Kubernetes Cluster Architecture Specification

Apex Enterprise Platform — Production Cluster `apex-prod-mer1`

Meridian Cloud • Region mer-east-1 • Kubernetes v1.29

Document ID	APX-K8S-ARCH-2026	**Version**	1.3.0
Classification	Confidential — Internal Use	**Status**	Approved Baseline
Owner	Apex Platform Engineering	**Approved By**	Dana Whitfield, VP Infrastructure
Effective Date	August 2, 2026	**Next Review**	February 2, 2027

Document Purpose

This specification defines the reference architecture for `apex-prod-mer1`, the production Kubernetes cluster underlying the Apex Enterprise Platform. It covers node topology, the container networking interface (CNI), ingress and traffic management, stateful storage, and the security policies enforced across all tenant workloads. It is the authoritative baseline for platform engineers deploying or auditing cluster changes; deviations require an approved architecture decision record.

1 Introduction

1.1 Purpose

The Apex Enterprise Platform runs its production workloads — the Vertex core application, the Synapse data and analytics platform, and the Nexa customer portal — on a single shared Kubernetes cluster operated by Apex Platform Engineering. This document specifies the architecture of that cluster: how nodes are pooled and scaled, how pod networking and ingress are implemented, how stateful services persist data safely, and how the security controls that bound every namespace are enforced. It is written for the engineers who deploy into the cluster and the engineers who operate it, and it is the reference against which cluster changes are reviewed.

1.2 Scope

This specification covers the production cluster `apex-prod-mer1` in Meridian Cloud's `mer-east-1` region. It is explicitly scoped to platform-level concerns rather than any single application's internal design.

- **In scope:** node pool composition and autoscaling, the CNI and IP address plan, ingress and TLS termination, storage classes and backup policy, and the security baseline (RBAC, admission control, network policy, secrets).
- **Out of scope:** application-level deployment manifests, CI/CD pipeline internals, and the staging and development clusters, which follow a reduced variant of this baseline documented separately.

1.3 Audience

Primary readers are platform engineers on the Apex infrastructure team, service owners on Vertex, Synapse, and Nexa who need to understand the guarantees the cluster provides, and auditors reviewing the security posture of production workloads.

1.4 Related Documents

Document	Reference	Owner
Meridian Cloud Landing Zone Design	APX-CLD-LZ-014	Platform Engineering
Incident Response & On-Call Runbook	APX-OPS-RUN-009	Site Reliability
Data Classification & Retention Policy	APX-SEC-POL-002	Security Engineering
Vertex, Synapse & Nexa Service Catalogue	APX-SVC-CAT-021	Product Engineering

2 Cluster Topology & Node Architecture

2.1 Control plane

`apex-prod-mer1` runs on Meridian Cloud's managed Kubernetes control plane, version 1.29. The control plane is replicated across three availability zones in `mer-east-1`, with a five-member etcd quorum maintained by the provider and snapshotted every fifteen minutes to object storage with a 30-day retention window. The Kubernetes API server is reachable through two endpoints: a private endpoint peered into the Apex corporate network for interactive `kubectl` access, and a public endpoint restricted by IP allowlist to the CI/CD deployment runners only. Neither endpoint accepts anonymous or basic-auth requests; all access is mediated by OIDC tokens issued through Synapse Auth.

2.2 Node pools

Worker capacity is split into three pools so that noisy, bursty, or privileged workloads cannot starve one another. Each pool is a separate autoscaling group with its own instance class, taints, and scaling bounds.

Pool	Purpose	Instance Class	Nodes	Taints / Labels
`system`	Add-ons: CoreDNS, Cilium operator, metrics-server, cert-manager	4 vCPU / 16 GiB	3 (fixed)	`CriticalAddonsOnly=true:NoSchedule`
`general-compute`	Vertex API, Nexa portal backend, Synapse ingestion workers	8 vCPU / 32 GiB	6–24	`node-role=general`
`stateful-data`	Synapse Postgres, Redis, Kafka	8 vCPU / 64 GiB	3–9	`workload=stateful:NoSchedule`

The `system` pool is fixed at three nodes, one per availability zone, and never scales down — add-ons that the rest of the cluster depends on must not be evicted by capacity pressure elsewhere. The `general-compute` and `stateful-data` pools scale independently via Cluster Autoscaler, bounded by the ranges above.

