

GCP Compute Service

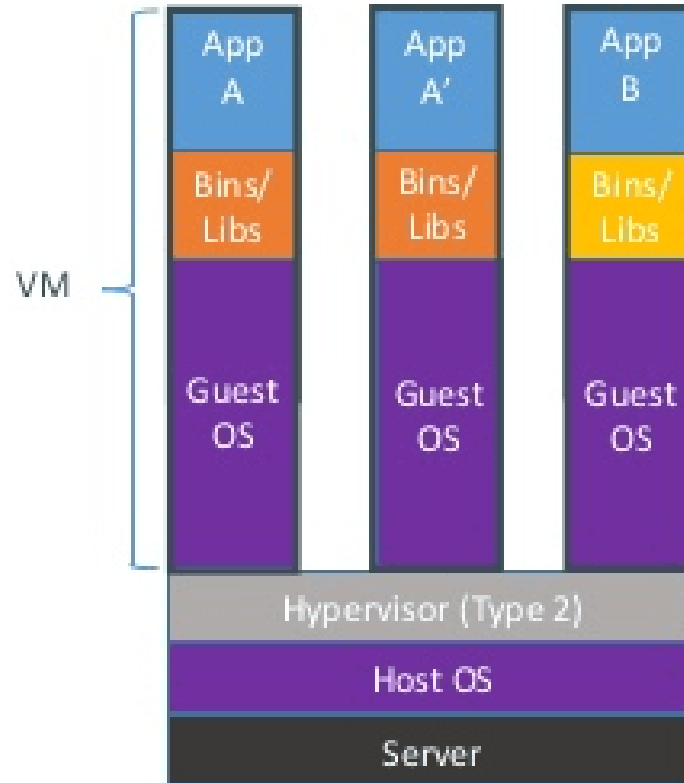
Cloud Container Engine-GKE

Deploy, manage, and scale containerized applications,
powered by **Kubernetes**

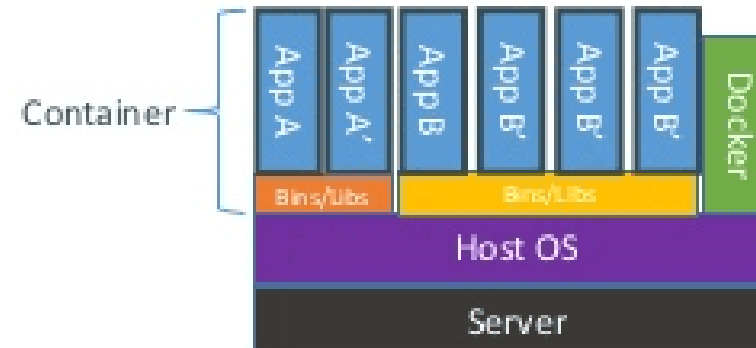


Google Cloud Platform

GKE: Container – Introduction



Containers are isolated, but share OS and, where appropriate, bins/libraries



Container Adv.

- **Write Once and run Anywhere**
- **Re-locatability**
- **Manage Applications and not hardware/machines**
- **Maintain Vendor Independence**
- **Decouple Applications from Dependencies**

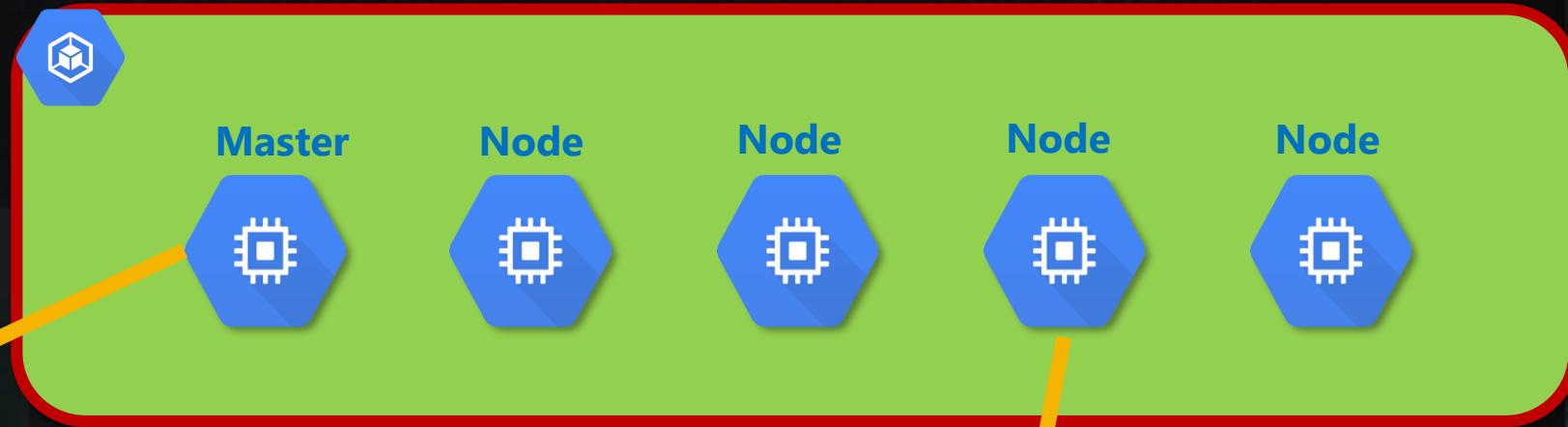


GKE: Introduction

Google Container Engine is a powerful **cluster manager** and **orchestration** system for running your Docker containers. Container Engine schedules your containers into the cluster, keeps them healthy and manages them automatically based on requirements you define (e.g. CPU and Memory)

