# NetApp User Group

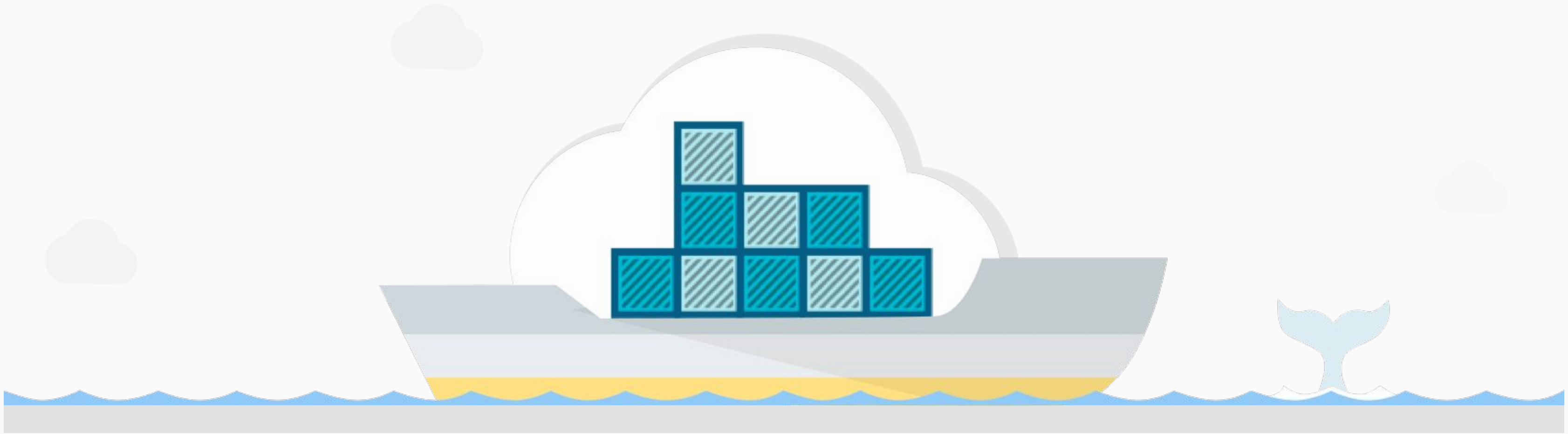
## Containers, Orchestration & NKS

Christopher Mende

# Containers

Logical Packaging Mechanism
Apps Abstracted from the Environment

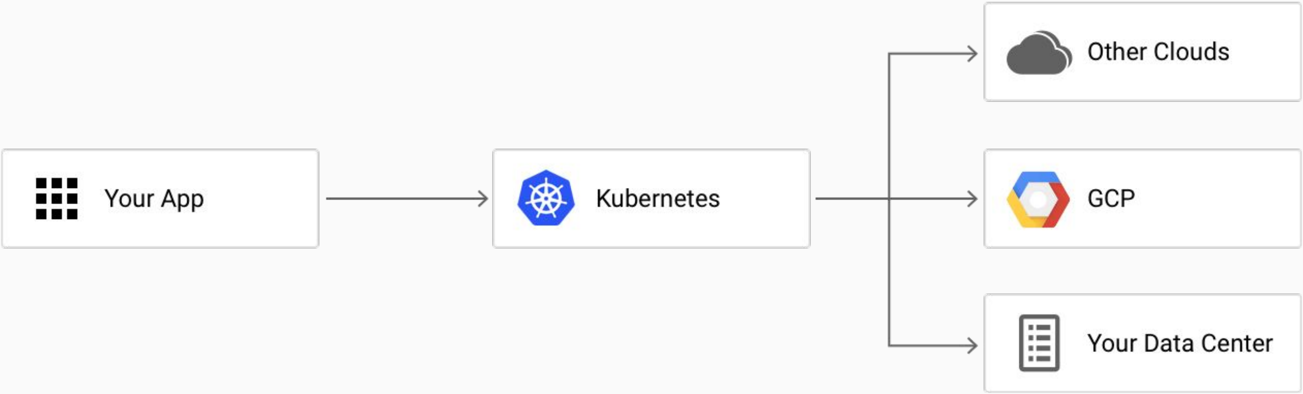
Separation of concerns from Ops & Dev
Deploy anywhere - cloud or priv DC



