Overview

Architecture **Kubernetes Support Matrix** **Glossary**

Release Notes

Architecture

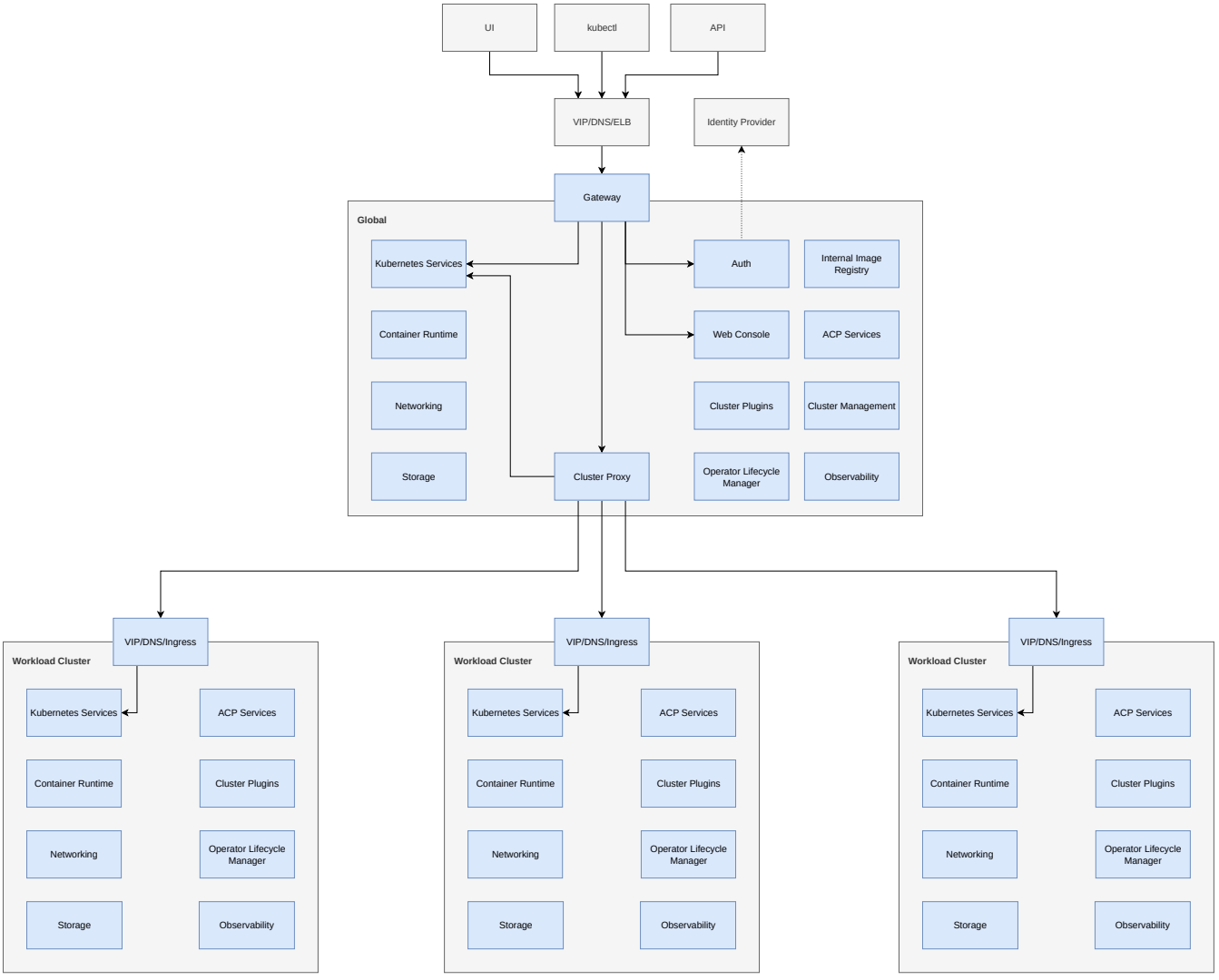
TOC

Introduction to Alauda Container Platform

The Alauda Container Platform (ACP) provides an enterprise-grade Kubernetes-based platform that enables organizations to build, deploy, and manage applications consistently across hybrid and multi-cloud environments. ACP integrates core Kubernetes capabilities with enhanced management, observability, and security services, offering a unified control plane and flexible workload clusters.