- Kubernetes is Open Source and based on google systems
- **GKE is Framework for Container Management and automation**
- Fully Managed Service – Kubernetes with SLA
- **Docker formats**
- Auto Scaling
- **Stackdriver logging and Monitoring**
- Cloud VPN Integration
- **Cloud IAM Integration**

GKE: Container Cluster



MASTER



DETAILS

Endpoint/doorway to Cluster

Kubernetes API server

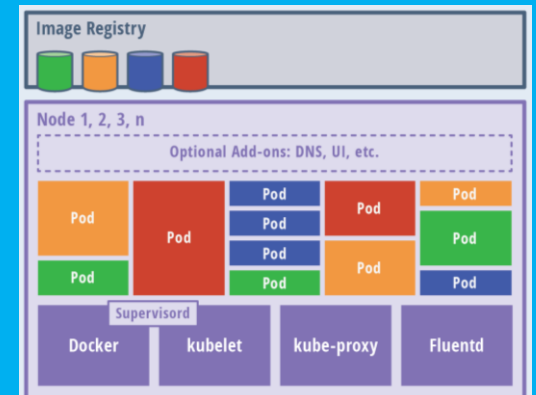
- Services REST requests
- Schedules pod creation/deletion on nodes
- Sync's pod info with service info

Cloud Service Integration

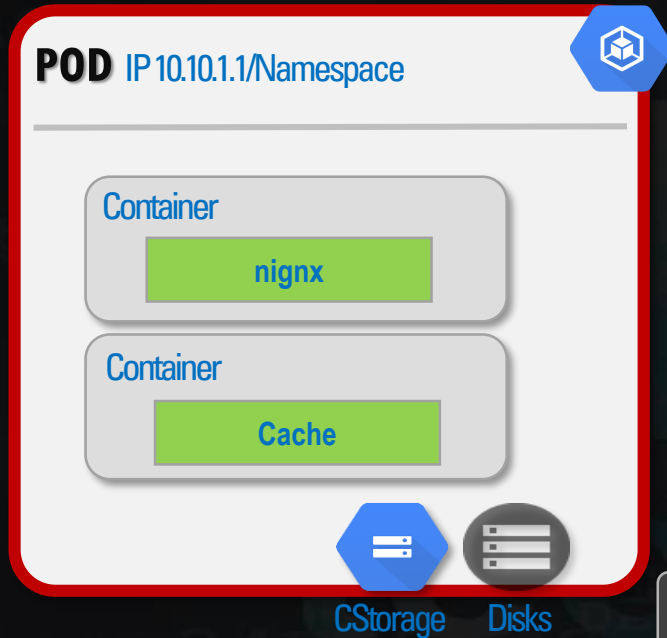
NODE

DETAILS

Docker Runtime
Kubelet Agent
Network Proxy
Runs POD



POD in Summary



POD

DETAILS

A pod is abstraction to represent as application

It holds one or more containers

The Containers in pod share

- A Single IP address
- A Single Name Space -> Localhost

