Google Cloud Platform

Containers

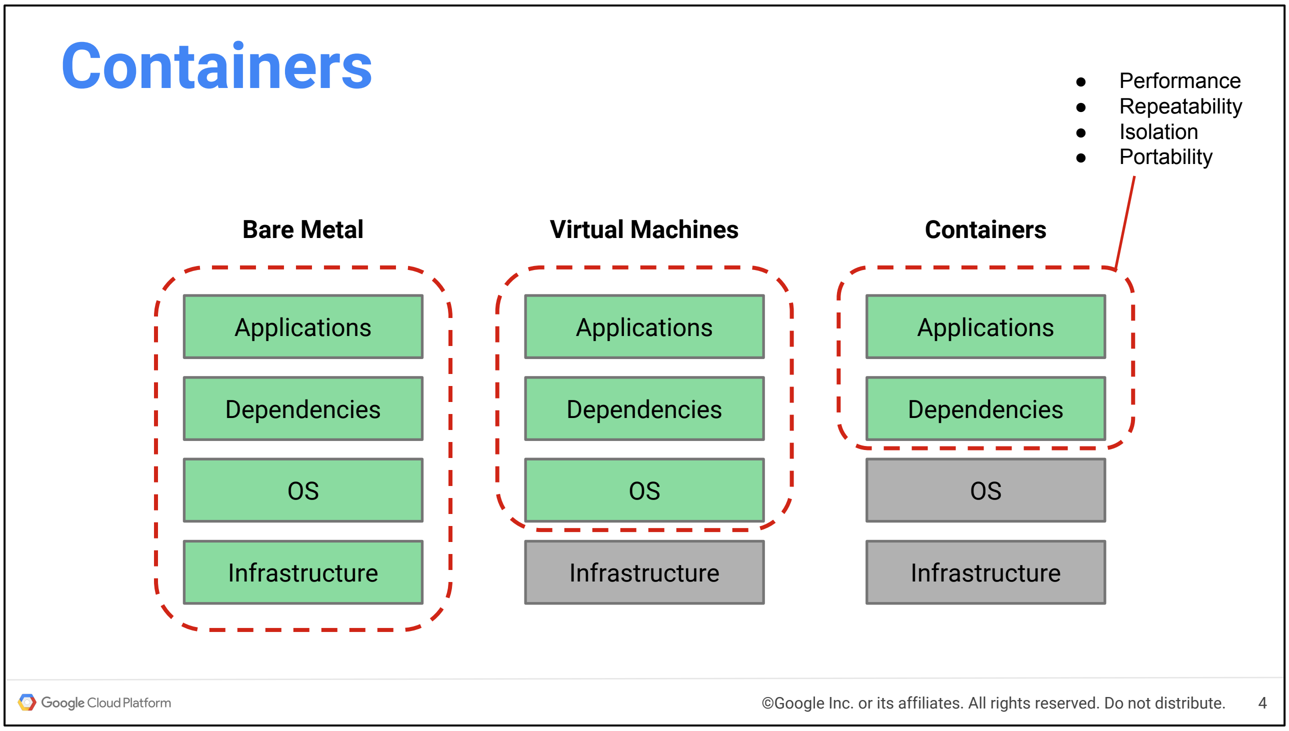
v 1.0

Agenda

1. Containers
2. Google Container Engine (GKE) (Kubernetes)
3. Google Container Registry
4. Lab
5. GKE or GAE or Containers on GCE ?
6. Review

GCP Compute & Processing Options

	Compute Engine	**Container Engine**	**App Engine Standard**	**App Engine Flexible**	**Cloud Functions**
Language support	Any	Any	Java 7 Python 2.7 Go PHP	Java 8 Python 2.7/3.5 Go Node.js Ruby Custom Runtimes	Triggers
Usage model	IaaS	IaaS PaaS	PaaS	PaaS	Microservices Architecture
	Server	Cluster	Autoscaling managed servers		Serverless
Primary use case	General Workloads	Container Workloads	Scalable web applications Mobile backend applications		Lightweight Event Actions



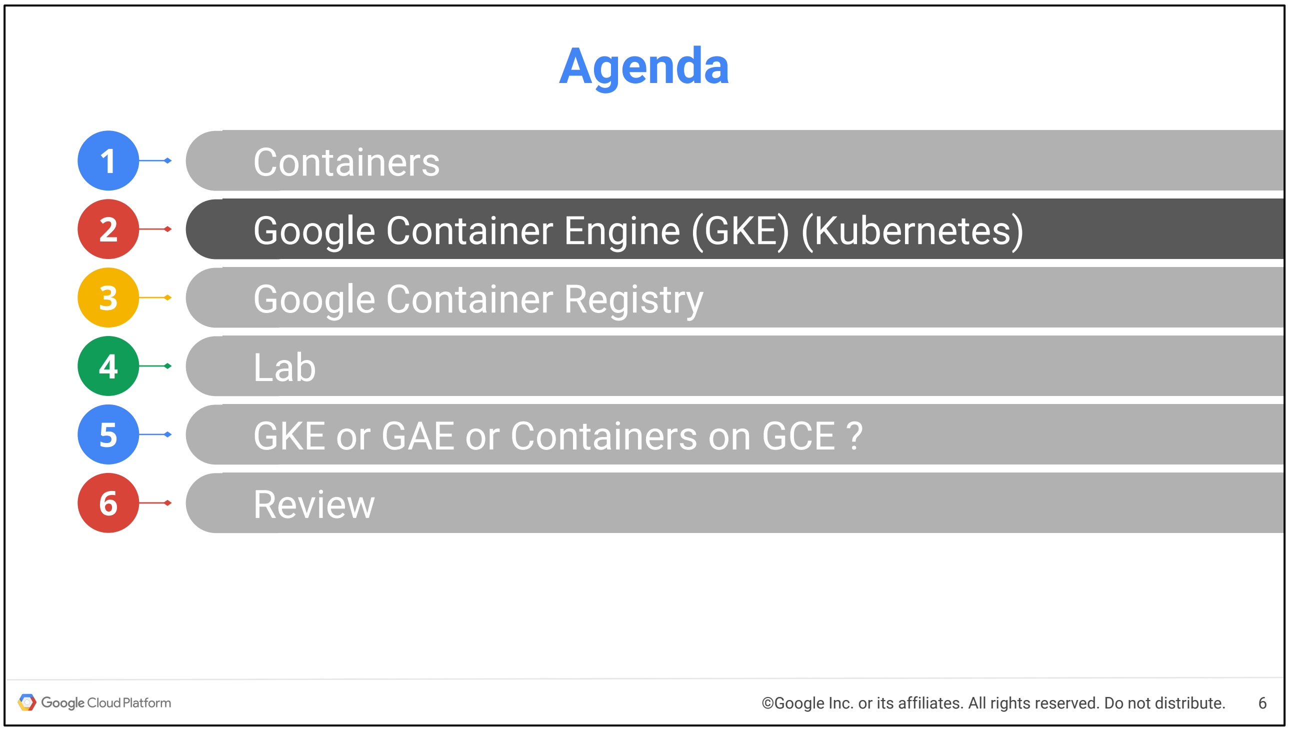
Containers

- Performance
- Repeatability
- Isolation
- Portability

Containers bundle application code and dependencies into a single unit, abstracting the application from the infrastructure.

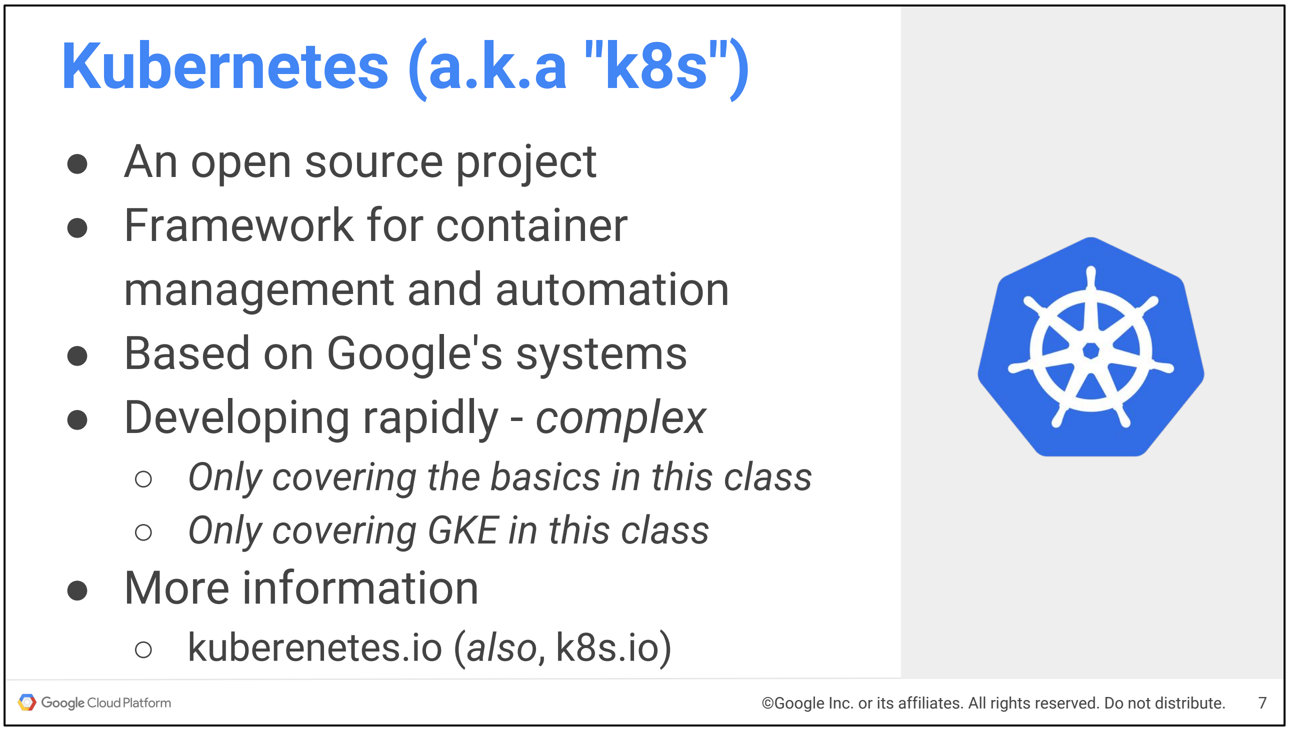
Benefits of Container-based Solutions

- Manage applications, not machines
- Maintain vendor independence
- Write once, run anywhere
  - Develop application on premise
  - Upload to cloud for production and scale
- Workload relocatability
  - Migrate to a new platform
- Decouple applications from dependencies
  - Ex: Namespaces, services, DNS, Secrets, specific APIs



Agenda

1. Containers
2. Google Container Engine (GKE) (Kubernetes)
3. Google Container Registry
4. Lab
5. GKE or GAE or Containers on GCE ?
6. Review



