Containers

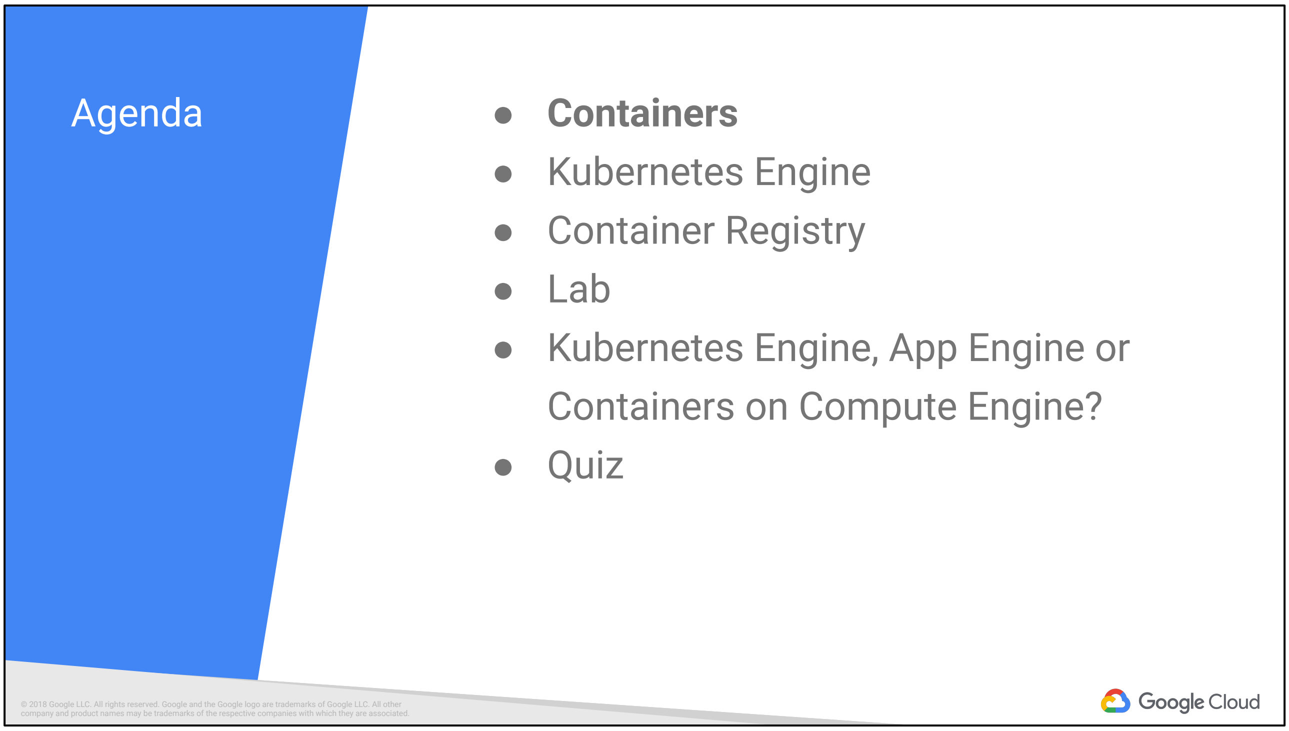
Architecting with GCP Fundamentals: Infrastructure

KUBERNETES ENGINE, CONTAINER REGISTRY

QWIKLABS KUBERNETES LOAD BALANCING

Google Cloud

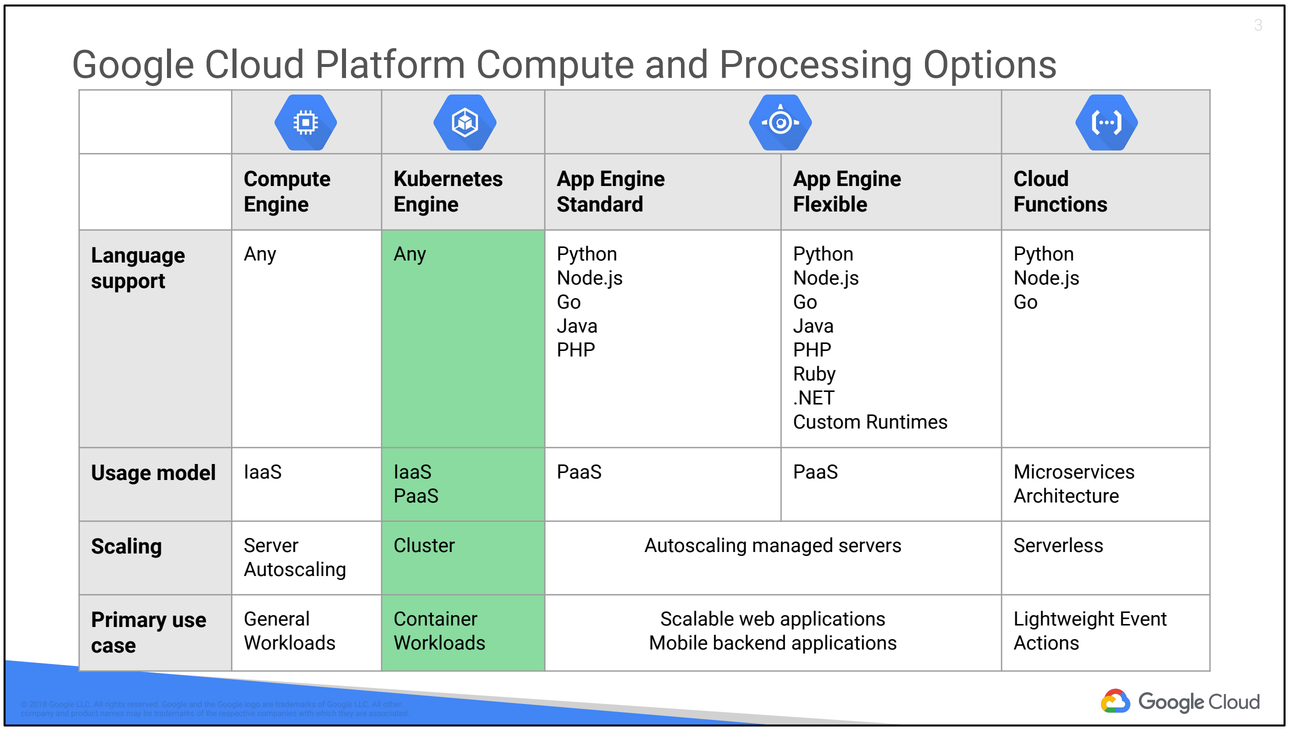
Last modified 2018-01-29



Agenda

- **Containers**
- Kubernetes Engine
- Container Registry
- Lab
- Kubernetes Engine, App Engine or Containers on Compute Engine?
- Quiz