# Kubernetes

Automated rollouts & rollbacks
Service health monitoring

Automatic scaling of services
Declarative management



# Kubernetes Breakdown

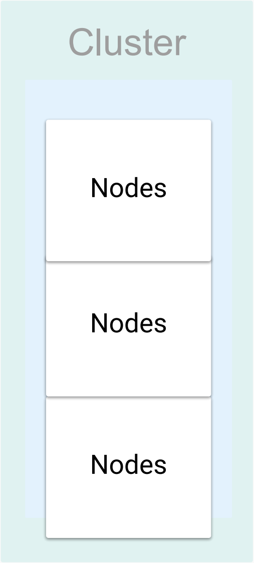
## Nodes

Nodes

Smallest unit of computing
Representation of a single machine in a cluster
Physical or Virtual

# Kubernetes Breakdown

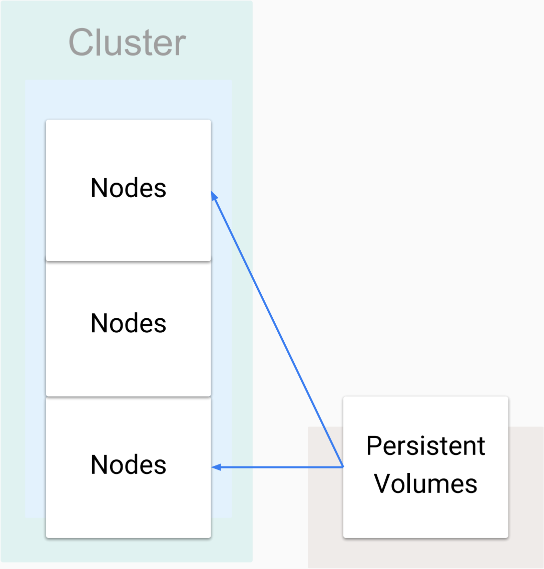
## Clusters



The BORG
A pool of independent nodes
Workloads are shifted as size changes
Working with clusters is the kubernetes way

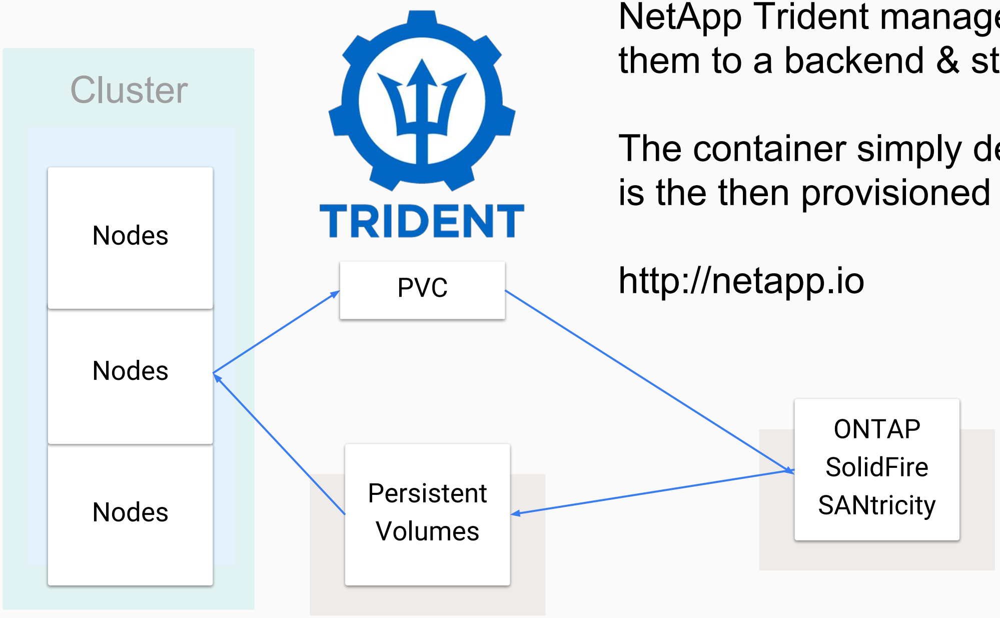
# Kubernetes Breakdown

## Persistent Volumes



Programs aren’t guaranteed to run on a specific node*
Local storage on a node is used as cache
Not all programs are ethereal, they require ‘state’
Persistent Volumes & Claims tackle this

## NetApp Trident



NetApp Trident manages PVCs and PVs by mapping them to a backend & storage class.