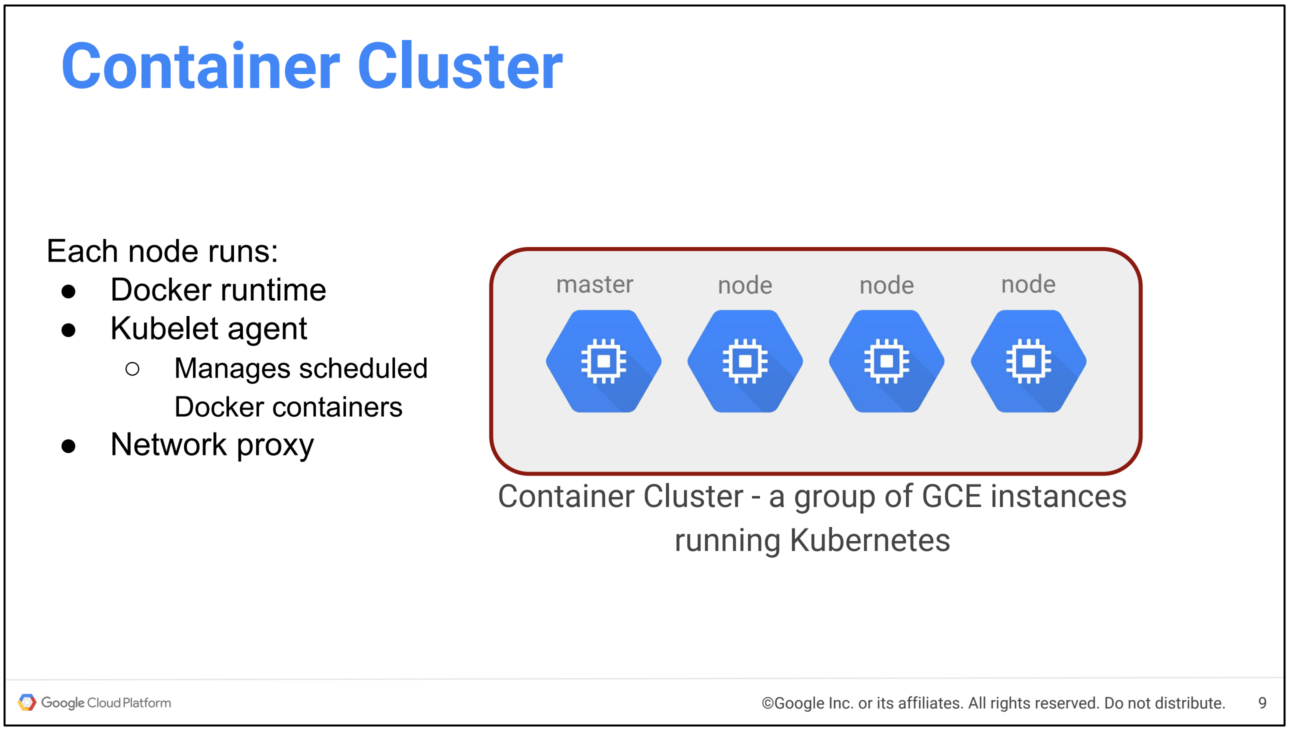
Kubernetes (a.k.a "k8s")

- An open source project
- Framework for container management and automation
- Based on Google's systems
- Developing rapidly - *complex*
  - *Only covering the basics in this class*
  - *Only covering GKE in this class*
- More information
  - kuberenetes.io (*also*, k8s.io)

https://kubernetes.io/
Kubernetes is greek for "helmsman" or "pilot". Projects started in 2014. Based on experience with Google's internal container management system.

Google Container Engine (GKE)

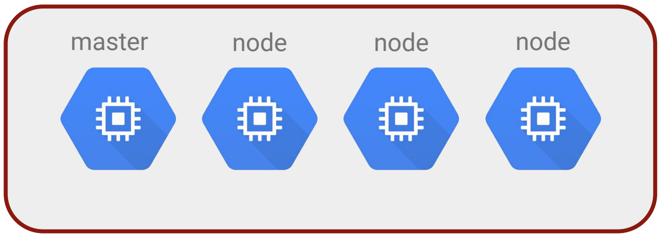
- Fully-managed service
  - Kubernetes software maintained
  - SLA
- Docker format containers
- Autoscaling (CPU or memory)
- Stackdriver logging and monitoring
- Cloud VPN integration
  - Hybrid cloud and on premise solutions
- Cloud IAM integration



Container Cluster

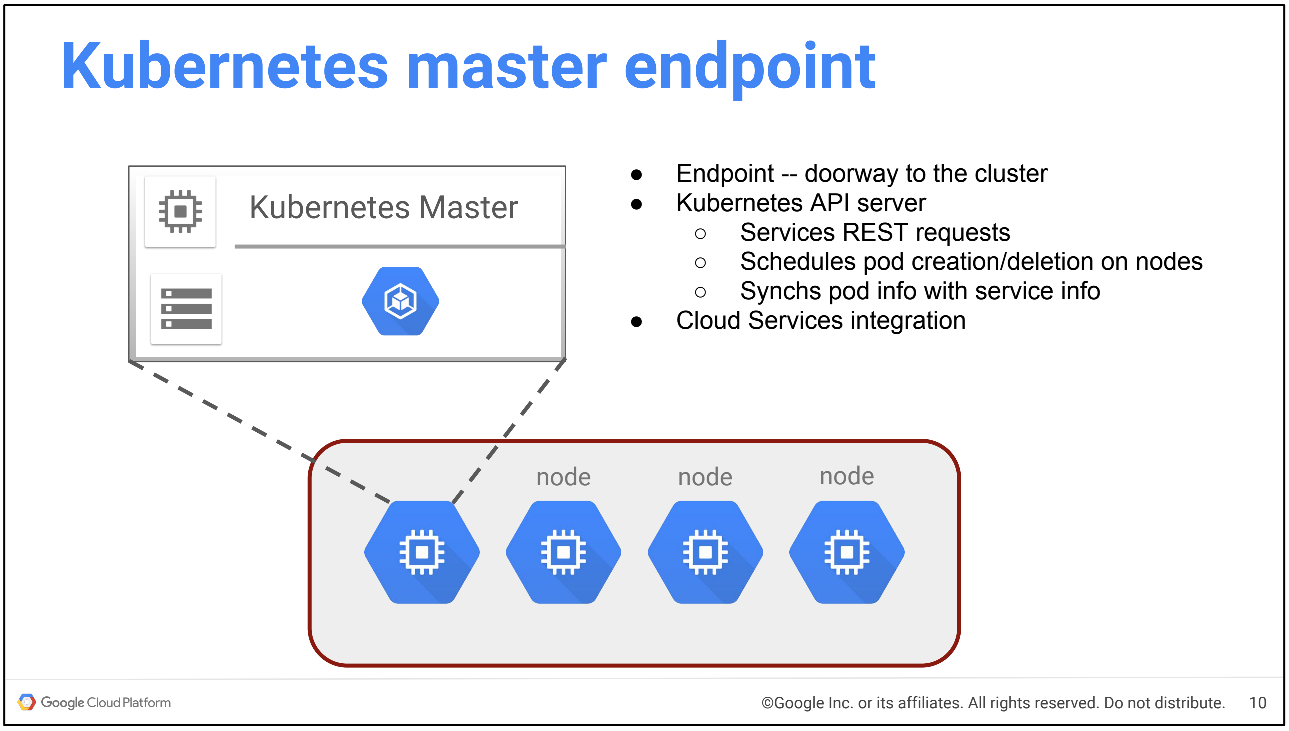
Each node runs:

- Docker runtime
- Kubelet agent
  - Manages scheduled Docker containers
- Network proxy

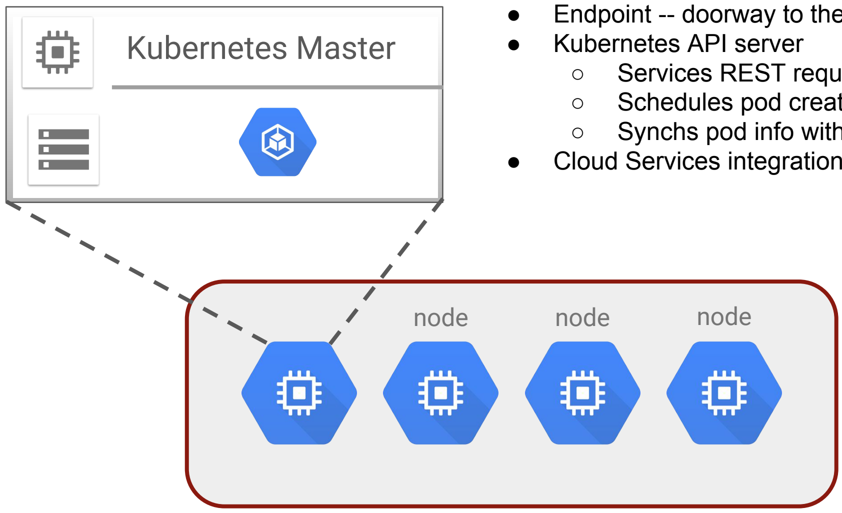


Container Cluster - a group of GCE instances running Kubernetes

A Container Cluster is a Google abstraction that relates Kubernetes to the GCE infrastructure.
A collection of GCE VM instances, consisting of the Kubernetes Master Endpoint and one or more node instances.



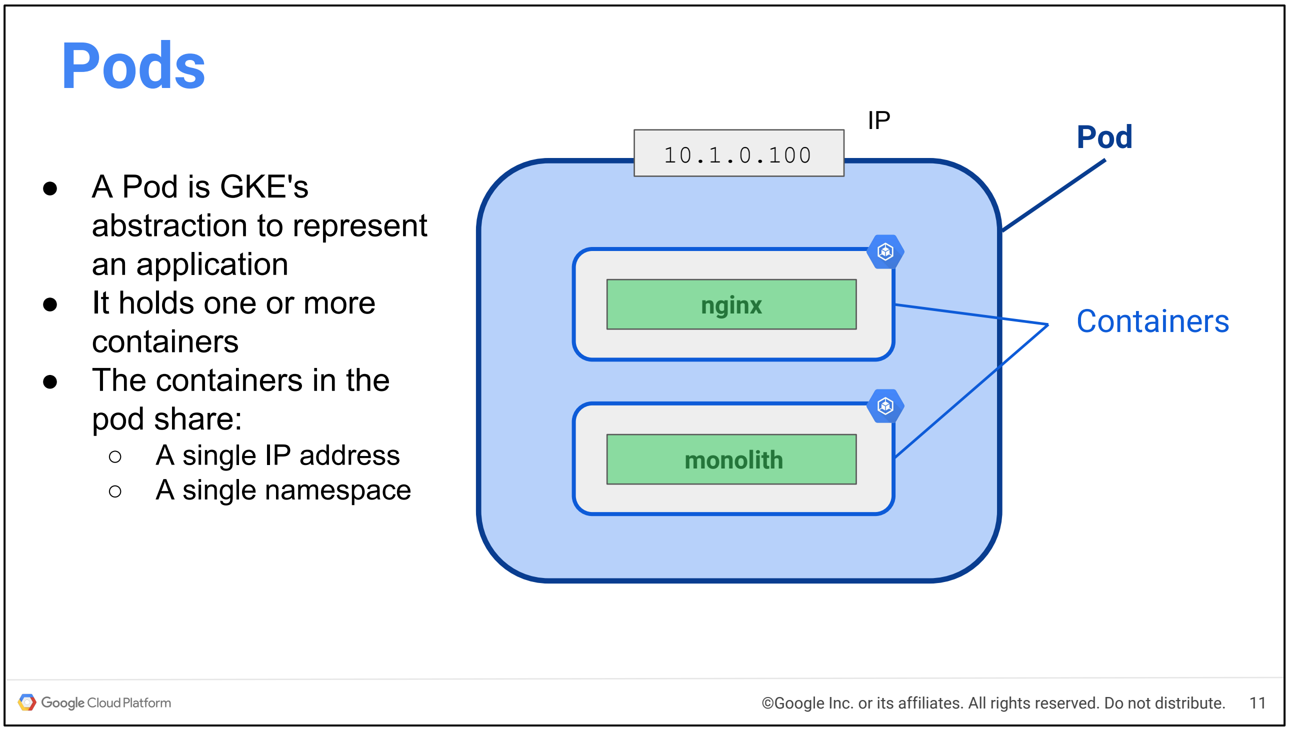
Kubernetes master endpoint



- Endpoint -- doorway to the cluster
- Kubernetes API server
  - Services REST requests
  - Schedules pod creation/deletion on nodes
  - Synchs pod info with service info
- Cloud Services integration