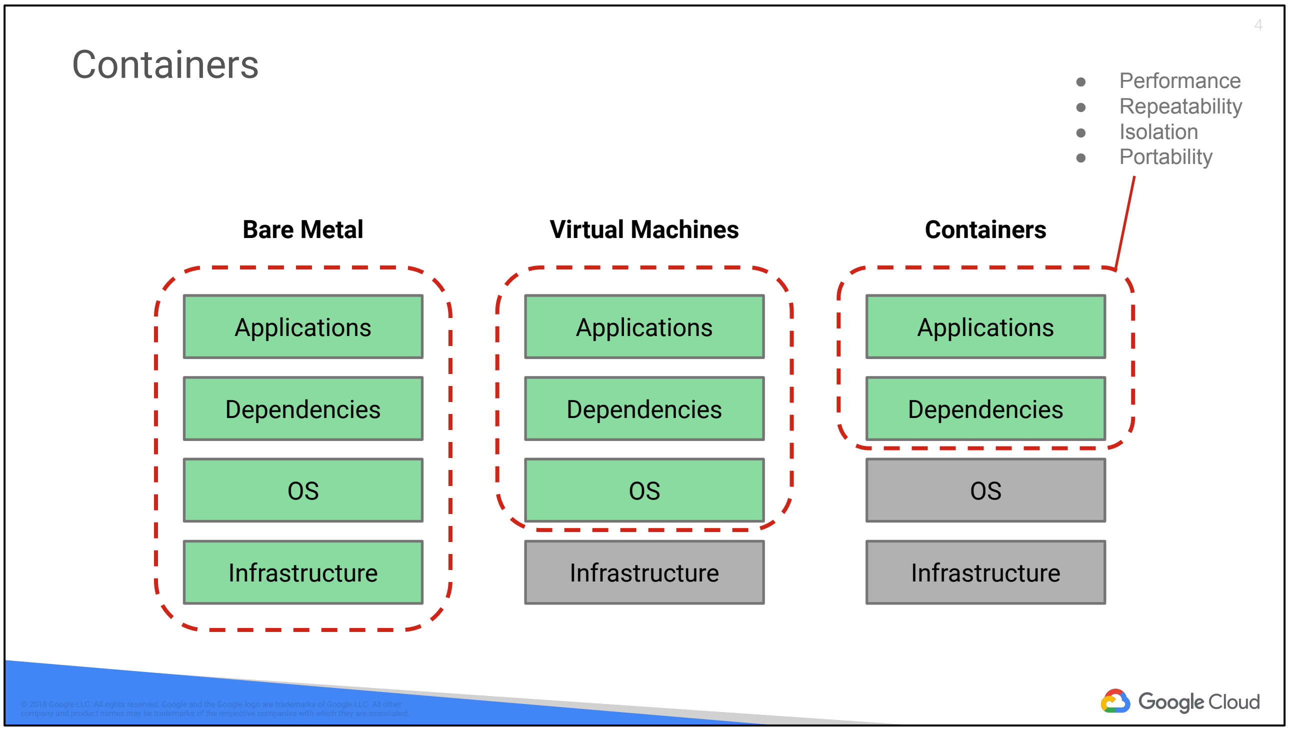
Google Cloud



Google Cloud Platform Compute and Processing Options

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

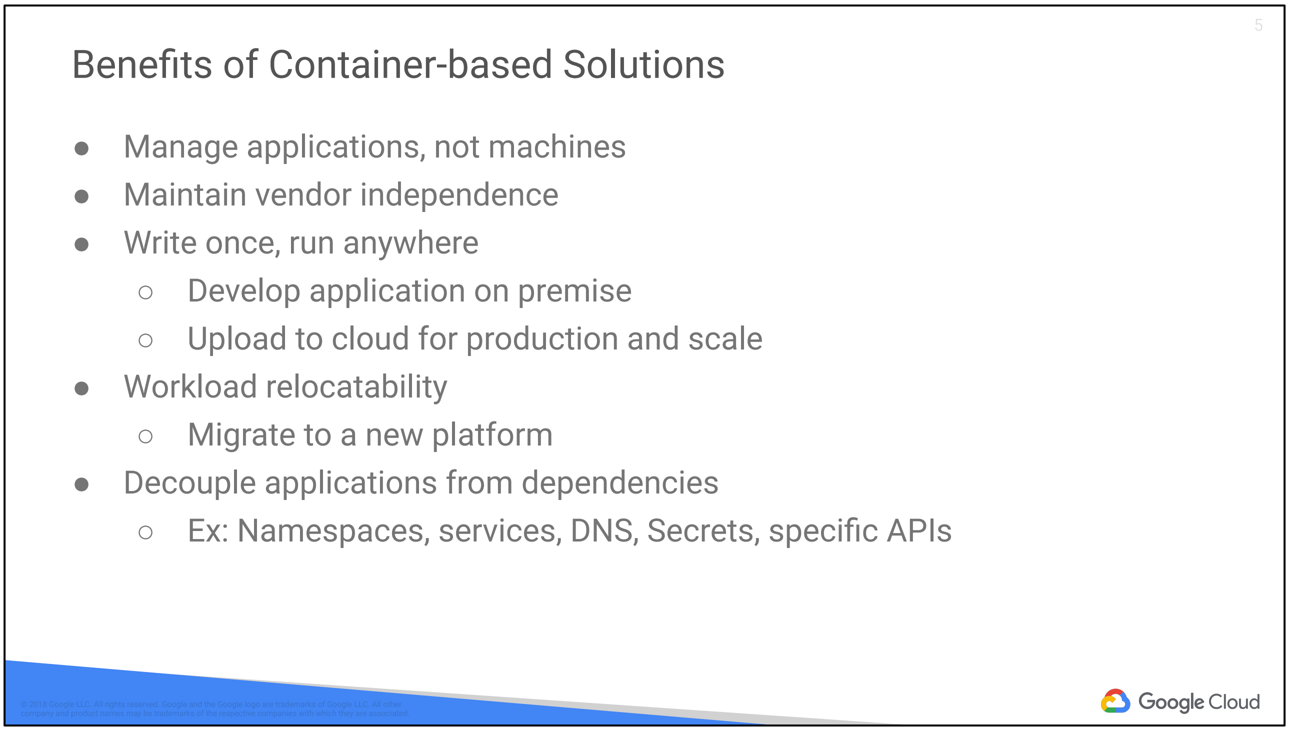
Google Cloud



Containers

- Performance
- Repeatability
- Isolation
- Portability

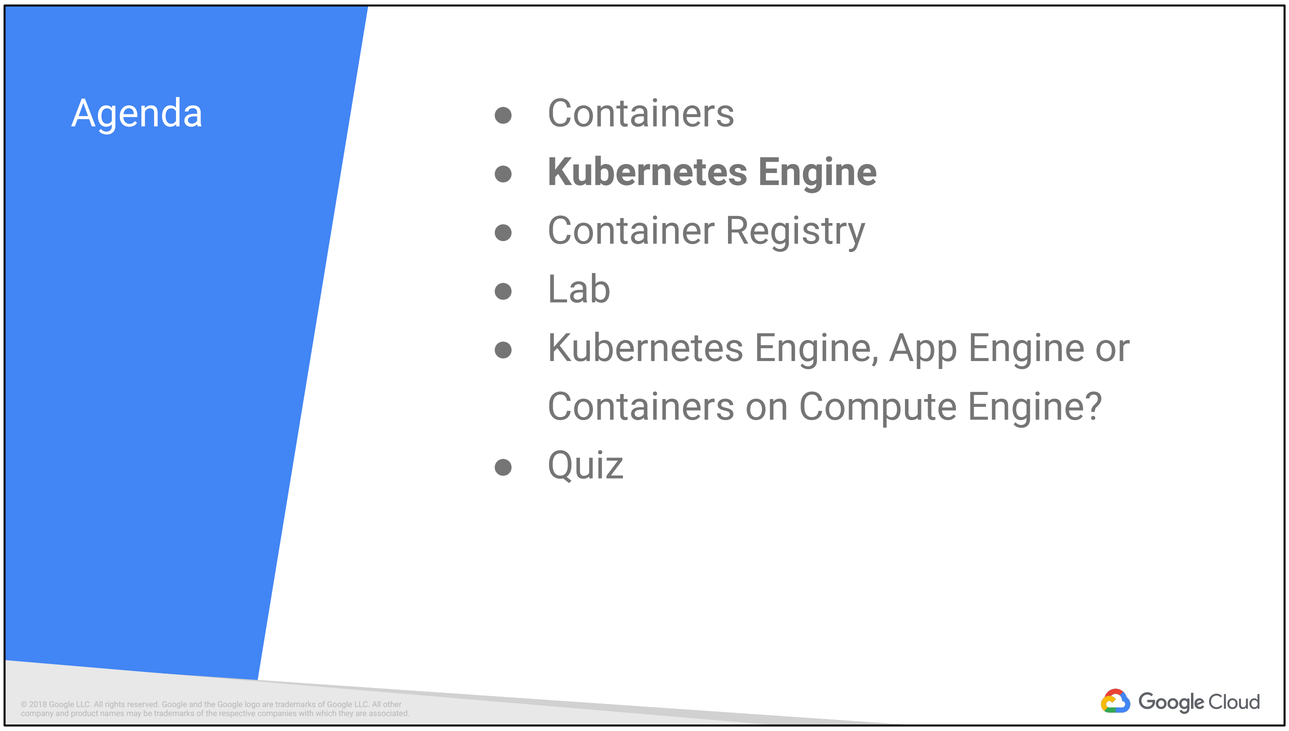
Containers bundle application code and dependencies into a single unit, abstracting the application from the infrastructure.



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

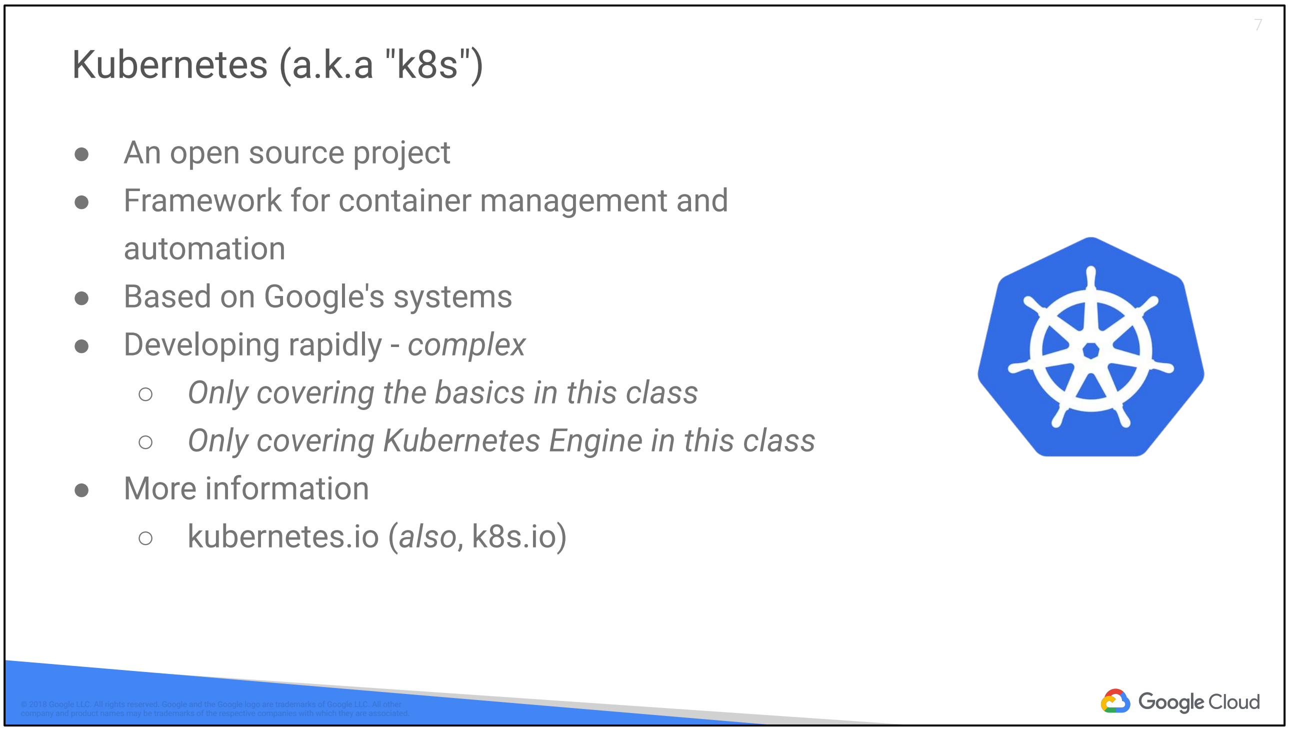
Google Cloud



Agenda

- Containers
- **Kubernetes Engine**
- Container Registry
- Lab
- Kubernetes Engine, App Engine or Containers on Compute Engine?
- Quiz

Google Cloud

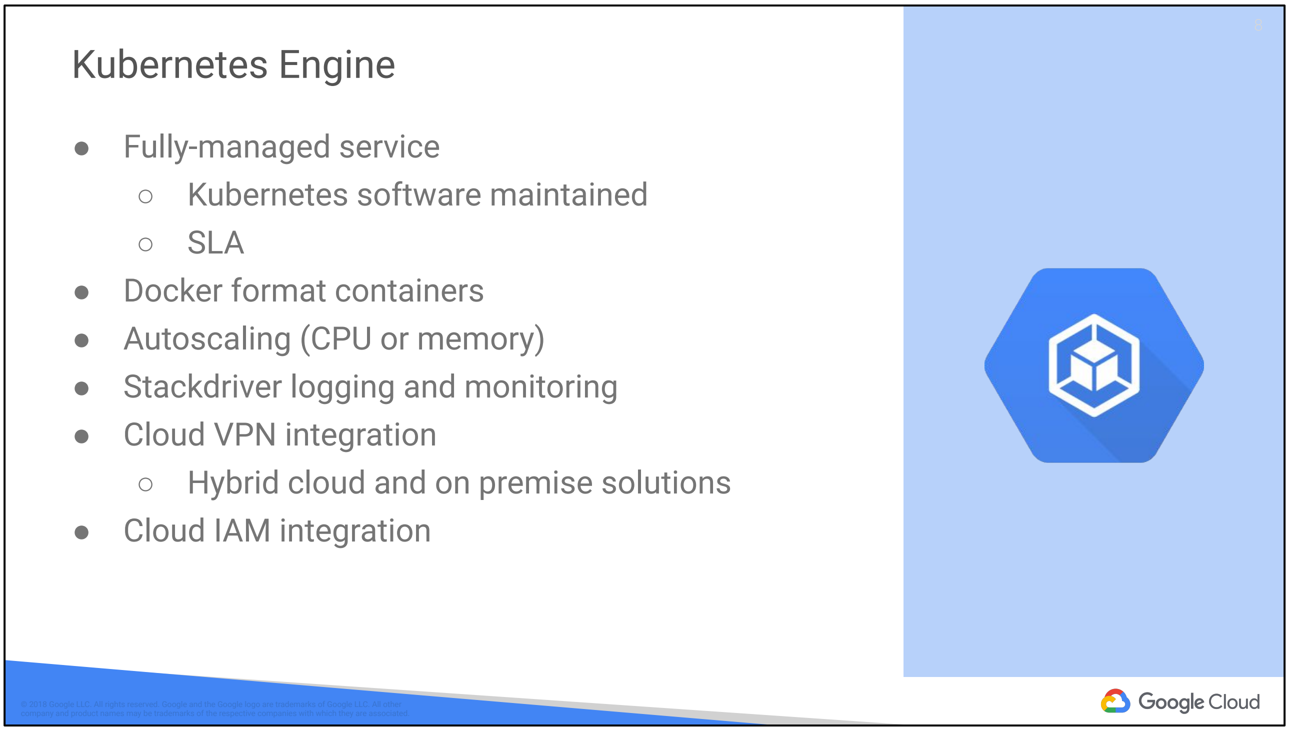


Kubernetes (a.k.a "k8s")