Data can be shared using CS or Disks

POD LIFECYCLE

DETAILS

Pending - Pod is create but some containers are not running

Running — Pod is bound to node and running all containers

Succeeded — All Containers are terminated successfully

Failed - at least one container has terminated failure

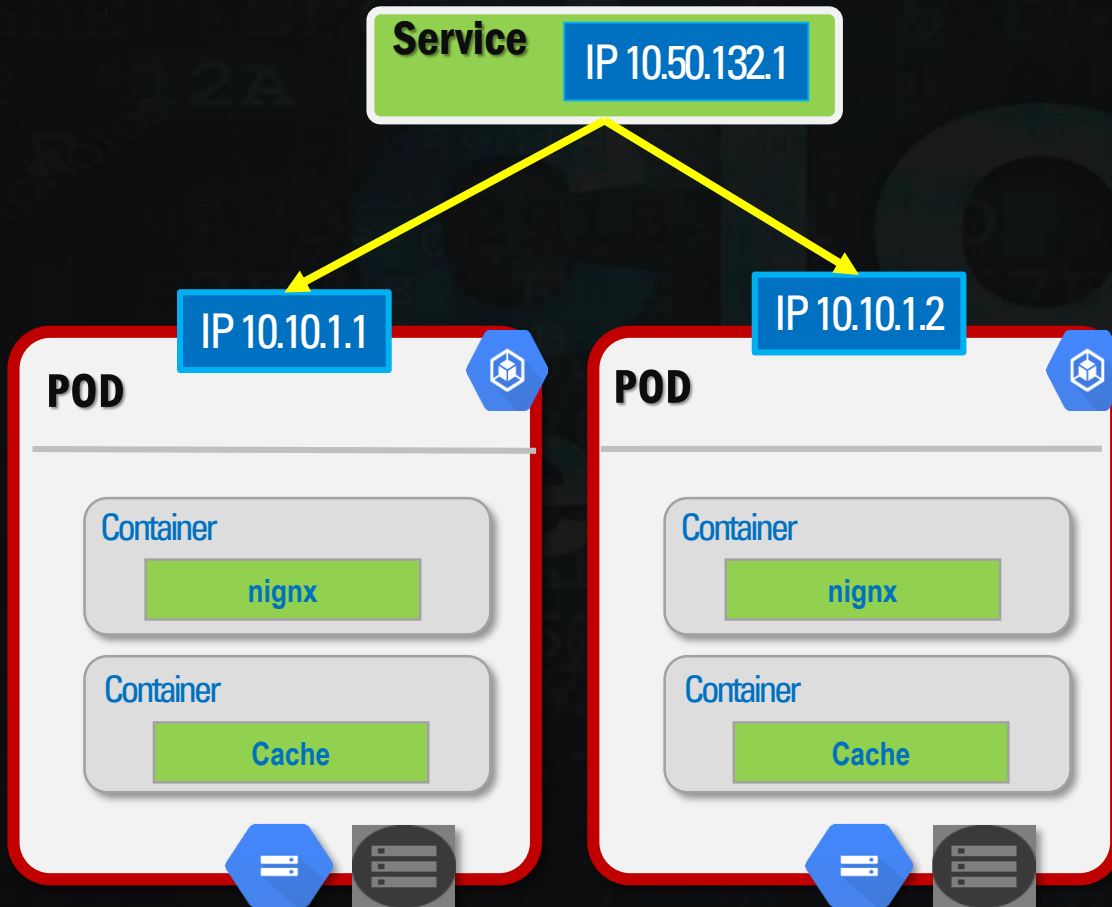
Unknown — Pod can not be determined

GKE Service- Load Balancer end point

GKE SERVICE

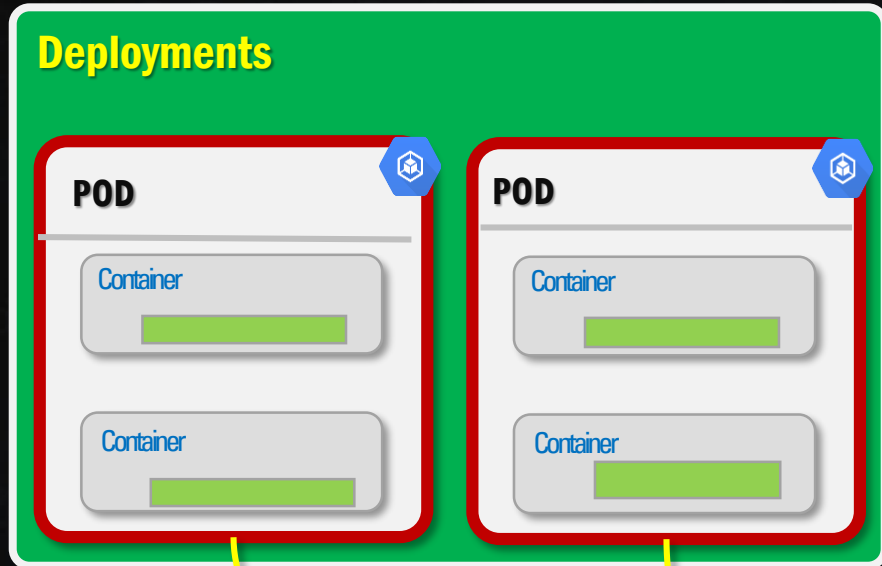
DETAILS

Acts as External Load Balancer and provides persistent internal and external IP for Pods..



GKE Deployment

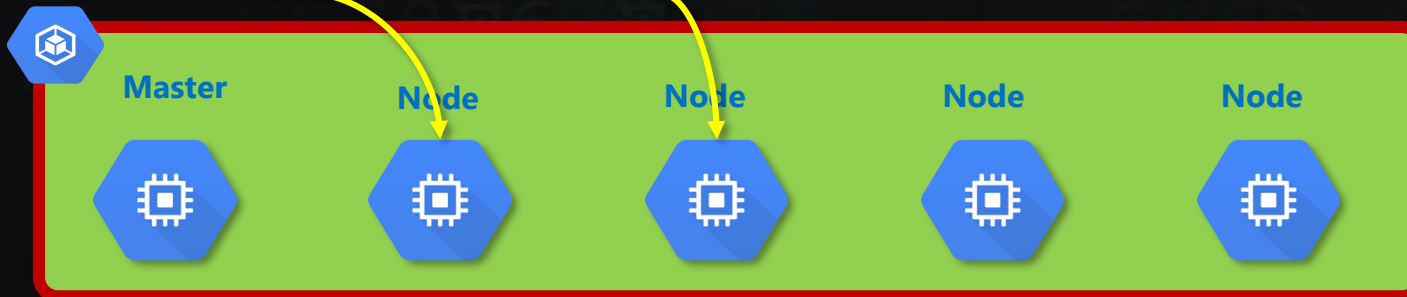
Deployments



DEPLOYMENT GROUP

DETAILS

- Define how HA and AUTOSCALE PODS.
- Create and Manage pod lifecycle
- Used for rolling updates, Changing Versions of application running in container.
- It takes care of service is available to user



