

The Journey to Cloud

From Microservices
to Kubernetes

Niklaus Hirt

DevOps Architect / Cloud Architect

nikh@ch.ibm.com



Who am I?

Niklaus Hirt

Passionate about tech for over 35 years

- High-school in Berne
- Degree in Computer Science at EPFL
- ELCA
- CAST
- IBM

✉ nikh@ch.ibm.com

🐦 @nhirt



Agenda – DevSecOps & Kubernetes Basic Concepts

Module 0: Prepare the Labs

Module 1: IBM Cloud PoV

Module 2: Multicloud Management

BREAK

Module 3: Microservices

Module 4: Containers with Docker

Module 5: Kubernetes

Module 6: Let's get real

LUNCH

Agenda – DevSecOps & Kubernetes Basic Concepts

Module 7: Docker Hands-On

Module 8: Kubernetes Hands-On

Module 9: Kubernetes Security

Module 10: Kubernetes Security Hands-On (Optional)

Wrap-up

END of course – 16:00

Agenda – DevSecOps & Kubernetes Basic Concepts

Module 0: Prepare the Labs

Module 1: Kubernetes Mesh Networking

Module 2: Hands-on Istio

Lunch

Module 3: Kubernetes Best Practices

Module 4: Q&A

END of course – 14:30/15:00



Documentation

https://github.com/niklaushirt/k8s_training_public

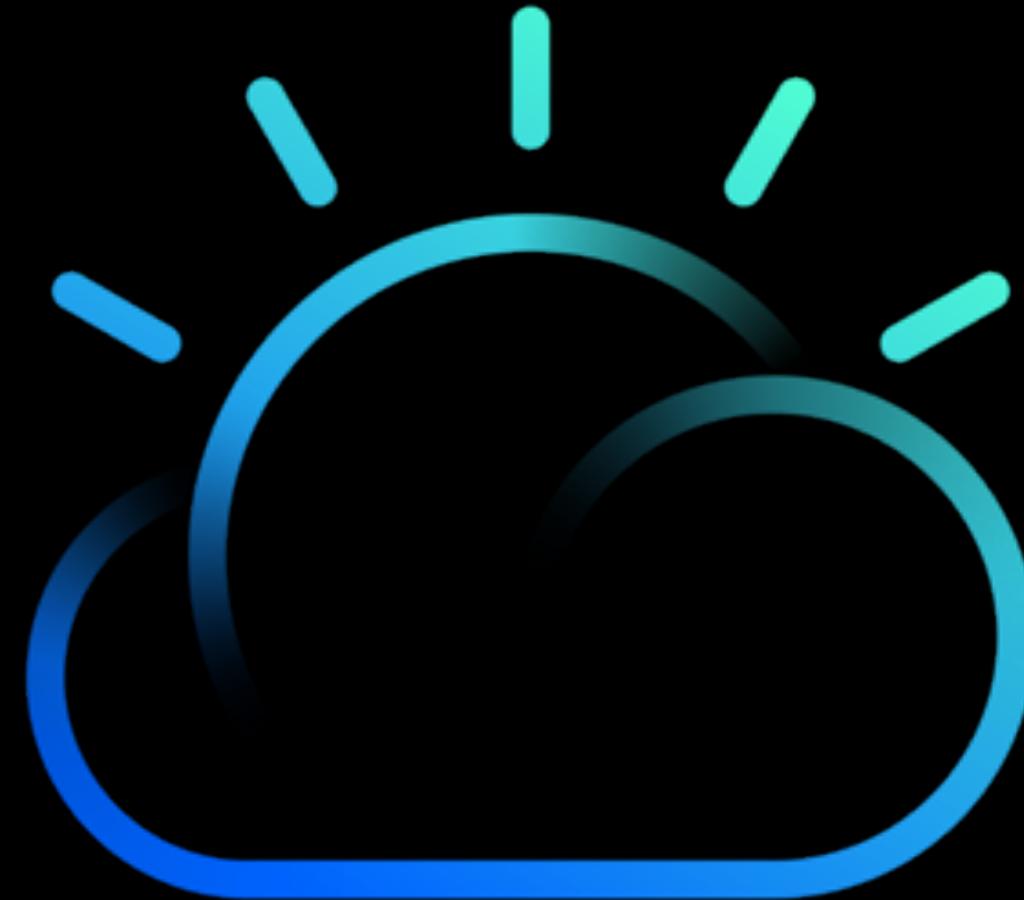
Sources

<https://github.com/niklaushirt/training>

Always-on training environment

<http://158.177.137.195:31999/>

Day 01





°° The Journey to Cloud
Prepare the Labs



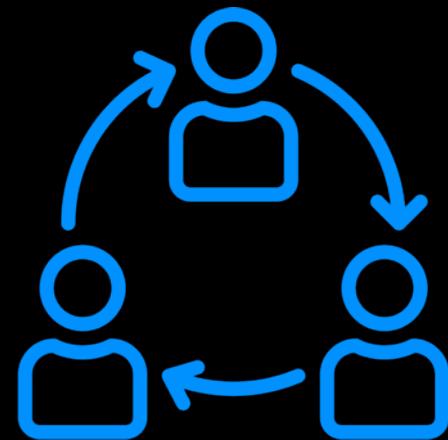
IBM Cloud

Session Objectives