The container simply defines a storage type, and this is the then provisioned via NFS or iSCSI.

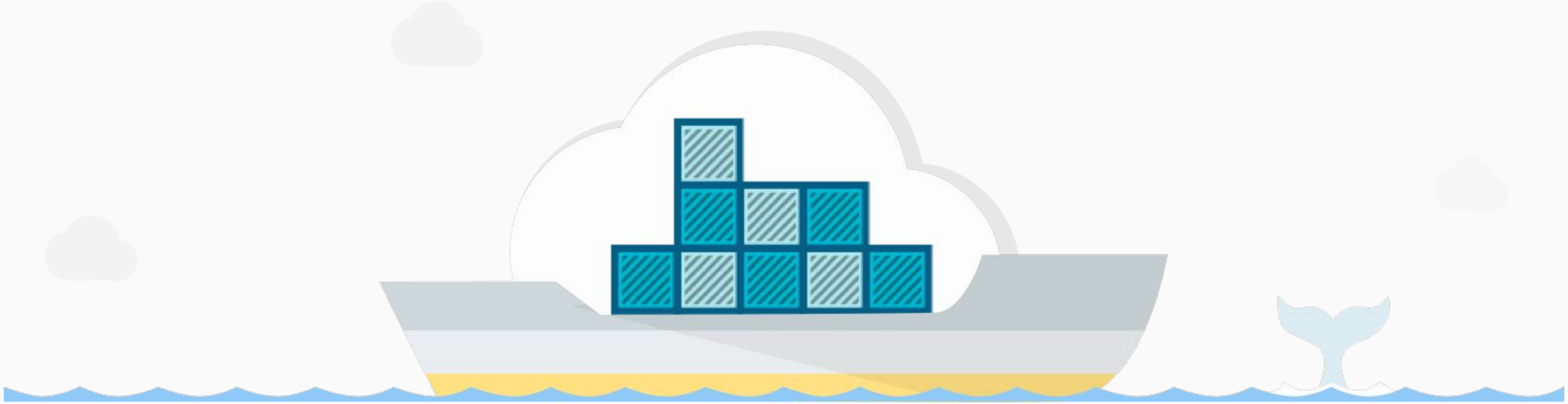
http://netapp.io

# Kubernetes Breakdown

## Containers!

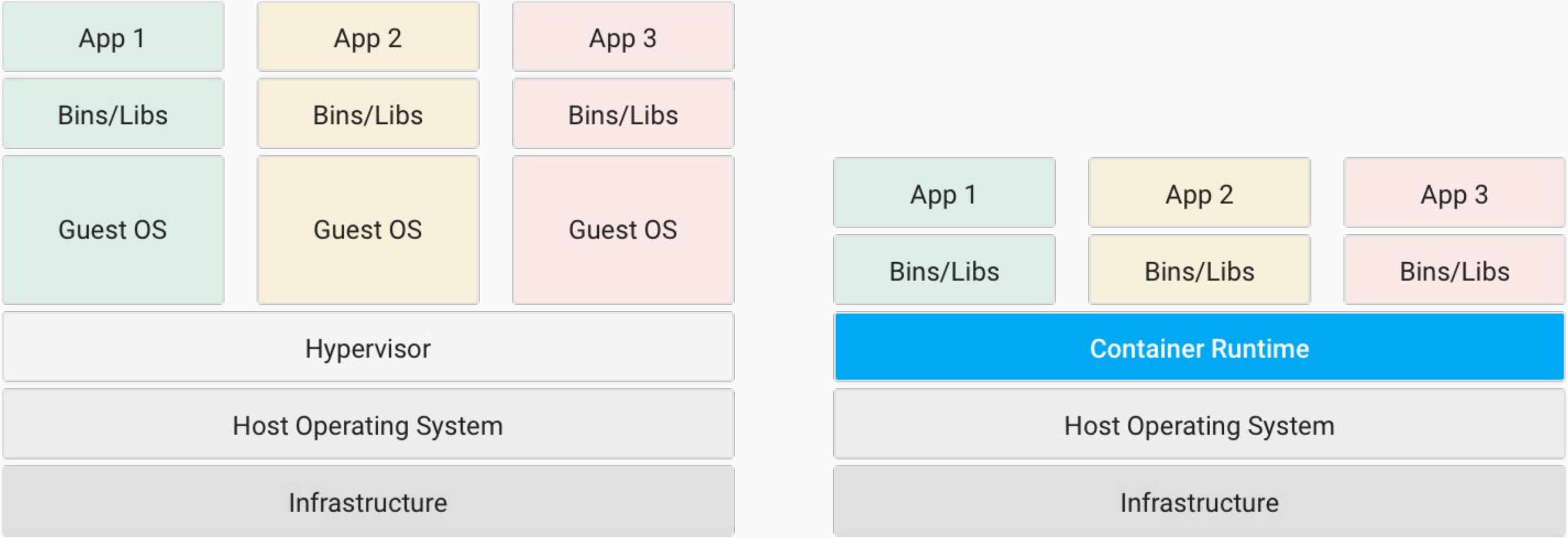
Logical Packaging Mechanism
Apps Abstracted from the Environment

Separation of concerns from Ops & Dev
Deploy anywhere - cloud or priv DC



# Kubernetes Breakdown

Containers & VMs are kinda the same, kinda not. Efficiency is the name of this game.



App 1 | App 2 | App 3