- 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 Kubernetes Engine in this class*
- More information
    - kubernetes.io (*also*, k8s.io)

Google Cloud

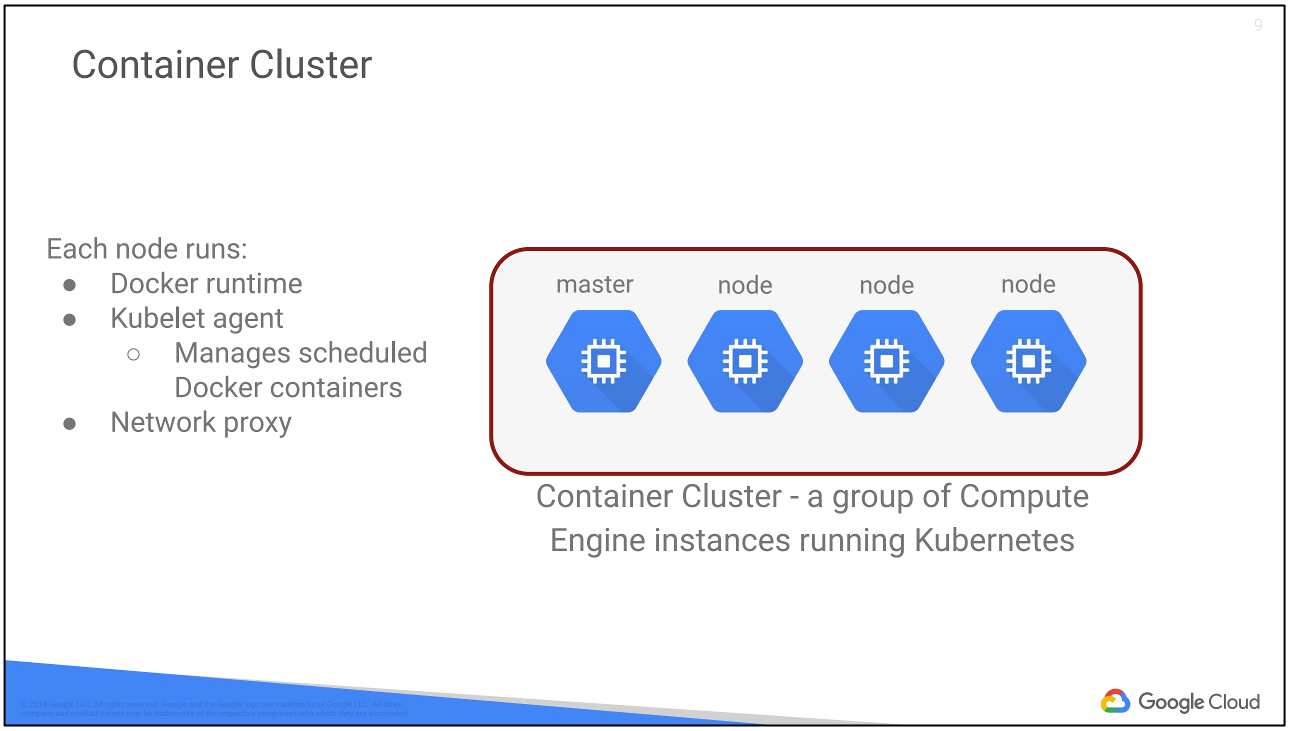
https://kubernetes.io/
Kubernetes is greek for "helmsman" or "pilot". Projects started in 2014. Based on experience with Google's internal container management system.



Kubernetes Engine

- 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

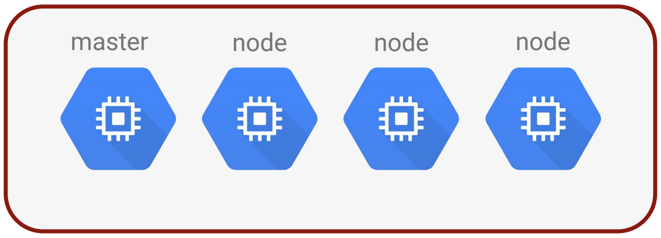
Cloud VPN integration: Reserve an IP address range for your container cluster, allowing your cluster IPs to coexist with private network IPs via Google Cloud VPN.



Container Cluster

Each node runs:

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

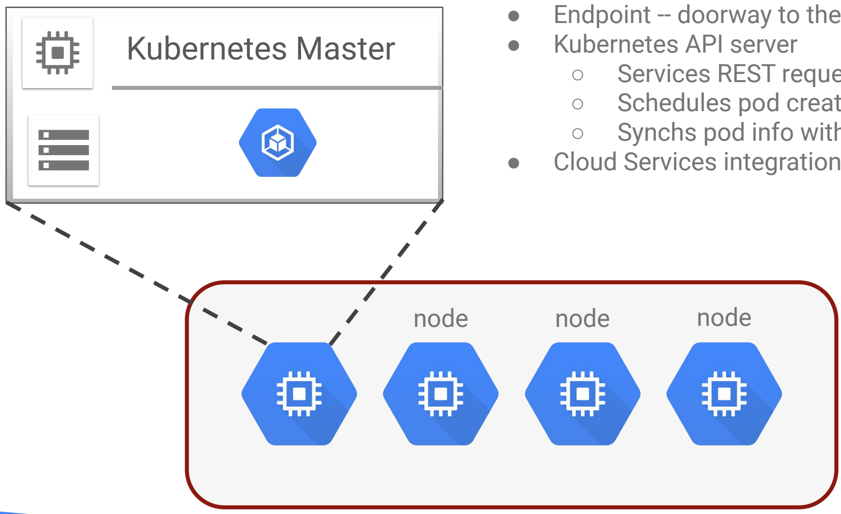


Container Cluster - a group of Compute Engine instances running Kubernetes

Google Cloud

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

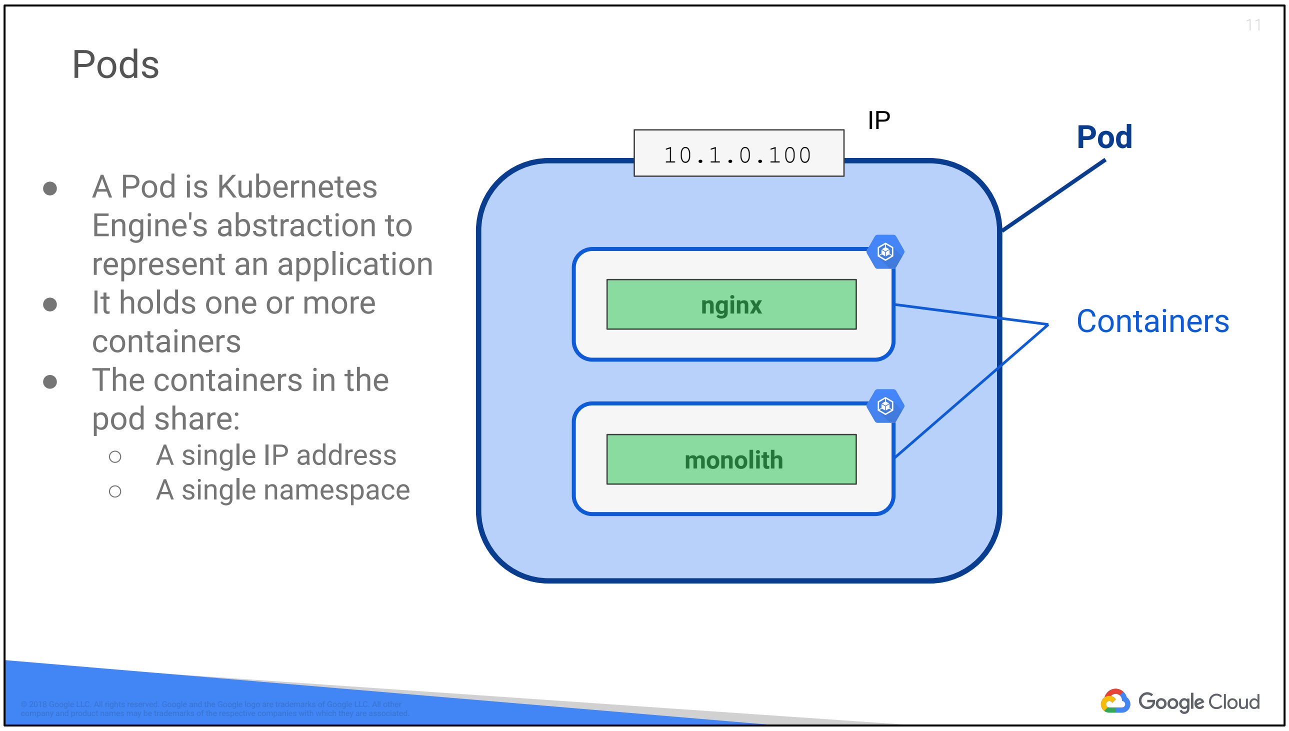
Kubernetes master endpoint



- 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

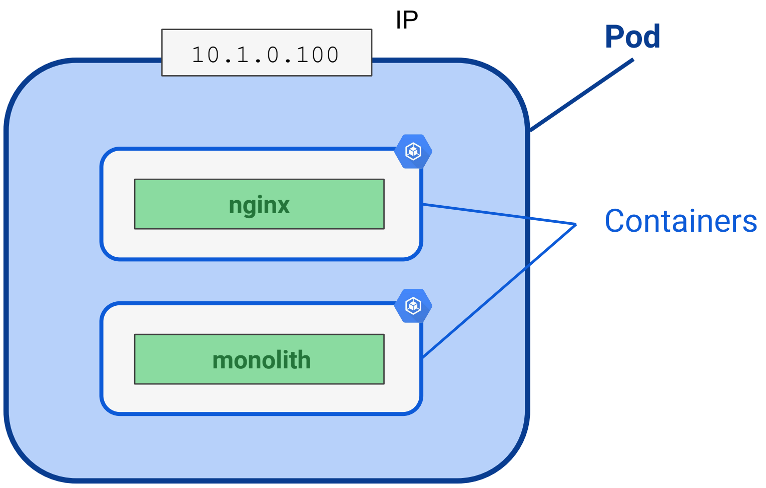
Google Cloud

Now that you understand the physical relationship between Compute Engine 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.