Attendees will be grouped in **teams** wherever it makes sense to facilitate collaborative work.



Following some lectures will be **hands-on** work that each team collaborates to complete.



Teams

black 31701

olive 31711

peru 31715

white 31702

brown 31712

chocolate 31716

red 31703

lightblue 31713

orchid 31717

blue 31704

orange 31708

gold 31718

yellow 31705

purple 31709

pink 31719

lime 31706

maroon 31710

violet 31720

cyan 31707

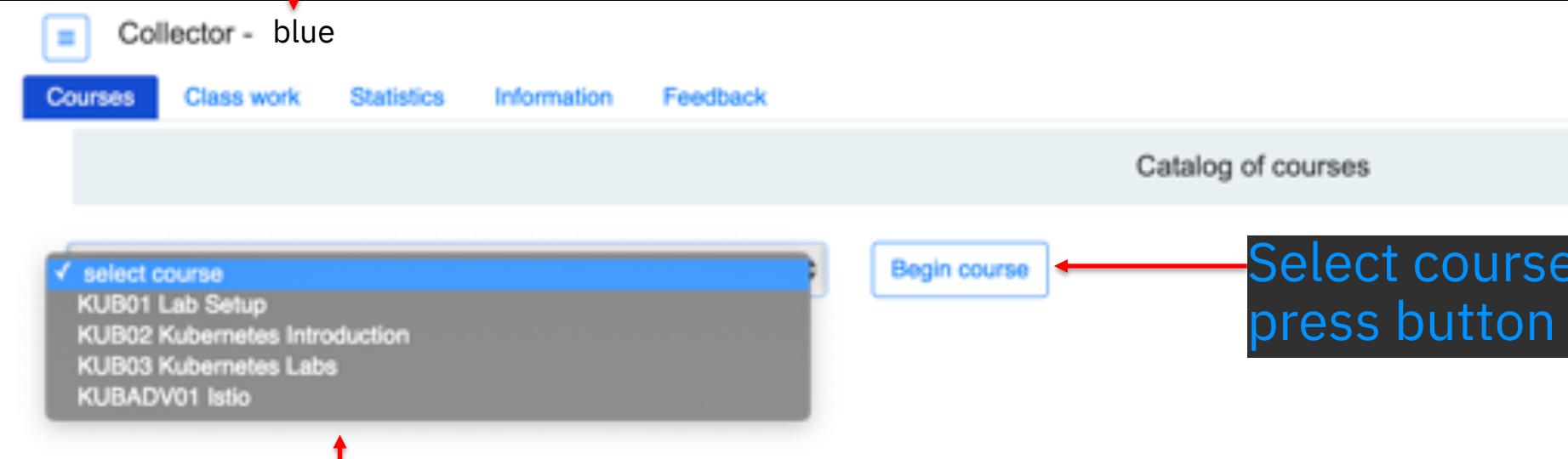
firebrick 31714

Collector - Accessing team web site

`http://158.177.137.195:{port#}`

Team name / color will be shown

blue **31704**



The screenshot shows a web browser window with the title "Collector - blue". The navigation bar includes links for "Courses", "Class work", "Statistics", "Information", and "Feedback". Below the navigation bar, a section titled "Catalog of courses" displays a list of courses under the heading "select course". The listed courses are: KUB01 Lab Setup, KUB02 Kubernetes Introduction, KUB03 Kubernetes Labs, and KUBADV01 Istio. To the right of the course list is a "Begin course" button. A large blue callout box with white text and a red border is positioned over the "Begin course" button, containing the instruction "Select course and press button to begin". Red arrows point from the text labels "Team name / color will be shown" and "Current course catalog" to their respective locations in the screenshot.