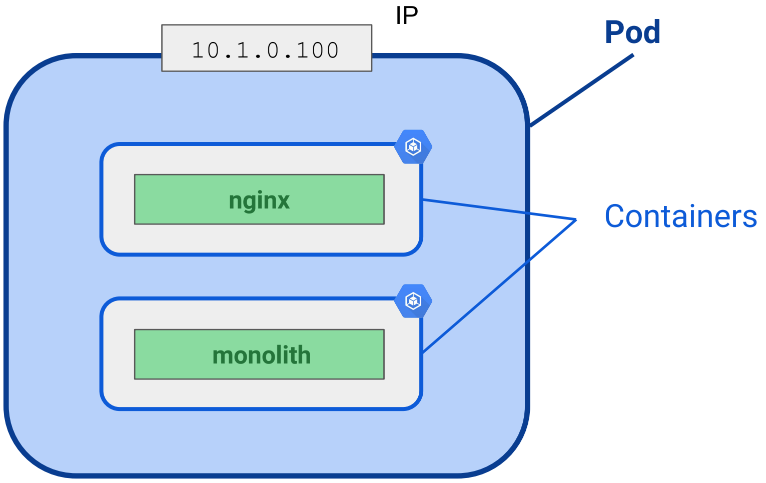
Google Cloud Platform

Now that you understand the physical relationship between GCE and Kubernetes, we need to focus on Kubernetes abstractions and understand how Kubernetes works. Then we will return to the discussion of containers and see how those Kubernetes abstractions interact with nodes.



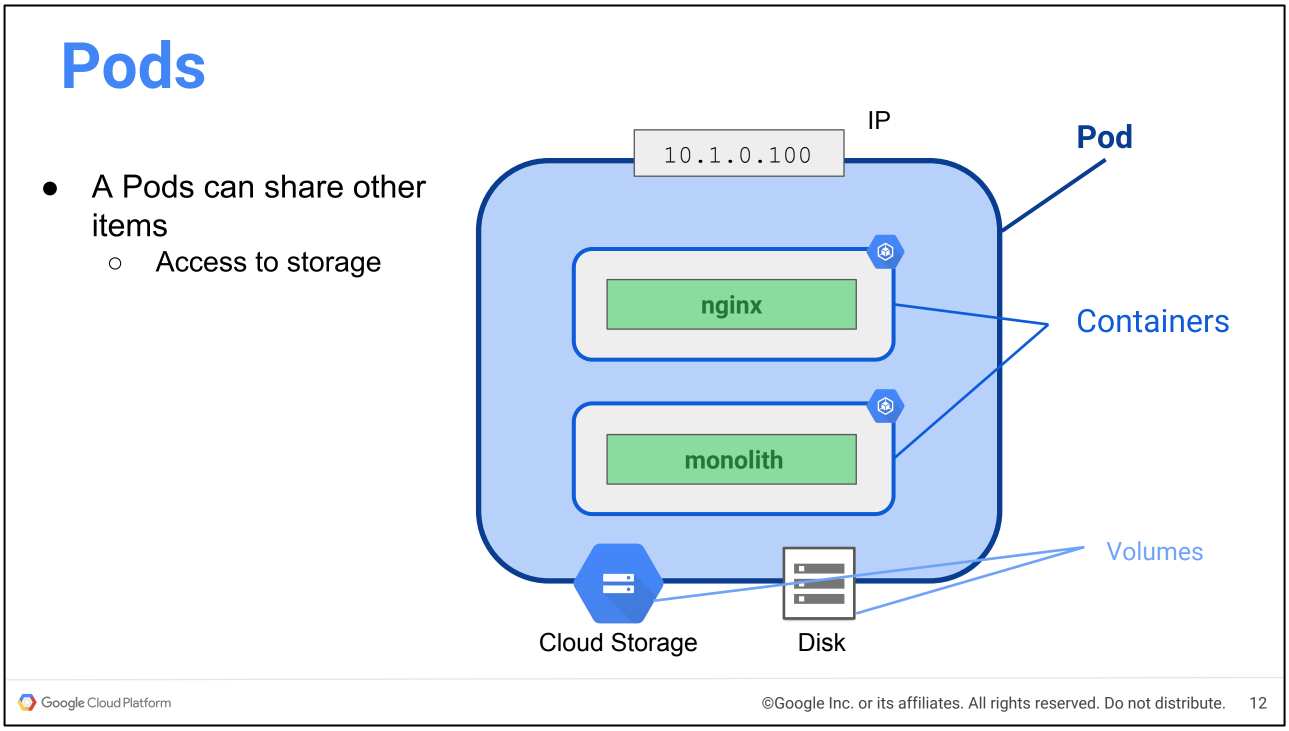
Pods

- A Pod is GKE's abstraction to represent an application
- It holds one or more containers
- The containers in the pod share:
  - A single IP address
  - A single namespace



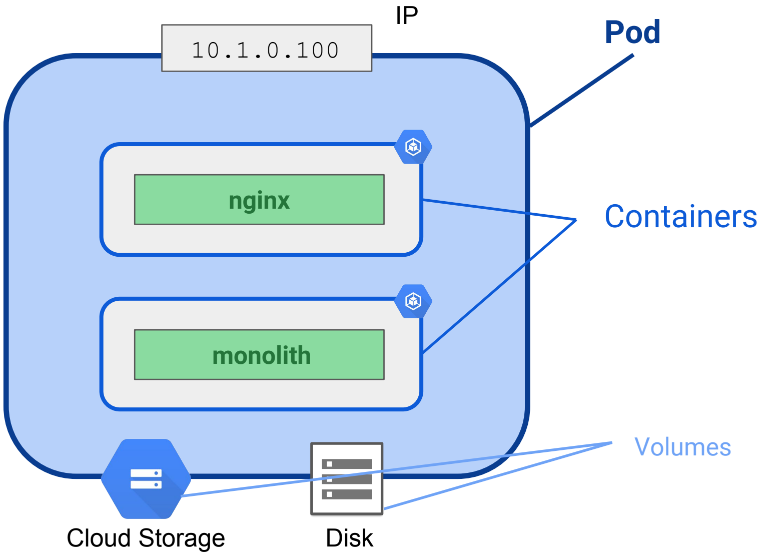
One purpose of GKE is to enable you to manage applications, not machines.
To accomplish this, you need to understand the GKE abstractions for applications.

Any data access mounted to a pod, called a Volume, is available to all containers in the pod.
Containers that are part of the same pod are guaranteed to be scheduled together on the same VM and can share state via local volumes.
Persistent Volumes, using persistent disks in GCE, survive instance and container restarts.



Pods

- A Pods can share other items
  - Access to storage

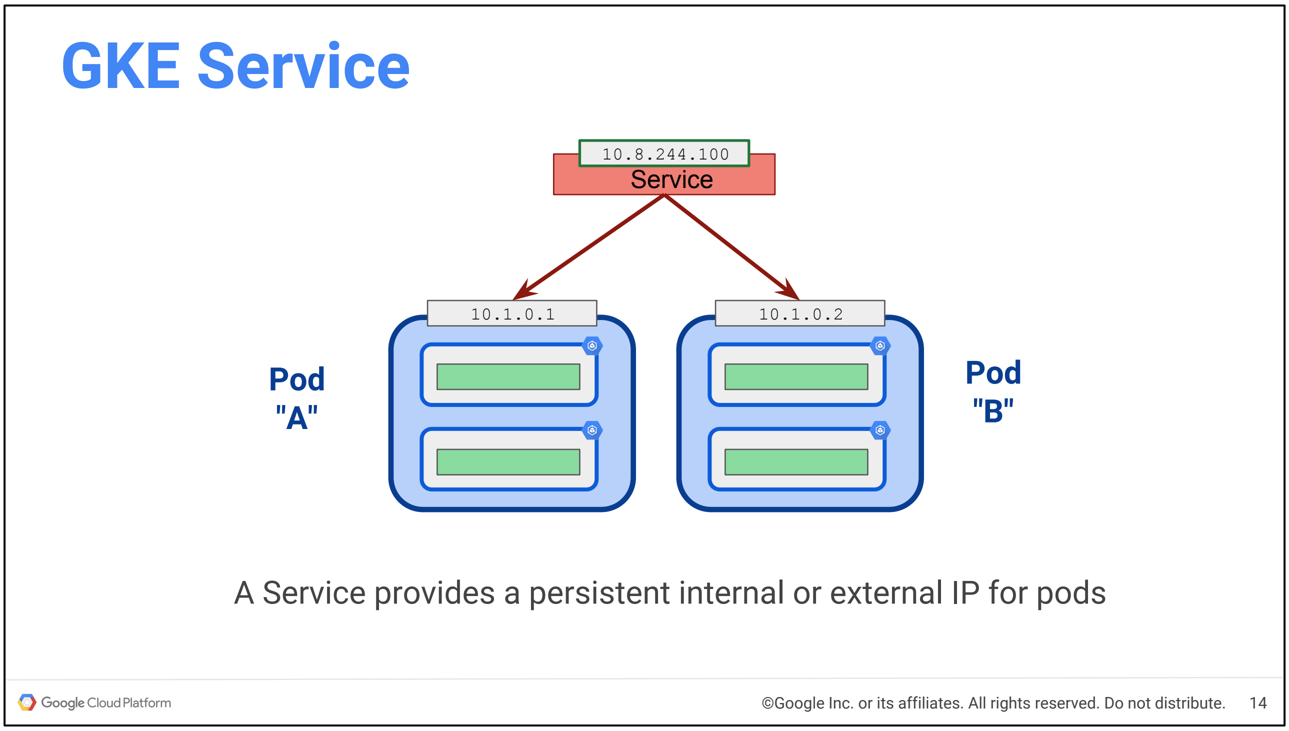


One purpose of GKE is to enable you to manage applications, not machines.
To accomplish this, you need to understand the GKE abstractions for applications.