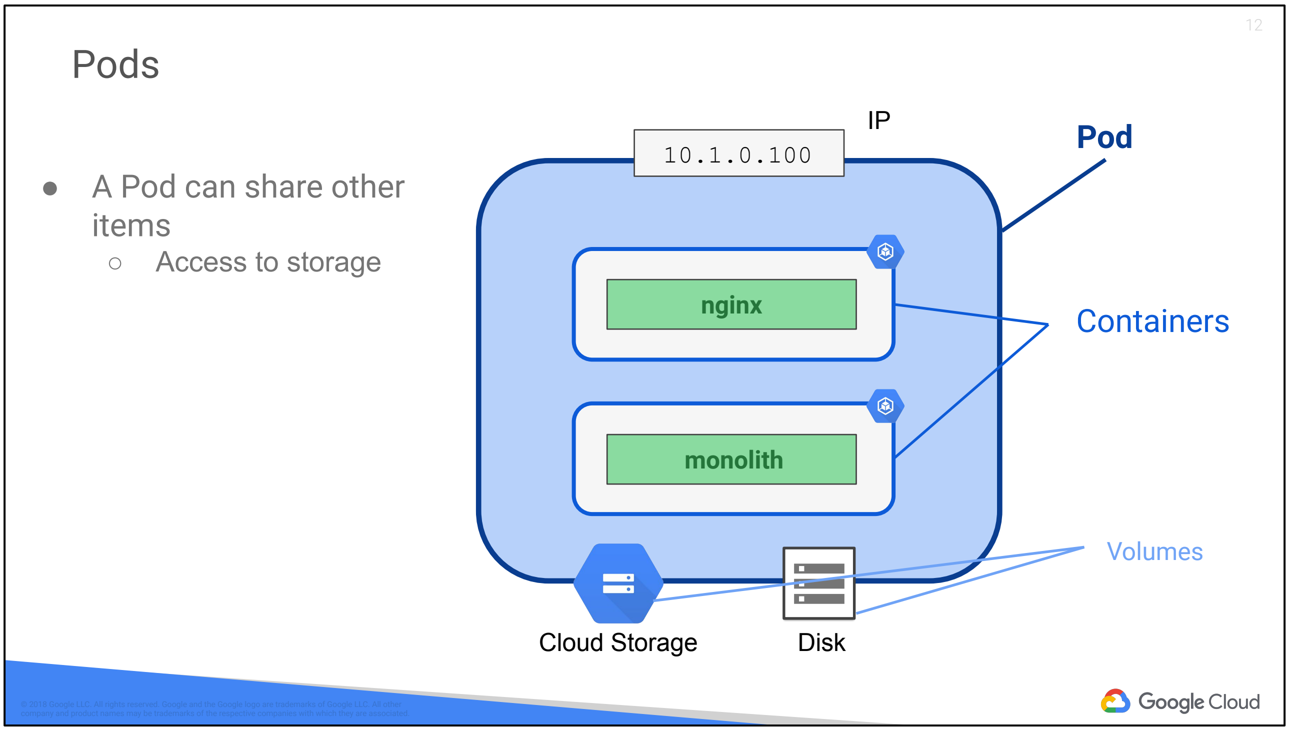


Pods

- A Pod is Kubernetes Engine'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

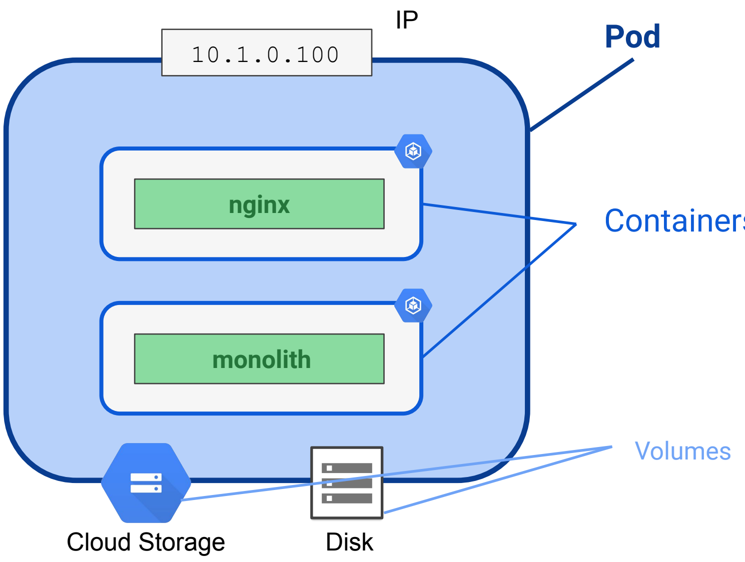


One purpose of Kubernetes Engine is to enable you to manage applications, not machines.
To accomplish this, you need to understand the Kubernetes Engine abstractions for applications.



Pods

- A Pod can share other items
    - Access to storage



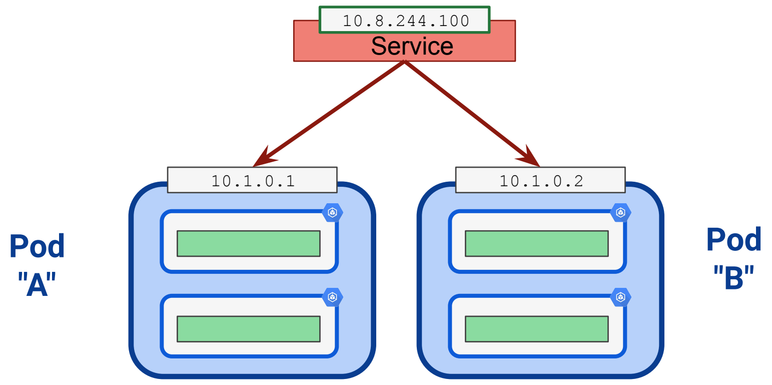
Any data access mounted to a pod, called a Volume, is available to all containers in the pod.
Containers that are part of the same pod are guaranteed to be scheduled together on the same VM and can share state via local volumes.
Persistent Volumes, using persistent disks in Compute Engine, survive instance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

Kubernetes Engine Labels

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

Google Cloud

Kubernetes Engine Service



A service provides a persistent internal or external IP for pods

Google Cloud

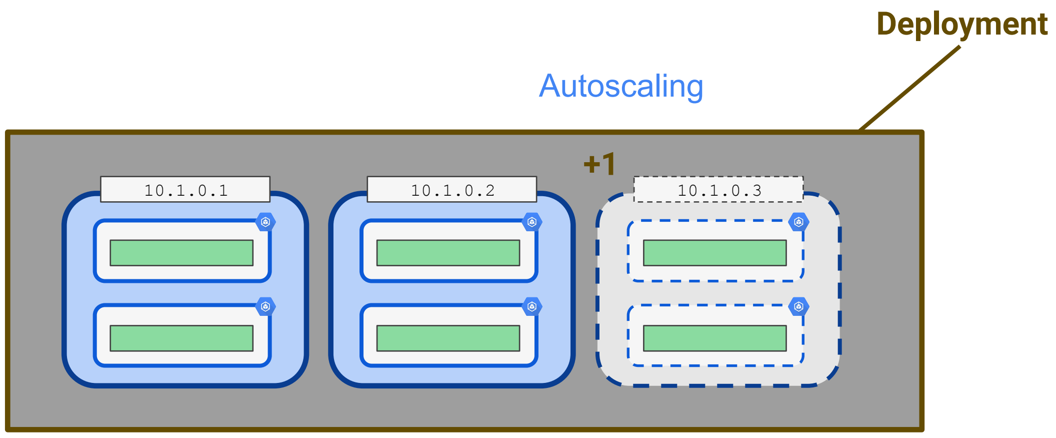
Here is an example to explain the Kubernetes Engine Service concept.

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, the pod'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.

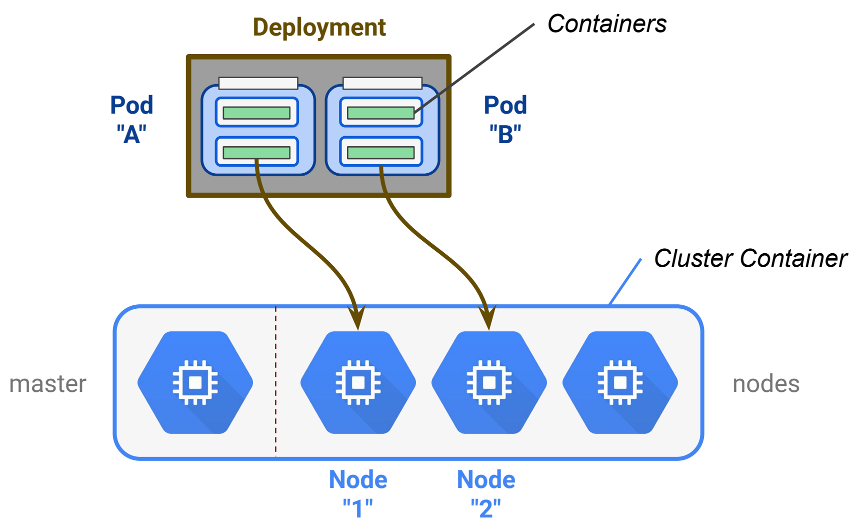
Kubernetes Engine Deployment



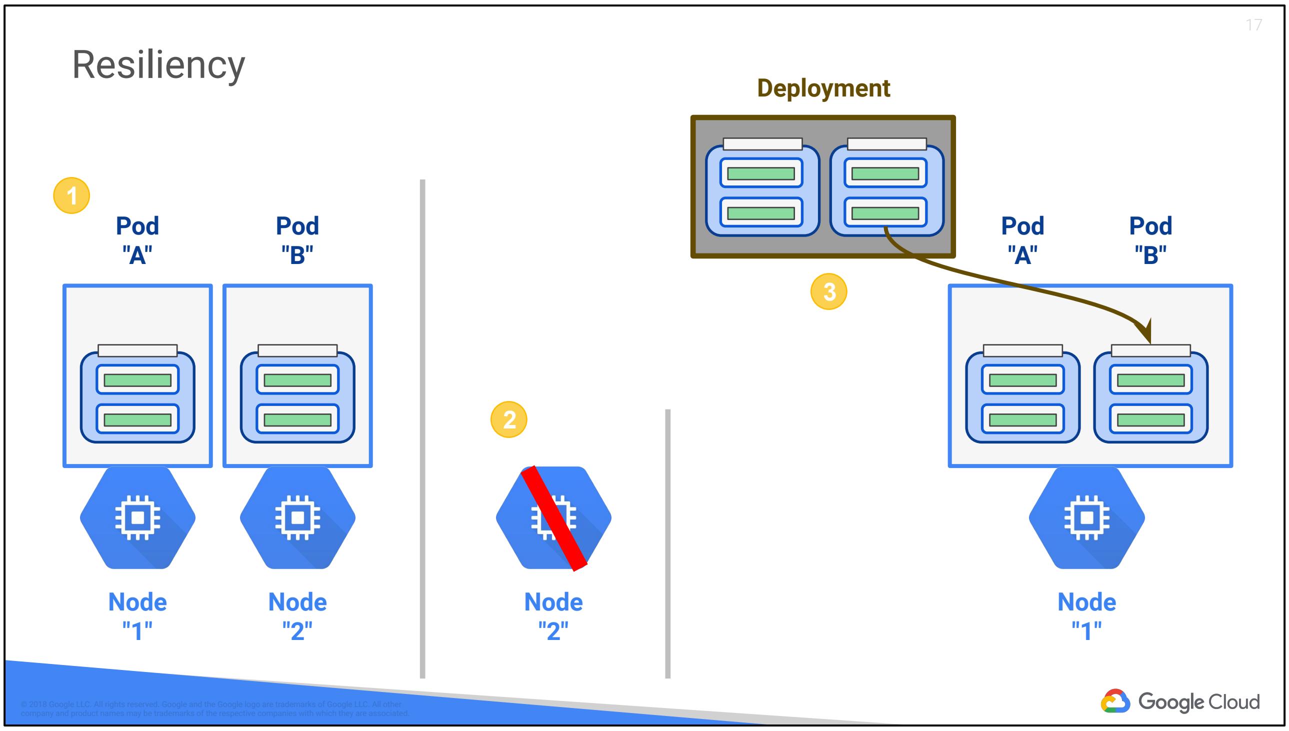
Google Cloud

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 Compute Engine.