2.3 Cluster topology diagram

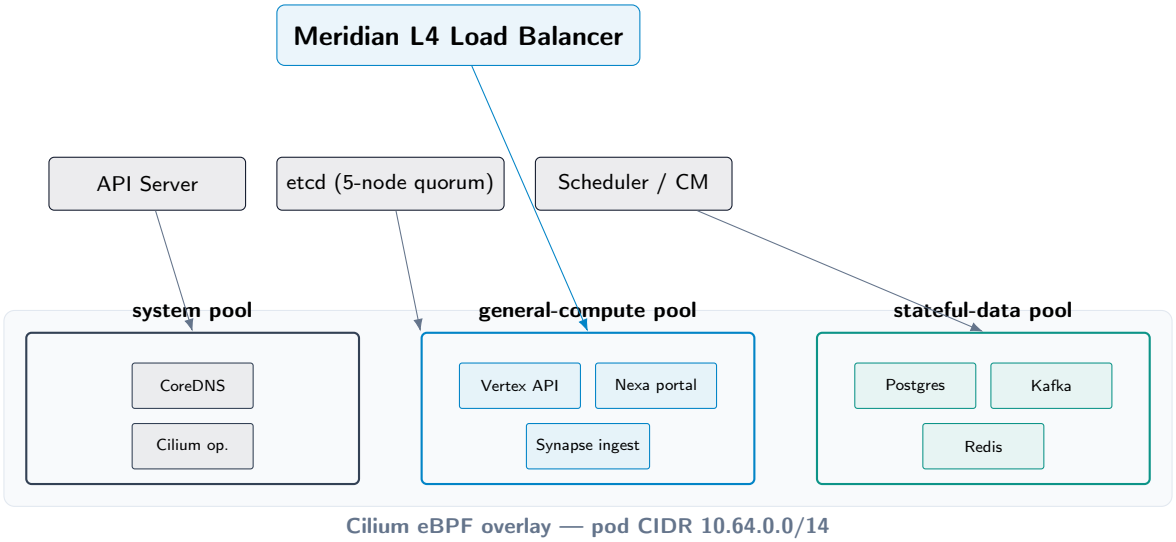

Figure 1: Control plane, node pools, and the shared CNI overlay for `apex-prod-mer1`.

2.4 Autoscaling policy

AD-04: Pool Segregation Over Bin Packing

Earlier revisions of this cluster ran all workloads in a single general-purpose pool. Stateful services were repeatedly evicted during scale-down events triggered by unrelated traffic spikes on the Nexa portal. Segregating pools by workload class costs some bin-packing efficiency but removes that failure mode entirely, and it lets `stateful-data` scale on a slower, more conservative policy than

general-compute.

Horizontal Pod Autoscalers govern individual Deployments; Cluster Autoscaler reconciles pool size against unschedulable pods every 15 seconds and will not scale a pool below its floor even when utilization is low, to preserve quorum for pool-local daemons.

3 Networking & CNI

3.1 CNI selection

apex-prod-mer1 uses Cilium 1.15 in eBPF mode with the kube-proxy replacement enabled. Cilium was chosen over the Meridian-native CNI for three reasons: native `kube-proxy` replacement removes a whole class of `iptables` scaling problems once the cluster passed roughly 400 services; Hubble gives per-flow network observability without a sidecar; and `CiliumNetworkPolicy` supports L7-aware rules (HTTP method and path matching) that the platform's default-deny model depends on. Every node runs the Cilium agent as a DaemonSet across all three pools, including `system`.

3.2 IP address management

Pod and service address ranges are allocated cluster-scope, not per-node, to keep the CIDR plan predictable as pools scale.

