Kubernetes for Data Engineers

Big Data on Kubernetes – [Day 1]

LUAN MORENO

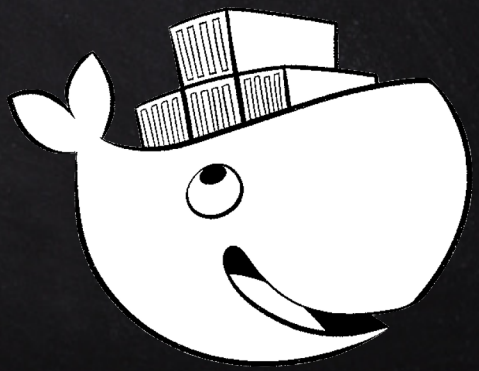
CEO & Data Architect
Data Engineer & MVP

MATEUS OLIVEIRA

Big Data Architect
Data In-Motion Specialist

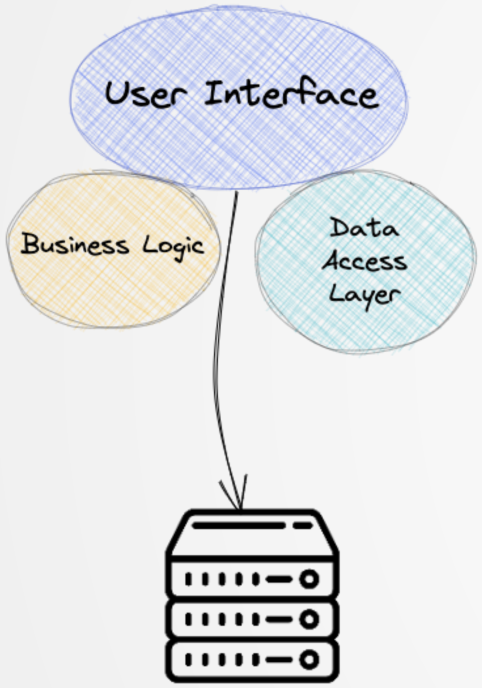
History of Containerized Applications and Container Orchestration Tools

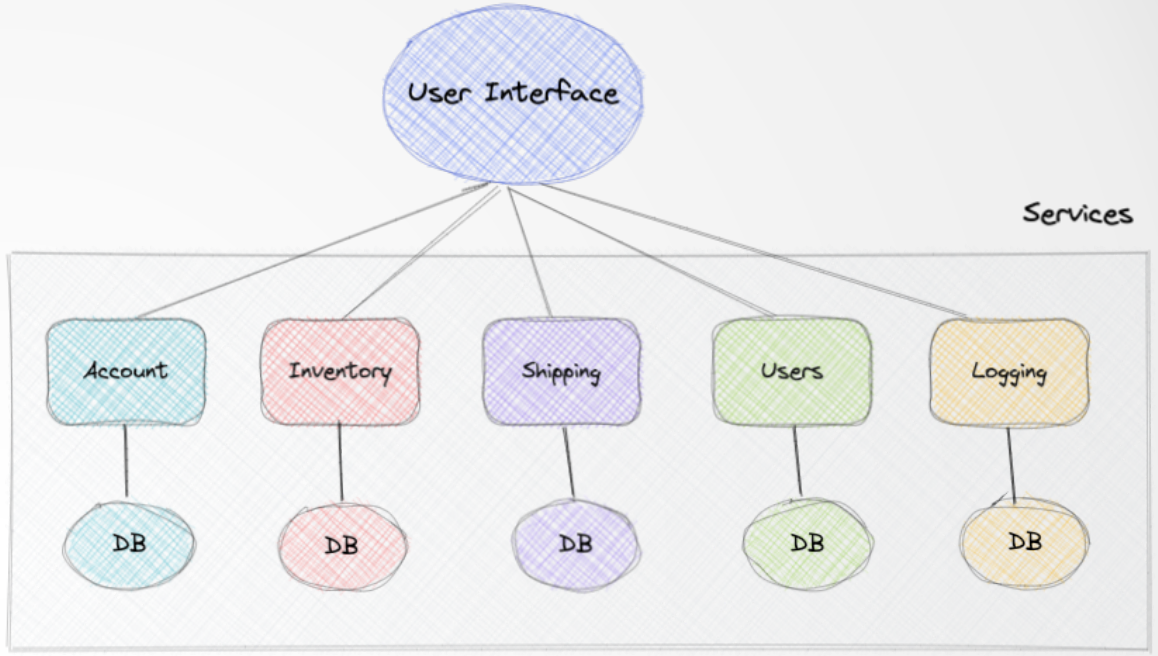
- Monolithic vs. Microservices
- The Era of Microservices & Big Data
- The Principles of Containerized Applications
- Stateless vs. Stateful Applications
- Container Orchestration Engines



Monolithic vs. Microservices

Single Unit Deployment vs. Smaller Independent Units for Speed Up Development Cycle





Drawbacks

- One Large System
- **Single Deployment Unit**
- Size of Application
- Redeploy Application for Update
- Impact of Changes
- Continuous Deployment
- Reliability

Characteristics

- Decomposing Application
- Manageable Services for Faster Development
- **Develop Independently Behavior**
- Adopting New Technologies
- Scale Components Individually

Drawbacks

- Distributed System
- **Different Architecture Paradigms**
- Performing Transactions
- Testing Application
- Deployment of Service

The Era of Microservices & Big Data

Real-Time Analytics at Scale by Enabling Services to Communicate in a Unified and Central Hub

Service Mesh

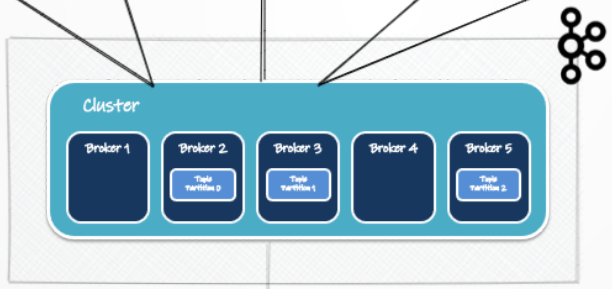
- Software Architecture
- Dedicated Infrastructure Layer
- **Service-to-Service Communication**
- Sidecar Proxy
- Service Discovery
- Load Balancing
- Encryption
- Authentication & Authorization
- Monitoring and Logging
- Circuit Breaker Pattern

Microservices

Independently Developed for Fast Shipping and Delivery

Producers

Communication Channel between services for Fast Event & Data Access



Apache Kafka

Open-Source Distributed Event Streaming Platform for High-Performance Data Pipelines, Streaming Analytics, Data Integration, and Mission-Critical Applications

Data Mesh

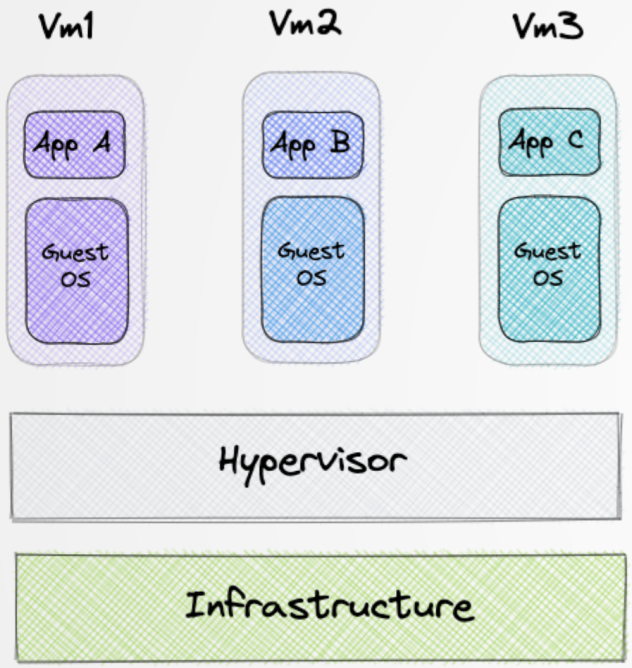
- Design Modern Data Architectures
- Data-Centric and Data Management
- Data Ownership by Domain (1)
- Data as a Product (2)
- Data Everywhere Self-Serve (3)
- Data Governed (4)
- **Specific Business Domain**

Data Pipelines

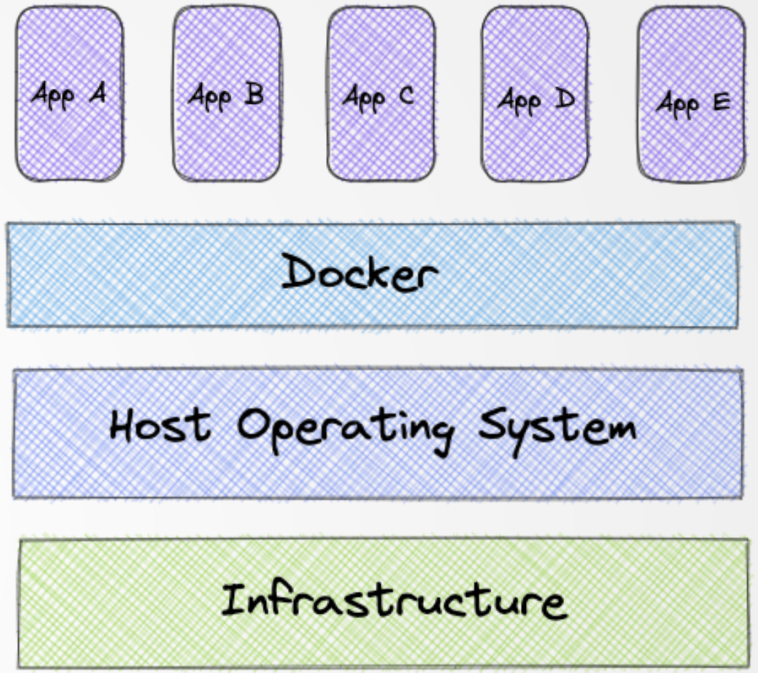
Big Data Technologies to Process Data in Real, Near and Batch Oriented Fashion

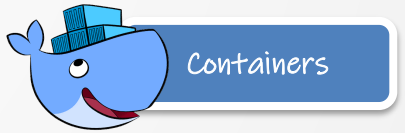
Virtual Machines vs. Containers

VM - Abstracts Machine and Containers - Abstracts Applications



- Enables Virtualization Process = Hypervisor
- Allow Multiple OS
- Ineffective Resource Usage
- Dependent of a Guest OS
- Not Portable

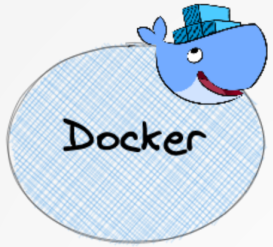




- Images from MB to GB
- Portability
- Effective Resource Usage
- Easier for Maintenance
- Highly-Scalable
- The State of Art