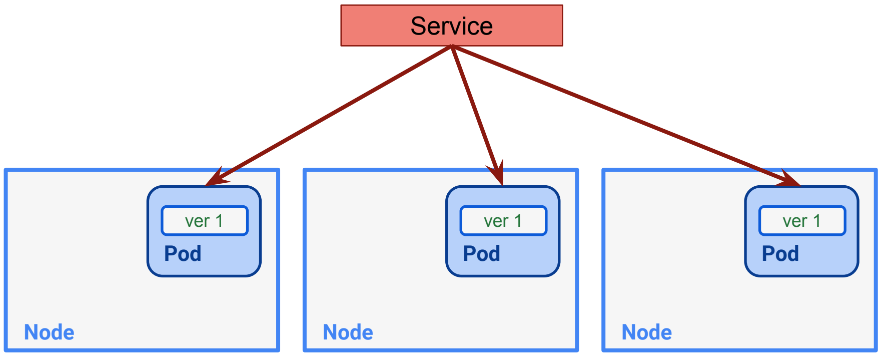


Resiliency

(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

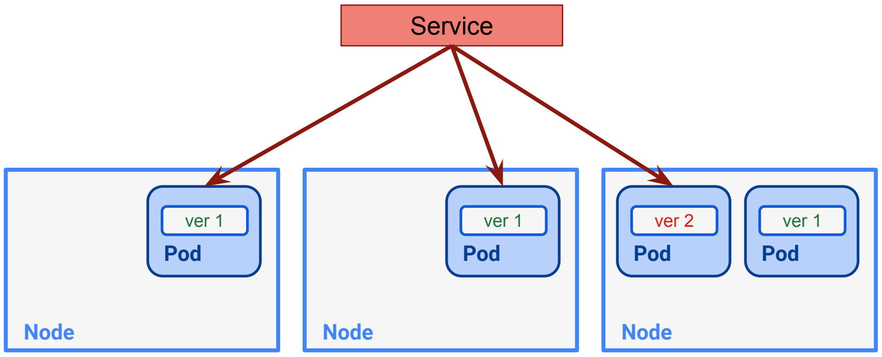
Deployments always drive the overall system towards the current state, using the resources available in the cluster.

Rolling Update (1 of 3)



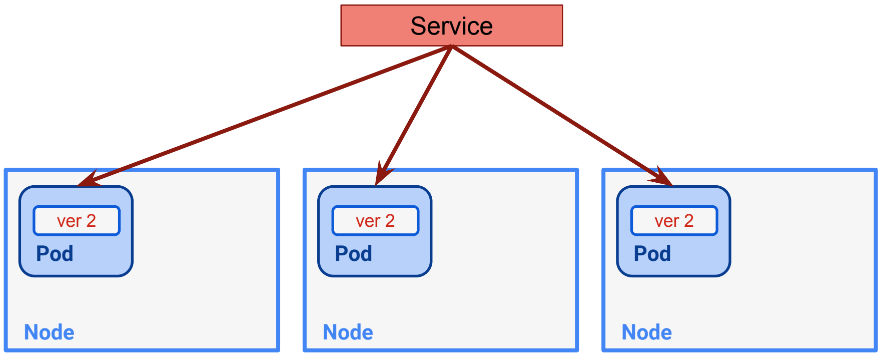
Version 1 of the software is running on three nodes.
Deployment is making sure we keep three pods running.
A service is shown so you can see the IP changes as well as the node changes.
The goal here is to deploy version 2 of the software, using the Kubernetes Engine Rolling Update feature.

Rolling Update (2 of 3)



A second pod is launched on Node 3 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)

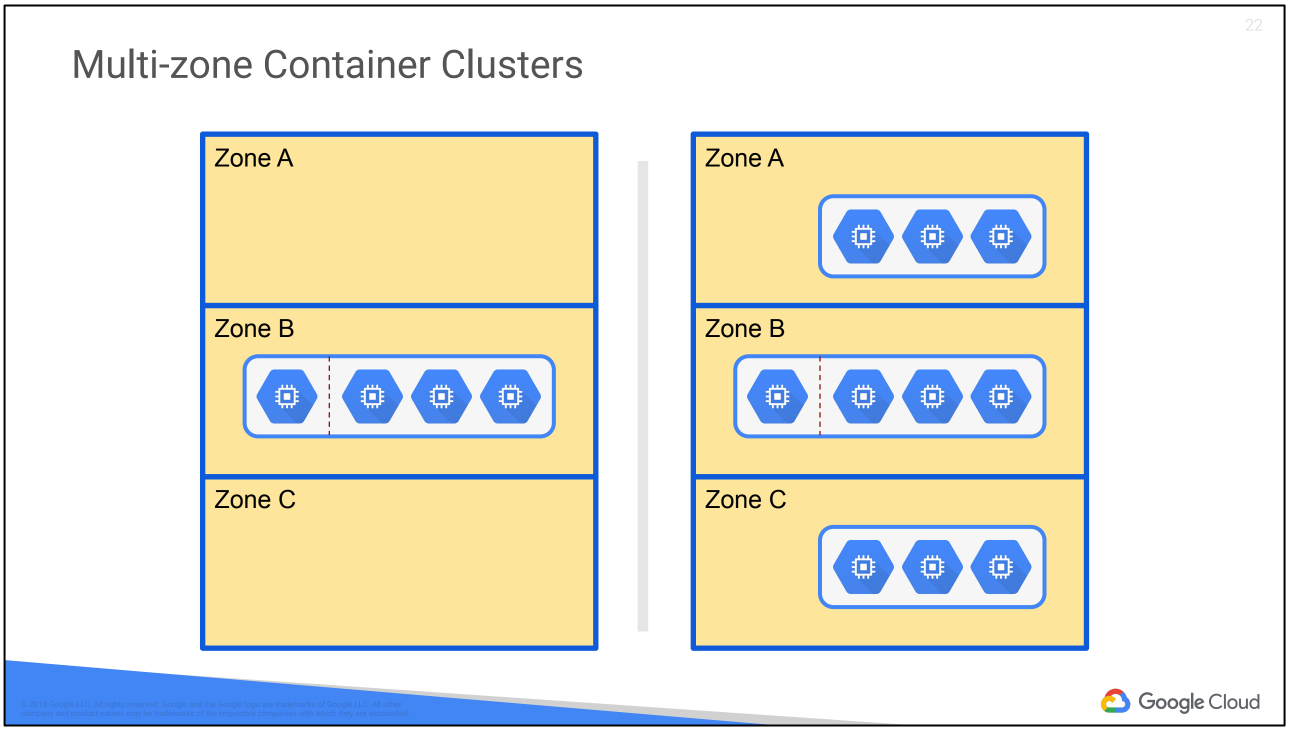


The process continues until all version 1 nodes have been replaced, one at a time with version 2 pods.

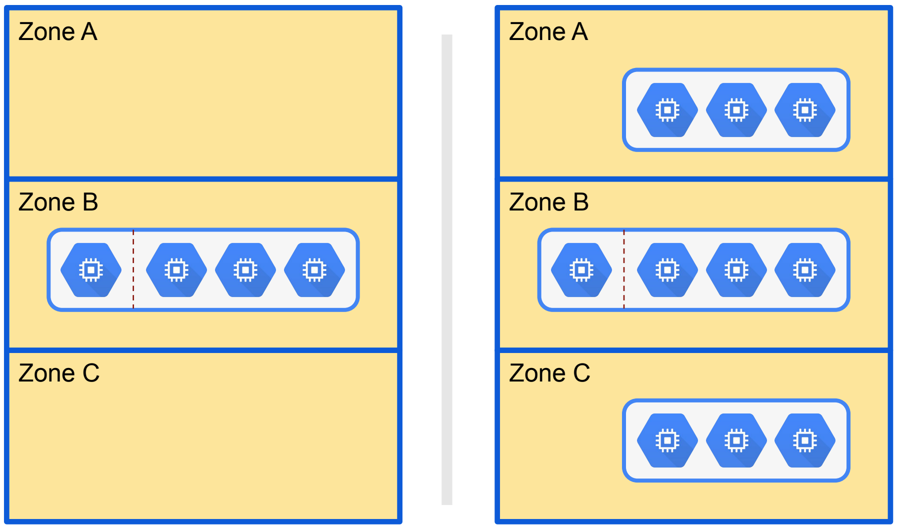
IAM support

Roles

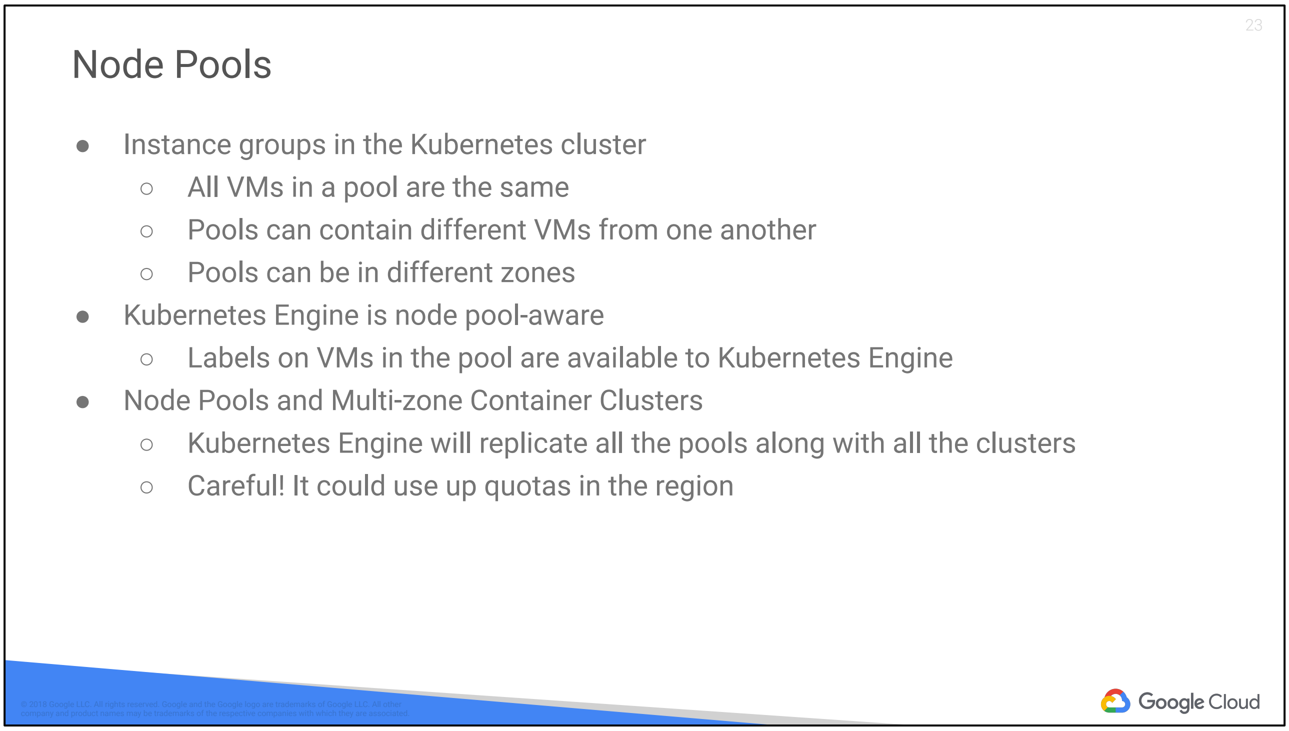
- `container.admin`
  - Full management of Container Clusters and their Kubernetes API objects
- `container.clusterAdmin`
  - Management of Cluster Containers
- `container.developer`
  - Full access to Kubernetes API objects inside Container Clusters
- `container.viewer`
  - Read-only access to Kubernetes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