Bins/Libs | Bins/Libs | Bins/Libs

Guest OS | Guest OS | Guest OS

Hypervisor

Host Operating System

Infrastructure

**Virtual Machines**

App 1 | App 2 | App 3

Bins/Libs | Bins/Libs | Bins/Libs

Container Runtime

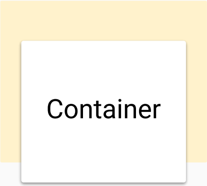
Host Operating System

Infrastructure

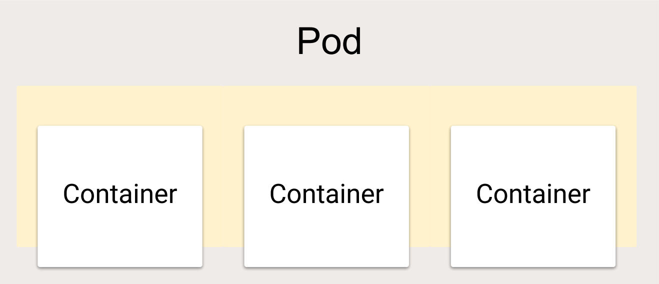
**Containers**

# Kubernetes Breakdown

## Programs & Containers



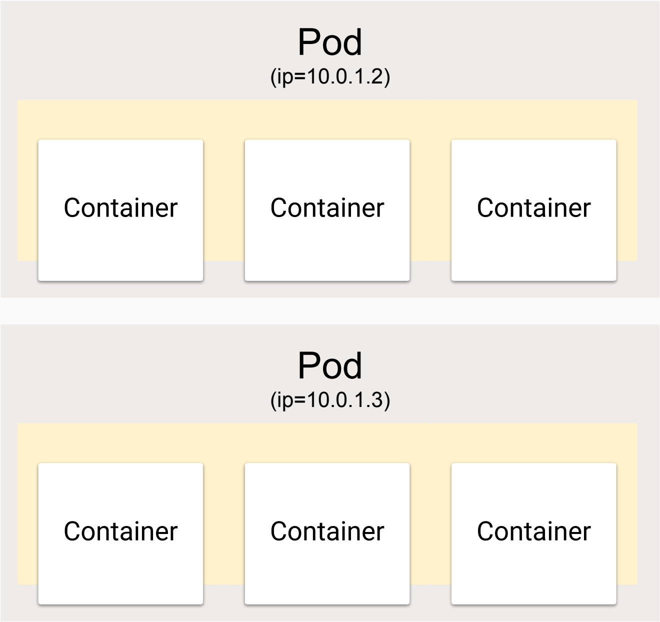
Programs are packaged as containers in kubernetes
You should limit to one process per container