GKE in Summary



- ✓ **Fully Managed**
- ✓ **Auto Scale**
- ✓ **Auto Upgrade**
- ✓ **Auto Repair**
- ✓ **Hybrid Networking**
- ✓ **Integrated Logging & Monitoring**
- ✓ **Stateful application support**
- ✓ **Docker Image Support**
- ✓ **Resource Limits**
- ✓ **Private Container Registry**
- ✓ **Identity & Access Management**
- ✓ **Resource-Optimized Deployments**
- ✓ **Reliable and Self-Healing**
- ✓ **Provide configuration as Desired System State**

**Not So
Important
for Exam**

GKE Other Concepts

Google Compute Service - GCP



GKE : Main Concepts

**Container
Cluster**

**Cluster
Federation**

**Cluster
AutoScaling**

**Cluster
Labeling**

Node Pools

**Node Auto
Upgrade**

Node Repair

Node Images

**Load
Balancing**

**IAM & Access
Control**

**IP Agent/
Rotation**

Other conf.

GKE : Node Pools !

NODE POOLS

DETAILS

Pool of actual individual nodes **like Instance Group**

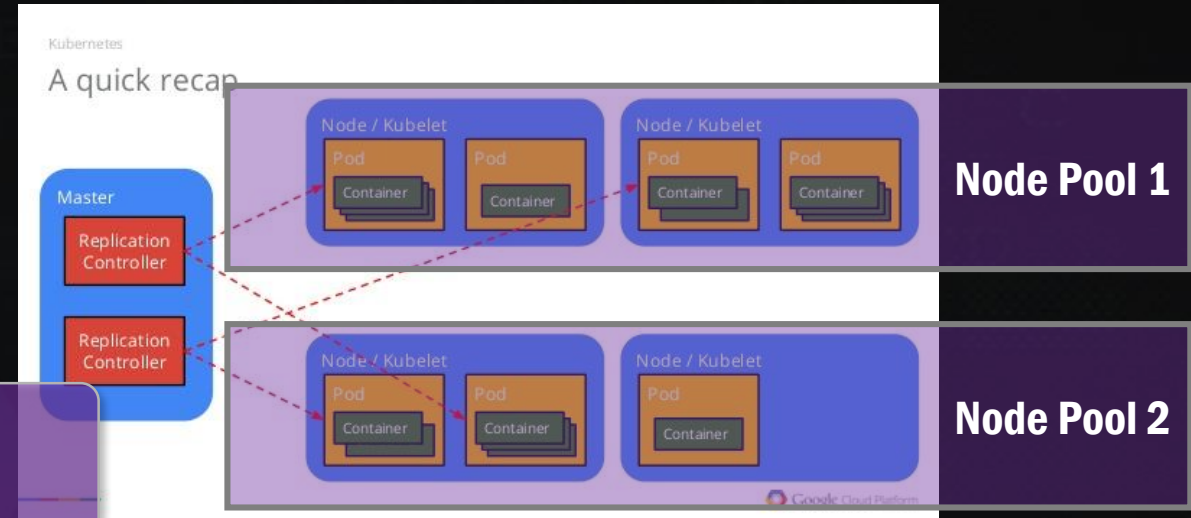
Pool can contain different VM from one another

Can be in different Zones

GKE is node pool aware. (use Labels on vm)

Node Pool and Multizone Container Cluster

- GKE replicate all the pools along all the cluster
- Watch for Quota





GKE : Multi-zone clusters !

Multi-zone clusters are a way to have nodes from **different zones in a **single cluster**, all controlled by the same master**

Multi-zone Container Engine clusters are primarily used as a way to improve availability of your application in the unlikely event of a zone outage

GKE : Cluster AutoScaling

You can enable Container Engine's cluster autoscaler to have Container Engine automatically resize clusters based on the demands of the workloads you want to run.

Works per node pool basis ..