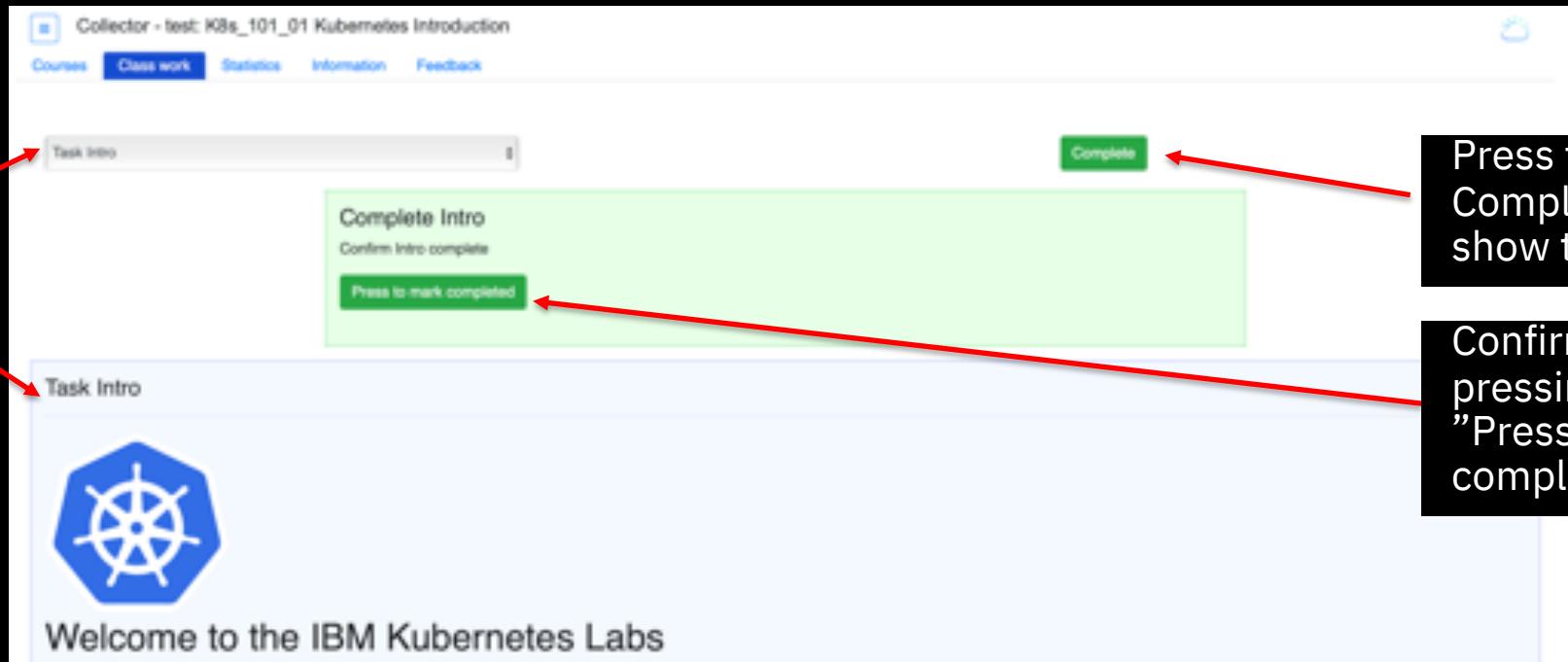
Team name / color will be shown

Current course catalog

Select course and press button to begin

Collector – Class work

Select class work and the blue portion of the screen is shown



Press the green Complete button to show the green portion.