In Kubernetes, containers aren’t run directly
Containers are grouped into pods
A pod shares same resources and local network

# Kubernetes Breakdown

## POD Details



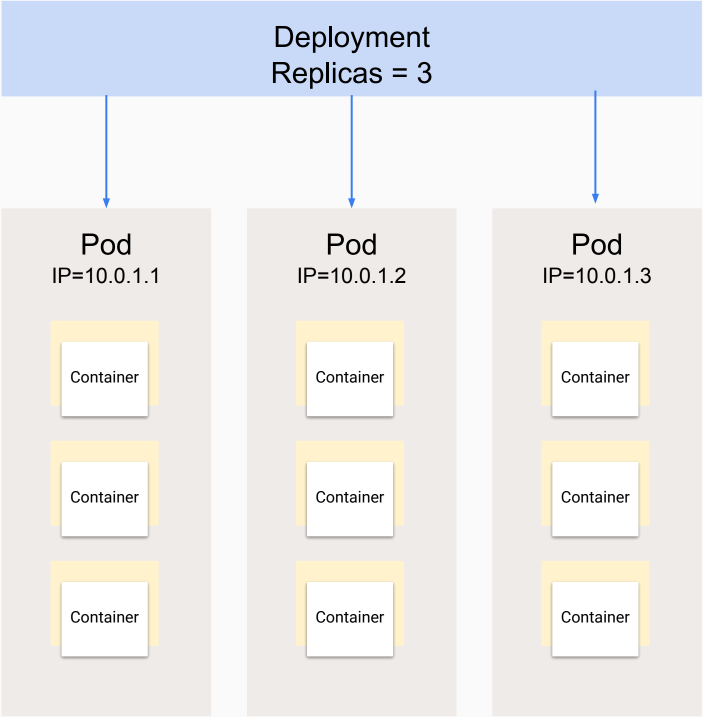
Pods contain multiple containers, but not too many!
Pods are the unit of “replication”
Pods run on a single node

Availability, performance & scale are handled by replicas.

Pods are also key to enabling one of the core benefits of microservices - ability to roll forward and back using multiple methods.

# Kubernetes Breakdown

## Deployments



Pods are also not run directly by kubernetes!

They are run via deployments, which specify how many replicas.

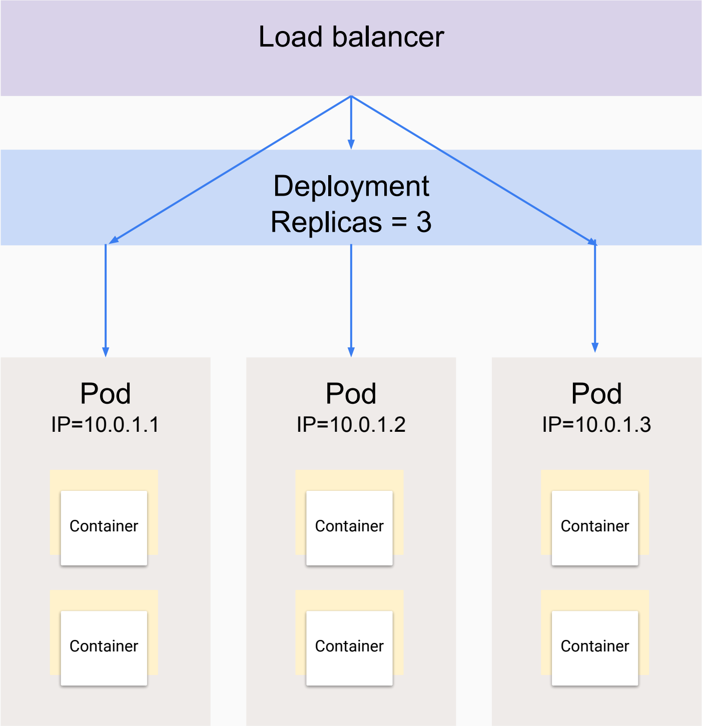
When a deployment is spun up, the number of replicas is specified.