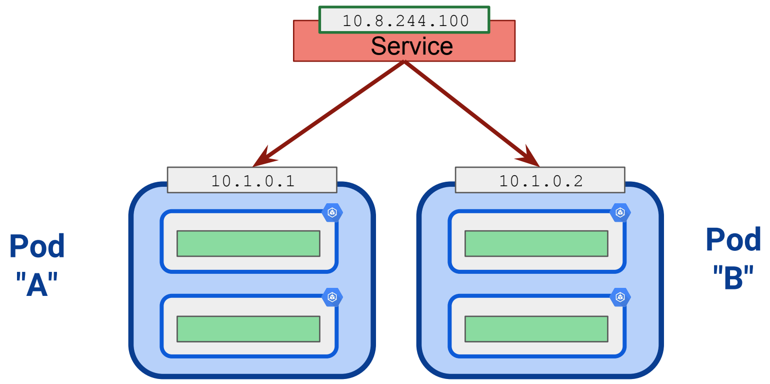
Any data access mounted to a pod, called a Volume, is available to all containers in the pod.
Containers that are part of the same pod are guaranteed to be scheduled together on the same VM and can share state via local volumes.
Persistent Volumes, using persistent disks in GCE, survive instance and container restarts.

GKE Labels

- Arbitrary key:value pairs
  - Applied to pods and other objects
- Used by GKE for orchestration
- label selector
  - A query on the labels
  - Labels matching selector have operations applied



GKE Service



A Service provides a persistent internal or external IP for pods

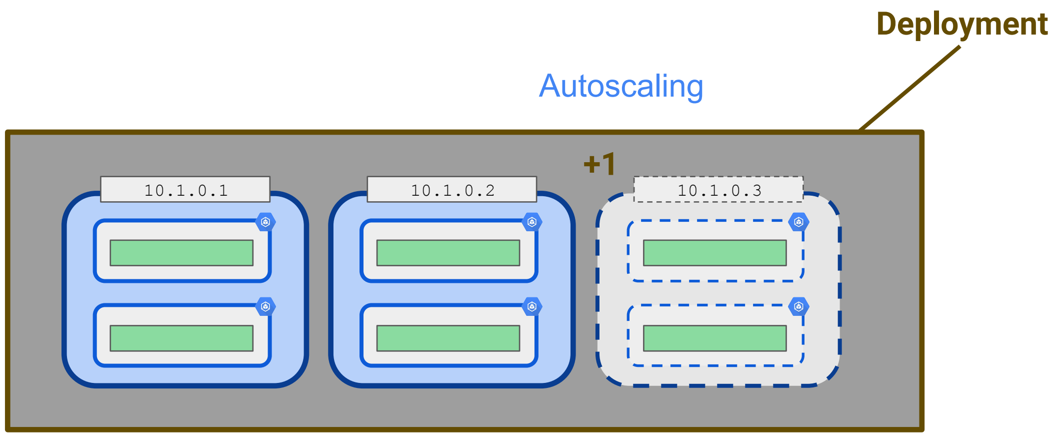
Here is an example to explain the GKE Service concept.

Problem
An application needs to communicate with a group of pods, "A", and "B".
What happens if a pod has to be restarted? Examples -- if the pod fails some kind of check like a health or readiness check.
When the pod restarts it will be dynamically assigned a new internal IP.

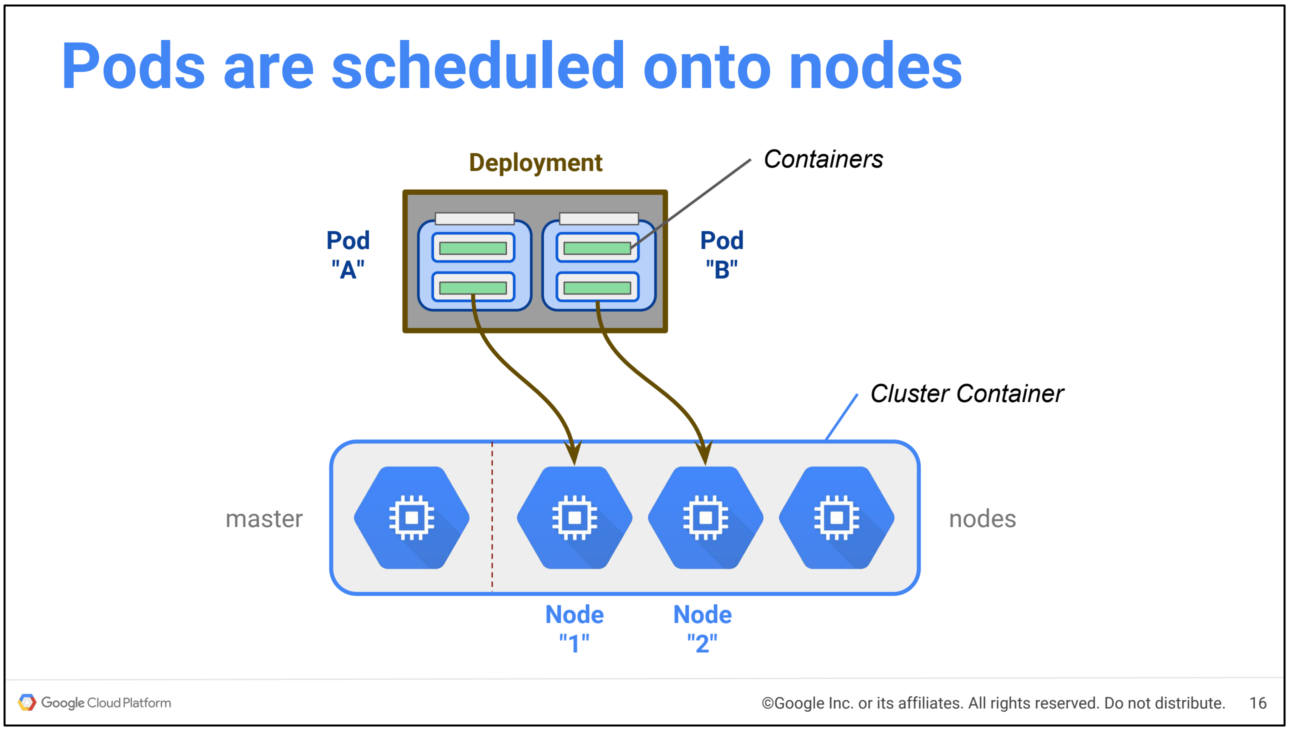
Solution
The service has a persistent IP, which can be either an internal IP or an external IP.
The application communicates with the pods through the service IP.
The services is connected to the pods by label. If a pod has the correct label it will be connected to the service.
If a pod restarts, it's label causes it to be picked up by the service.

Services come in different types. They can expose internally, externally, and they can load balance incoming requests to the pods in the group.
They combine the functions of a group and a load balancer.

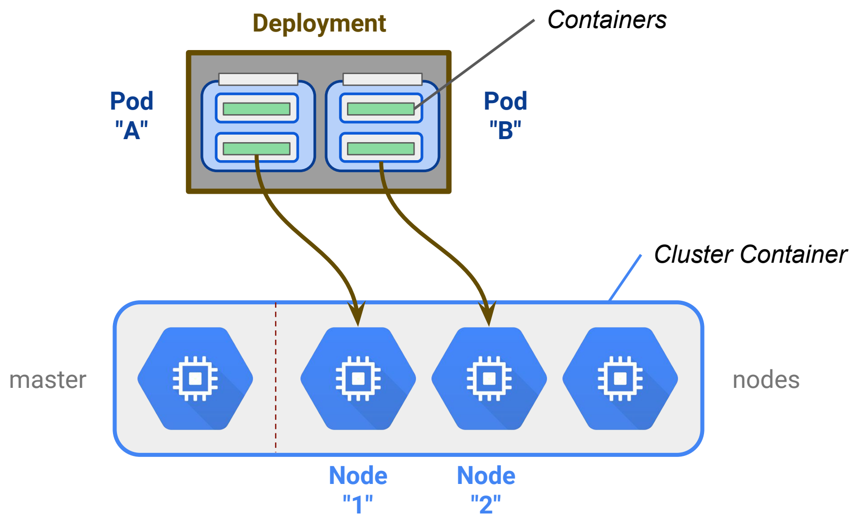
GKE Deployment



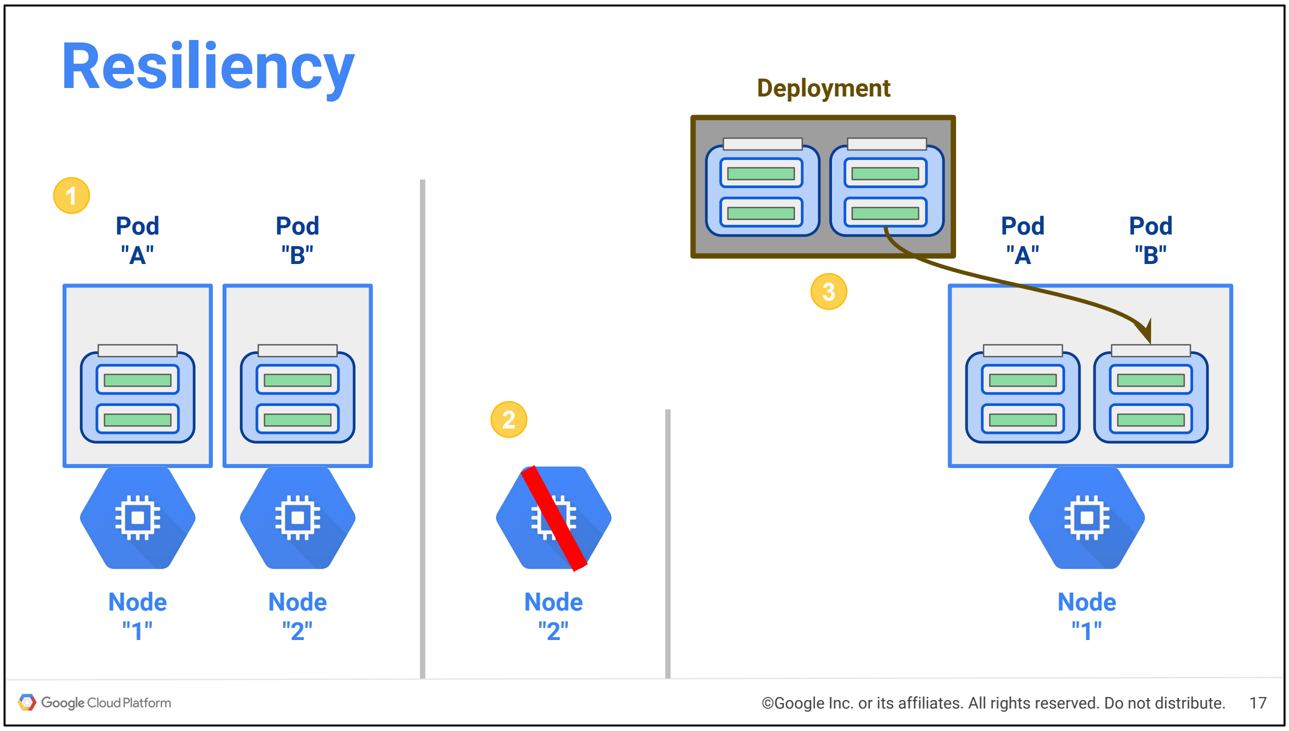
The deployment is analogous to an autoscaler. It drives the number of pods that are actually running towards the ideal state of the number that we want to be running. Deployments scale the number of pods.