Docker

Use OS-Level Virtualization to
Deliver Software in Packages



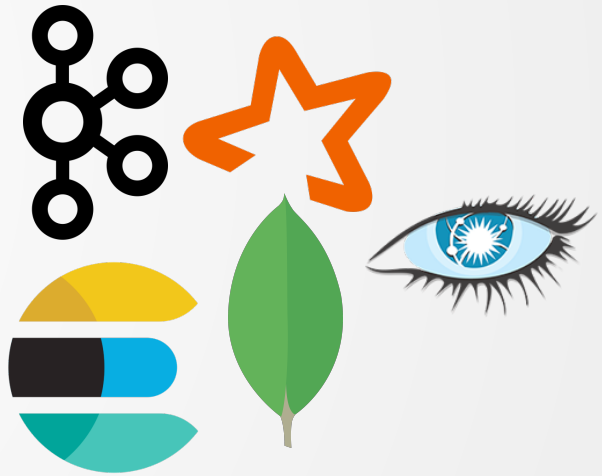
13 Million + Developers

7 Million + Applications

13 Billion + Monthly Image Downloads

Information

- First Started in **2013**, Developed by **Docker Inc**
- Platform as a Service (PaaS)
- OS-Level Virtualization



Docker Desktop

Developer Productivity Tools and a
Local Kubernetes Environment

Docker Hub

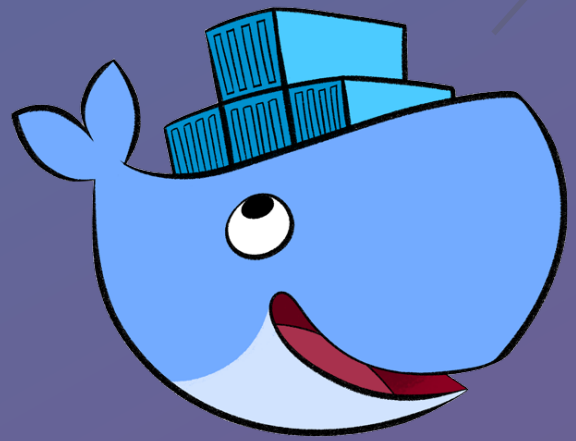
Cloud-Based Application
Registry and Development
Team Collaboration Services

Tools

- **Docker Compose** = Running Multi-Container Docker Applications
- **Docker Swarm** = Native Clustering Functionality
- **Docker Volume** = Copy or Create a File in a Container

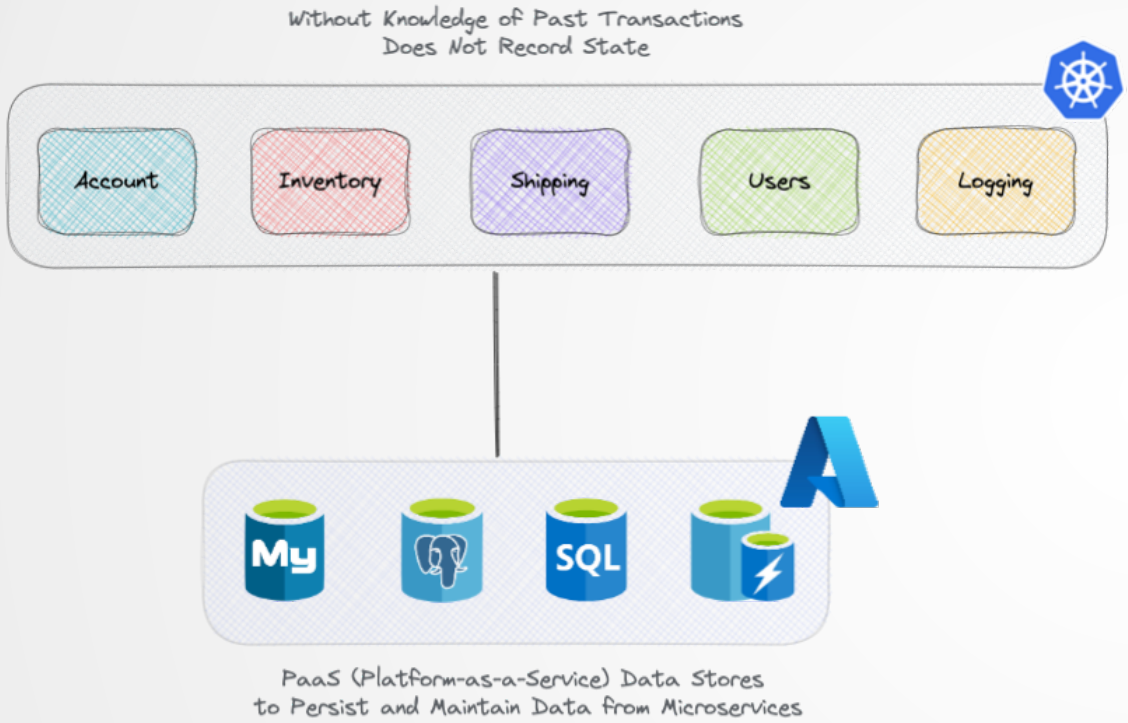
Demo

Docker 101



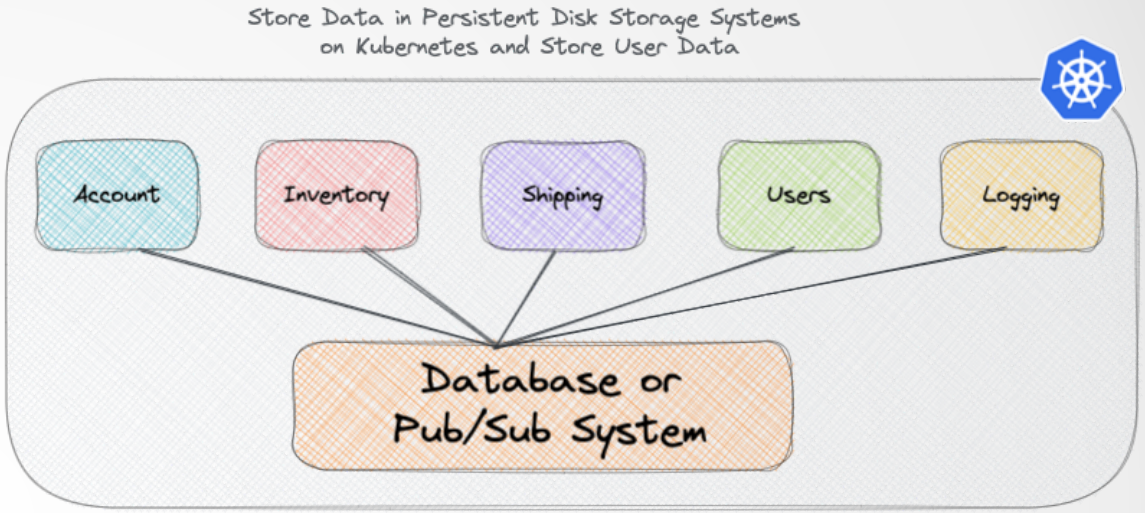
Stateless vs. Stateful Applications

Stateless - Does Not Save Client Data Generated and Stateful - Save Data and State



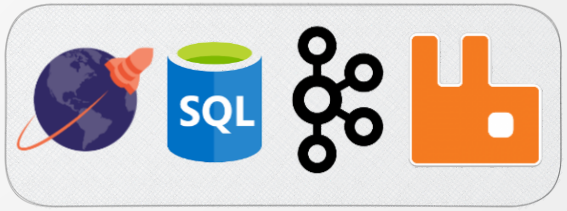
Stateless

- Design, Implementation and Architecture is Simple
- Handles Crash Efficiently
- Scaling is Fast and Easy
- **Recommendation** - Kubernetes and Istio



Stateless

- Complex Design and Architecture
- Storage System Design is Key
- Resource Isolation for Better SLA
- Used by Modern Data Apps
- **Recommendation** - Kubernetes and YugaByteDB



Container Orchestration Engines

Automates Deployment, Management, Scaling and Networking of Container Engines

Docker Swarm

- Maintained by Docker
- Multiple Docker Instances
- Tied with Docker API
- **Used to Spin Up Multiple Containers**

OpenShift

- **Developed by Red Hat**
- Hybrid Proposal
- Commercial Product
- Kubernetes Encapsulated
- Uses CRI-O Runtime
- Image Repository Out-of-Box

Kubernetes

- **First Open-Source Orchestration Engine**
- Originally Designed at Google
- Maintained by CNCF Corporation
- Runtimes - Docker, Containerd, CRI-IO
- Container Runtime Interface (CRI) - Standard

Kubernetes Concepts

- Introduction and Overview
- Cluster Architecture
- Master & Worker
- Workloads
- Service
- Ingress
- Storage
- Configuration

Introduction & Overview

Most Used Orchestration Engine Nowadays. Deployed at Google's Infrastructure, Azure and AWS Data Centers

De-Facto Operating System
for Cloud Native Products

IaaS, PaaS and SaaS Softwares
for Easy and Fast Development

Characteristics

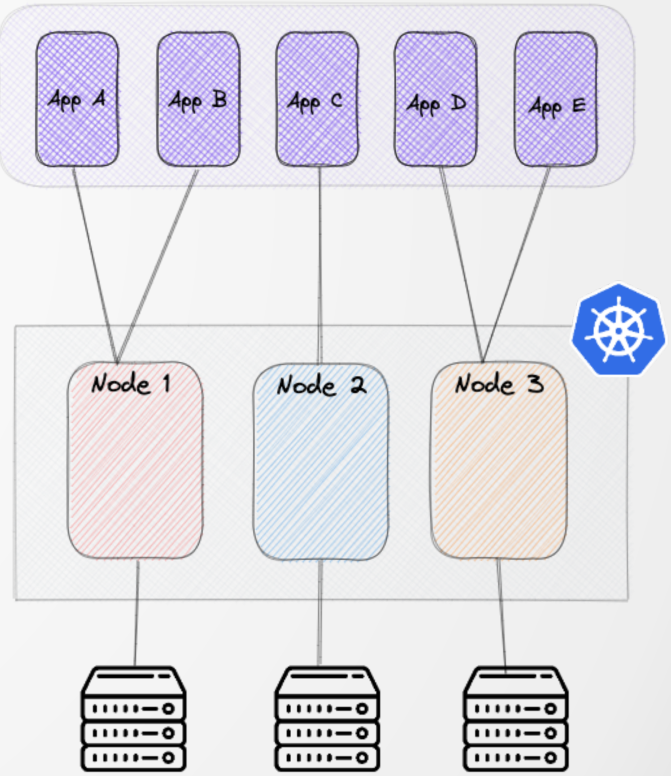
- Originally Developed by Google (Borg & Omega)
- First Open-Source Orchestrator Platform - 2017
- Self-Healing Infrastructure Backbone System
- **Velocity, Scaling, Abstraction and Efficiency**

Main Features

- **Service Discovery & Load Balancing**
- Storage Orchestration
- Automated Rollouts and Rollbacks
- Automatic Bin Packing
- Secret & Configuration Management