The architecture follows a **hub-and-spoke** model, consisting of a `global` cluster and multiple workload clusters. This design provides centralized governance while allowing independent workload execution and scalability.

For canonical definitions of platform-wide terms such as `global` cluster, workload cluster, and cluster plugin, see Glossary.



Core Architectural Components

Global Cluster

The `global` cluster serves as the centralized management and control hub of ACP. It provides platform-wide services such as authentication, policy management, cluster lifecycle operations, and observability. It's also a central hub for multi-cluster management and provides cross-cluster functionality.

Key components include:

- **Gateway** Acts as the main entry point to the platform. It manages API requests from the UI, CLI (kubectl), and automation tools, routing them to appropriate backend services.
- **Authentication and Authorization (Auth)** Integrates with external Identity Providers (**IdPs**) to provide Single Sign-On (SSO) and RBAC-based access control.
- **Web Console** Provides a web-based interface for ACP. It interfaces with platform APIs through the gateway.
- **Cluster Management** Handles the registration, provisioning, and lifecycle management of workload clusters.
- **ACP Services**
- **Operator Lifecycle Manager (OLM) and Cluster Plugins** Manages the installation, updates, and lifecycle of operators and cluster extensions.
- **Internal Image Registry** Offers an out-of-box integrated container image repository with role-based access.
- **Observability** Provides centralized logging, metrics, and tracing for both the `global` and workload clusters.
- **Cluster Proxy** Enables secure communication between the `global` cluster and workload clusters.

Workload Cluster

Workload clusters are Kubernetes-based environments managed by the `global` cluster. Each workload cluster runs isolated application workloads and inherits governance and configuration from the central control plane.

External Integrations

- **Identity Provider (IdP)** Supports federated authentication via standard protocols (OIDC, SAML) for unified user management.
- **API and CLI Access** Users can interact with ACP through RESTful APIs, the web console, or command-line tools like `kubectl` and `ac`.

- **Load Balancer (VIP/DNS/SLB)** Provides high availability and traffic distribution to the Gateway and ingress endpoints of the `global` and workload Clusters.

Scalability and High Availability

ACP is designed for horizontal scalability and high availability:

- Each component can be deployed redundantly to eliminate single points of failure.
- The `global` cluster supports managing dozens to hundreds of workload clusters.
- Workload clusters can scale independently according to workload demand.
- The use of VIP/DNS/Ingress ensures seamless routing and failover.

Functional Perspective

Alauda Container Platform (ACP)'s complete functionality consists of **ACP Core** and extensions based on two technical stacks: **Operator** and **Cluster Plugin**.

- **ACP Core**
  The minimal deliverable unit of ACP, providing core capabilities such as cluster management, container orchestration, projects, and user administration.
  - Meets the highest security standards
  - Delivers maximum stability
  - Offers the longest support lifecycle
- **Extensions**
  Extensions in both the Operator and Cluster Plugin stacks can be classified into:
  - **Aligned** – Life cycle strategy consisting of multiple maintenance streams, with alignment to ACP.
  - **Agnostic** – Life cycle strategy consisting of multiple maintenance streams, released independently from ACP.

  For more details about extensions, see Extend.

Technical Perspective

Platform Component Runtime All platform components run as containers within a Kubernetes management cluster (the `global` cluster).

High Availability Architecture

- The `global` cluster typically consists of at least three control plane nodes and multiple worker nodes
- High availability of etcd is central to cluster HA; see *Key Component High Availability Mechanisms* for details
- Load balancing can be provided by an external load balancer or a self-built VIP inside the cluster

Request Routing

- Client requests first pass through the load balancer or self-built VIP
- Requests are forwarded to **ALB** (the platform's default Kubernetes Ingress Gateway) running on designated ingress nodes (or control-plane nodes if configured)
- ALB routes traffic to the target component pods according to configured rules

Replica Strategy

- Core components run with at least two replicas
- Key components (such as registry, MinIO, ALB) run with three replicas