Pods are scheduled onto nodes



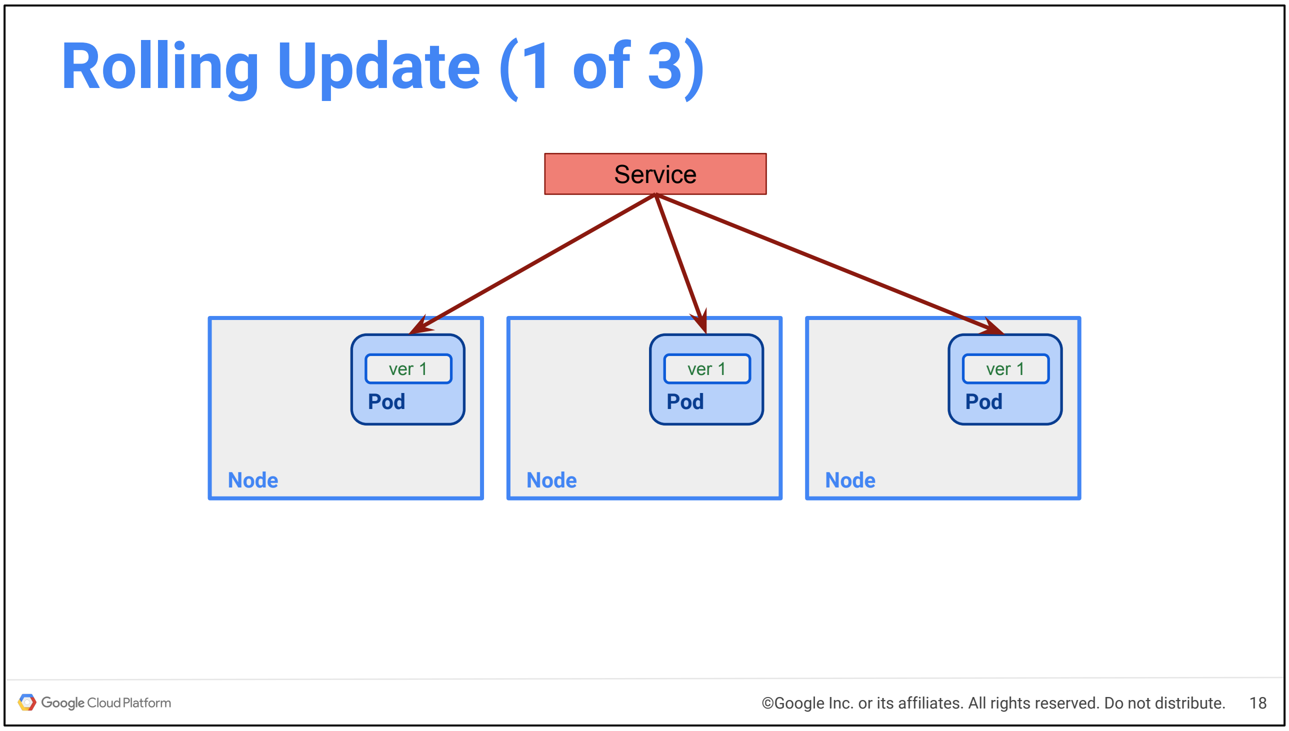
Deployments handle the scheduling of the pods onto the machines, which are called Nodes.
So now that you understand Pods and Deployments, we will return to the relationship with GCE.



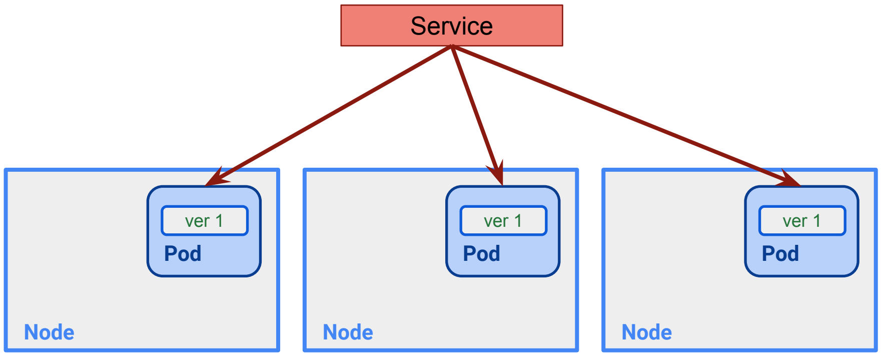
Resiliency

(1) Pod A is running on Node 1, Pod B is running on Node 2.
(2) Node 2 is lost
(3) Deployment redeploys Pod B to another node in the cluster, in this case Node 1

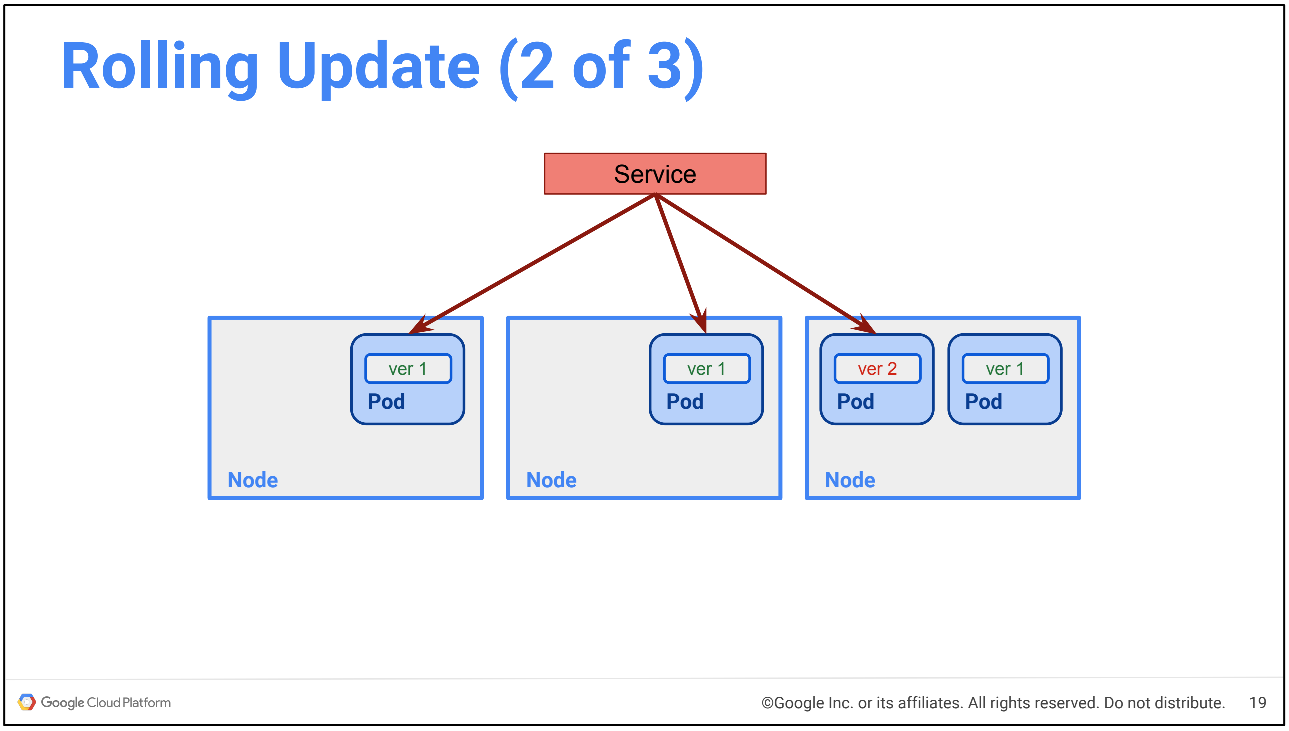
Deployments always drive the overall system towards the current state, using the resources available in the cluster.



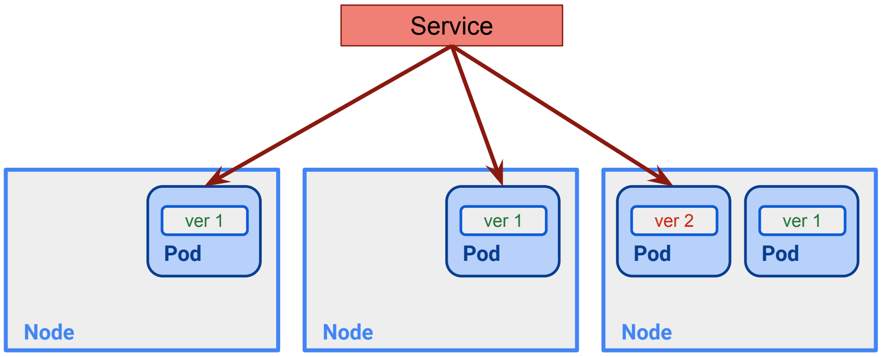
Rolling Update (1 of 3)



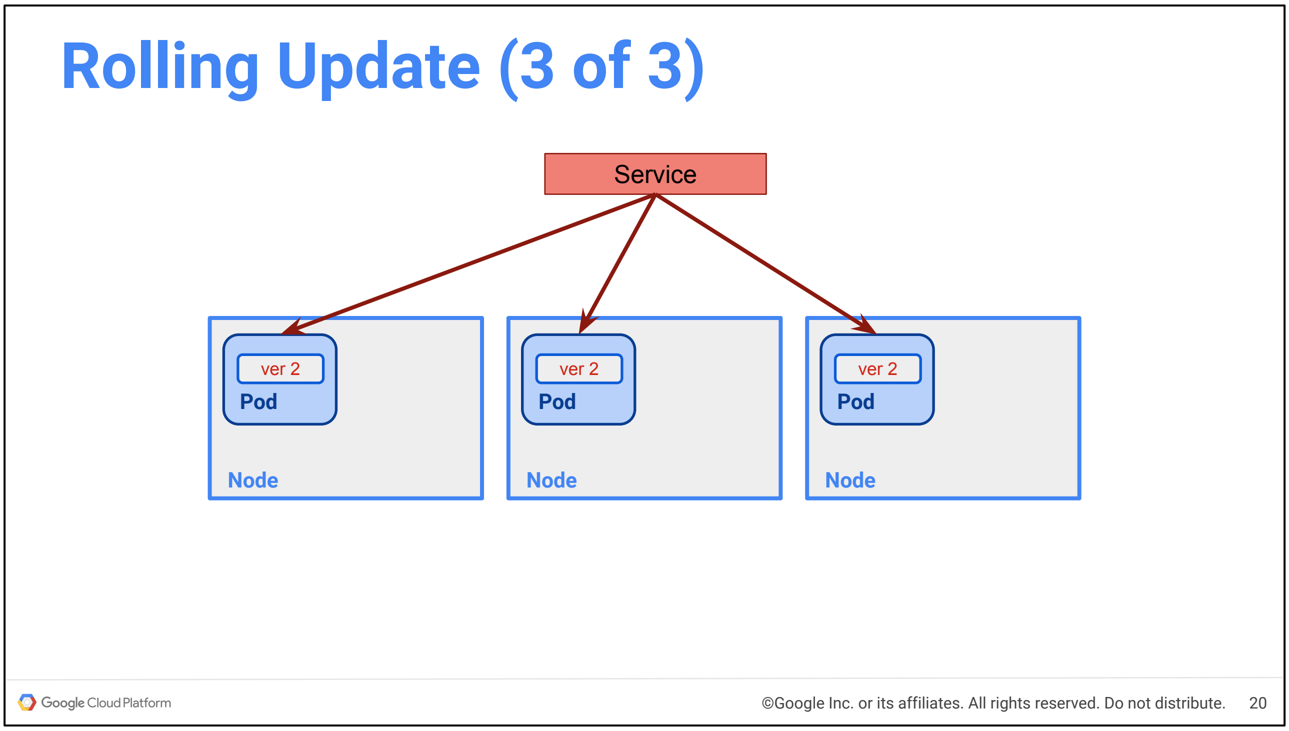
Version 1 of the software is running on three nodes.
Deployment is making sure we keep three pods running.
A service is shown so you can see the IP changes as well as the node changes.
The goal here is to deploy version 2 of the software, using the GKE Rolling Update feature.