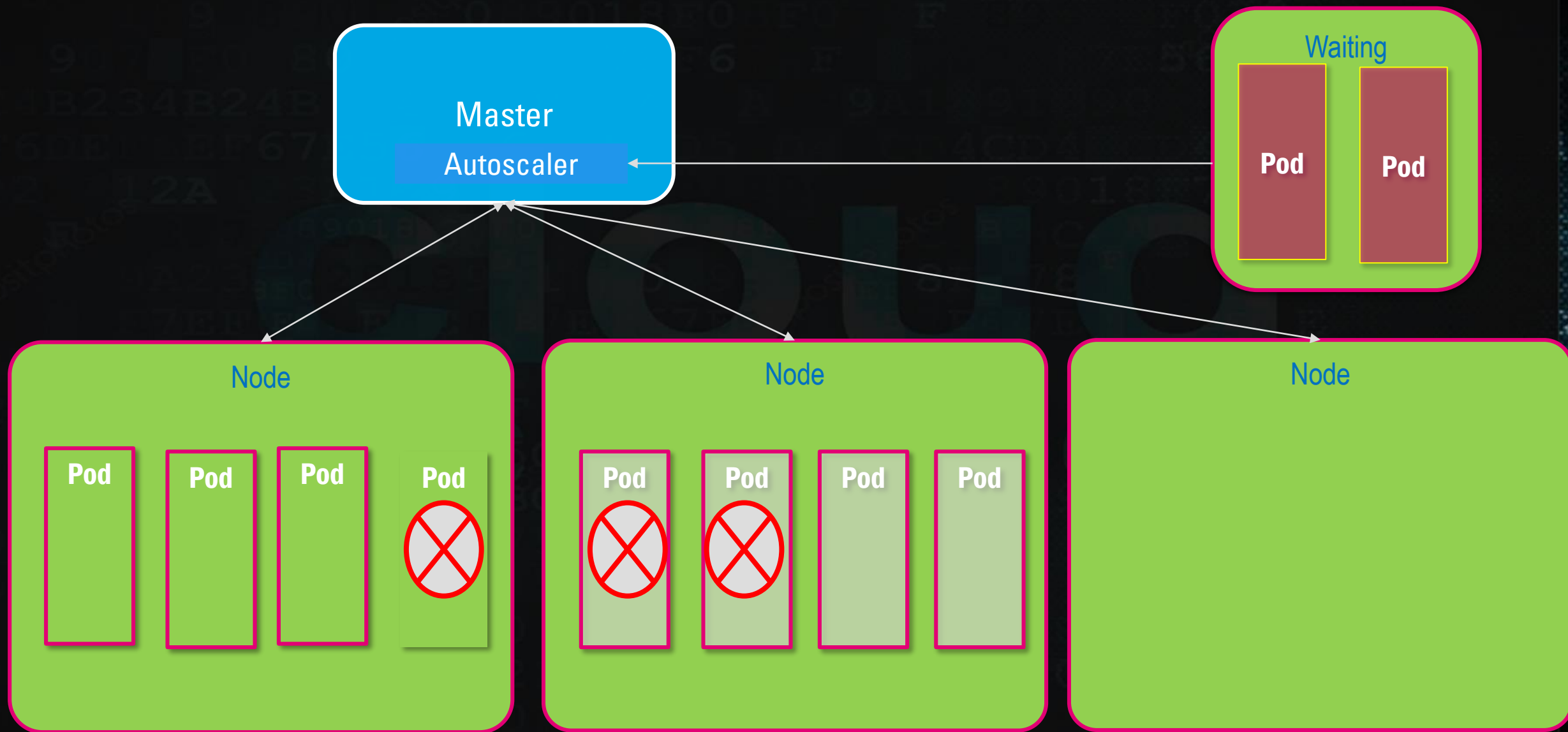
Deployments defines how many instances of pod will be run, Cluster auto scaler looks at resource requirements for current load and add additional instances of pod if demand increases and reduce if demand goes down but maintains min replica which is defined in deployments

Cluster Auto Scaling and Balancing

Google Compute 201



Google Cloud Platform



GKE : Cluster AutoScaling – Operating Criteria

Cluster autoscaler makes the following assumptions when **resizing a node pool**:

All **replicated pods can be restarted** on some other node, possibly causing a **brief disruption**. If your services are not disruption-tolerant, using autoscaling is not recommended.

Users or administrators are not manually managing nodes; it may override any manual node management operations you perform.

All nodes in a single node pool have the **same set of labels**.

Cluster autoscaler considers the relative cost of each instance type in the node pool and attempts to expand the **least expensive possible node pool**.

GKE : Cluster AutoScaling – Limitations

Cluster autoscaler should not yet be used with large clusters (more than 100-150 nodes).



GKE : Internal Load Balancing

The Internal load balancer creates a private (RFC1918) LoadBalancer Ingress IP address in the cluster for receiving traffic on the network within the same compute region.

You create an Internal load balancer by adding a LoadBalancer spec to your cluster's Service configuration file.

If your cluster does not have a Service configuration file, you'll need to create one. Then, you can create the Internal load balancer using the Kubernetes **kubectl command-line interface.**

GKE : Internal Load Balancing

Restrictions for Internal load balancers

- **Kubernetes version 1.7.2 or higher.**
- **ILB can only serve traffic on one type of protocol (TCP or UDP), and it uses the protocol of the first port specified in the Service definition.**
- **ILB not accessible from within the Kubernetes cluster (Nodes and Pods).**
- **ILB are only accessible from within the same network and region.**
- **ILB cannot currently be used with custom subnets;**
- **While the clusterIP remains unchanged, Internal load balancer IP addresses cannot be reserved, so changes made to ports, protocols, or session affinity may cause this IP addresses change.**

Limits

Each Node of the cluster is a backend instance and will count against the backend instance limitation.



GKE : Alpha Clusters !

Alpha Cluster is a **short-lived** cluster that is not covered by the Container Engine SLA and cannot be upgraded, but has all Kubernetes APIs and features enabled.