Fault Tolerance & Self-healing

- Achieved through cooperation between kubelet, kube-controller-manager, kube-scheduler, kube-proxy, ALB, and other components
- Includes health checks, failover, and traffic redirection

Data Storage & Recovery

- Control-plane configuration and platform state are stored in etcd as Kubernetes resources
- In catastrophic failures, recovery can be performed from etcd snapshots

Primary / Standby Disaster Recovery

- Two separate `global` clusters: **Primary Cluster** and **Standby Cluster**
- The disaster recovery mechanism is based on real-time synchronization of etcd data from the Primary Cluster to the Standby Cluster.
- If the Primary Cluster becomes unavailable due to a failure, services can quickly switch to the Standby Cluster.

Key Component High Availability Mechanisms

etcd

- Deployed on three (or five) control plane nodes
- Uses the RAFT protocol for leader election and data replication
- Three-node deployments tolerate up to one node failure; five-node deployments tolerate up to two
- Supports local and remote S3 snapshot backups

Monitoring Components

- **Prometheus**: Multiple instances, deduplication with Thanos Query, and cross-region redundancy
- **VictoriaMetrics**: Cluster mode with distributed VMStorage, VMInsert, and VMSelect components

Logging Components

- **Nevermore** collects logs and audit data
- **Kafka / Elasticsearch / Razor / Lanaya** are deployed in distributed and multi-replica modes

Networking Components (CNI)

- **Kube-OVN / Calico / Flannel**: Achieve HA via stateless DaemonSets or triple-replica control plane components

ALB

- Operator deployed with three replicas, leader election enabled
- Instance-level health checks and load balancing

Self-built VIP

- High-availability virtual IP based on Keepalived
- Supports heartbeat detection and active-standby failover

Harbor

- ALB-based load balancing
- PostgreSQL with Patroni HA
- Redis Sentinel mode
- Stateless services deployed in multiple replicas

Registry and MinIO

- Registry deployed with three replicas
- MinIO in distributed mode with erasure coding, data redundancy, and automatic recovery

Kubernetes Support Matrix

This document provides the Kubernetes version support matrix for ACP. This information is critical when creating clusters, upgrading ACP, and managing third-party clusters.

TOC

Overview

ACP supports multiple Kubernetes versions across different ACP releases. Understanding the supported versions is essential for:

- **Creating clusters** – Determine which Kubernetes versions can be used when provisioning new clusters
- **Upgrading ACP** – Ensure all workload clusters meet the documented compatible-version requirements before upgrading the global cluster

- **Managing third-party clusters** – Verify that public cloud or CNCF-compliant Kubernetes clusters are within the supported management range

Version Support Matrix

The following table shows the Kubernetes version support for each ACP release.

> **INFO**
>
> The table lists ACP minor versions and does not distinguish between patch versions. Patch versions only include bug fixes and security updates, so the Kubernetes minor versions remain consistent across all patch versions within the same minor release.
>
> **Starting from ACP 4.1**, each ACP release supports only **one Kubernetes version** for cluster creation. This ensures consistency and simplifies the upgrade path for new clusters.

ACP Version	Supported for Cluster Creation	Compatible Versions
ACP 4.3	1.34	1.34, 1.33, 1.32, 1.31
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

ACP 4.3 Notes

- ACP 4.3 adds support for Kubernetes 1.34 for platform-managed cluster scenarios.
- For upgrades to ACP 4.3, the workload-cluster compatible versions are 1.34, 1.33, 1.32, and 1.31.
- This means environments upgrading from ACP 4.0 to ACP 4.3 can keep workload clusters on Kubernetes 1.31 through 1.34 while upgrading the global cluster.

Third-Party Cluster Management Range

- For third-party clusters, ACP 4.3 accepts Kubernetes versions in the range `>=1.19.0 <1.35.0`.
- This management range is separate from the Compatible Versions column, which is the authoritative prerequisite for upgrading the ACP global cluster.
- Product documentation continues to list only the Kubernetes versions that have passed product validation for third-party cluster support and the default Extend baseline.
- Product validation for the Extend baseline covers the following capability areas:
  - Installing and using Operators
  - Installing and using Cluster Plugins
  - ClickHouse-based logging
  - VictoriaMetrics-based monitoring
- This does not mean that all specific Operators or Cluster Plugins are covered by product validation.
- For specific Operators or Cluster Plugins outside this baseline, refer to the relevant product documentation or contact technical support.