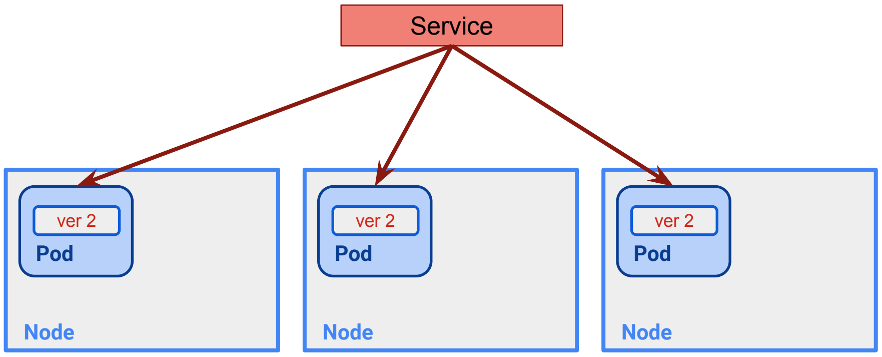
Rolling Update (2 of 3)



A second pod is launched on Node 1 with the new version of the application.
When it is operational, the Service picks up the pod (based on label) and begins serving to it.
We now have four pods and we only need three, so deployment manager shuts down the version 1 pod on node 3.



Rolling Update (3 of 3)

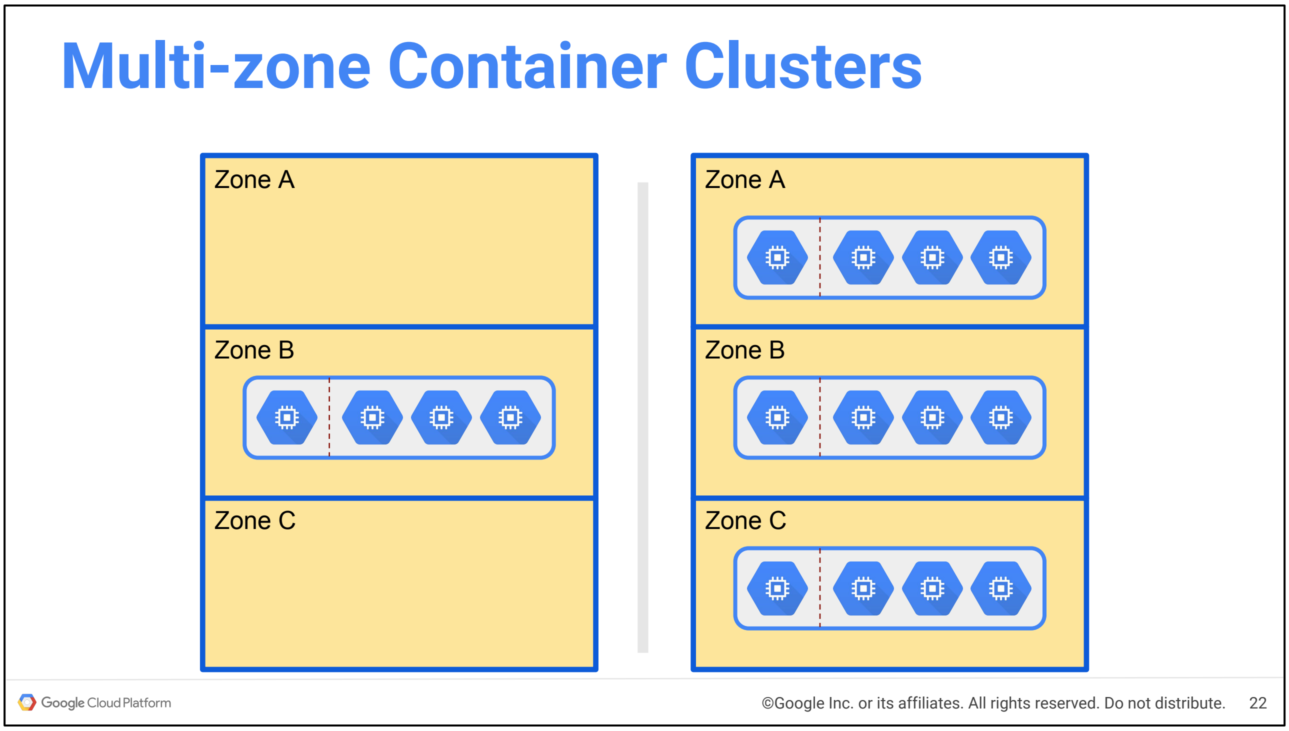


Google Cloud Platform

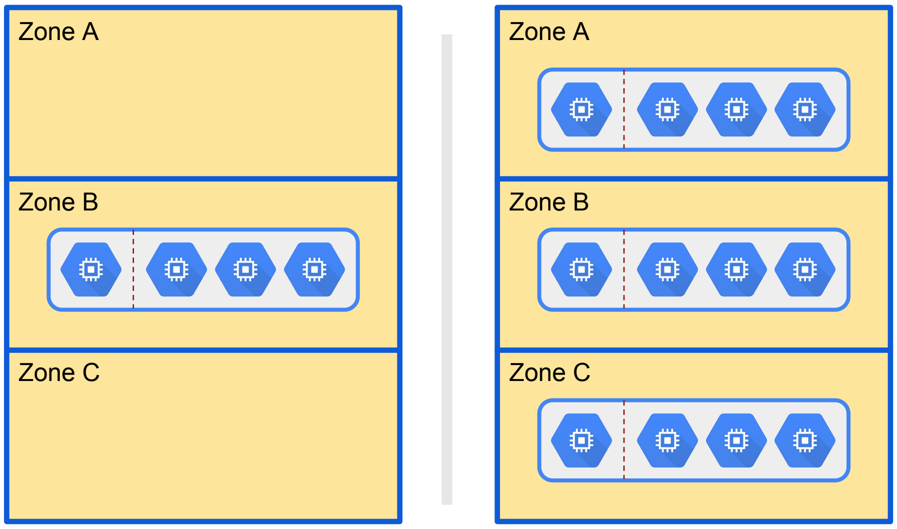
The process continues until all version 1 nodes have been replaced, one at a time with version 2 pods.

IAM support

- Roles
  - `container.admin`
    - Full management of Container Clusters and their Kubernetes API objects
  - `container.clusterAdmin`
    - Management of Cluster Containers
  - `container.developer`
    - Full access to Kubernetes API objects inside Container Clusters
  - `container.viewer`
    - Read-only access to GKE resources



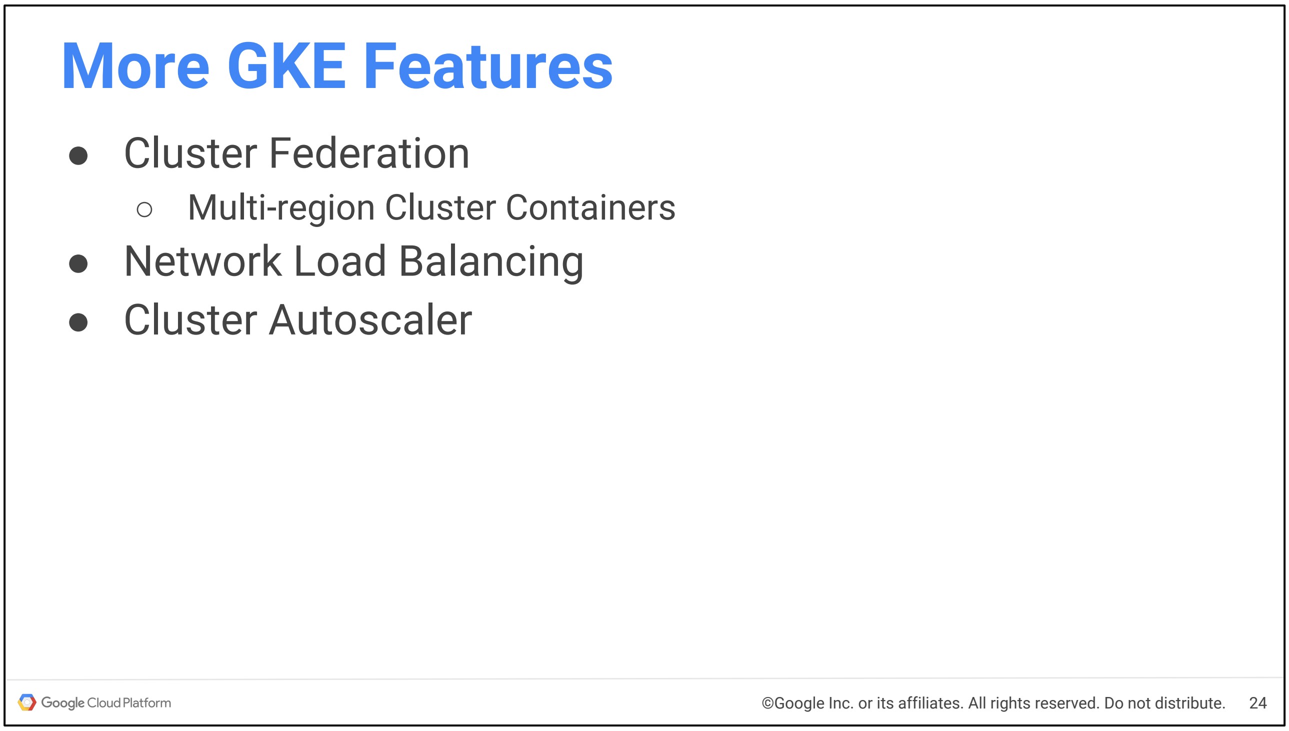
Multi-zone Container Clusters



When you enable Multi-zone Container Clusters, the container resources are replicated in the additional zones, and work is scheduled across all of them.
If one zone fails, the others can pick up the slack. In this case, any single zone is capable of running the entire application.
All of the zones are within the same region.