Alpha Clusters are the way to run stable Kubernetes releases with Alpha features **that may be less stable**.

Alpha clusters cannot be upgraded, and are automatically deleted after 30 days. Make sure any data is **moved off of the cluster before the expiration date**.



GKE : Cluster Labeling

In Container Engine, you apply labels at the Cluster level.

When you label a cluster, the label you have chosen propagates to all of the cluster's individual resources (such as Compute Engine instances and Persistent Disks).

For large clusters – it can take hour to label all resources in cluster.

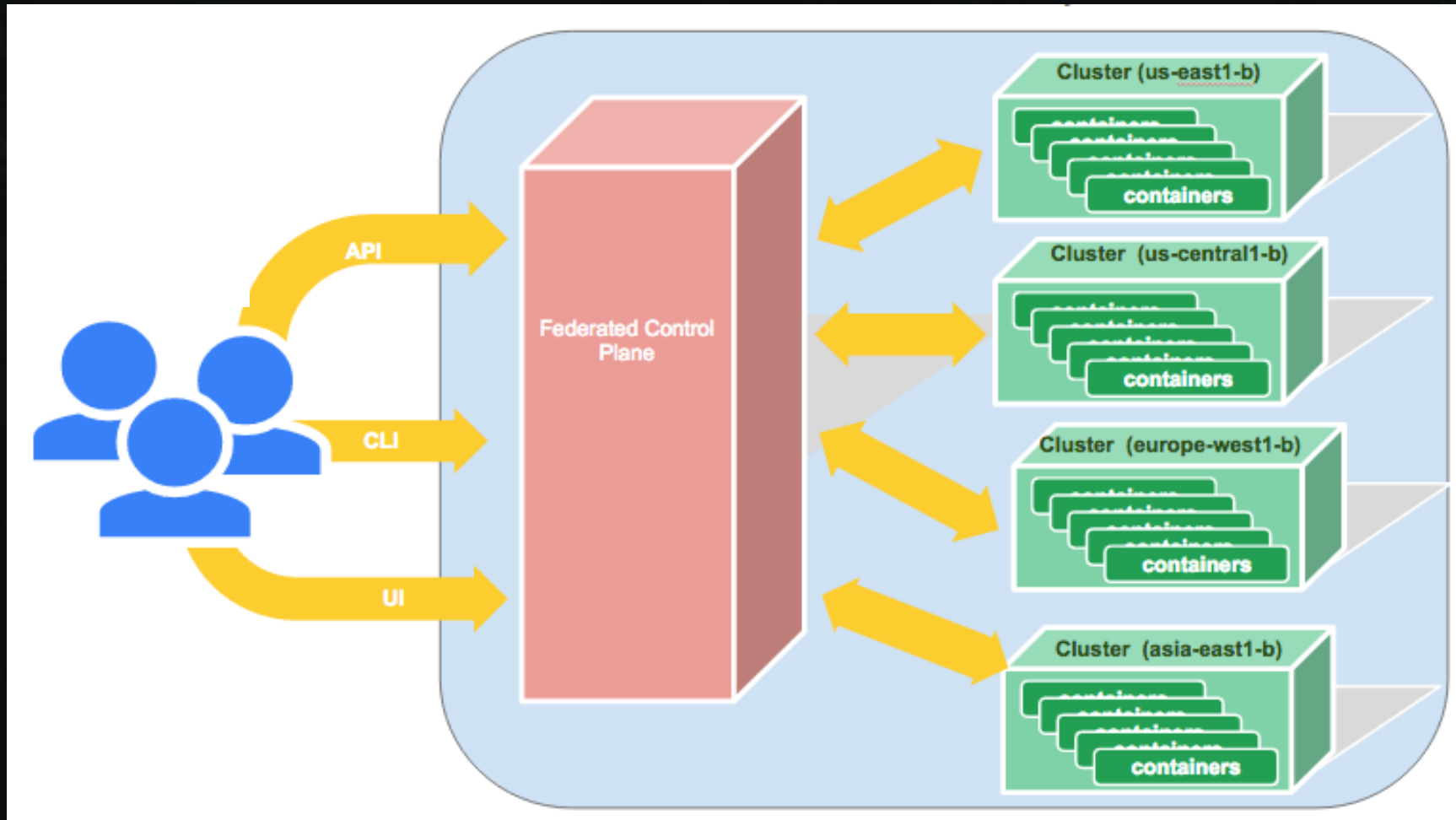


GKE : Kubernetes Cluster Federation

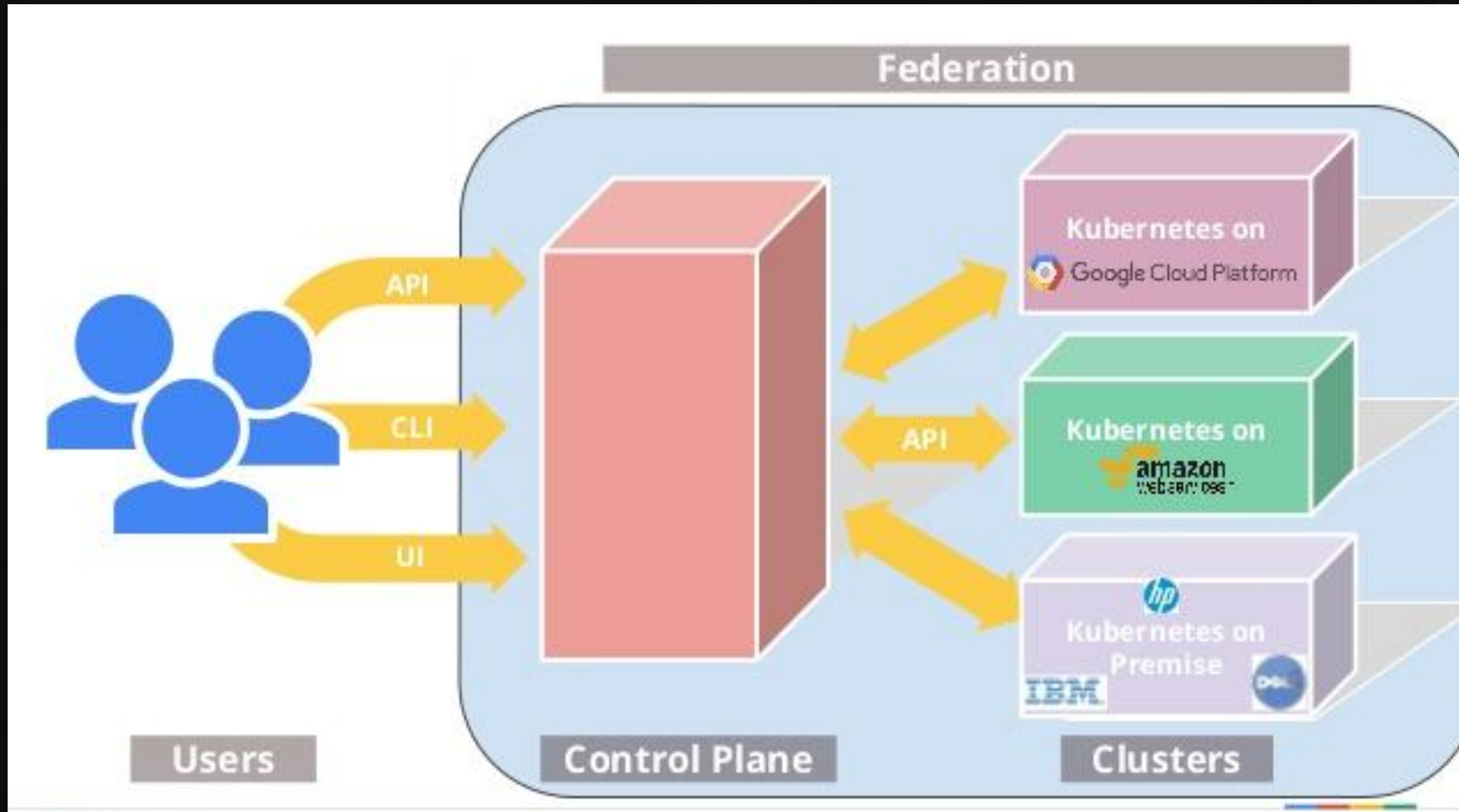
Kubernetes Cluster Federation enables users to federate multiple clusters across different regions, cloud providers, or on-premise installations into a single logical compute federation.

This simplifies the deployment of highly available, geographically distributed services and hybrid cloud scenarios.

GKE : Kubernetes Cluster Federation



GKE : Kubernetes Cluster Federation





GKE : Identity and Access Management (IAM)

Every Google Container Engine API call requires that the account making the request has the necessary IAM permissions. For example,

gcloud container clusters update cluster-1

Will only succeed if the caller has `container.clusters.update` permission on cluster-

kubectl get pods

Requires that the caller has `container.pods.list` permission on the cluster.

gcloud container clusters create cluster-1 --project project-123

requires that the caller have `container.clusters.create` permission on project-123 as well as `iam.serviceAccounts.actAs` on the default service account in project-123. This can be accomplished by assigning both the Container Engine Admin and Service Account Actor roles.



GKE : IAM

IAM Roles e.g.

Role	Description	Permissions
roles/container.admin	Full management of Container Clusters and their Kubernetes API objects.	container.*.*
roles/container.clusterAdmin	Management of Container Clusters.	container.clusters.{create delete get list update} container.operations.*

IAM Permissions for each Container Engine API method

Method	Required Permission(s)
projects.zones.clusters.create	container.clusters.create on the containing Cloud project, and iam.serviceAccounts.actAs on the specified service account.
projects.zones.clusters.delete	container.clusters.delete on the requested cluster.



GKE : IAM

When to use

IAM Allows each project member to use their own credentials tied to their own IAM permissions, instead of the shared cluster certificate.

This provides revocable cluster access, and allows users to be given different levels of access and hence fine control is shared managed.

Authentication Modes

Container Engine users can authenticate to the Kubernetes API on their cluster using Google OAuth2 access tokens.