Upgrade Requirements

For ACP 4.3 and later, workload clusters only need to remain within the documented compatible version range before upgrading the ACP global cluster. For ACP 4.3, this means Kubernetes 1.31 through 1.34.

In ACP 4.2 and earlier, **all** workload clusters must be upgraded to the **latest** Kubernetes version in the compatible versions list **before** upgrading the ACP global cluster.

Glossary

This glossary defines canonical platform-wide terms used across Alauda Container Platform documentation. It focuses on concepts that appear in multiple sections of the product. Terms that apply only to a single workflow or subsystem should remain documented in their local pages.

TOC

Platform and Cluster Terms

Term	Definition	Related doc
Global Cluster	The centralized management and control hub of ACP. In the platform's hub-and-spoke architecture, it provides platform-wide services such as	Architecture

Term	Definition	Related doc
	authentication, policy management, cluster lifecycle operations, and observability.	
Workload Cluster	A Kubernetes-based environment managed by the `global` cluster. A workload cluster runs isolated application workloads and inherits governance and configuration from the central control plane.	Architecture
Platform-Provisioned Infrastructure	A cluster management model in which the platform provisions both machines and node operating systems, and manages the full cluster lifecycle. In this model, all nodes use an immutable operating system.	Clusters Overview
User-Provisioned Infrastructure	A cluster management model in which users provide pre-provisioned physical or virtual machines. The platform manages Kubernetes on those nodes, while node operating system management remains under user control.	Clusters Overview
Hosted Control Plane (HCP)	A deployment model in which each cluster has its own dedicated control plane, while multiple control planes are hosted as workloads on a dedicated management cluster. This model separates the control plane from worker nodes to reduce resource consumption and improve multi-cluster scalability.	About Hosted Control Plane
Managed Cluster	An existing cluster brought under the platform for centralized governance and operations. In ACP, managed clusters include existing standard Kubernetes clusters and selected public cloud clusters that are onboarded through import or registration workflows.	Managed Clusters Overview
Immutable OS	An immutable operating system used for platform-managed nodes in platform-provisioned environments. Node state is kept consistent and recoverable by treating the operating system layer as read-only and centrally managed.	Clusters Overview

Term	Definition	Related doc
Immutable Infrastructure	A cluster provisioning and operating model in which node configurations are baked into images and remain unchanged after deployment. Cluster upgrades and configuration changes are applied by replacing nodes with new images.	About Immutable Infrastructure
Project	A platform governance unit that isolates resources and personnel for a tenant or team. A project can span multiple associated clusters and acts as the management boundary for quotas, policies, and namespace ownership.	Create Project
Namespace	A Kubernetes namespace managed directly or indirectly by the platform. In ACP, a namespace can be created within or imported into a project so that it inherits project-level governance and visibility.	Importing Namespaces
Control Plane	The Kubernetes management layer that runs core cluster components such as the API server, scheduler, and controller manager.	Architecture
Control Plane Node	A node that runs Kubernetes control plane components used for cluster management. Use this term instead of outdated alternatives such as "master node".	Architecture
Worker Node	A node that runs application workloads and supporting platform components. Use this term instead of outdated alternatives such as "slave node".	Architecture

Identity and Access Terms

Term	Definition	Related doc
Identity Provider (IdP)	An external identity system that authenticates users for the platform, such as LDAP, Active Directory, or an OpenID Connect provider.	Accessing the Web Console

Term	Definition	Related doc
OpenID Connect (OIDC)	An identity layer built on OAuth 2.0 that ACP uses in several authentication and authorization scenarios.	Disabling the PKCE Plain Method