Node Pools

- Instance groups in the Kubernetes cluster
  - All VMs in a pool are the same
  - Pools can contain different VMs from one another
  - Pools can be in different zones
- GKE is node pool-aware
  - Labels on VMs in the pool make them available to GKE
- Node Pools and Multi-zone Container Clusters
  - GKE will replicate all the pools along with all the clusters
  - Careful! It could use up quotas in the region



More GKE Features

- Cluster Federation
  - Multi-region Cluster Containers
- Network Load Balancing
- Cluster Autoscaler

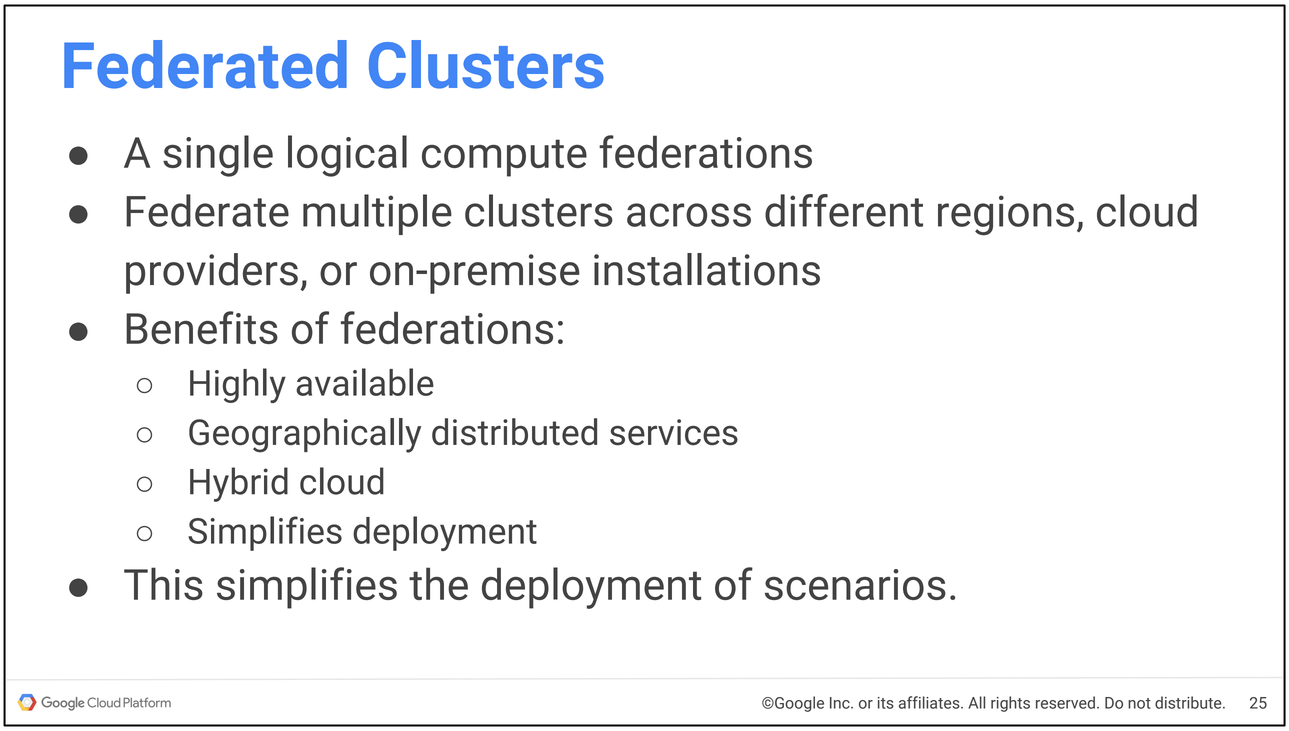
Google Cloud Platform

Cluster Federation
Cluster Federation -- Enables clusters across multiple regions, including other cloud providers or on premise Kubernetes installations.
Useful for super high availability or for super scalability.
https://cloud.google.com/container-engine/docs/cluster-federation

About Secrets:
https://cloud.google.com/sql/docs/mysql/connect-container-engine

About Network and HTTP Load Balancing
https://cloud.google.com/container-engine/docs/tutorials/http-balancer

Cluster Autoscaler
https://cloud.google.com/container-engine/docs/cluster-autoscaler



Federated Clusters

- A single logical compute federations
- Federate multiple clusters across different regions, cloud providers, or on-premise installations
- Benefits of federations:
  - Highly available
  - Geographically distributed services
  - Hybrid cloud
  - Simplifies deployment
- This simplifies the deployment of scenarios.

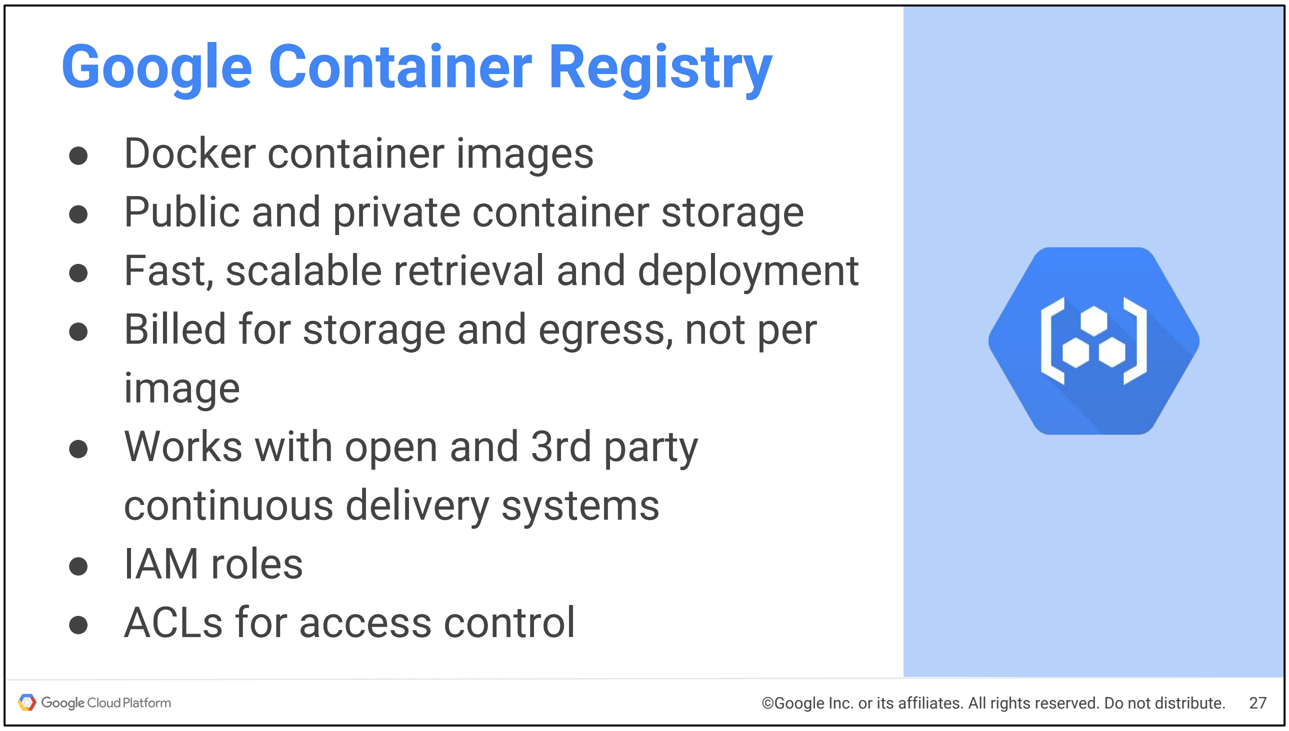
https://cloud.google.com/container-engine/docs/cluster-federation

Cluster Federation is useful when you want to deploy resources across more than one cluster, region or cloud provider. You may want to do this to enable high availability, offer greater geographic coverage for your app, use more than one cloud provider, combine cloud provider and on-premise solutions, or for ultra-high scalability.

Cluster Federation is also helpful when you want resources to be contactable in a consistent manner from both inside and outside their clusters, without incurring unnecessary latency or bandwidth cost penalties, or being susceptible to individual cluster outages.

Agenda

1. Containers
2. Google Container Engine (GKE) (Kubernetes)
3. Google Container Registry
4. Lab
5. GKE or GAE or Containers on GCE ?
6. Review

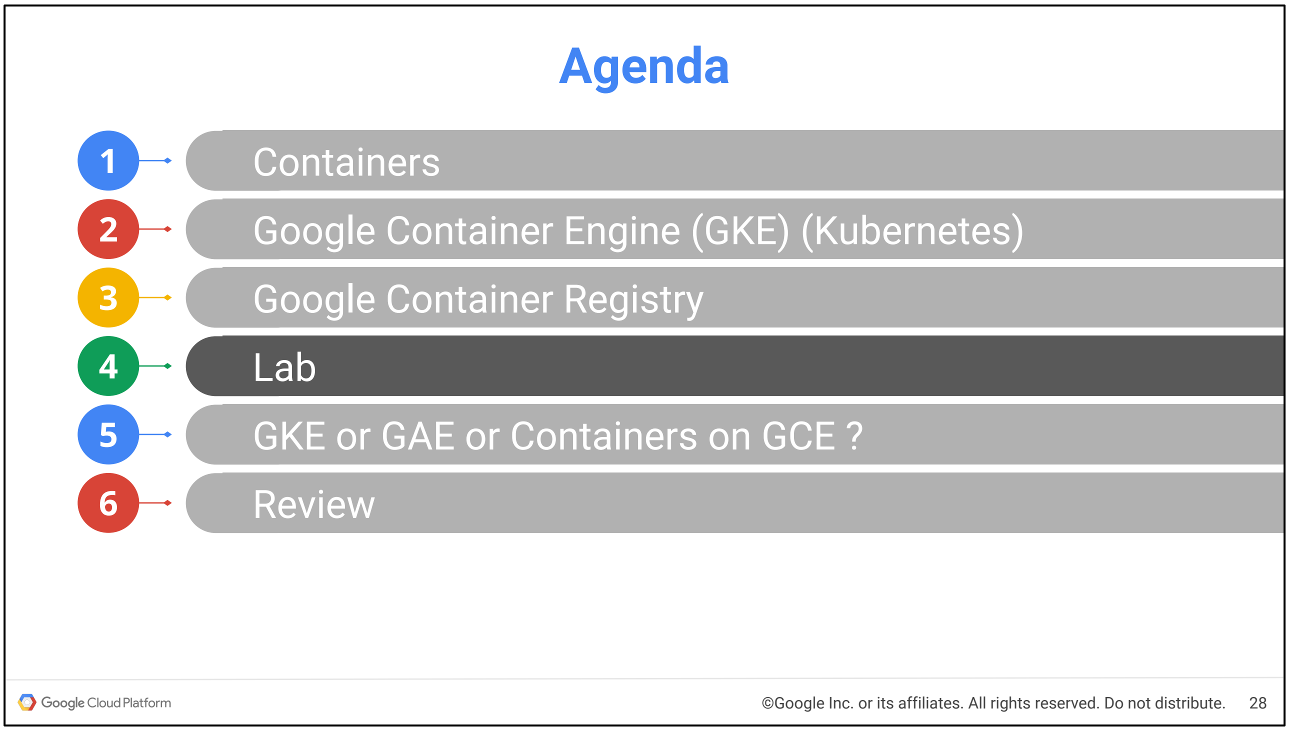


Google Container Registry

- Docker container images
- Public and private container storage
- Fast, scalable retrieval and deployment
- Billed for storage and egress, not per image
- Works with open and 3rd party continuous delivery systems
- IAM roles
- ACLs for access control