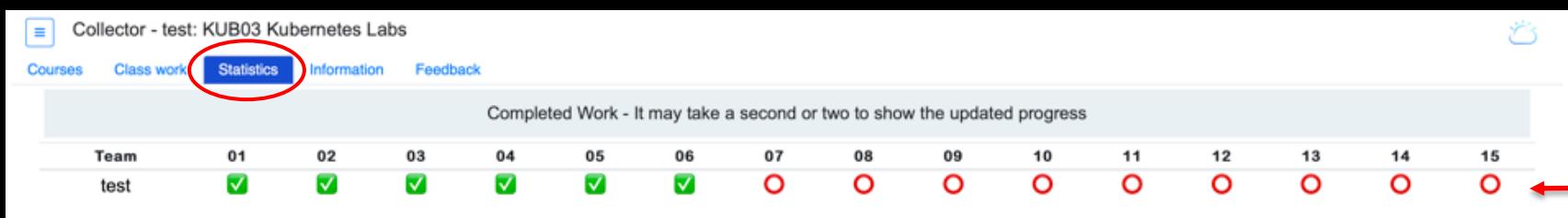
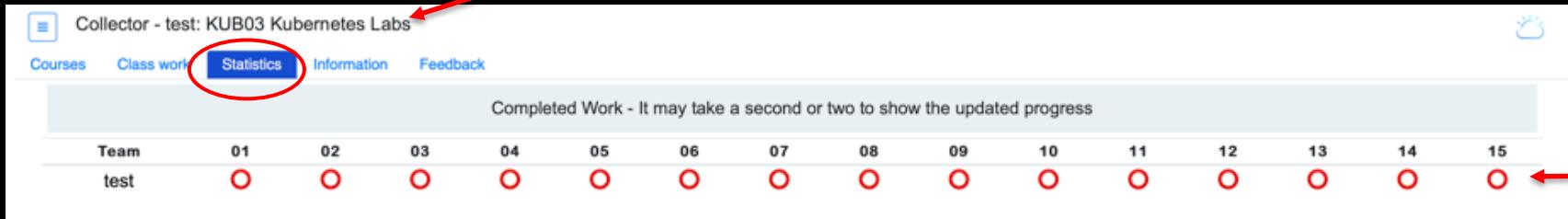
Confirm completion by pressing the green "Press to mark completed" button.



The Complete Button might not show instantly depending on the course settings

Collector – Track course completed work

Course title



Green checkmark - item is completed

Red circle - item is waiting to be completed

The number of items tracked will change based on the current course selected.

Collector – Instructor Dashboard

Remaining Time for the Lab

Collector - instructor: K8s_101_01 Kubernetes Introduction

Remaining time: 0h 29m 50s

Courses Class work Statistics Information Feedback Insight

Completed Work - It may take a second or two to show the updated progress

Team	01	02	03	04	05	06
instructor	✓	○	○	○	○	0
lightblue	✓	✓	✓	✓	✓	1
olive	✓	✓	○	○	○	2
peru	○	○	○	○	○	3
chocolate	○	○	○	○	○	4
pink	○	○	○	○	○	5
violet	○	○	○	○	○	6