Extension and Packaging Terms

Term	Definition	Related doc
Operator	An extension mechanism built on Kubernetes custom resources and controllers that automates lifecycle management for complex applications or services. In ACP, Operators are managed through Operator Lifecycle Manager.	Operator
Operator Lifecycle Manager (OLM)	The operator management framework that handles Operator installation, upgrades, channel subscriptions, dependency resolution, and related custom resources such as `CatalogSource`, `Subscription`, and `InstallPlan`.	Operator
OperatorHub	The platform interface for discovering, installing, upgrading, and managing Operators through OLM.	Operator
Cluster Plugin	The platform's extension mechanism for chart-based plugins. Cluster plugins are managed through the `ModulePlugin`, `ModuleConfig`, and `ModuleInfo` custom resources.	Cluster Plugin

Networking and Access Terms

Term	Definition	Related doc
Ingress	A Kubernetes resource that exposes HTTP and HTTPS routes from outside the cluster to internal services. ACP uses Ingress as one of its main north-south traffic entry models.	Configure Ingresses
Gateway API	The Kubernetes networking API family that defines role-oriented resources for advanced L4 and L7 routing. In ACP, Gateway API is positioned as a next-generation traffic management model alongside Service and Ingress.	Networking Overview
Service	In Kubernetes, a Service is a method for exposing a network application that runs as one or more Pods in a cluster. In ACP, Service is a core service-discovery and traffic-exposure primitive, including `ClusterIP`, `NodePort`, and `LoadBalancer` types.	Configure Services
LoadBalancer	A Service type that exposes a Service through an external load balancer. This usually requires either a cloud-provider integration or a separately provided load-balancing component.	Configure Services
Platform Access Address	The external address used to access platform services such as the web console and platform APIs. It can be the same as the Cluster Endpoint or a separate address for external access scenarios.	Install
Cluster Endpoint	The address used by cluster components and administrators to reach the target cluster control plane endpoint. It is the primary control-plane access entry during installation and later operations.	Install
Self-built VIP	The built-in virtual IP option used when an external load balancer is not provided for the Cluster Endpoint.	Install

Disaster Recovery and Upgrade Terms

Term	Definition	Related doc
Global Cluster Disaster Recovery	The disaster recovery model for the `global` cluster in which a primary global cluster and a standby global cluster are kept ready for failover through etcd data synchronization and coordinated operational procedures.	Global Cluster Disaster Recovery
Cluster Version Operator (CVO)	The operator-based upgrade workflow and controller used to coordinate target version, preflight status, and execution progress for `global` and workload cluster upgrades.	Upgrade Overview

Usage Notes

- Use this page as the canonical source for ACP-wide terms that appear across multiple documentation sections.
- Keep page-local `## Terminology` sections for workflow-specific or subsystem-specific terms that are not reused broadly across the product.
- The **Term** column uses a normalized display style for readability.
- Keep official feature names, protocol names, UI labels, and API-facing names in their official capitalization, such as `OperatorHub`, `Platform Access Address`, `ClusterIP`, `Self-built VIP`, and `OpenID Connect (OIDC)`.
- Favor product concepts, platform models, and high-value cross-section entry terms over generic engineering vocabulary.
- Expand an acronym on first mention when needed, then use the acronym consistently.
- When a term is already defined by Kubernetes or OpenShift, use the upstream meaning first and add ACP-specific context only when needed.

Release Notes

TOC

4.3.0

Issued: 2026-04-16

Features and Enhancements

Support for Kubernetes 1.34

ACP 4.3 adds support for **Kubernetes 1.34** for platform-managed cluster scenarios.

For upgrades to ACP 4.3, the workload-cluster compatible versions are 1.34, 1.33, 1.32, and 1.31. This compatible-version requirement determines whether the `global` cluster can be upgraded and is separate from the third-party cluster management range.

For more information, see Kubernetes Support Matrix.

CVO-Based Cluster Upgrade Workflow

ACP 4.3 introduces a Cluster Version Operator (CVO)-based upgrade workflow for both `global` and workload clusters.

Key capabilities include:

- Preparing upgrade artifacts and the upgrade controller with `bash upgrade.sh`
- Running preflight checks before execution
- Requesting upgrades from the Web Console or by updating `ClusterVersionShadow.spec.desiredUpdate`
- Inspecting conditions, preflight results, stages, and history from `cvsh.status`

ACP CLI also introduces upgrade-oriented administrator commands such as `ac adm upgrade`, `ac adm upgrade status`, `--to-latest`, `--to`, and `--allow-explicit-upgrade` for requesting and troubleshooting workload cluster upgrades from the current context.