Range	CIDR	Notes
Node subnet	`10.60.0.0/20`	Three subnets, one per AZ, in the Meridian production VPC
Pod CIDR	`10.64.0.0/14`	Cilium cluster-scope IPAM; `/24` allocated per node on demand
Service CIDR	`10.96.0.0/12`	`ClusterIP` allocation range
Hubble relay	`10.60.15.0/28`	Fixed range for the observability control plane

3.3 Network policy model

The cluster default is deny-all: every namespace is created with a baseline `CiliumNetworkPolicy` that drops ingress and egress not explicitly allowed. Application teams add scoped allow rules through their own manifests rather than requesting exceptions from platform engineering, which keeps ownership of the policy with the team that understands the traffic. Three properties are enforced regardless of what a team writes:

- **DNS is always permitted** to the in-cluster CoreDNS service, so a misconfigured egress rule cannot silently break name resolution.
- **Cross-namespace traffic requires an explicit label selector** — there is no cluster-wide allow rule that lets one team's pods reach another's by IP alone.
- **Egress to the public internet is denied by default** outside the `general-compute` pool; workloads that need it request an egress gateway route, which is logged and rate-limited.

Known Limitation

Hubble flow logs are retained for 72 hours in-cluster before rolling over. Longer-term flow analysis for incident investigation depends on the export pipeline into Synapse, which currently lags live traffic by up to 10 minutes during peak ingestion windows.

4 Ingress & Traffic Management

4.1 Ingress architecture

External traffic reaches the cluster through the Meridian L4 load balancer, which terminates TCP and forwards to an NGINX Ingress Controller deployment running two replicas per availability zone in the `general-compute` pool. NGINX performs virtual-host routing and TLS termination; it does not implement business logic. `cert-manager` issues and rotates certificates automatically, watching `Ingress` and `Certificate` resources across every namespace.

Class	Controller	Exposure	TLS Source	Consumers
`nginx-public`	NGINX v1.10	External	Public CA via ACME	Nexa customer portal
`nginx-internal`	NGINX v1.10	VPC-only	Apex internal CA	Vertex admin console, Synapse dashboards
`nginx-partner`	NGINX v1.10	Allowlisted	Public CA via ACME	Partner API integrations

4.2 TLS & certificate lifecycle

Certificates are requested with a 90-day validity and renewed automatically at 30 days remaining. The internal CA used for `nginx-internal` is rooted in Meridian KMS and rotated annually; its intermediate is distributed to internal clients through the platform's trust-bundle ConfigMap, refreshed on every node at boot. mTLS between the ingress controller and upstream pods is required for any service handling account or payment data.

4.3 Rate limiting & resilience

Each `Ingress` resource carries a per-route rate limit and connection cap set by the owning team, with platform-wide defaults applied when a route omits them (100 requests/second, burst 50). NGINX is configured with a 30-second upstream timeout and three retries on connection-reset errors only, never on timeouts, to avoid amplifying load into an already-degraded backend.

Incident History: Noisy-Neighbor Saturation

In March 2026, an unrate-limited reporting endpoint on Synapse consumed the shared ingress connection pool during a customer's bulk export, degrading Vertex checkout latency for eleven minutes. Per-route rate limits became mandatory for every new `Ingress` manifest after this incident, enforced by an admission policy described in Section 6.

5 Stateful Storage

5.1 CSI driver & storage classes

Persistent volumes are provisioned through the Meridian Block Store CSI driver. Three storage classes cover the platform's workloads; none are the cluster default, so every `PersistentVolumeClaim` must name one explicitly.

StorageClass	Media	Reclaim	Expansion	Snapshot Schedule
`fast-ssd`	NVMe SSD	Retain	Online	Every 6h, 14-day retention
`standard-ssd`	SSD	Delete	Online	Daily, 7-day retention
`archive`	Object-backed	Retain	Offline	Weekly, 180-day retention

`fast-ssd` backs the Synapse Postgres primaries and the Kafka log volumes in the `stateful-data` pool; `standard-ssd` covers Redis and general application caches; `archive` holds compacted Synapse analytics exports that are read infrequently.