JTC90 Lab Setup

EVERYBODY

Task 2: Setup Kubectl

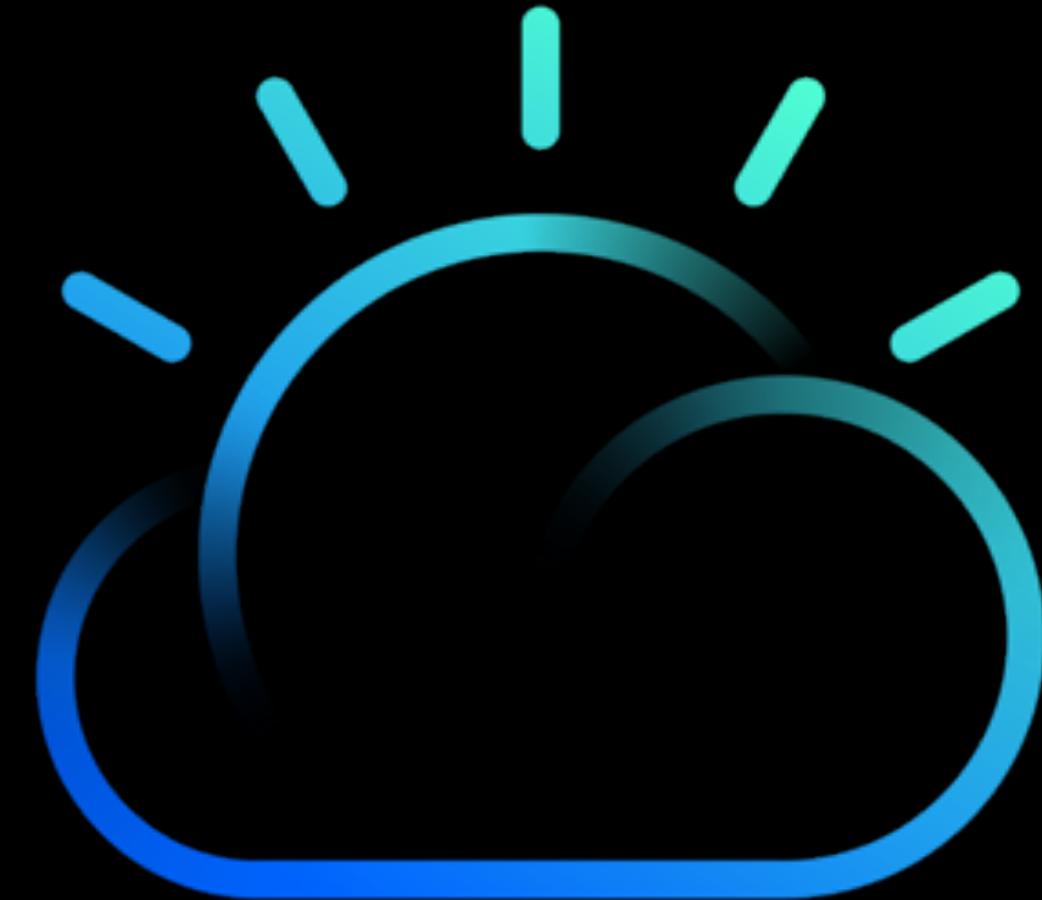
OK

Task 3: Setup git

Task 4: Final Check

?

QUESTIONS?