For operational guidance, see Upgrade.

Standalone Cluster Plugin Upgrade

ACP 4.3 adds standalone upgrade support for cluster plugins that use the `Aligned` or `Agnostic` life cycle.

The **Cluster Plugins** page now shows the plugin life cycle, and eligible plugins can be upgraded independently from the list page or details page. `Core` plugins continue to follow cluster upgrades.

MicroOS-Based Global Clusters on Huawei DCS

ACP 4.3 allows administrators to create the `global` cluster on Huawei DCS with MicroOS-based immutable infrastructure. This extends the immutable operating model from workload clusters to platform installation scenarios on DCS.

For more information, see About Immutable Infrastructure.

Huawei Cloud Stack Support in Immutable Infrastructure

ACP 4.3 adds Immutable Infrastructure support for Huawei Cloud Stack (HCS). The HCS provider documentation now covers provider overview, installation, cluster creation, node management, cluster upgrades, and provider APIs in the Immutable Infrastructure documentation set.

For more information, see About Immutable Infrastructure.

VMware vSphere Support in the 4.3 Cycle

ACP 4.3 begins introducing Immutable Infrastructure support for VMware vSphere. The provider work is now tracked in the Immutable Infrastructure documentation set, while the public installation details and finalized plugin naming are still being published.

For more information, see About Immutable Infrastructure.

New Web Console Preview Entry

ACP Core now provides the top-navigation anchor required by the next-generation Web Console experience. When Alauda Container Platform Web Console Base is installed on the `global` cluster, users in the **Container Platform** and **Administrator** views can open the new console through a **Preview Next-Gen Console** entry in a separate browser tab.

The experience is designed for gradual migration and works with the Web Console Base plugin on the global cluster and the Web Console Collector plugin on workload clusters.

Containerd 2.0 Baseline

ACP 4.3 upgrades the platform runtime baseline to containerd 2.0. Review runtime-dependent operational procedures before upgrading environments that rely on customized containerd configuration.

Expanded Third-Party Cluster Management Range

For third-party clusters, ACP 4.3 now accepts Kubernetes versions in the range `>=1.19.0 <1.35.0`.

This management range is separate from the compatible Kubernetes versions used to determine whether the `global` cluster can be upgraded.

Product documentation continues to publish only the Kubernetes versions that have passed product validation for third-party cluster support and the default Extend baseline.

Product validation for the Extend baseline covers the following capability areas:

- Installing and using Operators
- Installing and using Cluster Plugins
- ClickHouse-based logging
- VictoriaMetrics-based monitoring

This does not mean that all specific Operators or Cluster Plugins are covered by product validation.

For specific Operators or Cluster Plugins outside this baseline, refer to the relevant product documentation or contact technical support.

For more information, see Kubernetes Support Matrix and Import Standard Kubernetes Cluster.

Expanded Monitoring Plugin Configuration

ACP 4.3 expands the configuration options for the monitoring plugins, making it easier to adapt monitoring deployments to infra-node placement and different storage layouts.

For ACP Monitoring with VictoriaMetrics, administrators can now:

- Configure plugin-level node selectors and tolerations for workload placement on dedicated infra nodes
- Configure the data storage directory for VictoriaMetrics when `Storage Type` is `LocalVolume`
- Remove the previous three-node limit for VictoriaMetrics deployments

For ACP Monitoring with Prometheus, administrators can now configure plugin-level node selectors and tolerations, so monitoring workloads can be scheduled to dedicated infra nodes through plugin configuration.

> **WARNING**
>
> If you previously used patch resources or override-based customizations to define node selectors or tolerations separately, you need to update the plugin configuration after upgrading to ACP 4.3. After the updated plugin configuration takes effect, you must remove the related patch resources or override settings.

For operational guidance, see Installation and Planning Infra Nodes for Monitoring.