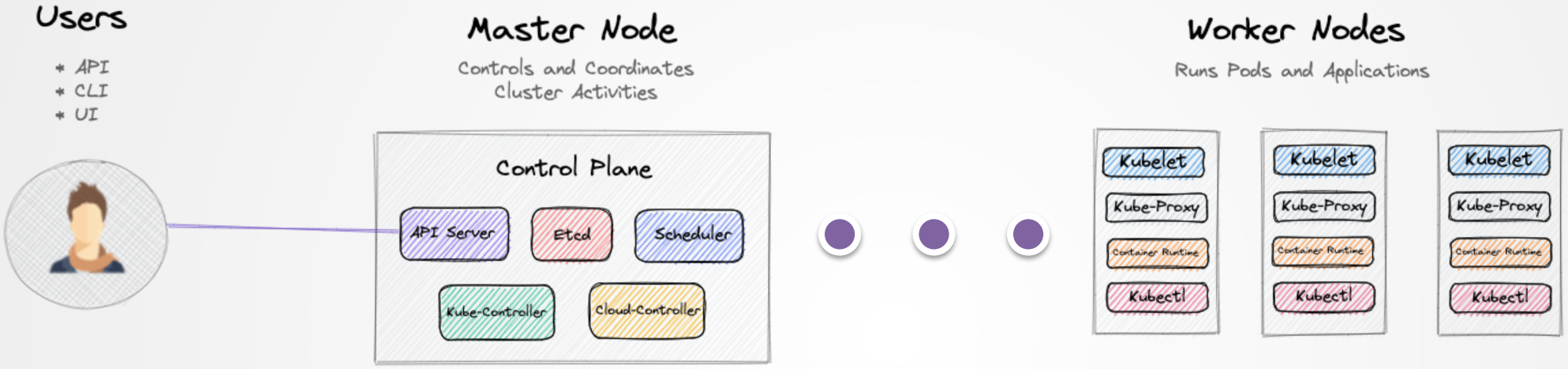
Revolution
The Cloud Native World

* First - Creation of Cloud Computing
* Second - Dawn of DevOps
* Third - Containers



Cluster Architecture - Control Plane and Nodes

Consists of a Set of Worker Machines (Slave) Called Nodes that Runs the Containerized Applications and an Intelligent Control Plane Area



Factors Taken

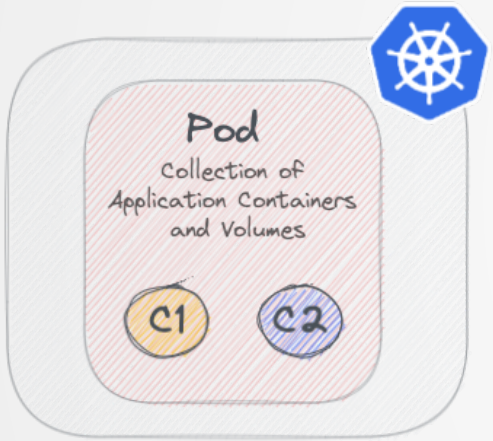
- Individual and Collective Resource
- Hardware, Software
- Policy Constraints
- Affinity and Anti-Affinity
- Data Locality
- Inter-Workload Interference
- Deadlines

API Server	Frontend Server for Handling API Requests
Eted	Key-Value Store Database
Scheduler	Watcher for Newly Created Pods with no Assigned Node
Kube-Controller	Running Resource Controller Processes - Deployments
Cloud-Controller	Interacts with Cloud Providers - LBs, Volumes

Kubelet	Agent Running on Each Node, Checks Container Health and State
Kube-Proxy	Networking to Route Request Between Pods on Different Nodes and Between Pods & Internet
Container Runtime	Starts and Stops Containers and Handles Communications, Usually Docker (Rkt, CRI-O)
Kubectl	Command-Line Tool for Interacting with Kubernetes Objects

Pod & Manifests

Smallest Execution Unit in Kubernetes, Ephemeral by Nature, Automatically Create a New Replica Transparently to Continue Operations




```yaml
apiVersion: v1
kind: Pod
metadata:
  name: mc1
spec:
  volumes:
    - name: html
      emptyDir: {}
  containers:
    - name: 1st
      image: nginx
      volumeMounts:
        - name: html
          mountPath: /usr/share/nginx/html
    - name: 2nd
      image: debian
      volumeMounts:
        - name: html
          mountPath: /html
      command:
        - /bin/sh
        - '-c'
      args:
        - while true; do date >> /html/index.html; sleep 1; done
```

Manifests

Manifests = API Object Description

- Deployment
- ConfigMaps, Secret
- DaemonSet
- JSON or YAML

Group of Whales

Phases

- **Pending** = Accepted by Cluster, Waiting, Downloading or Waiting Process
- **Running** = Bound to a Node, All Containers Created Successfully
- **Succeeded** = All Containers in Pod have Terminated in Success
- **Failed** = Exited with Non-Zero Status or Terminated by the System
- **Unknow** = State was not Obtained, Occurs due to Error in Communicating with Node

Workloads

Applications Running, Singles or Several Components
Working Together Inside of Pods

Specify How Many Identical Pods are Needed

Deployment & ReplicaSet

```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: frontend
  labels:
    app: guestbook
    tier: frontend
spec:
  replicas: 3
  selector:
    matchLabels:
      tier: frontend
  template:
    metadata:
      labels:
        tier: frontend
    spec:
      containers:
      - name: php-redis
```

Groups of Replicated Pods (Copy in All Nodes)

DaemonSet

```
apiVersion: apps/v1
  kind: DaemonSet
  metadata:
    name: fluentd-elasticsearch
... spec: ...
    template:
      ...
      spec:
        containers:
        - name: fluentd-elasticsearch
...
```

Pods with Sticky Identity and Persistent Storage

StatefulSet

```
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: web
spec:
  selector:
    matchLabels:
      app: nginx
  serviceName: "nginx"
  replicas: 3
  template:
    metadata:
      labels:
        app: nginx
    spec:
      terminationGracePeriodSeconds: 10
      containers:
      - name: nginx
        image: k8s.gcr.io/nginx-slim:0.8
        ports:
        - container Port: 80
          name: web
        volumeMounts:
        - name: www
          mountPath: /usr/share/nginx/html
  volumeClaimTemplates:
  - metadata:
      name: www
    spec:
      accessModes: [ "ReadWriteOnce" ]
      storageClassName: "my-storage-class"
      resources:
        requests:
          storage: 1Gi
```

Runs a Pod for a Specified Number of Times

Job

```
apiVersion: batch/v1
  kind: Job
  metadata:
    name: queue-worker
  spec:
    completions: 1
    parallelism: 10
    template:
      metadata:
        name: queue-worker
      spec:
        containers:
...
```

Schedule Jobs by Cron Daemon

Cronjob

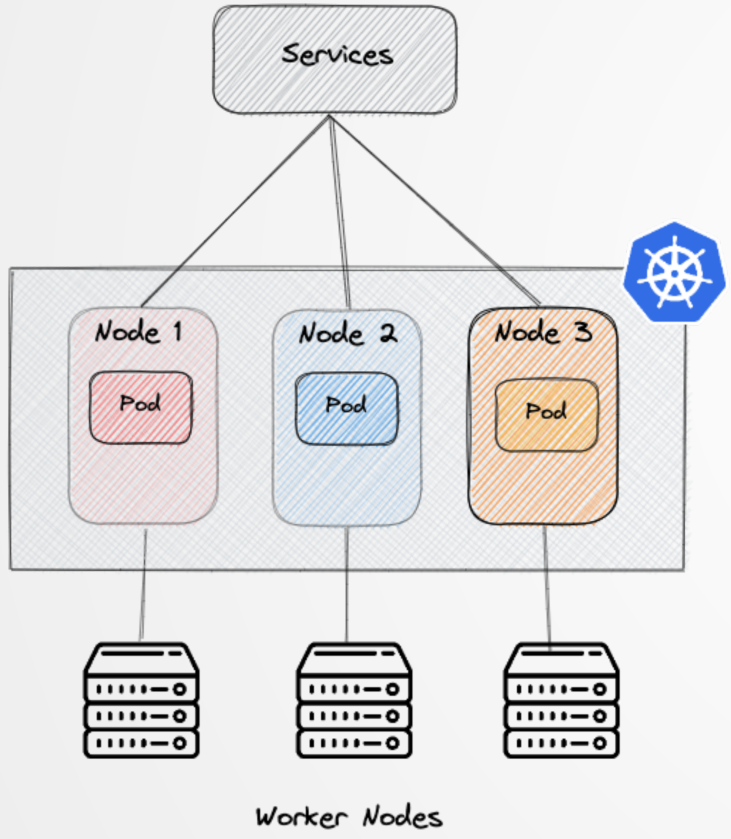
```
apiVersion: batch/v1beta1
  kind: CronJob
  metadata:
    name: demo-cron
  spec:
    schedule: "*/1 * * * *"
    jobTemplate:
    spec: ...
```

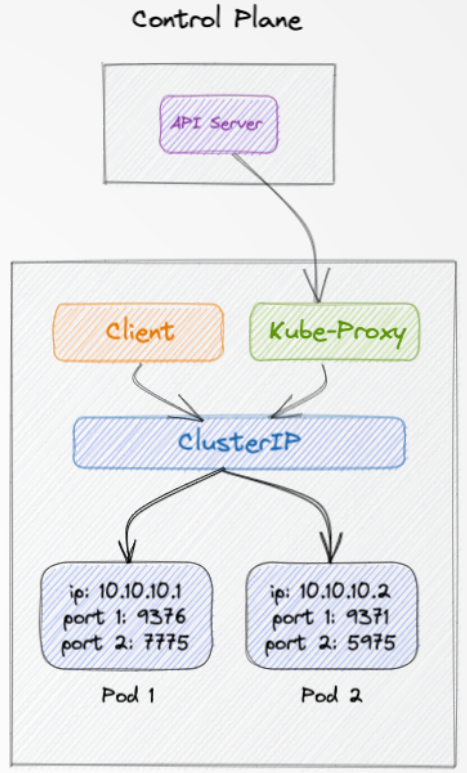
Pod Controllers

Built-In Workload Resources

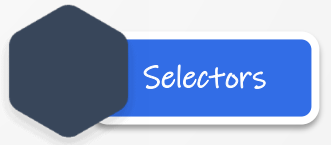
Service

Abstract Exposure of an Application Running on a Set of Pods as a Network Service. Pods Receives IP Addresses and a Single DNS Name






```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: MyApp
  ports:
    - protocol: TCP
      port: 80
      targetPort: 9376
```