The Journey to Cloud **IBM Cloud PoV**

01



IBM Cloud



disruption

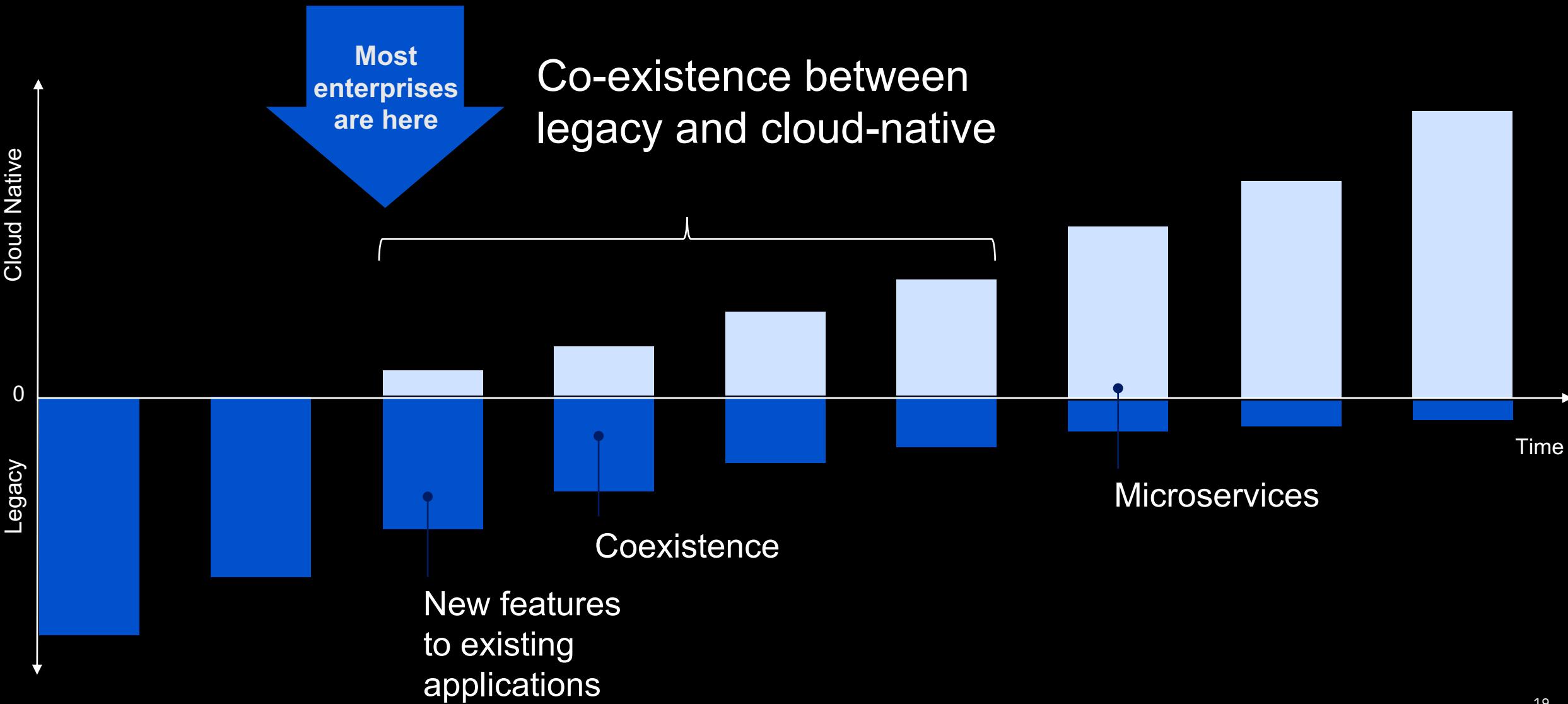
dɪs'ruptʃn/
noun

Business. a **radical change** in an industry, business strategy, etc., especially involving the introduction of a **new product or service** that creates a **new market**