StatefulSet Cross-Cluster Application Disaster Recovery Solution

This release introduces cross-cluster disaster recovery capabilities for stateful applications. Built on an Active-Passive dual-center architecture, it combines **Alauda Build of VolSync**

asynchronous data sync and **GitOps** configuration distribution to achieve minute-level RTO failover.

Key Highlights:

- The primary cluster handles all `read/write` traffic; the standby cluster maintains a warm data replica via periodic rsync snapshots (RPO > 0).
- Supports three operational scenarios: planned migration, emergency failover, and failback.
- The standby cluster runs with `replicas=0` by default; storage and compute resources remain in cold standby, handling no business traffic.
- Suitable for workloads without strict zero-data-loss requirements (RPO = 0). For financial or transactional core applications, use native database replication instead.

For more details, see: Cross-Cluster Application Disaster Recovery for Stateful Applications ↗

Alauda Container Platform Registry - Image Management Enhancements

This release introduces `ac images` and `ac adm prune images` commands, enabling full lifecycle management of Registry images from the command line.

- `ac get images`: List images in the registry. Results are scoped to namespaces the current user has permissions for, with support for namespace filtering and multiple output formats (`table`, `json`, `yaml`, `wide`).
- `ac delete images`: Delete one or more image tags by registry path. Built-in namespace permission checks; runs in dry-run mode by default to preview the impact, and requires `--confirm` to perform actual deletion.
- `ac adm prune images`: Admin command to prune image manifests that are not referenced by any cluster Pod. Flexible pruning policies include retention duration, retention count, allowlist, and `--all` scope. Optionally triggers Registry GC after pruning. Also supports scheduled cleanup via CronJob.

For more details, see: Cluster Image Registry Cleanup: Administrator Guide for Manual and Scheduled Tasks ↗

Alauda Container Platform Project Application Essential (Alpha)

This release introduces the **Alauda Container Platform Project Application Essential** plugin, built on the brand-new **Next-Gen Console** frontend framework. Deployed on the `global` cluster, it delivers cross-cluster application orchestration and full lifecycle management from a project-centric perspective, fully respecting user permissions.

Key Highlights:

- **Cross-cluster orchestration**: Unified deployment of applications to multiple member clusters within a single project.
- **Full lifecycle management**: Supports `create`, `update`, `scale`, `rollback`, `delete`, with real-time sync of application status across clusters.
- **Project isolation**: All operations scoped to the project boundary, ensuring natural isolation between projects.
- **Permission-aware**: Strictly enforces RBAC permissions, displaying only resources the user is authorized to access.

Underlay and Egress Gateway Enhancements

ACP 4.3 expands core CNI networking capabilities around underlay access and egress gateway operations.

Key enhancements include:

- Better high-availability and fast-switching design for egress gateway workloads, reducing service impact during node maintenance or failover.
- Resource protection guidance and platform support for egress gateway Pods, helping reduce the risk of node resource contention under traffic spikes or replica growth.
- Support for configuring taints for egress gateway workloads, allowing better placement isolation on dedicated nodes.
- Support for managing VLAN sub-interfaces for underlay NICs.
- Added YAML editing support for subnet resources.
- Added support for node selector settings for centralized gateways.
- Added subnet CRD support for centralized gateway scenarios.

These enhancements make ACP more adaptable to complex enterprise networking environments and simplify migration from earlier exposure models to underlay-based designs.

Gateway API Enhancements

ACP 4.3 strengthens **Gateway API** as a key Layer 7 load balancing capability in the platform.

Key enhancements include:

- Support for host-network-based gateway deployment scenarios.
- Support for exposing services through `metalLB + Envoy Gateway proxy + underlay`, so business traffic can avoid the management network.
- Support for custom VIP addresses for Gateway API, helping keep service exposure addresses stable across rebuilds or lifecycle changes.

Stateful Application Disaster Recovery with PVC-Based Protection

ACP 4.3 introduces stronger disaster recovery capabilities for stateful workloads, including **PVC-based disaster recovery** and support for **VolSync-based backup and restore workflows** for storage-backed applications such as MinIO.

This enhancement improves cross-cluster recovery readiness for stateful applications and provides a more practical protection path for storage-heavy production environments.

Ceph Storage Management Enhancements

ACP 4.3 improves storage operations and Ceph-based workload support.