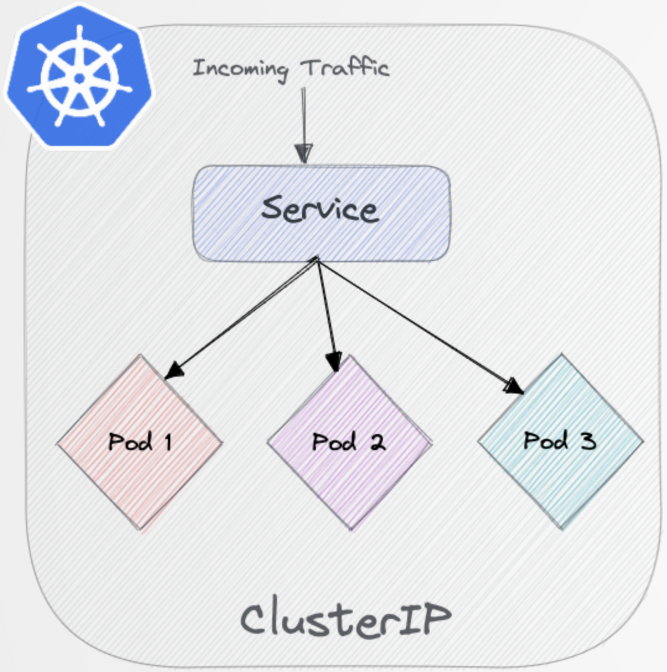


Selectors = Identify a Set of Objects

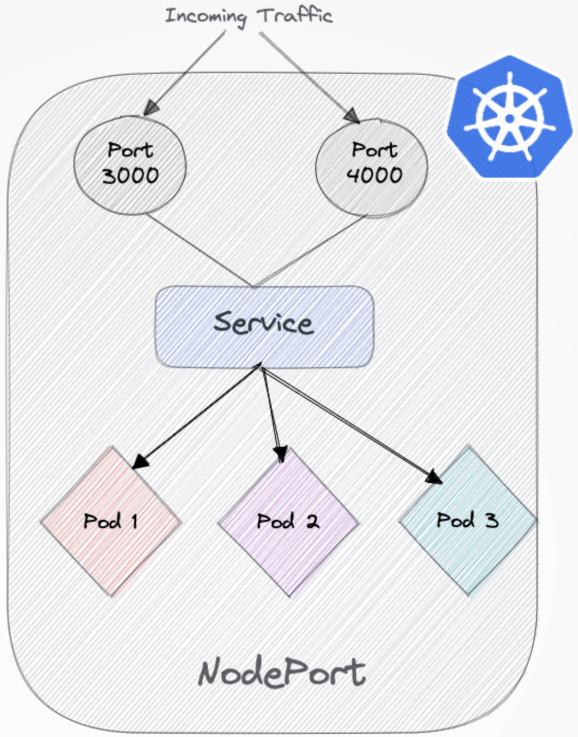
- Equality and Set-Based
- Multiple Requirements

Service Types

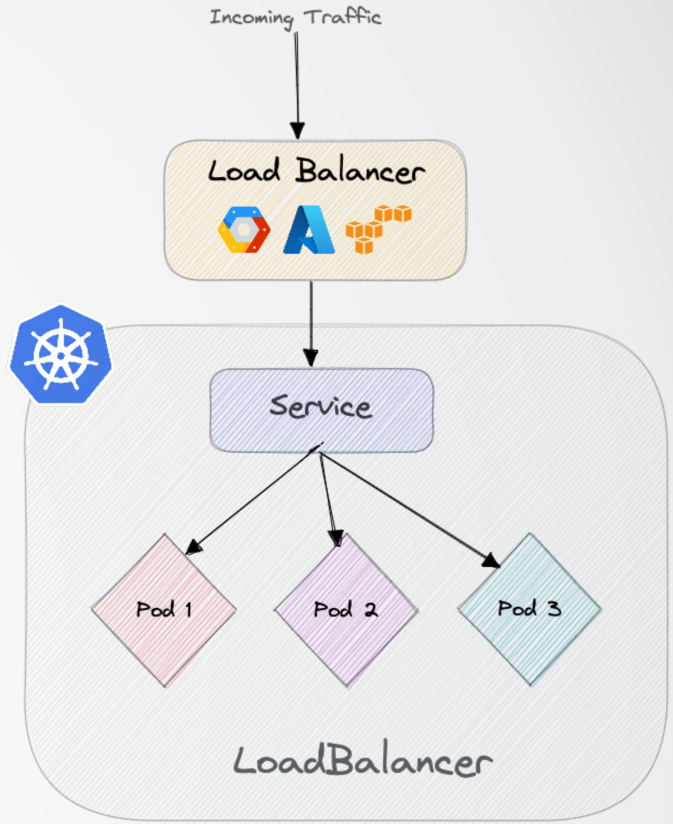
Expose a Service onto an External API Address, Outside of Kubernetes Cluster for Internet-Based Access




```yaml
apiVersion: v1
kind: Service
metadata:
  name: Backend
spec:
  type: ClusterIP
  ports:
    - targetPort: 80
      port: 80
```



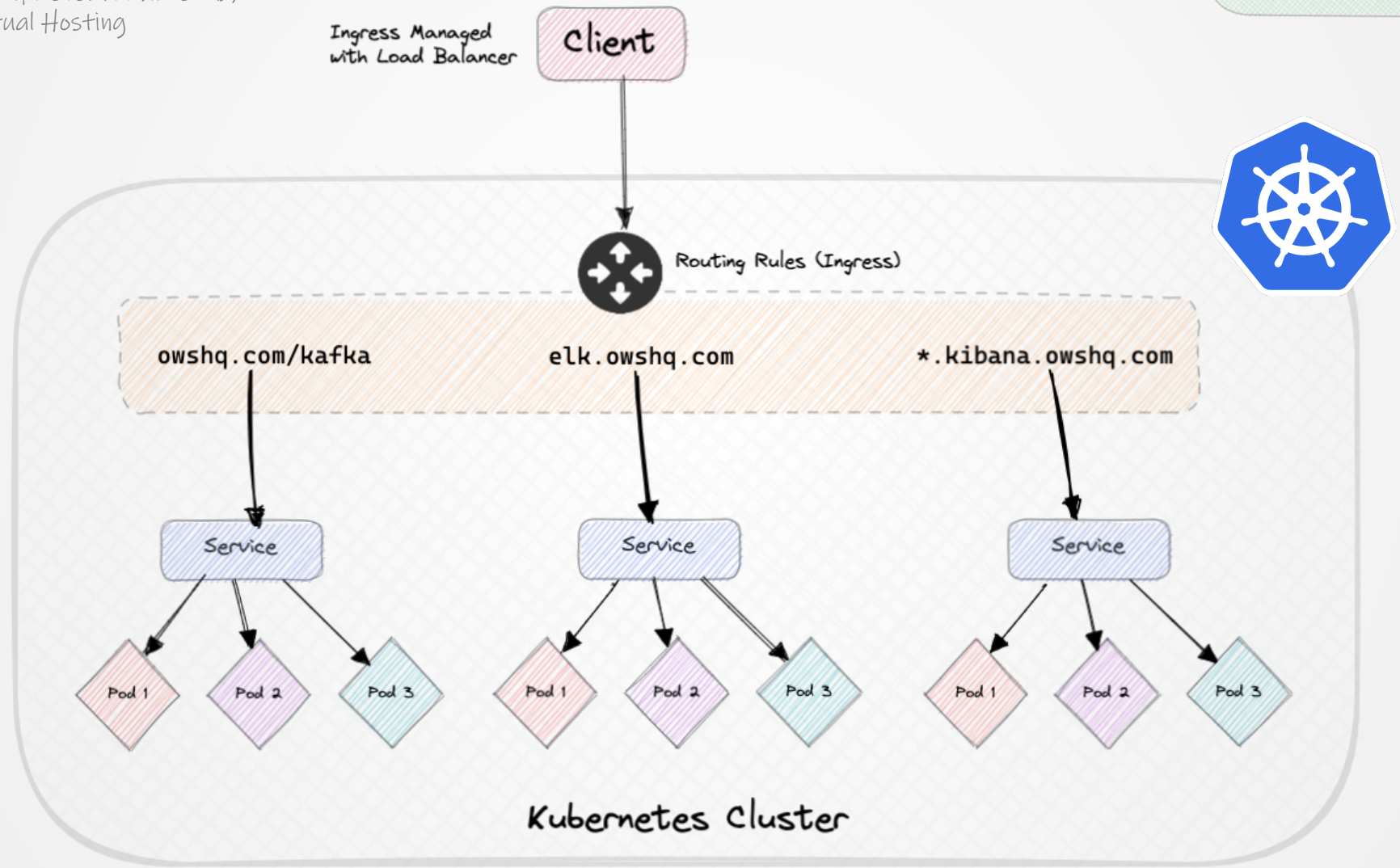

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  type: NodePort
  selector:
    app: MyApp
  ports:
      - port: 80
        targetPort: 80
        nodePort: 30007
```




```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: MyApp
  ports:
    - protocol: TCP
      port: 80
      targetPort: 9376
  clusterIP: 10.0.171.239
  type: LoadBalancer
status:
  loadBalancer:
    ingress:
    - ip: 192.0.2.127
```

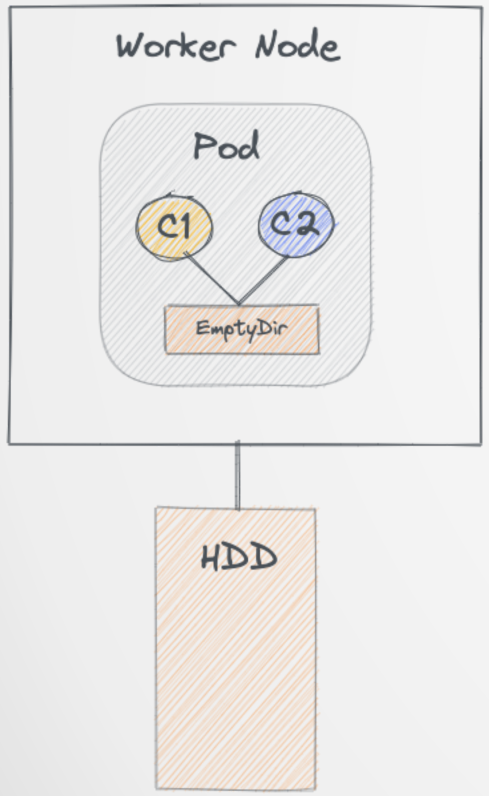
Ingress

Manages External Access to Services in a Cluster, Typically HTTP Requests. Provides LB, SSL and Name-Based Virtual Hosting



Volumes & Types

Pod Restarts in Clean State, Wipe Out Data, Ability to Use Different Volumes Simultaneously, Destroys Ephemeral but do not Destroy Persistent Volumes



Ephemeral Types

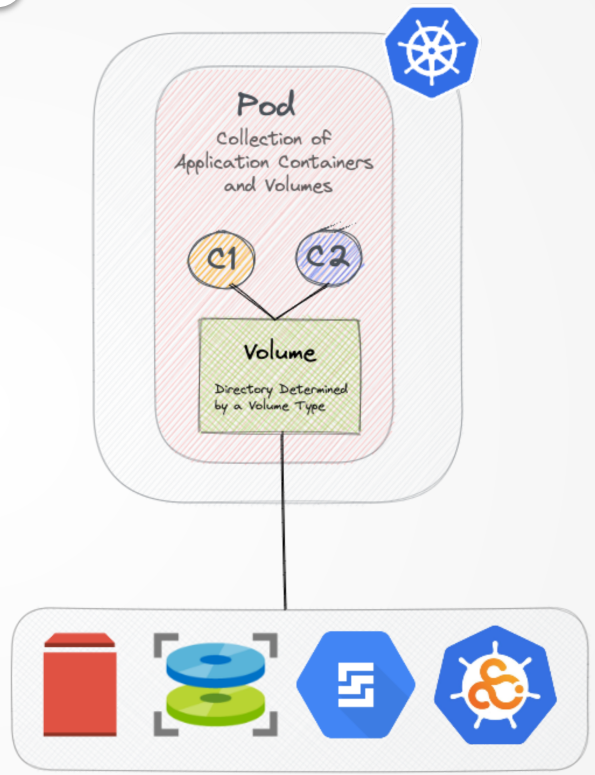
Data not Stored Persistently Across Restarts

* emptyDir
* CSI ephemeral volumes
* generic ephemeral volumes