Multi-zone Container Clusters



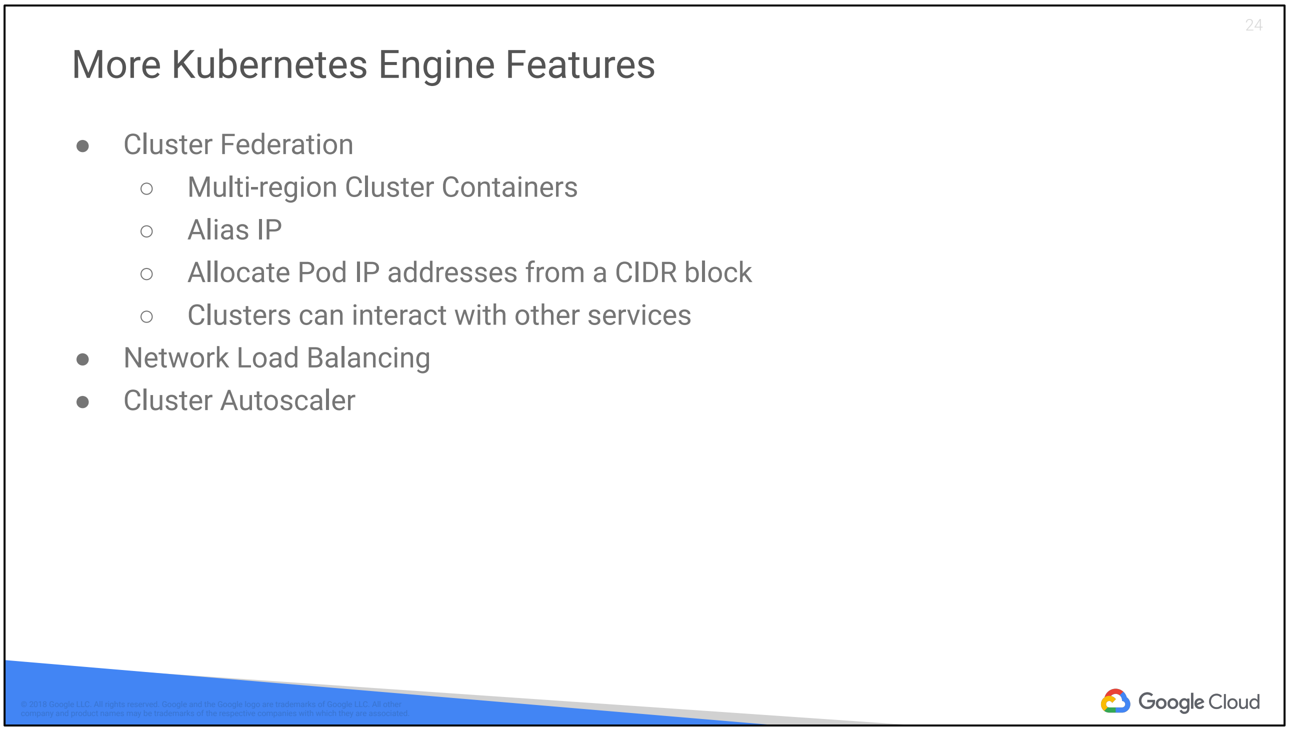
When you enable Multi-zone Container Clusters, the container resources are replicated in the additional zones, and work is scheduled across all of them.
If one zone fails, the others can pick up the slack. In this case, any single zone is capable of running the entire application.
All of the zones are 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
- Kubernetes Engine is node pool-aware
    - Labels on VMs in the pool are available to Kubernetes Engine
- Node Pools and Multi-zone Container Clusters
    - Kubernetes Engine will replicate all the pools along with all the clusters
    - Careful! It could use up quotas in the region

Google Cloud



More Kubernetes Engine Features

- Cluster Federation
  - Multi-region Cluster Containers
  - Alias IP
  - Allocate Pod IP addresses from a CIDR block
  - Clusters can interact with other services
- Network Load Balancing
- Cluster Autoscaler

Google Cloud

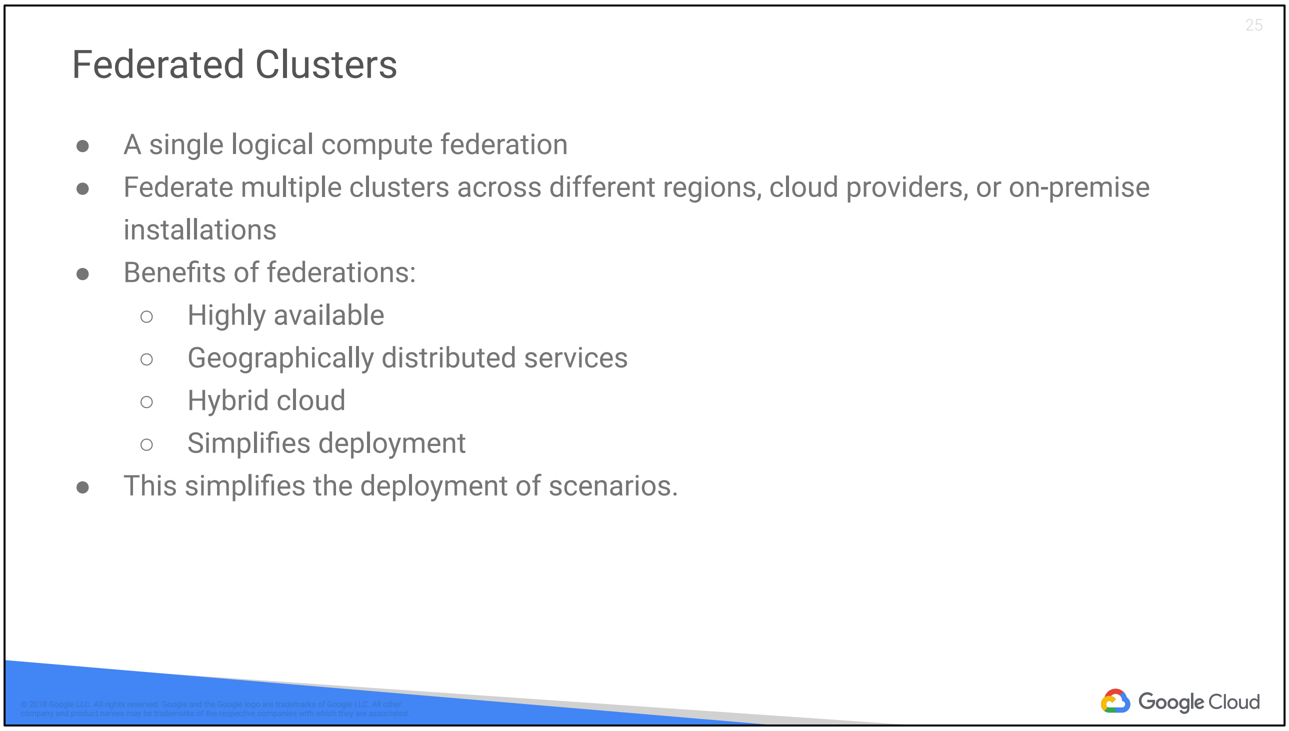
Cluster Federation
Cluster Federation -- Enables clusters across multiple regions, including other cloud providers or on premise Kubernetes installations.
Useful for super high availability or for super scalability.
https://cloud.google.com/kubernetes-engine/docs/cluster-federation

With IP aliases, Kubernetes Engine clusters can allocate Pod IP addresses from a CIDR block known to Google Cloud Platform. This allows your cluster to interact with other Cloud Platform products and entities, and also allows more scalable clusters.
More about IP Aliases:
https://cloud.google.com/kubernetes-engine/docs/ip-aliases

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

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

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



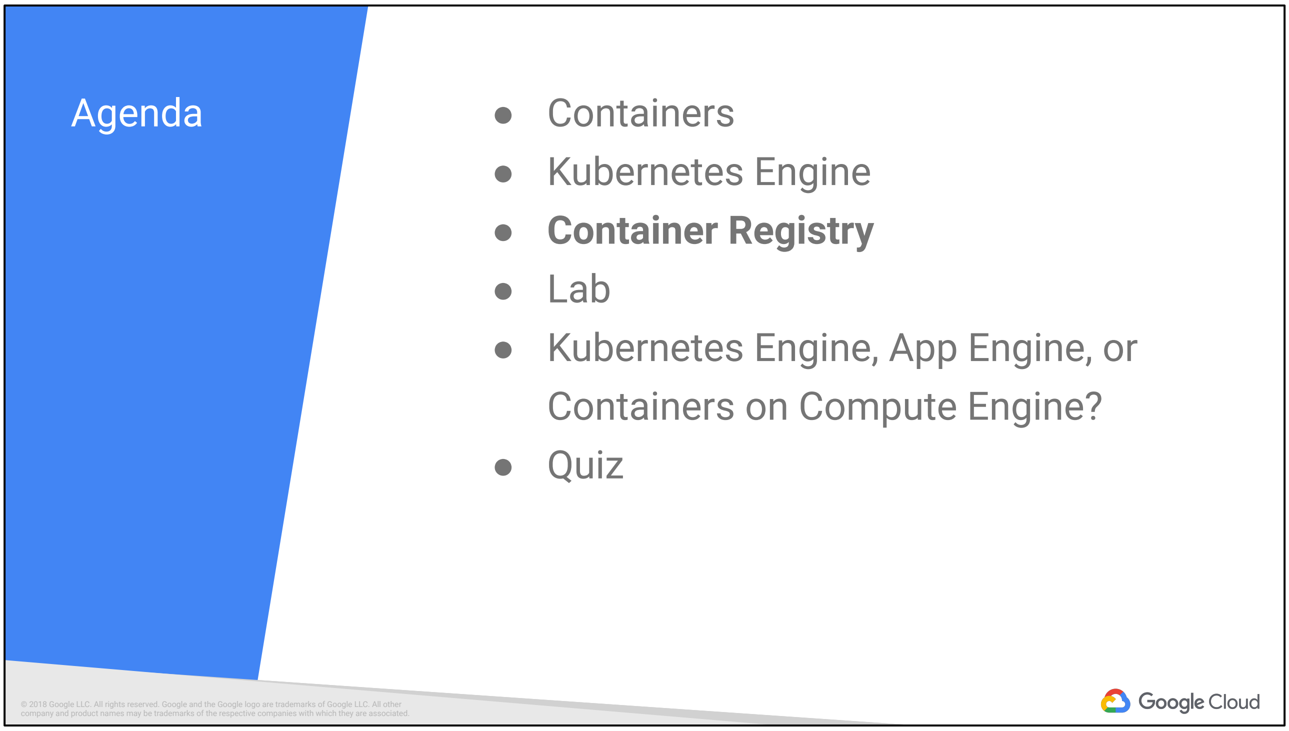
Federated Clusters

- A single logical compute federation
- 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/kubernetes-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.