Key enhancements include:

- Added support for placing disks into different Ceph pools from the UI.
- Improved operational support for Ceph disk replacement scenarios.

These changes improve day-2 storage operations and make Ceph-based environments easier to manage in production.

Virtualization Platform Enhancements

ACP 4.3 delivers several important virtualization-related improvements.

Key enhancements include:

- Improved VM creation and display workflows.
- Added support for Astra Linux in virtualization-related scenarios.
- Added support for multi-NIC and NIC hot-plug capabilities for virtual machines.

These enhancements improve virtualization usability and expand guest workload compatibility in enterprise environments.

Deprecated and Removed Features

Decommissioning of Operation Statistics

The metering and billing plugins are now generally available and fully cover the capabilities previously provided by the Operations Statistics feature. Therefore, the top-level **Operations Statistics** entry under **Platform Management** will be removed.

- For newly deployed platforms, Operations Statistics components are no longer installed. If you need metering or billing capabilities, use the **Cost Management** plugin.
- For upgraded platforms, metering collection by Operations Statistics stops after the upgrade, while historical data remains available. If you need data cleanup or migration, submit a support request.

Fixed Issues

- Fixed an issue where the olm-registry pod would continuously restart, preventing the OperatorHub from functioning properly. This was caused by the `seccompProfile: RuntimeDefault` security configuration added during CIS compliance hardening, which blocked the `clone` syscall required by CGO operations. The seccomp profile has been adjusted to allow necessary syscalls while maintaining security compliance.Fixed in ACP 4.3.0.
- Fixed a performance issue where the permission validation during native application creation became extremely slow (10+ seconds) when the cluster had 60+ operators installed.Fixed in ACP 4.3.0.
- When using the etcd backup feature provided by Alauda Container Platform Cluster Enhancer, if users configure to back up etcd to S3 storage, the plugin fails to retrieve the Secret object referenced in secretRef. The root cause was that the plugin lacked the

necessary RBAC permissions to read Secrets, resulting in S3 authentication information retrieval failure. This issue has been fixed in ACP 4.3.0.

- When using Alauda Container Platform Monitoring for VictoriaMetrics with multiple clusters sharing the same Storage, the alert rule cpaas-certificates-rule has two issues: alert notifications do not differentiate between clusters when triggered, and the rule monitors customer secrets instead of only platform certificates.
- Fix metis component storage limit configuration is too small and causes metis container to restart after exceeding the limit
- Fixed the issue where pushing container images with a large number of data layers (over 100) to the built-in image repository failed.
- Fixed an issue where imagePullSecret was not automatically injected when workloads used custom ServiceAccounts, resulting in image pull failures.
- Fixed an issue where Pods could not pull images when image-registry imagePullSecret auto-rotation used “create new Secret + delete old Secret”, and legacy Pods still referenced the old Secret but started only after it had expired.
- Fixed an exception triggered under specific scenarios during namespace creation. When entering the namespace creation page, if the page request response is slow, the default selected cluster information may not be available upon initial page access, triggering errors in other page interfaces and causing the project quotas on the page to fail to display correctly.
- The text in the real-time logging component has been adjusted: Logging has ended => End of logs
- Fixed an issue where line breaks were inconsistent between Windows and Mac when editing configmaps.

Known Issues

- When using violet push to upload a chart package, the push operation may complete successfully, but the package does not appear in the public-charts repository.
  Workaround: Push the chart package again.
- When using violet push to upload a chart package, the push operation may complete successfully, but the package does not appear in the public-charts repository.
  Workaround: Push the chart package again.

- Application creation failure triggered by the defaultMode field in YAML.
  Affected Path: Alauda Container Platform → Application Management → Application List → Create from YAML. Submitting YAML containing the defaultMode field (typically used for ConfigMap/Secret volume mount permissions) triggers validation errors and causes deployment failure.
  Workaround: Manually remove all defaultMode declarations before application creation.
- When pre-delete post-delete hook is set in helm chart.
  When the delete template application is executed and the chart is uninstalled, the hook execution fails for some reasons, thus the application cannot be deleted. It is necessary to investigate the cause and give priority to solving the problem of hook execution failure.