```
apiVersion: v1 kind: Pod
...
spec:
    volumes:
    - name: cache-volume
      emptyDir: {}
    containers:
    - name: demo
      image: cloudnatived/demo:hello
      volumeMounts:
    - mountPath: /cache
      name: cache-volume
```

Worker Node

Physical or Virtual Machines



Volume Type

Directory Determined by a Volume Type

* AWS EBS
* Azure Disk
* Google Persistent Disk
* Container Storage Interface

Container Storage Interface (CSI)

Standard for Exposing Arbitrary Block and File Storage Systems for Container Orchestration Systems (COs)

CSI = Container Storage Interface

- Promoted GA in V1.13 Release
- Powerful Volume Plugin System
- Before Embedded on Binaries
- Expose Arbitrary Block and File Storage

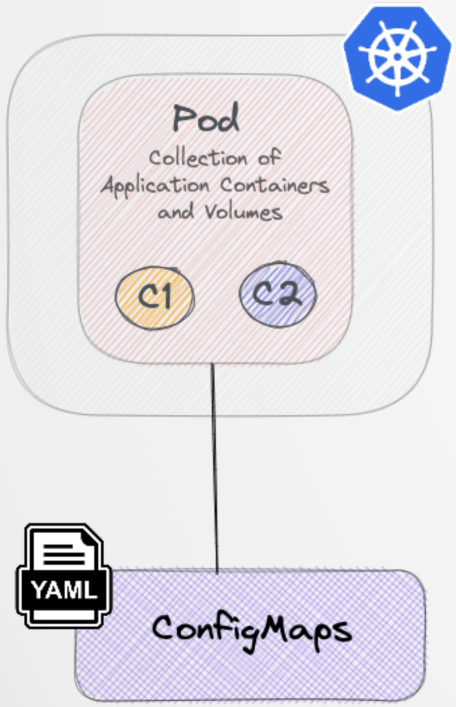
- Enable Dynamic Provisioning

Configuration - ConfigMaps & Secrets

ConfigMaps = Used to Store Non-Confidential Data in Key-Value
Secrets = Sensitive Information

ConfigMaps

Store Non-Confidential Data in Key-Value Pairs
Consumed as Variables, CLI Arguments or
as Configuration Files in a Volume



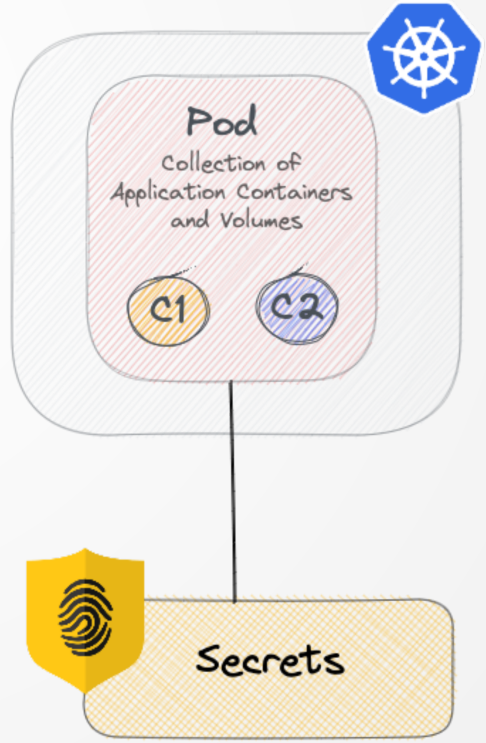

```
kind: ConfigMap
metadata:
  name: game-demo
data:
  player_initial_lives: "3"
  ui_properties_file_name: "user-interface.properties"

  game.properties: |
    enemy.types=aliens,monsters
    player.maximum-lives=5

  user-interface.properties: |
    color.good=purple
    color.bad=yellow
    allow.textmode=true
```

Secrets

Sensitive Data such as Password, Token or a Key.
Store Confidential Data Outside of Application Code




