1.7.23 (observed only once) and affects only individual pods when triggered. If encountered, restart containerd as a temporary workaround. This is a known issue in the containerd community, tracked at https://github.com/containerd/containerd/issues/6080.
- When upgrading clusters to Kubernetes 1.31, all pods in the cluster will restart. This behavior is caused by changes to the Pod spec fields in Kubernetes 1.31 and cannot be avoided. For more details, please refer to the Kubernetes issue: https://github.com/kubernetes/kubernetes/issues/129385

4.0.1

Fixed Issues

- Under high api-server pressure, the aggregate worker in kyverno-report-controller may occasionally fail to start, preventing proper creation of compliance reports. This results in

PolicyReport resources not being created, causing the Web Console to either display no compliance violation information or only partial report data. To troubleshoot, check the kyverno-report-controller pod logs for the presence of "starting worker aggregate-report-controller/worker" messages to verify proper operation. If the worker is not running, manually restart the kyverno-report-controller as a temporary solution.

Known Issues

- Fixed an issue where master nodes in HA clusters using Calico could not be deleted.
- Fixed an issue where performing a node drain on a public cloud Kubernetes cluster (such as ACK) managed by the platform failed with a 404 error.
- When upgrading from 3.18.0 to 4.0.1, running the upgrade script may fail with a timeout if the global cluster uses the built-in image registry with the protect-secret-files feature enabled. There is currently no available workaround.
- Occasionally, a pod may become stuck in the Terminating state and cannot be deleted by containerd. Although containerd attempts the deletion operation, the container remains in a pseudo-running state. The containerd logs show OCI "runtime exec failed: exec failed: cannot exec in a stopped container: unknown" while the container status appears as Running. This issue occurs very rarely in containerd 1.7.23 (observed only once) and affects only individual pods when triggered. If encountered, restart containerd as a temporary workaround. This is a known issue in the containerd community, tracked at https://github.com/containerd/containerd/issues/6080.
- When upgrading clusters to Kubernetes 1.31, all pods in the cluster will restart. This behavior is caused by changes to the Pod spec fields in Kubernetes 1.31 and cannot be avoided. For more details, please refer to the Kubernetes issue: https://github.com/kubernetes/kubernetes/issues/129385

4.0.0

Features and Enhancements

Installation and Upgrade: Modular Architecture

We've completely redesigned our platform's architecture to provide unprecedented flexibility, faster updates, and reduced operational overhead.

Streamlined Installation Our platform is now deployed via a lean core package containing only the essential components. Once the foundation is in place, customers can pick and choose exactly which Operators or cluster plugins they need—whether DevOps, Service Mesh, or other specialized features—and download, upload, and install them individually.

Targeted Patches

- Patch releases include only those components that actually require bug fixes.
- Components without fixes remain exactly as they are, ensuring the rest of the platform stays untouched.
- Customers apply patches through the platform’s built-in, standardized upgrade mechanism—rather than manually updating individual components—making maintenance and tracking far more straightforward.

Intelligent Upgrades

- During an upgrade, only components with new code are replaced and restarted.
- Unmodified components retain their existing versions and uptime.
- This minimizes downtime and shortens the maintenance window for a smoother upgrade experience.

Independent Component Versioning

- Most Operators follow their own release schedules, separate from the core platform.
- New features and fixes go live as soon as they’re ready—no need to wait for a full-platform update.
- This approach accelerates delivery and lets customers benefit from improvements faster.

Clusters: Declarative Cluster Lifecycle Management with Cluster API

On-premises clusters now leverage the Kubernetes Cluster API for fully declarative operations, including:

- Cluster creation
- Node scaling and joining

This seamless Cluster API integration fits directly into your IaC pipelines, enabling end-to-end, programmatic control over your cluster lifecycle.

Operator & Extension: Comprehensive Capability Visibility

Complete Operator Catalog

The OperatorHub now displays all supported Operators regardless of whether their packages have been uploaded to the platform. This enhancement:

- Provides full visibility into platform capabilities even in air-gapped environments
- Eliminates information gaps between what's available and what's known to users
- Reduces discovery friction when exploring platform capabilities

Version Flexibility

Users can now select specific Operator versions during installation rather than being limited to only the latest version, providing greater control over component compatibility and upgrade paths.

Web Console Extensions

Operators now support anchor-based Web Console extensions, allowing functionality-specific frontend images to be included within Operators and seamlessly integrated into the platform's Web Console.

Cluster Plugin Enhancements

All improvements to Operator visibility, version selection, and Web Console extension capabilities also apply to cluster plugins, ensuring consistent user experience across all platform extensions.

Log query logic optimization

The log query page has been optimized to solve the experience and performance problems users encounter when using the log query function:

- The original radio box has been replaced with the advanced search component. Now you can use the log search as you use the GIT search.
- Independent query conditions for log content

- The location of the time query criteria has been adjusted. Now you will not reset your log filter criteria when you adjust the time range.
- Optimized the log query API to improve the overall query performance

ElasticSearch upgrade to 8.17

We upgraded the version of ElasticSearch to 8.17 to follow up the functions and improvements of the community.

ALB authentication

ALB now support various authentication mechanism, which allows user to handle authentication at Ingress level instead of implementing it in each backend application.