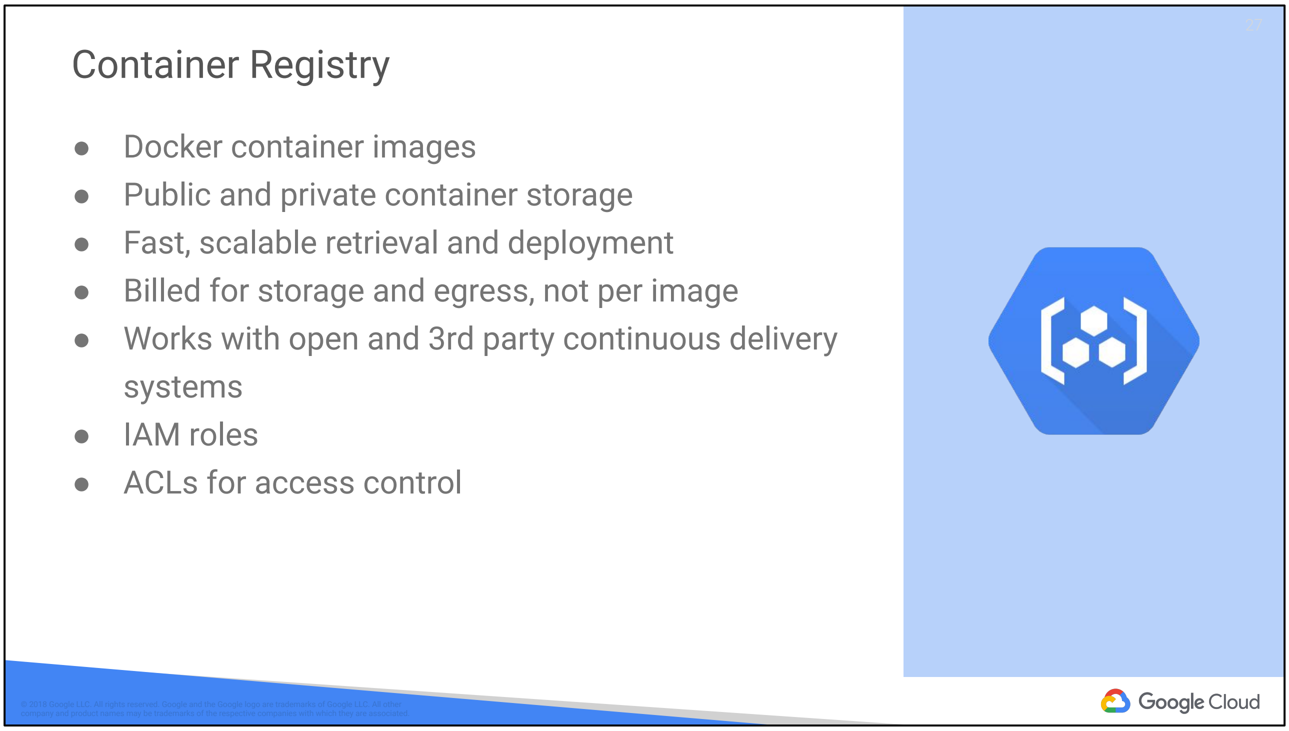
Cluster Federation is also helpful when you want resources to be contactable in a consistent manner from both inside and outside their clusters, without incurring unnecessary latency or bandwidth cost penalties, or being susceptible to individual cluster outages.



Agenda

- Containers
- Kubernetes Engine
- **Container Registry**
- Lab
- Kubernetes Engine, App Engine, or Containers on Compute Engine?
- Quiz

Google Cloud



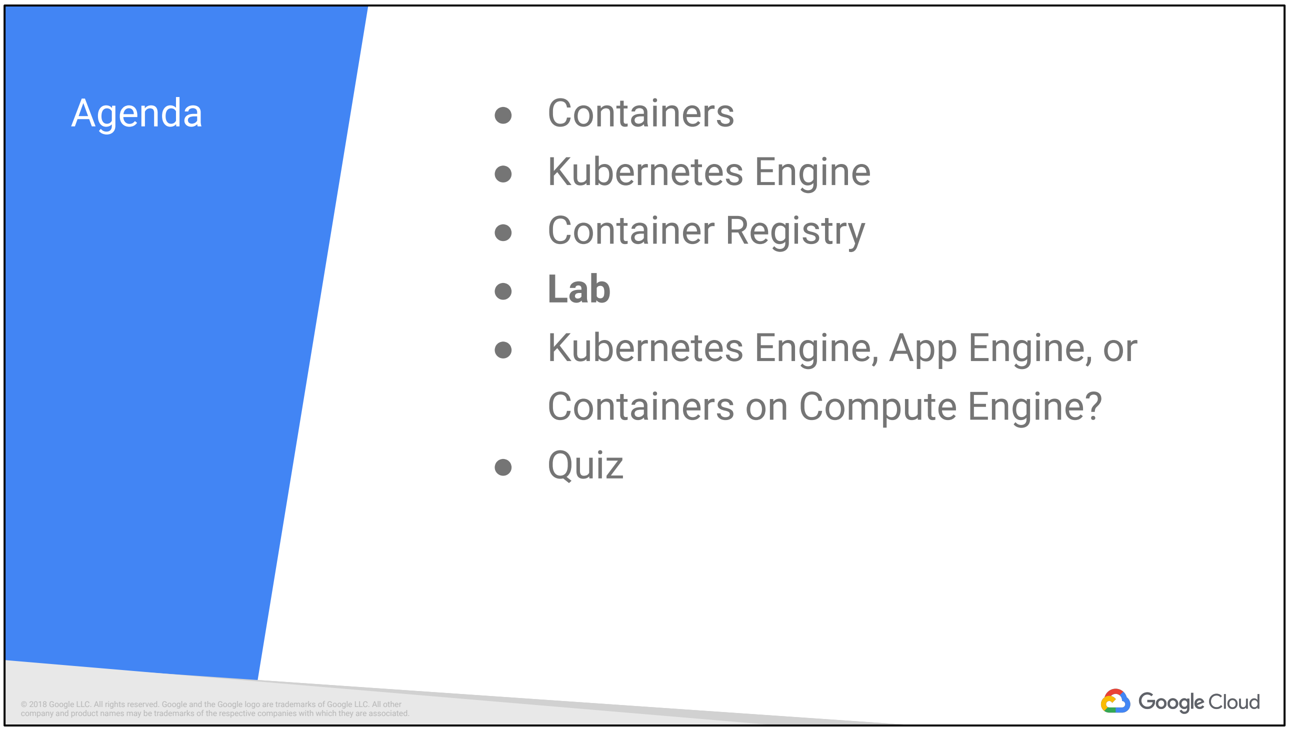
Container Registry

- 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 Cloud

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 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 that machine is a Google Compute Engine instance or your own hardware.



Agenda

- Containers
- Kubernetes Engine
- Container Registry
- **Lab**
- Kubernetes Engine, App Engine, or Containers on Compute Engine?
- Quiz

Google Cloud

Lab: Kubernetes Load Balancing

Objectives

In this lab, you learn how to perform the following tasks:

- Create a Kubernetes cluster using Google Kubernetes Engine with the built-in network load balancer.
- Deploy nginx into the cluster and verify that the application is working.
- Undeploy the application.
- Re-deploy the cluster using ingress to connect it to a Google Compute Engine HTTP(s) load balancer.
- Redeploy and test.

Completion: 60 minutes

Access: 120 minutes

Kubernetes Engine

kubernetes

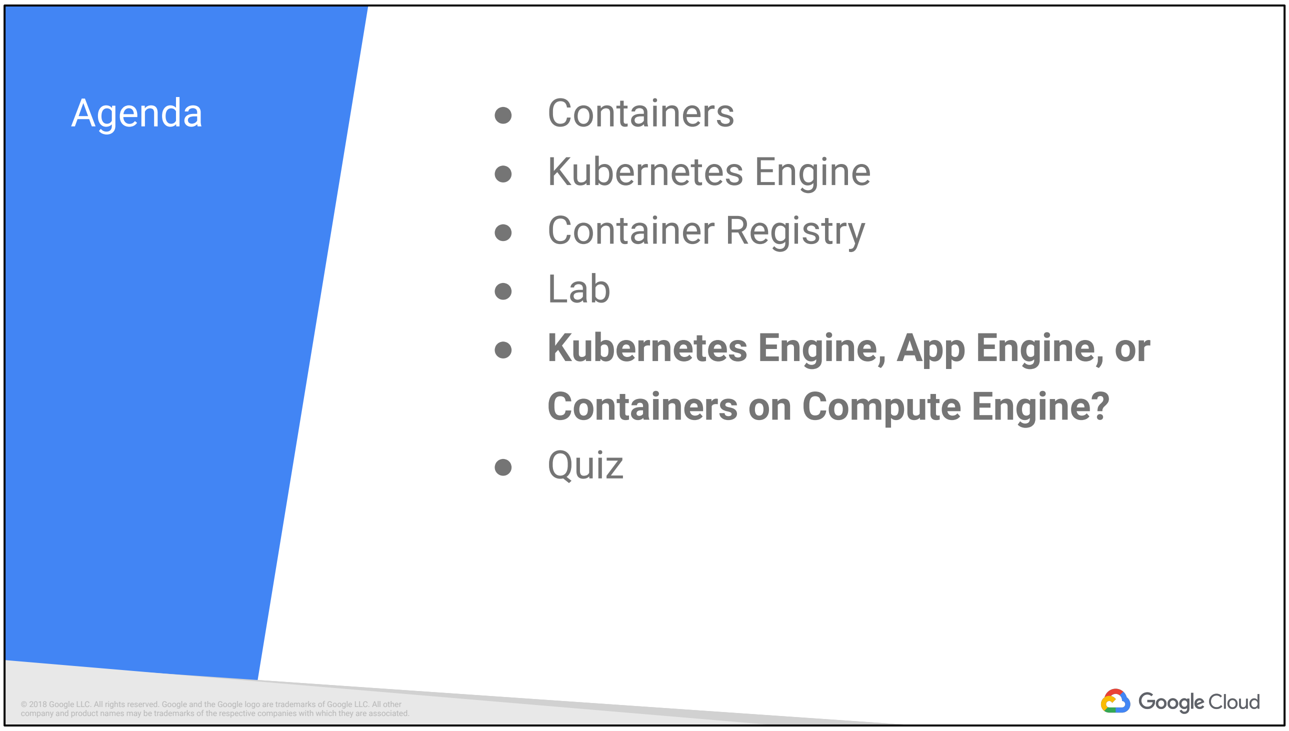
Google Cloud

Lab Review

In this lab, you:

- Deployed a Container application using Kubernetes and Google Kubernetes Engine.
- Tested the configuration of your application.
- Undeployed it and then re-deployed it behind a Google Compute Engine HTTP(S) Load Balancer using the Ingress extension to Kubernetes.