```
apiVersion: v1
kind: Secret
metadata:
  name: test-secret
data:
  username: bXktYXBw
  password: Mzk1MjgkdmRnNOpi
```

Demo

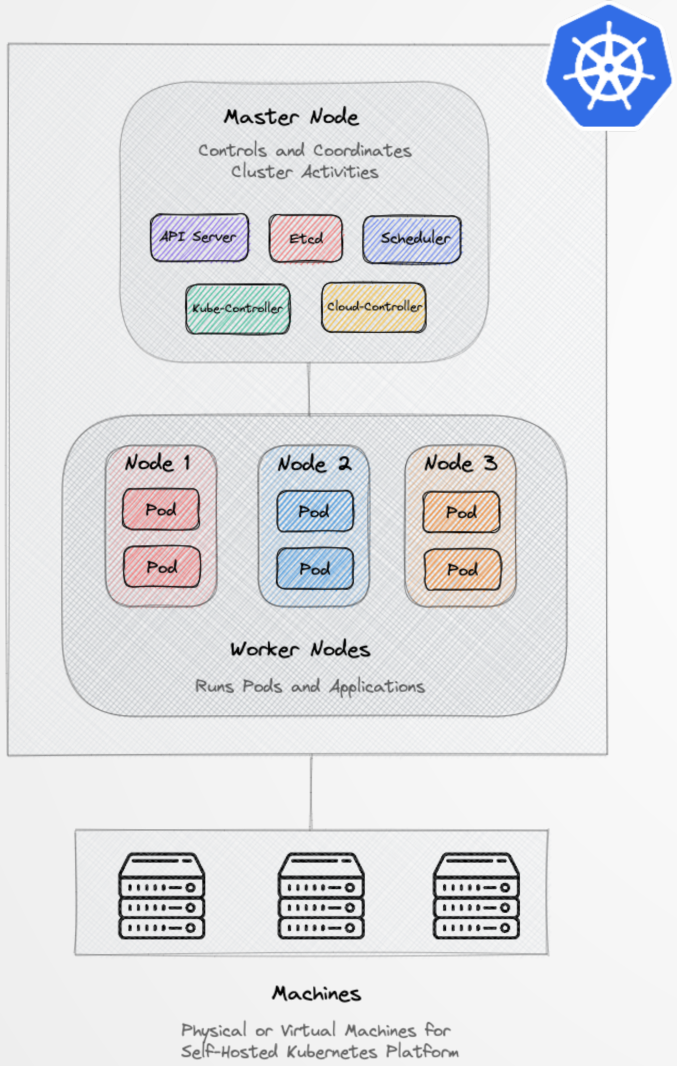
Kubernetes 101

Kubernetes as a Managed Offering

- Self-Hosted Kubernetes
- Managed Kubernetes Services
- AKS, EKS and GKE
- Infrastructure-as-Code (IaC)

Self-Hosted Kubernetes

Usually Deployed in On-Premises Environments,
Hard to Maintain and Implement



Challenges

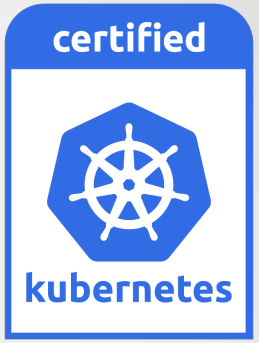
- High Availability
- Single Sign-On
- Multitenancy
- Resource Isolation
- Permission Management
- Upgrades
- Backups
- Package Management
- CI/CD Integration

Implement Kubernetes Self-Hosted from Scratch

- **$1 Million Dollars** in Engineer Salary
- Still Undergoing Process
- 6 Team Members Upstream Production
- 6 Months of Work

Kubernetes as a Managed Offering

Managed Production-Ready Solution Provided by Cloud Providers
for Deploying Your Containerized Applications



Azure Kubernetes Service (AKS)

Deploy and Scale Containers on
Managed Kubernetes

AKS

- Serverless Kubernetes Service
- Integrated CI/CD Experience
- Enterprise Grade Security and Governance
- Co-Founder of Kubernetes Brendan Burns

- Control-Plane is Free

Google Kubernetes Engine (GKE)

Managed, Production-Ready Environment
for Containerized Applications

GKE

- Fully-Managed by Google
- Running Anthos On-Premises
- Modes - Autopilot & Standard

- Control Plane - $0.10 per Cluster/Hour

Elastic Kubernetes Service (EKS)

Managed Container Service to Run
and Scale Containerized Applications

EKS

- Managed Kubernetes Service
- Fully-Managed and Integrated with AWS
- VPC Support
- Spot Instances

- Control Plane - $0.10 per Cluster/Hour

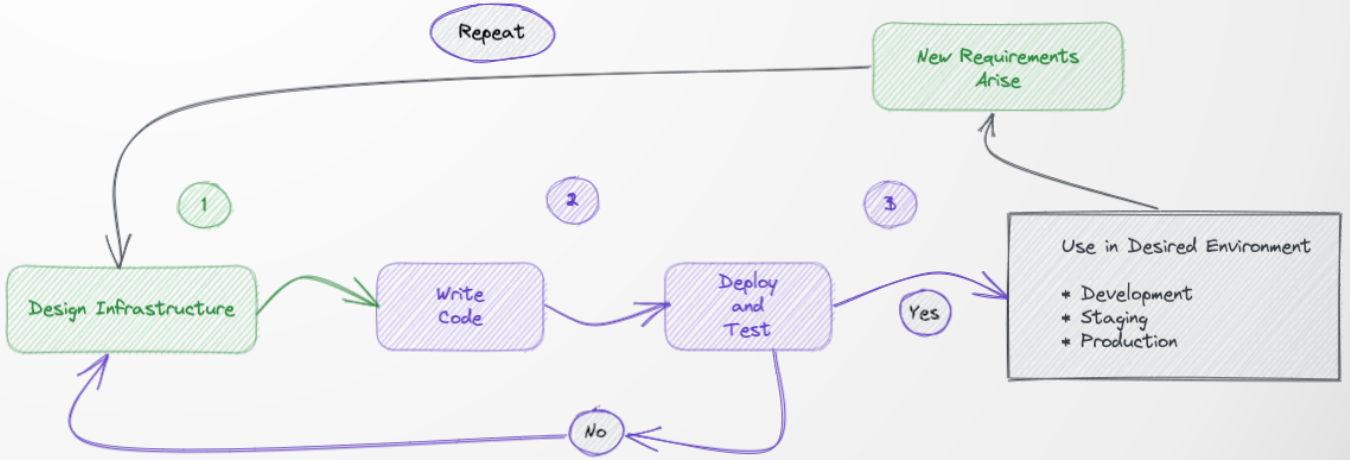
IaC for Data Engineers

Manage and Provision Resources using Code to Reduce Manual Process, Immutable Infrastructure Approach

* Define Infrastructure as Code using a Declarative Language
* Define, Deploy, Update & Destroy your Infrastructure
* DevOps Key Insight = Managed Process using Code

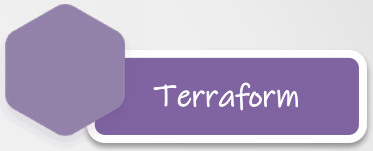
Benefits

- Self-Service
- Speed and Safety
- Documentation
- Version Control
- Validation
- Reuse



Terraform

Open-Source Infrastructure as Code Software. Provision Resources using a Declarative Configuration Language



- Created by HashiCorp
- Initial Release 2014
- Written in Go

Terraform

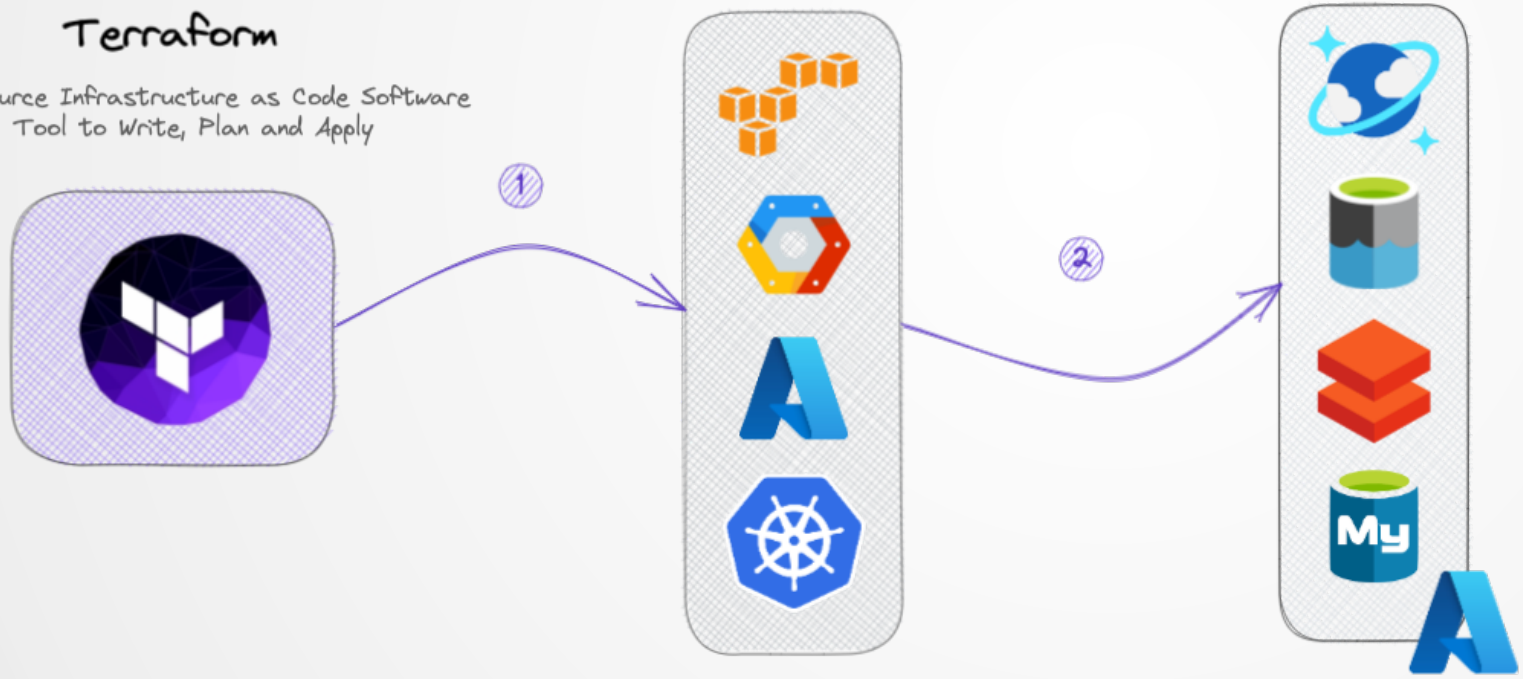
Open-Source Infrastructure as Code Software Tool to Write, Plan and Apply

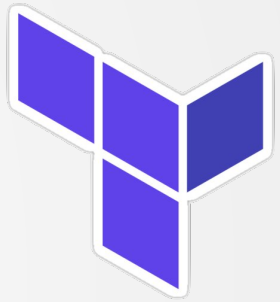
Providers

Providers are a Logical Abstraction of an Upstream API Understanding Interactions and Exposing Resources

Resources

Components Deployed on Provider, Managed and Controlled by Terraform State





Terraform Registry

Logical Abstraction of an Upstream API. Interacting with Providers to Expose Resources

- Official
- Verified
- Community

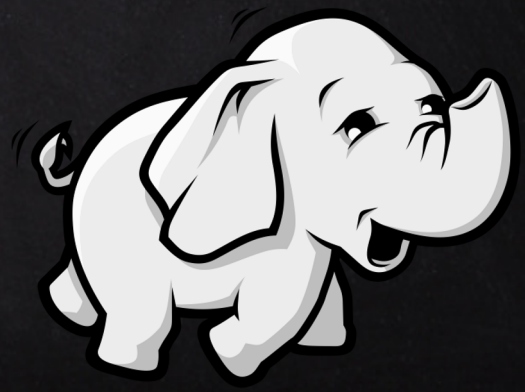
https://registry.terraform.io/browse/providers

Demo

Deploying Managed Kubernetes Services using Terraform

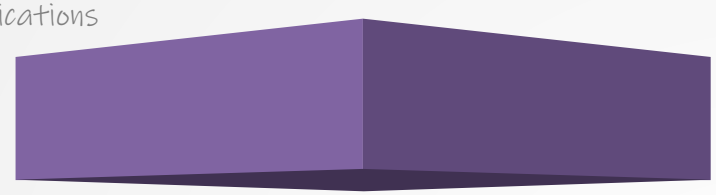
Debuting Kubernetes for Data Engineers

- Big Data Eras
- Kubernetes as Cornerstone Solution for Big Data Infrastructure
- Symbiosis of Kubernetes and Big Data Systems



Big Data Eras

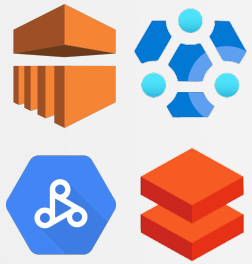
On-Premises, Cloud Computing with PaaS and SaaS, Kubernetes as Cornerstone for Big Data Applications

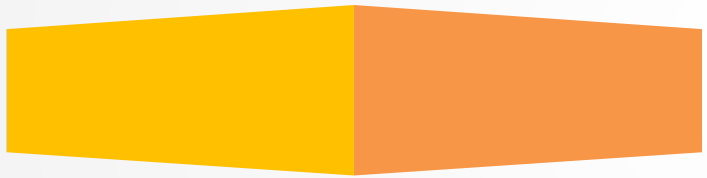


Kubernetes as Cornerstone Solution for Apps & Data ~ [2018]

- Multi-Cloud Strategy First
- Use of Managed Kubernetes Solutions with 99.99%
- Simplify Operations By using Kubernetes as Infrastructure Backbone
- Microservice Driven Approach ~ Containerize Environments
- Cheapest Solution for Application, DataStore & Big Data Solution

Cons ~ Steep Learning Curve, Mindset Change

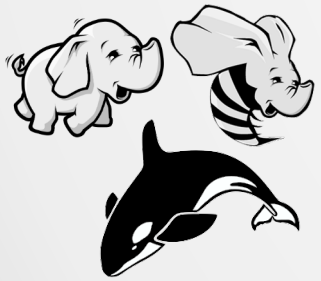


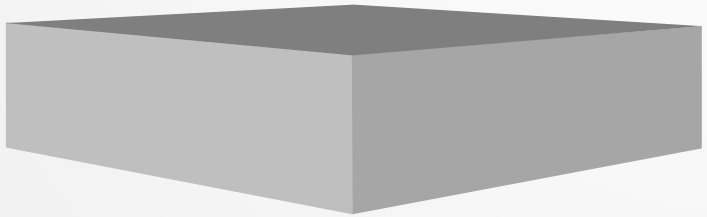


Use of PaaS & SaaS Software's ~ [2014]

- Solution Managed by Cloud Vendor
- Big Data as a Service ~ BDaaS
- Decoupling Computation from Storage
- Pay for Use Only
- Strategy to Reduce Overall Big Data Costs

Cons ~ Promise of Cost Reduction, Myriad of Options





Big Data on Premises using Vendors ~ [2009]

- Cloudera, MapR & HortonWorks
- Based on Open-Source Technologies
- On-Premises Data Centers using Physical Hardware
- Expensive & Dependency Wise
- Large Operations Team ~ Manage Clusters

Cons ~ Expensive, High-Level of Complexity, Burdensome

Kubernetes as Cornerstone Solution for Big Data Infrastructure