Just declare what you want the environment to be & kubernetes takes care of it.

Yes, this handles issues like failed nodes, etc.

Kubernetes Breakdown

## Letting Traffic In: Ingress



Ingress is the management of which ports are available to the outside world.

By default Pods are isolated from each other and the outside world.

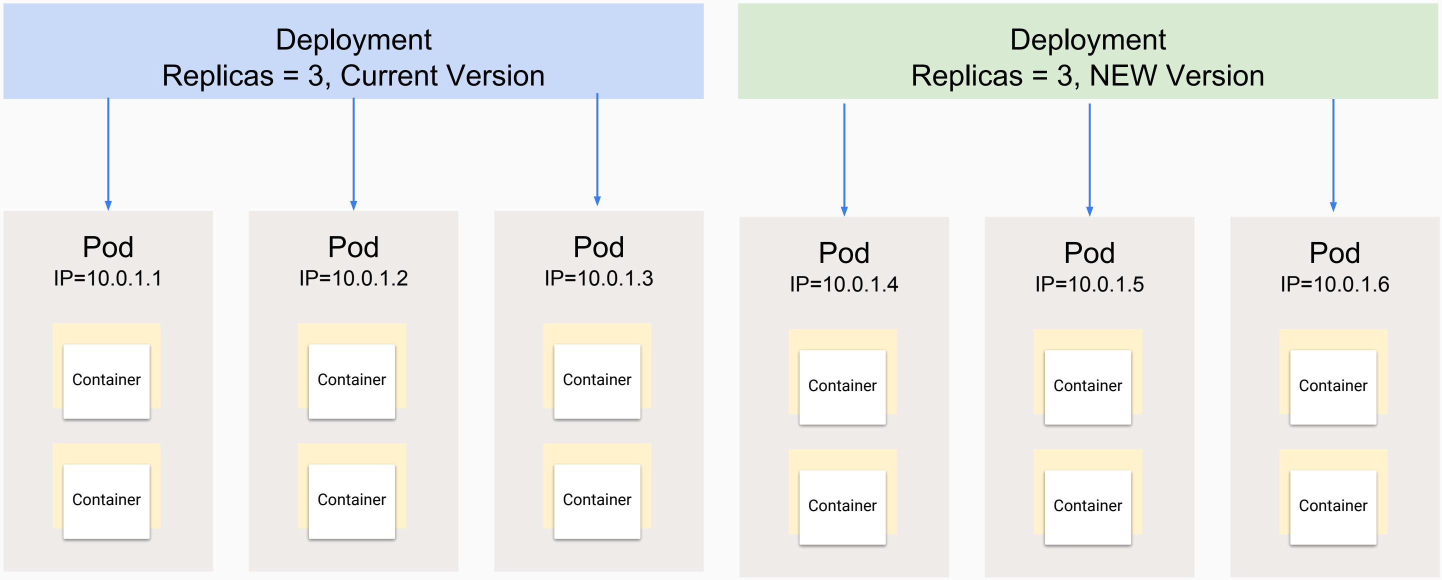
Using a load balancer is a common approach.

# Kubernetes Breakdown

How microservices helps roll out new app versions, migrations, testing, rollback.

## Blue-Green for Zero Downtime

1) Create identical pod
2) Update pod (green)
3) Check readiness
4) Switch trafic



