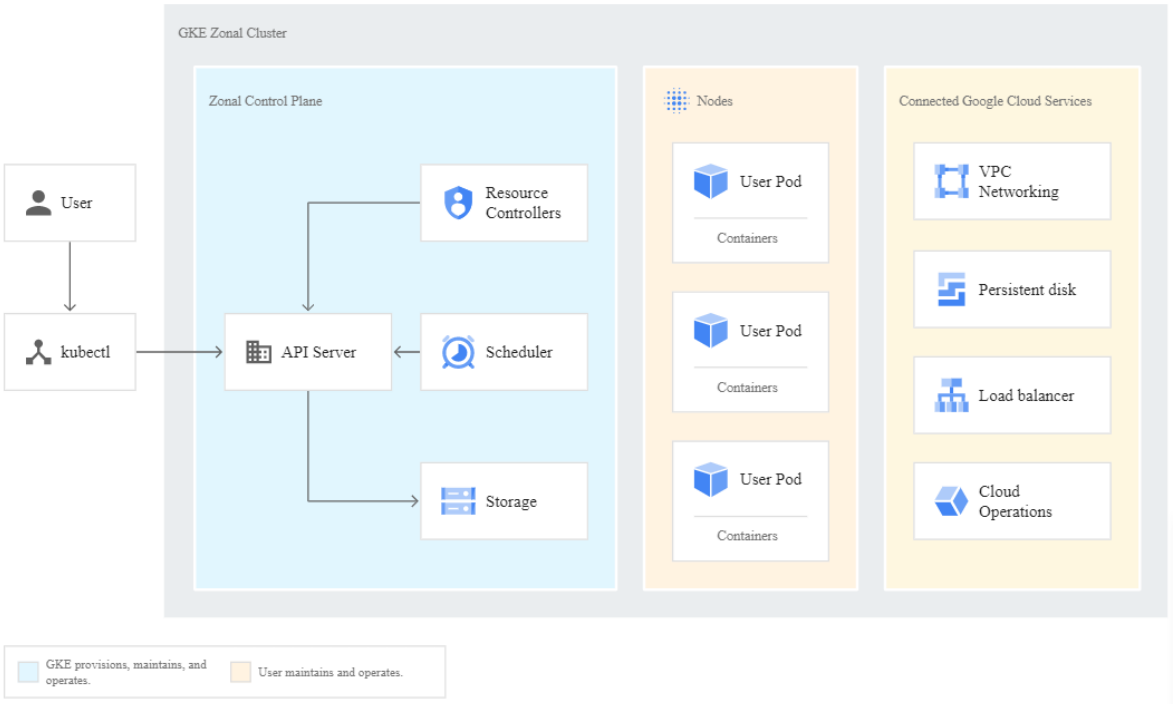
DEPLOYMENT MODEL

A cluster is the foundation of Google Kubernetes Engine (GKE): the Kubernetes objects that represent your containerized applications all run on top of a cluster.

In GKE, a cluster consists of at least one control plane and multiple worker machines called nodes. These control planes and node machines run the Kubernetes cluster orchestration system.

The following diagram provides an overview of the architecture for a zonal cluster in GKE:



Control plane

The control plane runs the control plane processes, including the Kubernetes API server, scheduler, and core resource controllers. The lifecycle of the control plane is managed by GKE when you create or delete a cluster. This includes

upgrades to the Kubernetes version running on the control plane, which GKE performs automatically, or manually at your request if you prefer to upgrade earlier than the automatic schedule.

Control plane and the Kubernetes API

The control plane is the unified endpoint for your cluster. You interact with the cluster through Kubernetes API calls, and the control plane runs the Kubernetes API Server process to handle those requests. You can make Kubernetes API calls directly via HTTP/gRPC, or indirectly, by running commands from the Kubernetes command-line client (kubectl) or by interacting with the UI in the Google Cloud console.

The API server process is the hub for all communication for the cluster. All internal cluster processes (such as the cluster nodes, system and components, application controllers) act as clients of the API server; the API server is the single "source of truth" for the entire cluster.

Control plane and node interaction

The control plane decides what runs on all of the cluster's nodes. The control plane schedules workloads, like containerized applications, and manages the workloads' lifecycle, scaling, and upgrades. The control plane also manages network and storage resources for those workloads.

The control plane and nodes communicate using Kubernetes APIs.

Control plane interactions with Artifact Registry and Container Registry

When you create or update a cluster, container images for the Kubernetes software running on the control plane (and nodes) are pulled from the pkg.dev Artifact Registry or the gcr.io Container Registry. An outage affecting these registries might cause the following types of failures:

- Creating new clusters fails during the outage.
- Upgrading clusters fail during the outage.
- Disruptions to workloads might occur even without user intervention, depending on the specific nature and duration of the outage.

In the event of a regional outage of the pkg.dev Artifact Registry or the gcr.io Container Registry, Google might redirect requests to a zone or region not affected by the outage.

To check the current status of Google Cloud services, go to the Google Cloud status dashboard.

Nodes

A cluster typically has one or more nodes, which are the worker machines that run your containerized applications and other workloads. The individual machines are Compute Engine VM instances that GKE creates on your behalf when you create a cluster.

Each node is managed from the control plane, which receives updates on each node's self-reported status. You can exercise some manual control over node lifecycle, or you can have GKE perform automatic repairs and automatic upgrades on your cluster's nodes.

A node runs the services necessary to support the containers that make up your cluster's workloads. These include the runtime and the Kubernetes node agent (kubelet), which communicates with the control plane and is responsible for starting and running containers scheduled on the node.

In GKE, there are also a number of special containers that run as per-node agents to provide functionality such as log collection and intra-cluster network connectivity.

Node machine type

Each node is of a standard Compute Engine machine type. The default type is e2-medium. You can select a different machine type when you create a cluster.

Node OS images

Each node runs a specialized OS image for running your containers. You can specify which OS image your clusters and node pools use.

Minimum CPU platform

When you create a cluster or node pool, you can specify a baseline minimum CPU platform for its nodes. Choosing a specific CPU platform can be advantageous for advanced or compute-intensive workloads. For more information, refer to Minimum CPU Platform.