When you create a new cluster, Container Engine configures kubectl to use Application Default Credentials to authenticate to the cluster.

Authorization is then controlled through Identity and Access Management,



GKE : Role Based Access Control

- **Container Engine's Role-Based Access Control (RBAC) lets you exercise fine-grained control over how users access the Kubernetes API resources running on your container cluster.**
- **You can use Role-Based Access Control to dynamically configure permissions for your cluster's users and define the kinds of resources with which they can interact**
- **You can create Role-Based Access Control permissions that apply to your entire cluster, or to specific namespaces within your cluster.**



GKE : Local SSD Support

- **Local SSDs are physical devices attached to the virtual machine instance. GKE has added the ability for nodes to be created with them, up to the machine limits and the project's quota**
- **Local SSDs provide higher throughput and lower latency than standard disks.**



GKE : Local SSD Support – Consideration

They come with a major caveat: because they are physically attached to the node's host virtual machine instance, any data stored in them only exists on that node.

A pod that writes to a local SSD may lose access to that data if it gets rescheduled away from that node. Also, upgrading a node causes the data to be completely lost. Please be aware of these restrictions as you use local SSDs

GKE : Node auto upgrade Container Engine

Node Auto Upgrades help you keep the nodes in your cluster or node pool up to date with the latest release version of Kubernetes. Auto Upgrades employ the same update mechanism as manual node upgrades.

Node Pools with Auto Upgrades enabled are automatically scheduled for an upgrade when a new Kubernetes version becomes available

Enable Auto upgrade

--enable-autoupgrade

Disable Auto upgrades

--no-enable-autoupgrade



GKE : Node Repair

- **Container Engine's Node Auto-Repair feature helps you keep the nodes in your cluster in a healthy, running state.**
- **When enabled, Container Engine makes periodic checks on the health state of each node in your cluster.**
- **If a node fails consecutive health checks over an extended time period (approximately 10 minutes), Container Engine initiates a repair process for that node**
- **Repair configured per node pool basis.**

GKE : Node Repair

Un Healthy Status means

- A node reports a **NotReady** status on consecutive checks over the given time threshold.
- A node does **not report any status** at all over the given time threshold.
- A node's boot disk is out of disk space for an extended time period.

You can manually check your node's health signals at any time by using the **kubectl get nodes** - command in the gcloud command-line tool.

Node Repair Process

If Container Engine detects that a node requires repair, that node will first be drained, and then Container Engine will re-create the node VM. The drain might not succeed if the node is unresponsive or is too unhealthy to process the drain command.



GKE : Node Images


When you create a Container Engine cluster or node pool, you can choose the operating system image that runs on each node. You can also upgrade an existing cluster to use a different node image type.

Available node images

- Container-Optimized OS from Google - **Linux 4.4 (backed by Google)**
- Ubuntu (beta) - **if your nodes require support for NFS, glusterfs, Sysdig, or Debian packages**
- container-vm open preview (deprecated)

You can select the node image you want to use when you create a new cluster, or you can upgrade an existing cluster to use a different node type.

GKE : Kubernetes UI

kubernetes

kube-system ▼

Workloads

+ DEPLOY APP

↑ UPLOAD YAML

Replication controllers

Name	Labels	Pods	Age	Images		
<div>✓</div> kube-dns-v11	<div>k8s-app: kube-dns</div> <div>kubernetes.io/service: true</div> <div>version: v11</div>	1 / 1	50 minutes	eu.gcr.io/go...-amd64:2.2.1 eu.gcr.io/go...ube2sky:1.14 eu.gcr.io/go...0-13-8c72f8c eu.gcr.io/goo...chealthz:1.0	☰	⋮
<div>✓</div> kubernetes-dashboard-v1.0.1	<div>k8s-app: kub...s-dashboard</div> <div>kubernetes.io/service: true</div> <div>version: v1.0.1</div>	1 / 1	50 minutes	eu.gcr.io/go...md64:v1.0.1	☰	⋮
<div>✓</div> l7-lb-controller-v0.6.0	<div>k8s-app: glbc</div> <div>kubernetes.io/service: true</div> <div>kubernetes.io/name: GLBC</div> <div>version: v0.6.0</div>	1 / 1	50 minutes	eu.gcr.io/go...tbackend:1.0 eu.gcr.io/goo...s/glbc:0.6.0	☰	⋮

Pods

Name	Status	Restarts	Age	Cluster IP	CPU (cores)	Memory (bytes)		
<div>✓</div> fluentd...9-36qx	Running	0	50 minutes	10.244.0.2	<div><div></div></div> 0.005	<div><div></div></div> 81.102 Mi	☰	⋮
<div>✓</div> fluentd...9-n4fn	Running	0	50 minutes	10.244.1.2	<div><div></div></div> 0.011	<div><div></div></div> 83.879 Mi	☰	⋮
<div>✓</div> fluentd...9-qahc	Running	0	50 minutes	10.244.2.2	<div><div></div></div> 0.000	<div><div></div></div> 80.100 Mi	☰	⋮

Google Compute Engine

Google Container Engine - GKE

Demo Next



Google Cloud Platform

End

cloud
technology