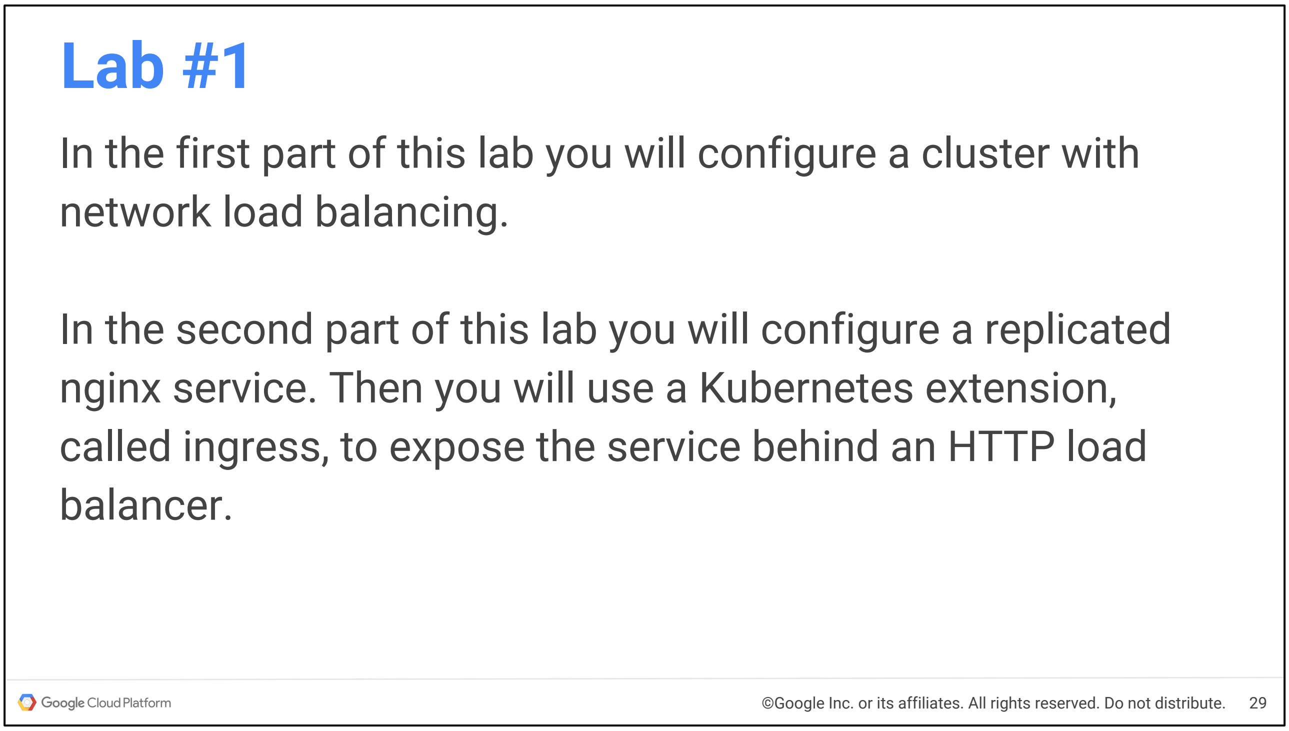
Google Container Registry provides secure, private Docker image storage on Google Cloud Platform.
While Docker provides a central registry to store public images, you may not want your images to be accessible to the world. In this case, you must use a private registry.

The Google Container Registry runs on Google Cloud Platform, so can be relied upon for consistent uptime and security. The registry can be accessed through an HTTPS endpoint, so you can pull images from any machine, whether it's a Google Compute Engine instance or your own hardware.



Agenda

1. Containers
2. Google Container Engine (GKE) (Kubernetes)
3. Google Container Registry
4. Lab
5. GKE or GAE or Containers on GCE ?
6. Review

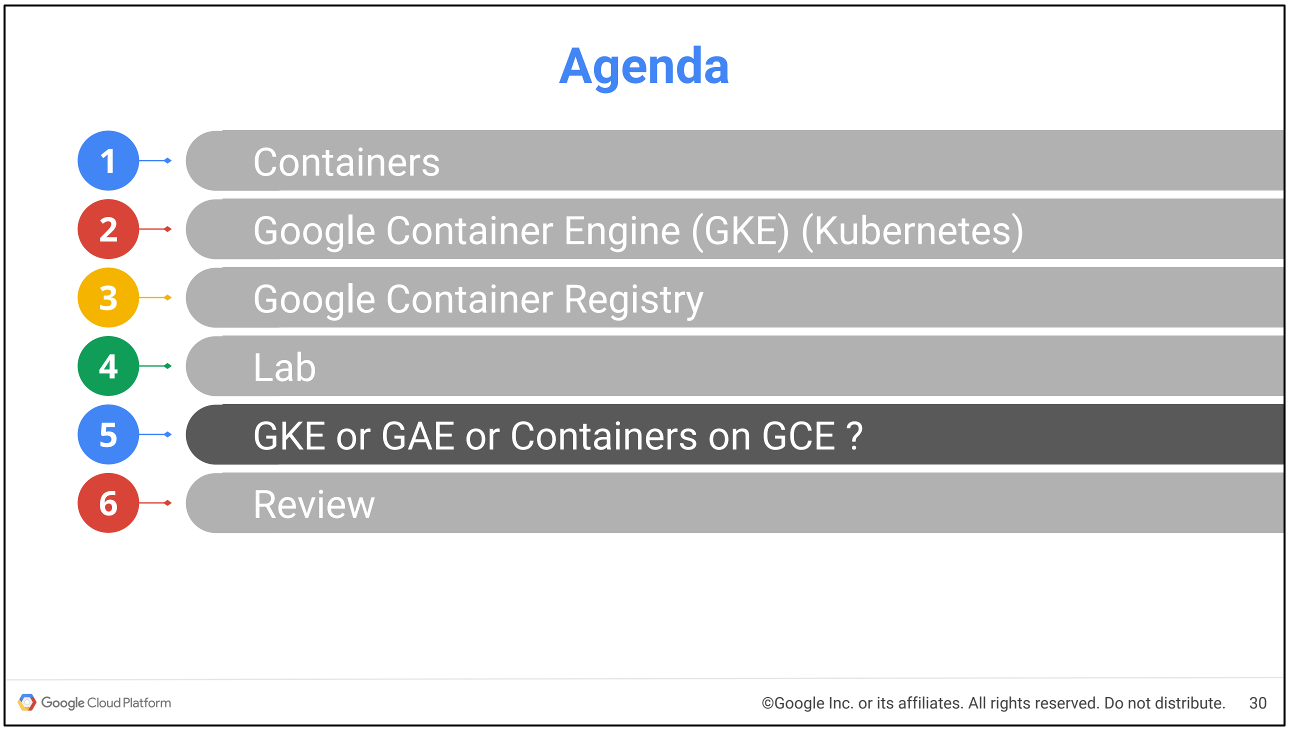


Lab #1

In the first part of this lab you will configure a cluster with network load balancing.

In the second part of this lab you will configure a replicated nginx service. Then you will use a Kubernetes extension, called ingress, to expose the service behind an HTTP load balancer.

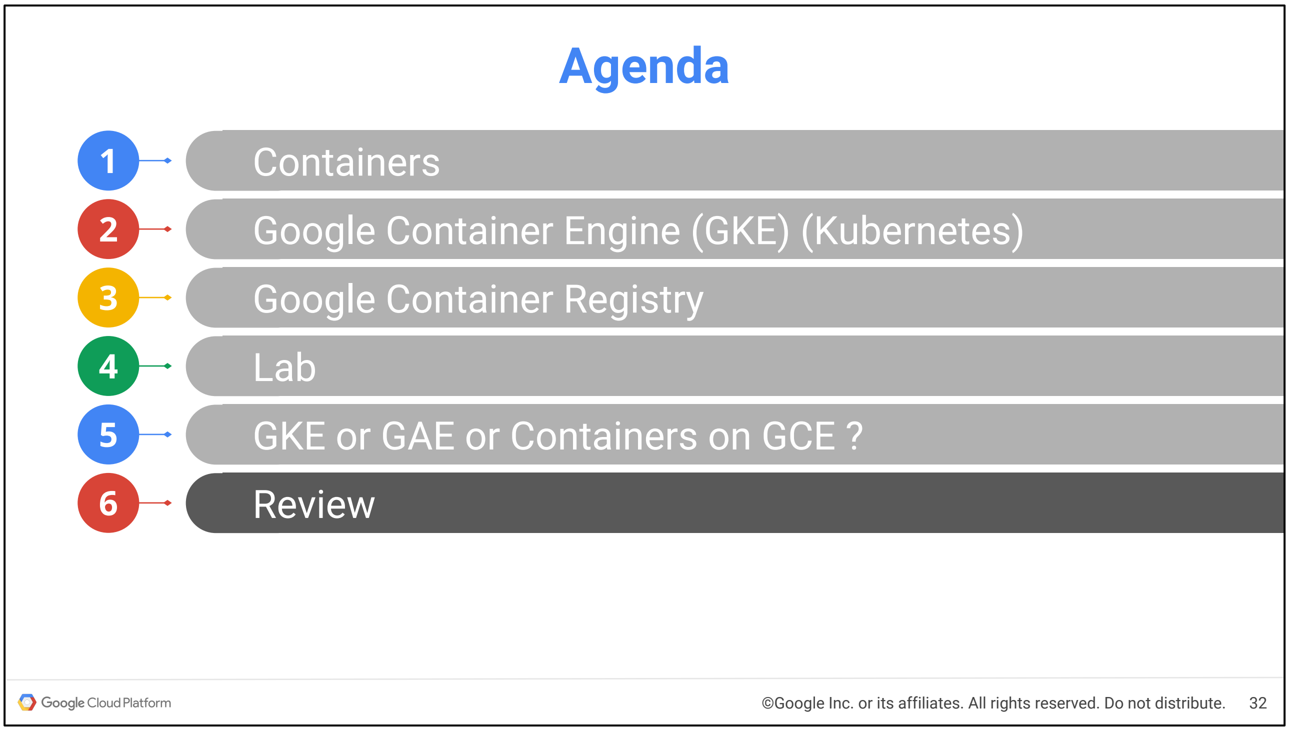
16-1 Kubernetes Load Balancing



Agenda

1. Containers
2. Google Container Engine (GKE) (Kubernetes)
3. Google Container Registry
4. Lab
5. GKE or GAE or Containers on GCE ?
6. Review

Google Cloud Platform



Agenda

1. Containers
2. Google Container Engine (GKE) (Kubernetes)
3. Google Container Registry
4. Lab
5. GKE or GAE or Containers on GCE ?
6. Review

More...

- Google Container Engine (GKE)
  - https://cloud.google.com/container-engine/docs/
- Google Container Registry
  - https://cloud.google.com/container-registry/docs/

cloud.google.com