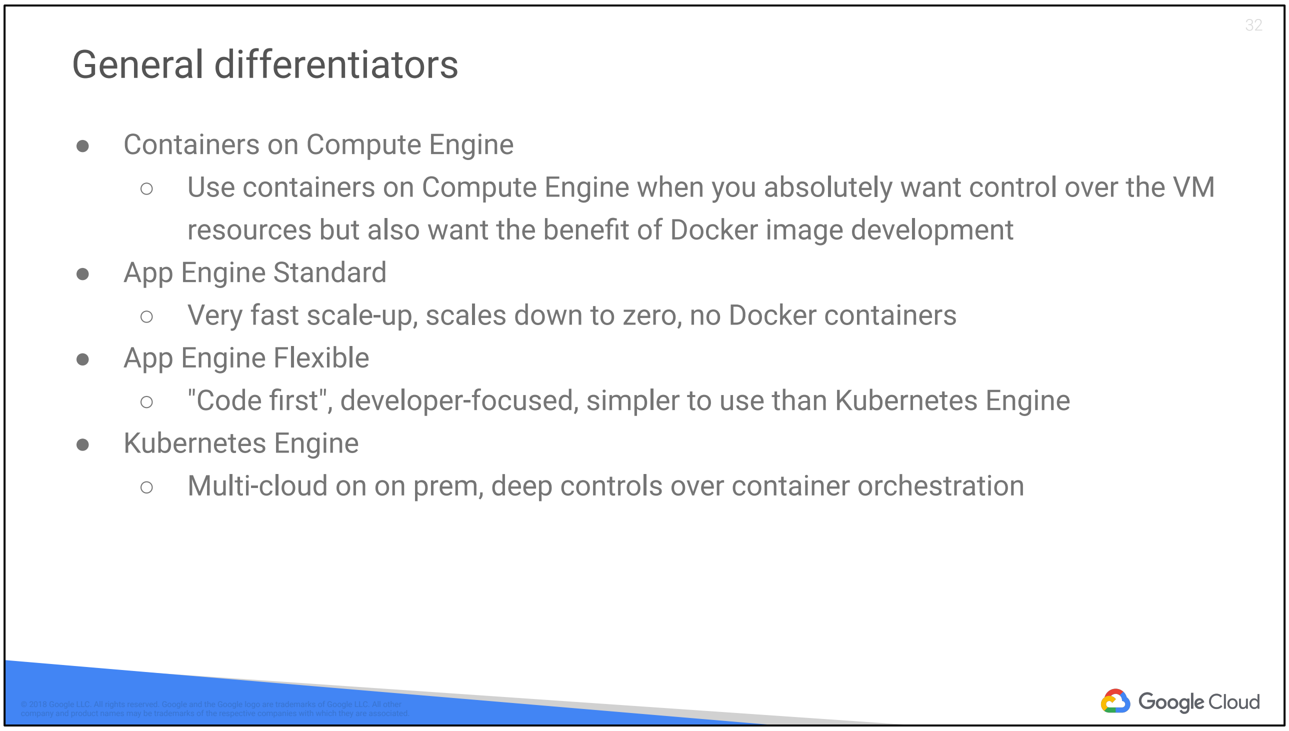
Google Cloud



Agenda

- Containers
- Kubernetes Engine
- Container Registry
- Lab
- **Kubernetes Engine, App Engine, or Containers on Compute Engine?**
- Quiz

Google Cloud



General differentiators

- Containers on Compute Engine
  - Use containers on Compute Engine when you absolutely want control over the VM resources but also want the benefit of Docker image development
- App Engine Standard
  - Very fast scale-up, scales down to zero, no Docker containers
- App Engine Flexible
  - "Code first", developer-focused, simpler to use than Kubernetes Engine
- Kubernetes Engine
  - Multi-cloud on on prem, deep controls over container orchestration

Google Cloud

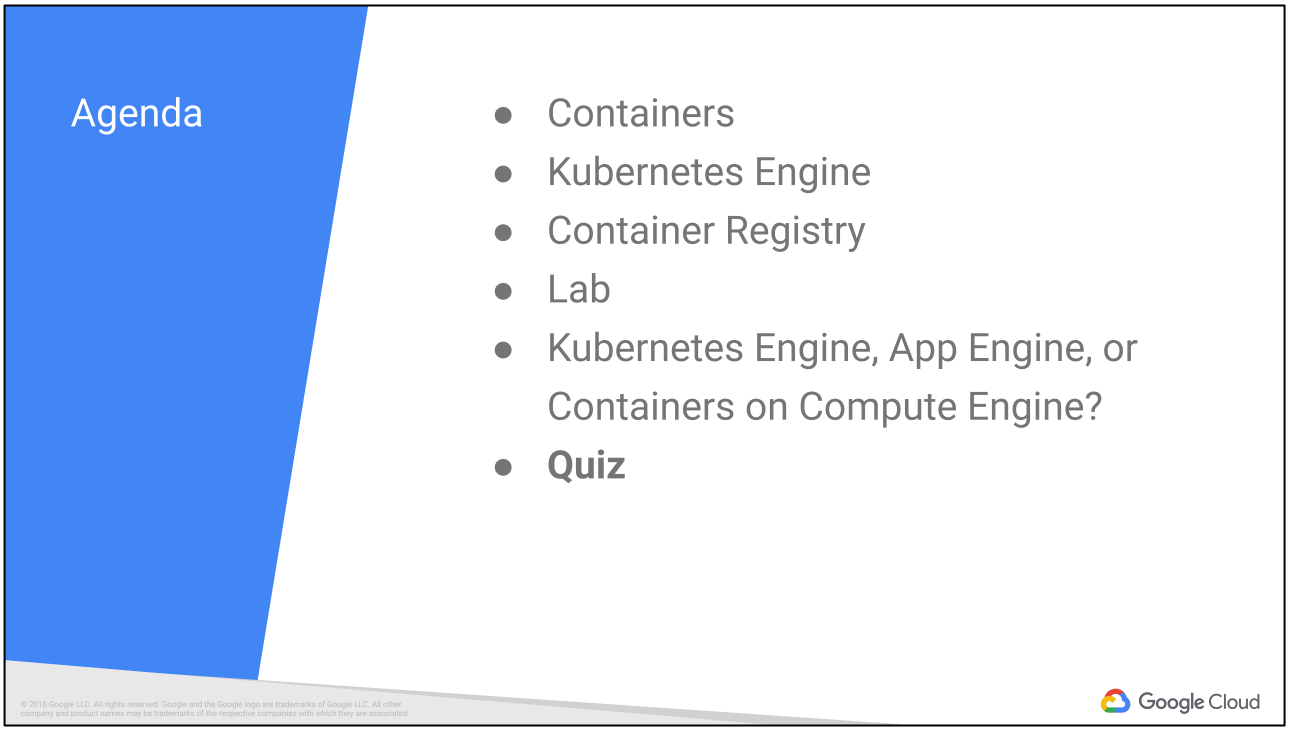
You should think of App Engine Flex as a developer focused, no operations compute infrastructure offering that abstracts the underlying infrastructure to a large extent in favor of developer friendliness. Flex runs docker containers in VMs under the hood, while providing convenient models of deployment and management.

Flex provides an experience that makes it easy to take your code, and directly deploy it to a production scale environment. It takes care of capturing logs, scaling, versioned upgrades, and traffic splitting, without the developer having to be aware of containers, config or settings.

If your organization is already familiar with Docker, and you want to run containerized workloads in production spread across pools of VMs - Kubernetes Engine may be a good fit. It provides deeper awareness of the underlying infrastructure than App Engine Flex, and may require an understanding of container orchestration concepts. Kubernetes Engine is also a great choice if you plan to run Kubernetes on premises or beyond Google Cloud Platform.

The good news is that if you start with App Engine Flexible and then decide you want to move to Kubernetes Engine, we make it really easy to generate the Dockerfile used under the hood by App Engine, which can be built into an image run by Kubernetes.

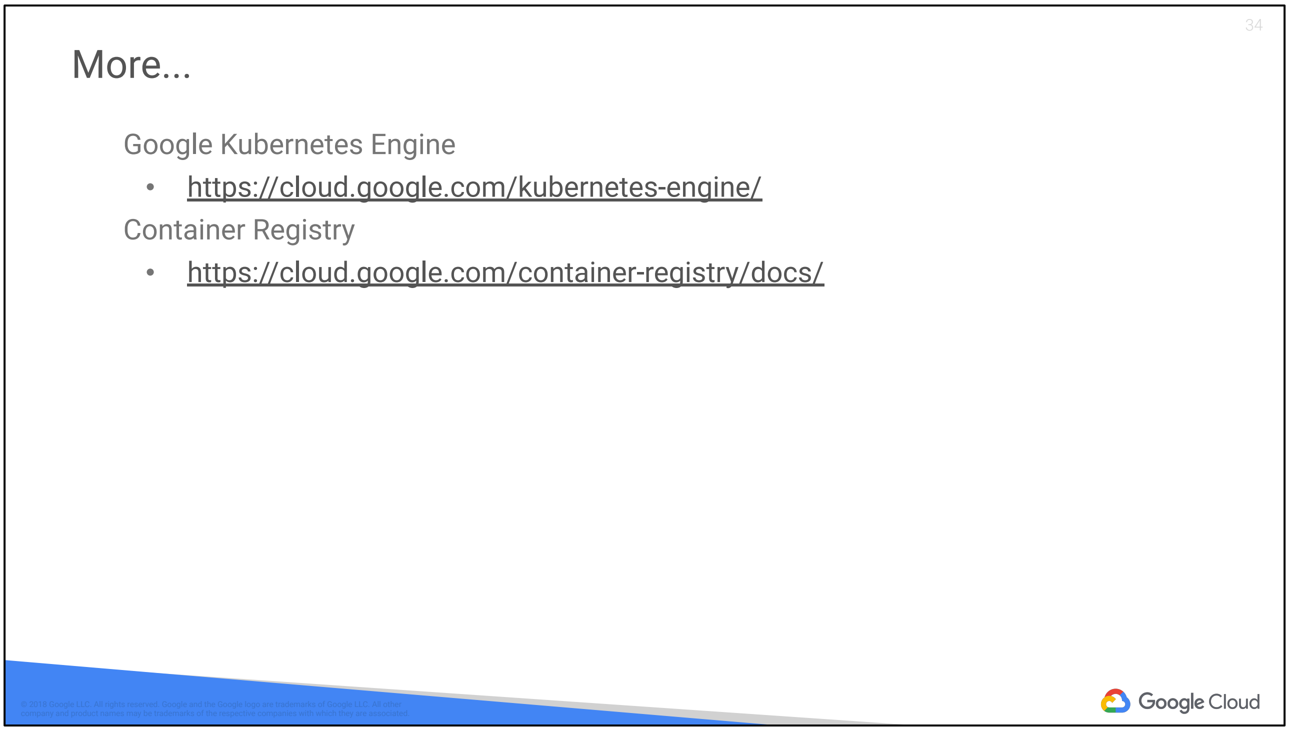
https://cloud.google.com/compute/docs/containers/
https://cloud.google.com/appengine/docs/flexible/
https://cloud.google.com/appengine/docs/the-appengine-environments



Agenda

- Containers
- Kubernetes Engine
- Container Registry
- Lab
- Kubernetes Engine, App Engine, or Containers on Compute Engine?
- **Quiz**

Google Cloud



More...

Google Kubernetes Engine

- https://cloud.google.com/kubernetes-engine/

Container Registry

- https://cloud.google.com/container-registry/docs/

Google Cloud

Google Cloud