5.2 Backup & disaster recovery

The Apex Backup Operator runs as a controller in the `system` pool, orchestrating CSI snapshots on the schedules above and replicating snapshot metadata — not the underlying blocks — to `mer-west-1` every hour. A full cross-region restore of the `stateful-data` pool is rehearsed quarterly against a scratch cluster; the two most recent rehearsals both completed within the recovery targets below.

Service	RPO	RTO
Synapse Postgres (primary ledger data)	15 minutes	45 minutes
Redis (cache, non-authoritative)	Best effort	10 minutes
Kafka (event log)	5 minutes	30 minutes

5.3 Stateful workload placement

Every StatefulSet that mounts a `fast-ssd` or `standard-ssd` volume carries the `workload=stateful` toleration for the `stateful-data` pool's taint (Section 2.2) and a pod anti-affinity rule spreading replicas across availability zones. This keeps a single AZ outage from taking down both the primary and its standby replica in the same failure domain.

6 Security Policies

6.1 Identity & RBAC

Human access is granted through OIDC-federated `Role` and `ClusterRoleBinding` objects mapped from Synapse Auth groups; there are no static `kubeconfig` credentials issued to individuals. Service-to-service access uses per-workload Kubernetes service accounts bound to the narrowest role that satisfies the workload, never the namespace default. Namespaces are grouped into three enforcement tiers, applied as Pod Security Admission labels.

Tier	Pod Security Standard	Default Network Policy	Applies To
`restricted`	Restricted	Deny-all baseline	Vertex, Nexa application namespaces
`baseline`	Baseline	Deny-all baseline	Synapse batch and analytics jobs
`privileged`	Privileged, per-workload exception	Deny-all baseline	`system` pool add-ons only

Every `privileged`-tier exception is reviewed by Security Engineering and time-boxed with an expiry label; an admission policy blocks any exception past its expiry from being reconciled.

6.2 Admission control & supply chain

Kyverno enforces cluster policy at admission, including the mandatory per-route rate limits referenced in Section 4.3, a ban on the `latest` image tag, and a requirement that every image be signed with `cosign` against the Apex build key before it can be scheduled. Trivy scans images on push to the internal registry and again nightly against everything currently running; a Critical finding blocks promotion to `apex-prod-mer1` until remediated or explicitly waived by Security Engineering.

6.3 Secrets management

Application secrets are stored in Meridian KMS and synchronized into the cluster by the External Secrets Operator, which materializes them as native `Secret` objects with a bounded TTL and re-fetches on rotation. No secret is committed to a Git repository or baked into a container image; CI enforces this with a pre-merge scan.

AD-07: Centralized Audit Logging

Kubernetes audit logs, Cilium flow logs, and admission decisions are shipped off-cluster to a write-once Synapse log store within seconds of generation, and retained for 400 days to satisfy the platform's compliance window. Nodes never hold more than 24 hours of local log history, so a compromised node cannot be used to tamper with the audit trail of its own compromise.

7 Appendix

7.1 Glossary

- **CNI** — Container Network Interface; the plugin responsible for pod networking. This cluster uses Cilium.
- **CSI** — Container Storage Interface; the plugin responsible for provisioning persistent volumes. This cluster uses the Meridian Block Store CSI driver.
- **HPA / VPA** — Horizontal / Vertical Pod Autoscaler; controllers that resize a workload's replica count or per-pod resource requests based on observed load.
- **PSA** — Pod Security Admission; the built-in Kubernetes controller that enforces the `restricted`, `baseline`, and `privileged` standards described in Section 6.1.
- **RPO / RTO** — Recovery Point / Time Objective; the maximum tolerable data loss and downtime, respectively, for a service during a disaster recovery event.

7.2 Revision history

Version	Date	Author	Summary
1.0.0	Nov 3, 2025	Dana Whitfield	Initial baseline: three-pool topology, Cilium CNI adopted.
1.1.0	Jan 19, 2026	Marcus Ilić	Added per-route rate limiting requirement after the Synapse ingress incident.
1.2.0	Apr 8, 2026	Priya Anand	Introduced mandatory image signing and the `privileged`-tier exception process.
1.3.0	Aug 2, 2026	Dana Whitfield	Extended audit log retention to 400 days; documented cross-region backup rehearsal results.