Deployment
Replicas = 3, Current Version

Pod
IP=10.0.1.1
Container
Container

Pod
IP=10.0.1.2
Container
Container

Pod
IP=10.0.1.3
Container
Container

Deployment
Replicas = 3, NEW Version

Pod
IP=10.0.1.4
Container
Container

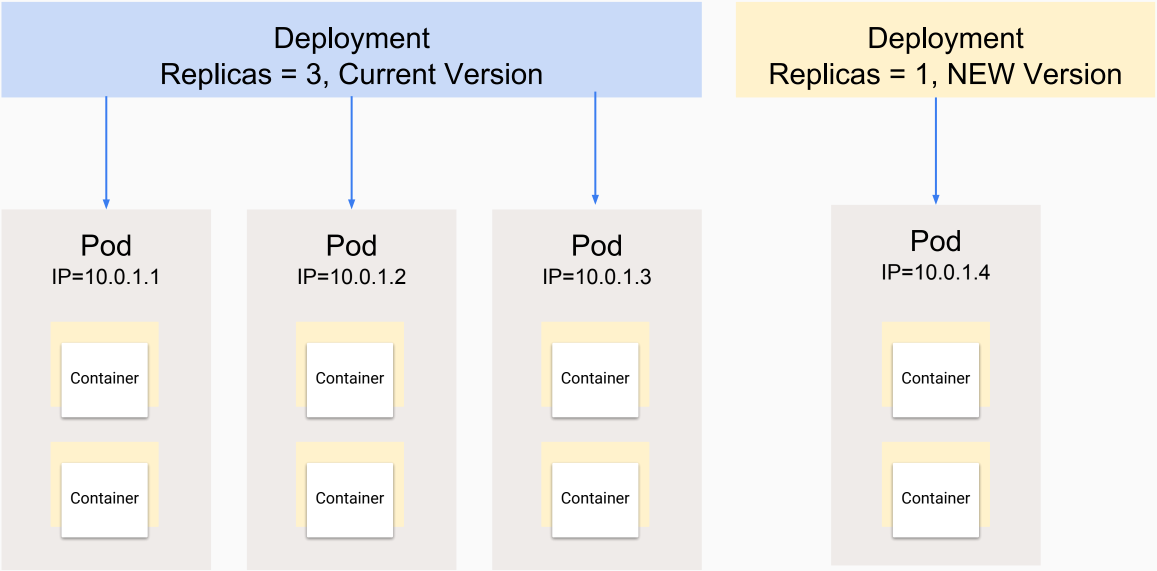
Pod
IP=10.0.1.5
Container
Container

Pod
IP=10.0.1.6
Container
Container

How microservices helps roll out new app versions, migrations, testing, rollback.

## Canary for Subset Testing

1) Create identical pod
2) Update pod (yellow)
3) Check readiness
4) Switch subset of users

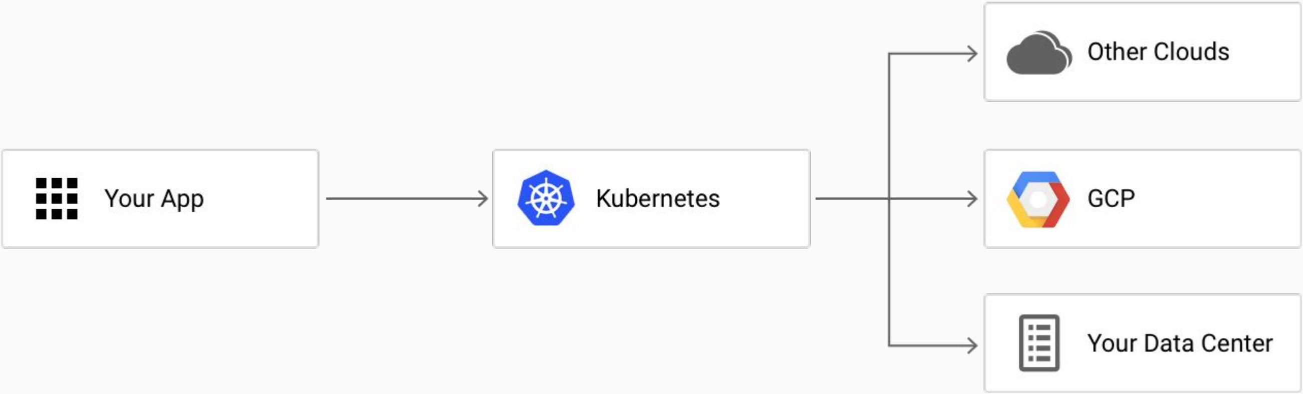


Which brings us to NetApp Kubernetes Service

The power of Kubernetes is expanded dramatically when you can apply its capabilities across clouds, geographies and private data centers.