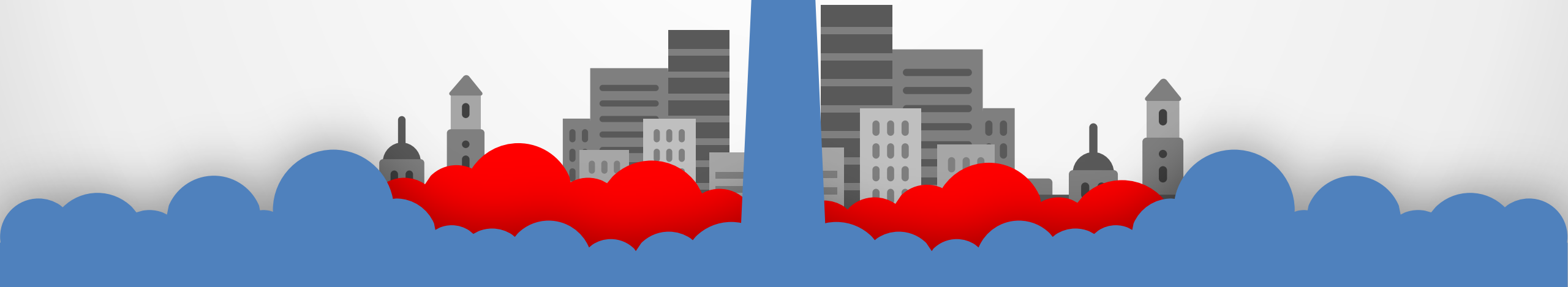
Third Generation of Big Data Deployments, Main Characteristics and Driving Points

Cost-Wise

Statefulsets

Current Gen of Big Data

Microservices



Symbiosis of Kubernetes and Big Data Systems

Main Components that Integrates Kubernetes and Big Data Systems Together in One-Single Unit

CRDs (Custom Resource) Operator Pattern

Custom Resources

- Software Extension
- Manage Applications & Components
- Principles of Control Loop
- Human Operator Intelligence

Automate Repeatable Tasks

Applying State using Storage Solutions

StatefulSets API

- Used for Distributed Applications
- Ordered Pod Creation
- Stable Network Identities
- Stable Storage using Storage Class
- PV and PVCs Claims

Sticky Identification of Pods

DoK - Data on Kubernetes Community

Data on Kubernetes Community

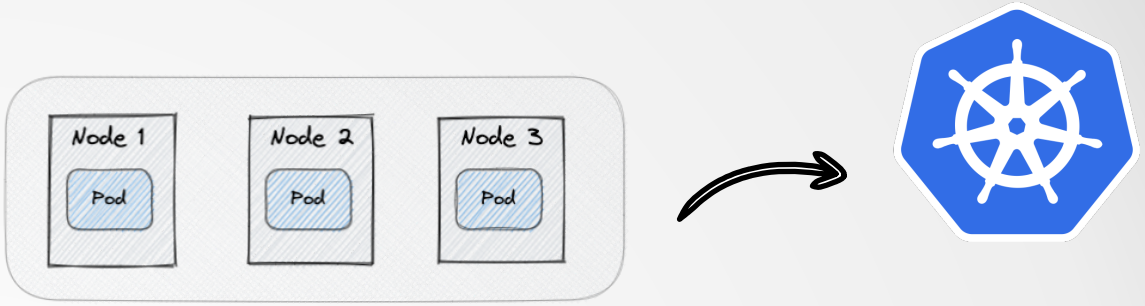
- Apply Best Practices
- Weekly Meetups
- Discussions

Big Data Channel

https://dokcommunity.slack.com/

SC, PV & PVC

Storage Class, Persistent Volume and Persistent Volume Claim for Stateful Applications



Persistent Volume Claim (PVC)

Persistent Volume Claim - PVC

- Pods Main Entry Communication
- Claim Space to PV

Request for Storage by User

Persistent Volume Claim (PVC)

Persistent Volume Claim (PVC)

Persistent Volume Claim (PVC)

* Request for Storage by User
* PVCs Consume PV Resources
* Claim Size for Volume Mount

Persistent Volume (PV)

Persistent Volume - PV

- Volume Plugins
- Lifecycle Independent
- Main Entry for Pod Requests

Abstracts Storage Layer

Persistent Volume (PV)

Persistent Volume (PV)

Persistent Volume (PV)

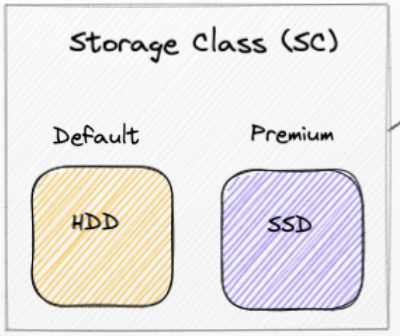
* Abstracts Storage Details
* Volume Plugins with Lifecycle Independent

Storage Class

Storage Class - SC

- Provide QoS Levels
- Backup Policies
- Concept of Profiles

Administrators Describe Classes



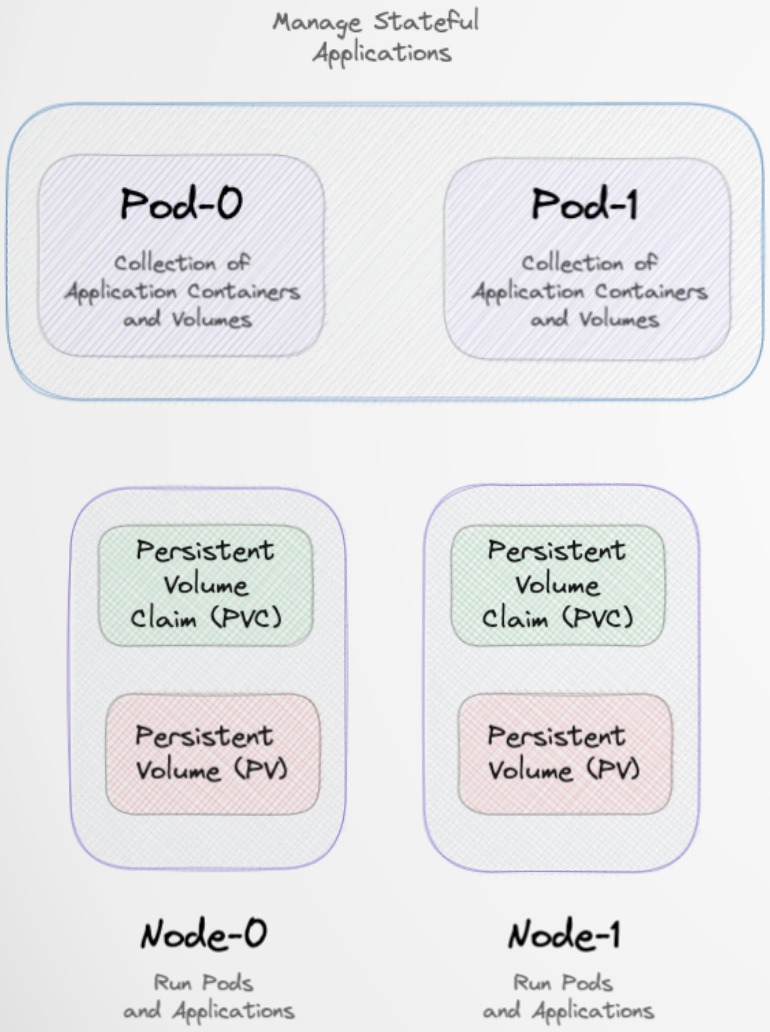
* Describe Classes of Storage
* Affect Quality-of-Service Levels
* Determined by Cluster Administrators

Directory Determined by a Volume Type

* AWS EBS
* Azure Disk
* Google Persistent Disk
* Container Storage Interface

StatefulSets

Manage Stateful Applications on Kubernetes. Provides Guarantees about Ordering and Uniqueness of Pods



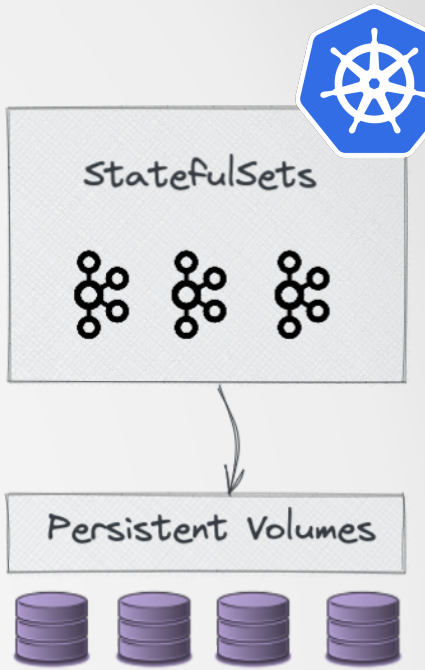
StatefulSets

Persistent Volumes

StatefulSets Explained

- Guarantees Ordering and Uniqueness of Pods
- Sticky Identity of Each Pod
- Use Deployment Type with Persistent Volumes
- Stable - Unique Network and Storage
- Ordered - Graceful Deployment and Rolling Updates

Used for Distributed Systems Deployment



PVC & PV Explained

Pod Request Claim using Persistent Volume Claim API
Persistent Volume (PV) Replies to Request from PVC
PV Sends Requests to Storage System

Maintain Data Persisted

Development Toolkit for Data Engineers

- Development Environment
- Kubernetes Package Options
- Source-Code Repository
- Container Registry Repository
- Toolbelt for Data Engineers

Development Environment

Local and Cloud Development Options for Data Engineers

Quickly Sets Up a Local Kubernetes Cluster on MacOS, Linux, and Windows. Helping Developers for Fast Application Shipment

Minikube

- MicroK8S
- Kind
- K3D

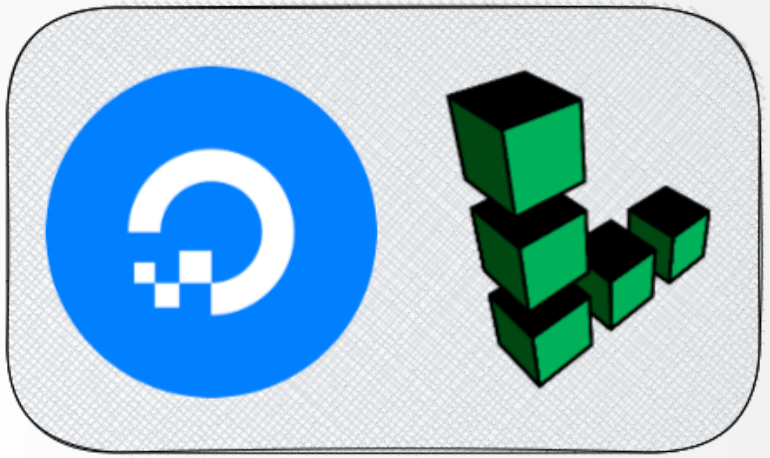
- 2 CPUs
- 2 GB of Memory
- 20 GB Space

Container or **Virtual Machine Manager**

Docker, Hyperkit, Hyper-V, KVM, Parallels, Podman, VirtualBox, VMware

Affordable Cloud Computing Services with Developer-Friendly Features and Scalability

* Digital Ocean
* Linode



Digital Ocean & Linode

Digital Ocean

- American Cloud Infrastructure
- Providers PaaS and SaaS Options
- Digital Kubernetes Engine (DKE)

https://try.digitalocean.com/freetrialoffer/

Linode

- American Privately-Owned Cloud Hosting
- Top Rated of 2021
- Simple, Affordable and Accesible
- Linode Kubernetes Engine (LKE)

https://www.linode.com/lp/free-credit-100/

Demo

Development Environment using Minikube and Digital Ocean

Kubernetes Packages

Manage Complex Manifests using Tools to Reduce Complexity and Enhancing Development Experience

Helm

Package Manager for Kubernetes. Manage Applications using Charts. Define, Install and Upgrade Complex Manifests

YAML Manifests

Human-Readable text-Based Format to Easily Specify Configuration-Type Information by using Maps of Name-Value Pairs and Lists of Items

Kustomize

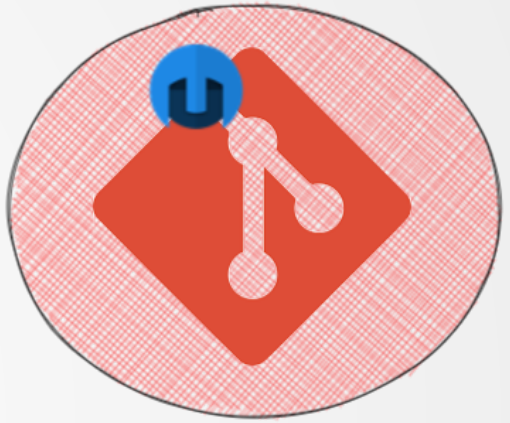
Native Configuration Management using a Template-Free to Customize and Simplify Out-of-the-Shelf Applications