ALB supports ingress-nginx annotations

This release adds support for common ingress-nginx annotations in ALB, including keepalive settings, timeout configurations, and HTTP redirects, enhancing compatibility with the community ingress-nginx.

Kubevirt live migration optimization

During the live migration process, the network interruption time has been reduced to less than 0.5 seconds, and existing TCP connections will not be disconnected. This optimization significantly improves the stability and reliability of virtual machine migrations in production environments.

LDAP/OIDC integration optimization

The LDAP/OIDC integration form fields have been adjusted, mainly including removal of unnecessary/duplicate fields and optimization of field descriptions. LDAP/OIDC integration now supports configuration through YAML, allowing user attribute mapping within the YAML file.

Source to Image (S2I) Support

- Added **Alauda Container Platform Builds** operator for automated image building from source code
- Supports Java/Go/Node.js/Python language stacks

- Streamlines application deployment via source code repositories

On-prem Registry Solution

- **ACP Registry** delivered lightweight Docker Registry with enterprise-ready features
- Provides out-of-the-box image management capabilities
- Simplifies application delivery

GitOps Module Refactoring

- Decoupled **ACP GitOps** into standalone cluster plugin architecture
- Upgraded Argo CD to v2.14.x version
- Enhanced GitOps-based application lifecycle management

Namespace-level Monitoring

- Introduced dynamic monitoring dashboards at namespace level
- Provides Applications/Workloads/Pods metrics visualization

Crossplane Integration

- Released **Alauda Build of Crossplane** distribution
- Implements app-centric provisioning via XRD compositions

Virtualization Updates

- Upgraded to KubeVirt 1.4 for enhanced virtualization capabilities
- Optimized image handling for faster VM provisioning
- Optimized VM live migration, now initiable directly from the UI with visible migration status
- Improved binding networking with dual-stack (IPv4/IPv6) support
- Added vTPM support to enhance VM security

Ceph Storage Updates

- Metro-DR with stretch cluster enables real-time data synchronization across availability zones

- Regional-DR with pool-based mirroring enhances data protection

TopoLVM Updates

- Added support for multipath device deployment, improving flexibility and stability

Fixed Issues

- Previously, after publishing a new Operator version, users had to wait 10 minutes before installing it. This waiting period has been reduced to 2 minutes, allowing faster installation of new Operator versions.

- On gpu nodes with multiple cards on a single node, gpu-manager occasionally exists, with unsuccessful scheduling issues for applications using vgpu.
- When using the pgpu plugin, you need to set the default runtimeclass on the gpu node to nvidia. if you don't, it may cause the application to not be able to request gpu resources properly.
- On a single GPU card, gpu-manager cannot create multiple inference services based on vllm, mlserver at the same time.
  On AI platforms, this issue occurs when gpu-manager is used to create multiple inference services; on container platforms, this issue does not occur when gpu-manager is used to create multiple smart applications.
- With mps, pods restart indefinitely when nodes are low on resources.

Known Issues

- Fixed an issue where master nodes in HA clusters using Calico could not be deleted.
- When upgrading from 3.18.0 to 4.0.1, running the upgrade script may fail with a timeout if the global cluster uses the built-in image registry with the protect-secret-files feature enabled. There is currently no available workaround.
- Occasionally, a pod may become stuck in the Terminating state and cannot be deleted by containerd. Although containerd attempts the deletion operation, the container remains in a pseudo-running state. The containerd logs show OCI "runtime exec failed: exec failed: cannot exec in a stopped container: unknown" while the container status appears as Running. This issue occurs very rarely in containerd 1.7.23 (observed only once) and affects only individual pods when triggered. If encountered, restart containerd as a

temporary workaround. This is a known issue in the containerd community, tracked at https://github.com/containerd/containerd/issues/6080.

- When upgrading clusters to Kubernetes 1.31, all pods in the cluster will restart. This behavior is caused by changes to the Pod spec fields in Kubernetes 1.31 and cannot be avoided. For more details, please refer to the Kubernetes issue: https://github.com/kubernetes/kubernetes/issues/129385
- Under high api-server pressure, the aggregate worker in kyverno-report-controller may occasionally fail to start, preventing proper creation of compliance reports. This results in PolicyReport resources not being created, causing the Web Console to either display no compliance violation information or only partial report data. To troubleshoot, check the kyverno-report-controller pod logs for the presence of "starting worker aggregate-report-controller/worker" messages to verify proper operation. If the worker is not running, manually restart the kyverno-report-controller as a temporary solution.
- The default pool .mgr created by ceph-mgr uses the default Crush Rule, which may fail to properly select OSDs in a stretched cluster. To resolve this, the .mgr pool must be created using CephBlockPool. However, due to timing uncertainties, ceph-mgr might attempt to create the .mgr pool before the Rook Operator completes its setup, leading to conflicts.
  If encountering this issue, restart the rook-ceph-mgr Pod to trigger reinitialization.
  If unresolved, manually clean up the conflicting .mgr pool and redeploy the cluster to ensure proper creation order.

No issues in this release.

- When the amount of logs in a single container is too large (standard output or file logs), it can happen that a log file reaches the rotate threshold and triggers a rotate, but the contents of the logs in it have not been captured yet, which results in the simultaneous capture of the old and new log files, and a chaotic log order.