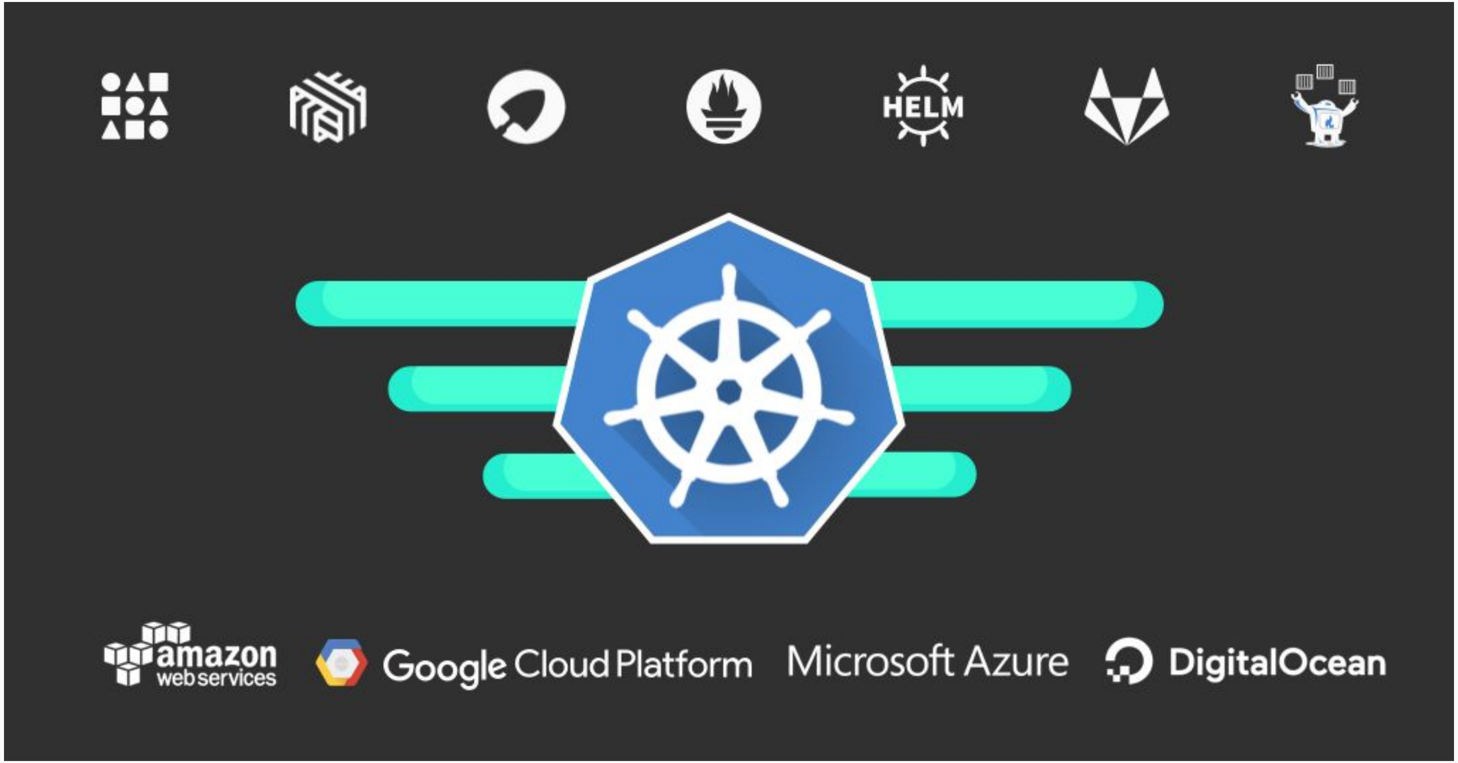
Standardize the way you deploy across disparate UI’s, technology stacks & pricing mechanisms.

Standardize application repositories (Helm Charts, Github repos)



# Stackpoint.io = NKS

Let’s do some hands on...



# Knowledge is information with causal effect.

-David Deutsch

Training:
www.coursera.org
www.acloudguru.com

**Reading:**
Medium.com
github
What do you like?

**Free practice & resources**
www.quiklabs.com
https://codelabs.developers.google.com