```
~/someApp
├── deployment.yaml
├── kustomization.yaml
└── service.yaml
```

```
~/someApp
├── base
│   ├── deployment.yaml
│   ├── kustomization.yaml
│   └── service.yaml
└── overlays
    ├── development
    │   ├── cpu_count.yaml
    │   ├── kustomization.yaml
    │   └── replica_count.yaml
    └── production
        ├── cpu_count.yaml
        ├── kustomization.yaml
        └── replica_count.yaml
```

ArtifactHub

Find, Install and Publish Kubernetes Packages
Web-Based Application for CNCF Projects
Helm Charts and Plugins



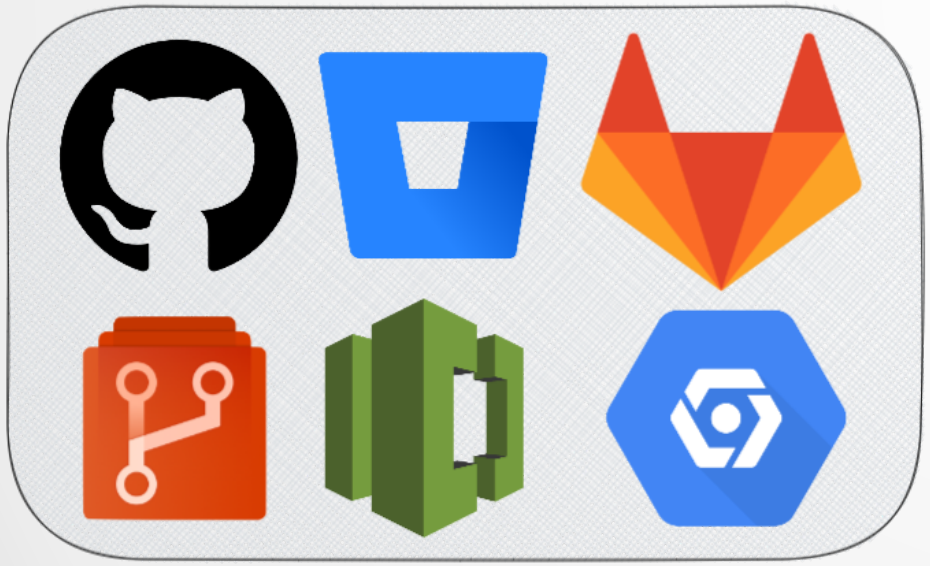
Source-Code & Container Registry Repository

Store Deployment Files and Containerized Applications Securely in a Repository for Sharing and to Speed Up Development Cycle

Source-Code Repository

Repository is a Data Structure that Stores Metadata for a Set of Files or Directory Structure

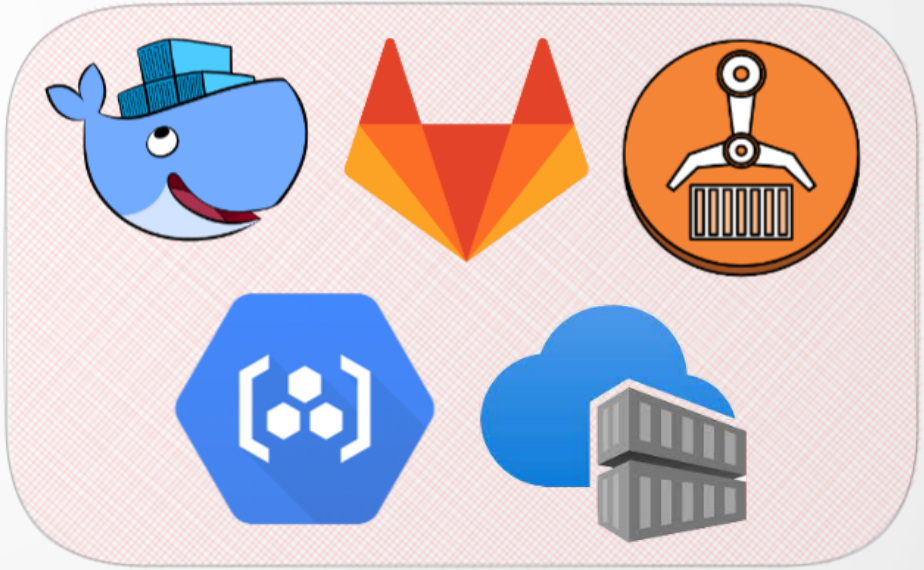
* GitHub
* BitBucket
* GitLab
* Azure Repos
* AWS CodeCommit
* Google Cloud Source



Container Registry

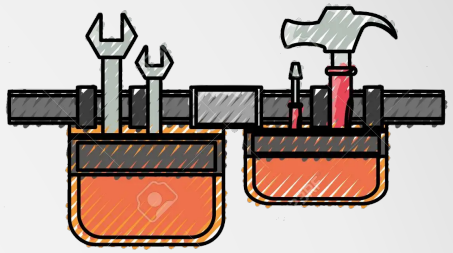
Repository or Collection Used to Store Container Images for Kubernetes, DevOps and Container-Based Application Development

* DockerHub
* GitLab Container Registry
* Amazon ECR
* Google Container Registry
* Azure Container Registry



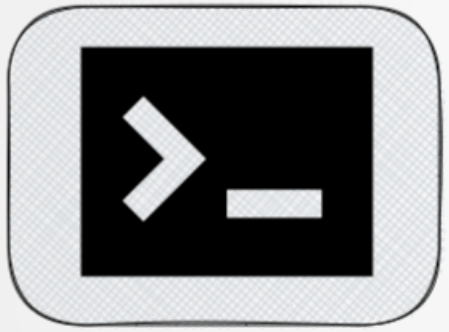
Toolbelt for Data Engineers

Different Tools and Applications to Help Data Engineers to Maintain and Monitor Applications inside of Kubernetes



Command-Line Interface

Operate Software and OS with Single Commands



* KubeCTL
* KubeCTX
* Kubens

Log and Troubleshooting

Resolve Problems Effortlessly using Tools to Reduce Complexity

* Stern

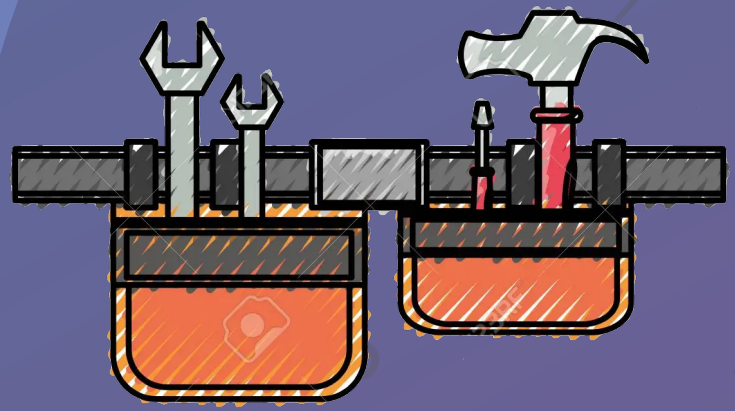
Administration & Maintenance

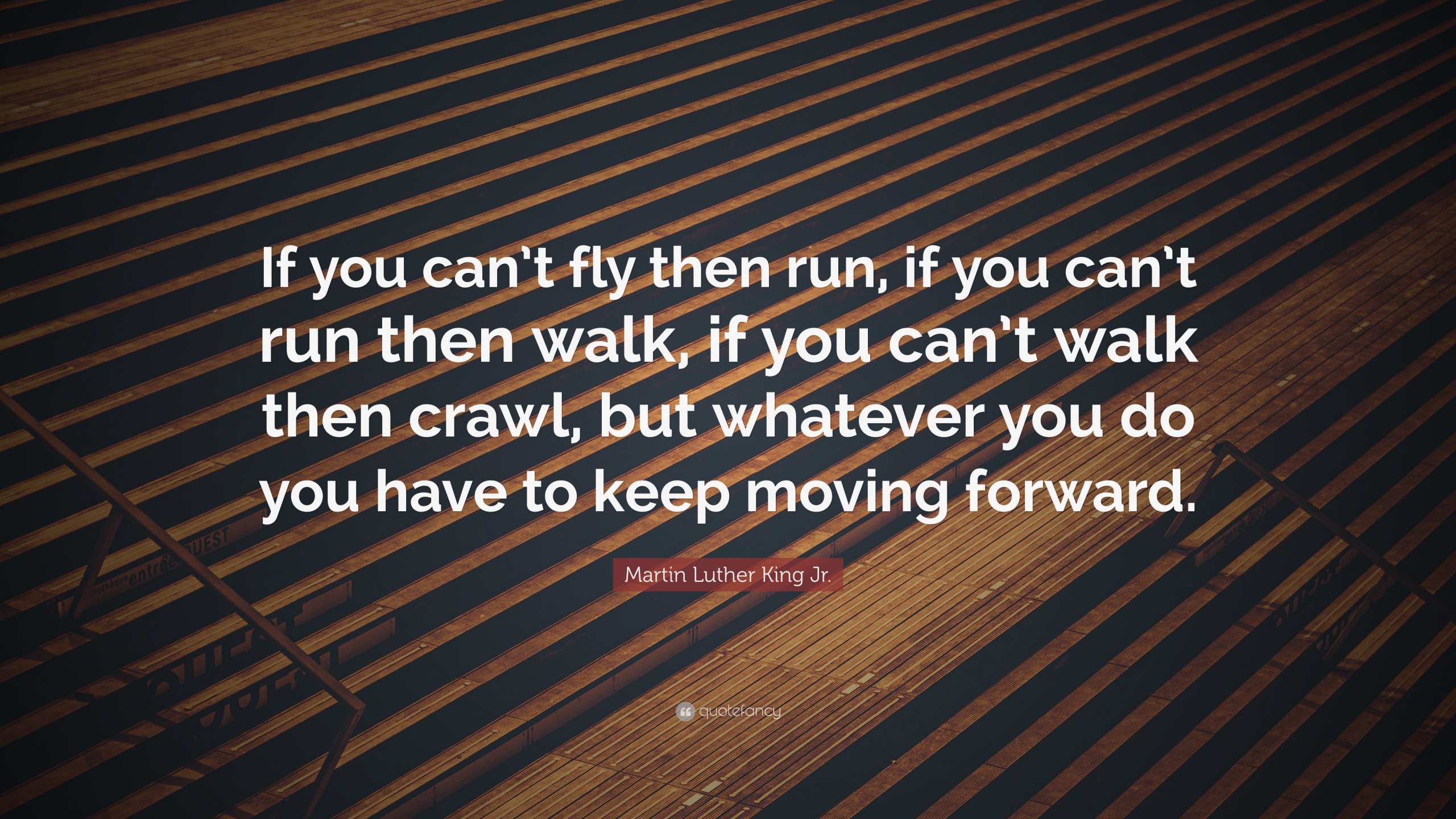
Monitor and Watch Kubernetes Cluster using best of breed tools

* Kube Forwarder
* K9S
* Lens

Demo

Development Toolkit for Data Engineers




If you can't fly then run, if you can't run then walk, if you can't walk then crawl, but whatever you do you have to keep moving forward.
Martin Luther King Jr.
quotefancy