Digital Transformation is the new normal

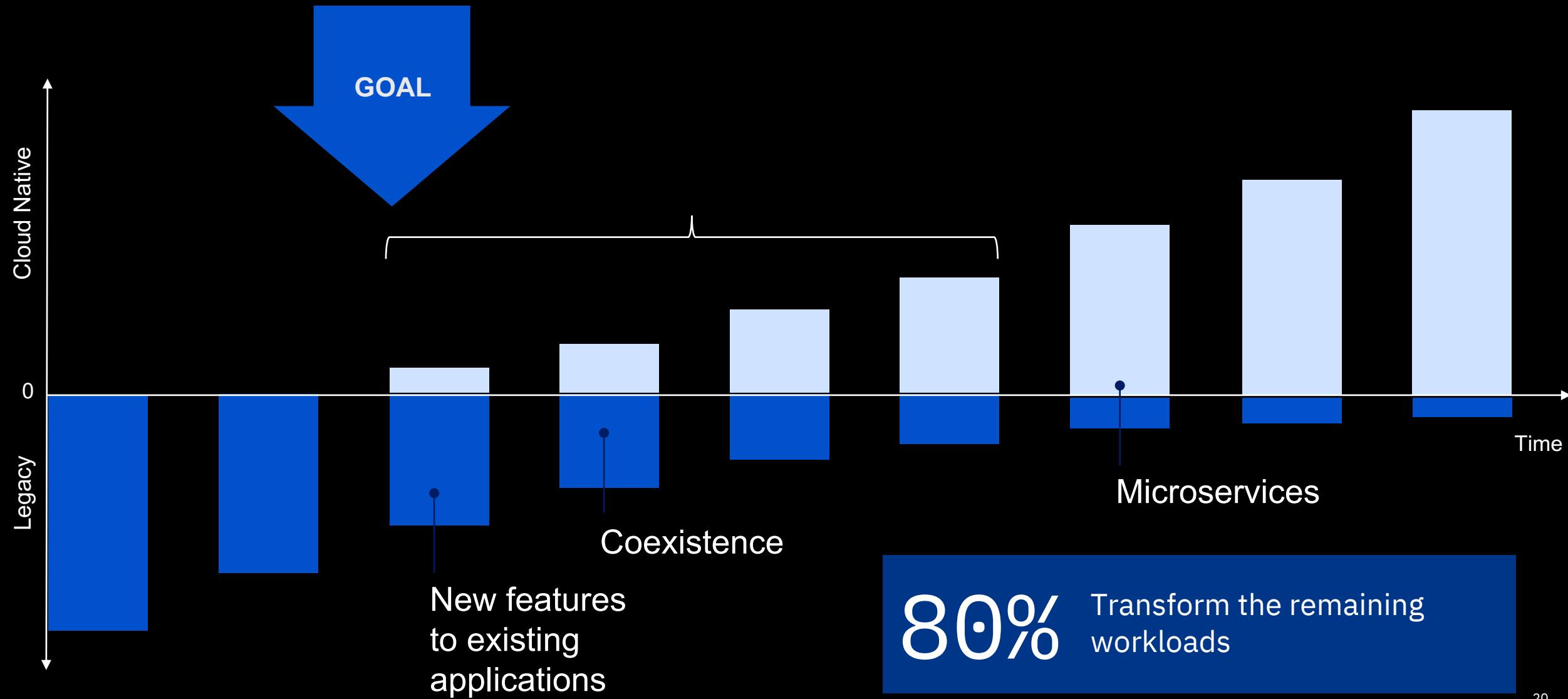
Digital Transformation - Way to go

Cloud native and legacy apps will co-exist for the next 10+ years

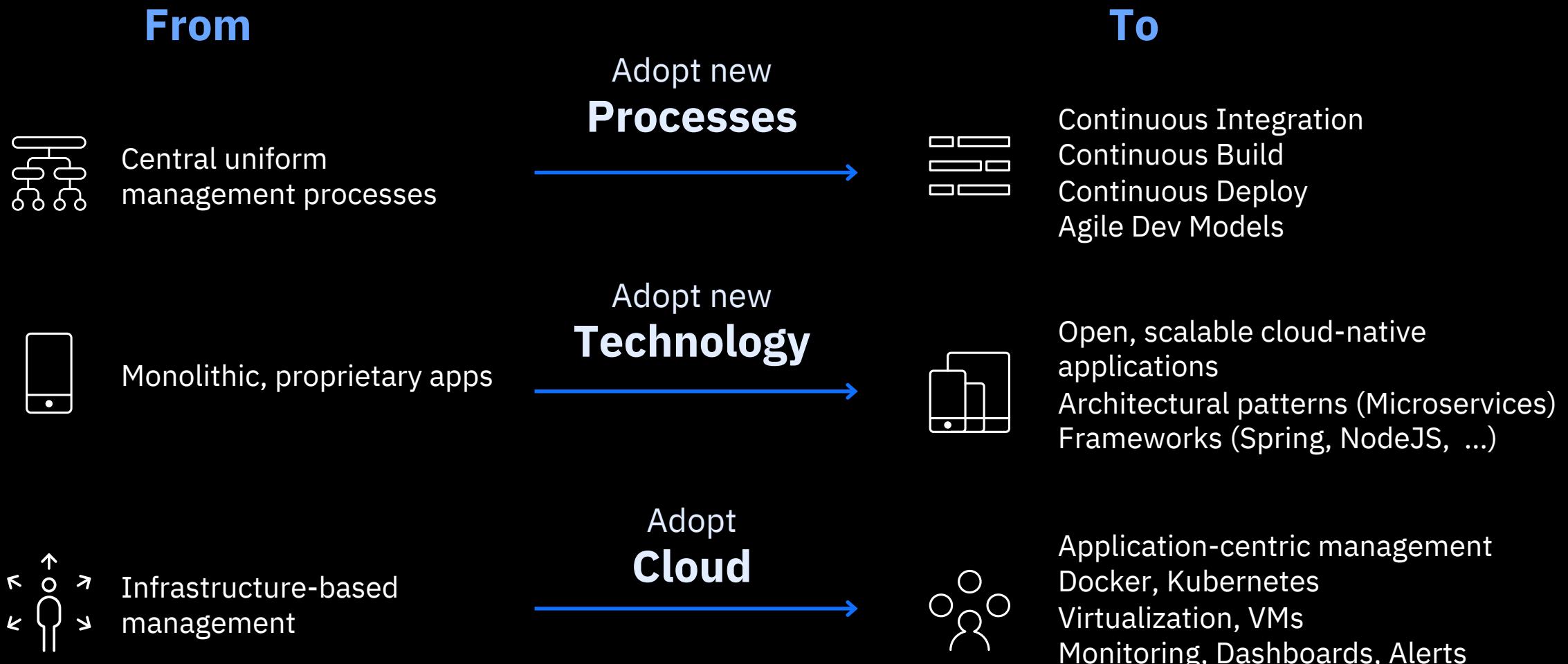


Digital Transformation - Journey to cloud

Cloud native and legacy apps will co-exist for the next 10+ years



Digital Transformation – How to get there



DEVOPS

What is DevOps

 just the saddest server
@sadserver

Follow ▾

DevOps is a software engineering culture and practice of putting horrors into containers and then talking about Kubernetes at conferences.

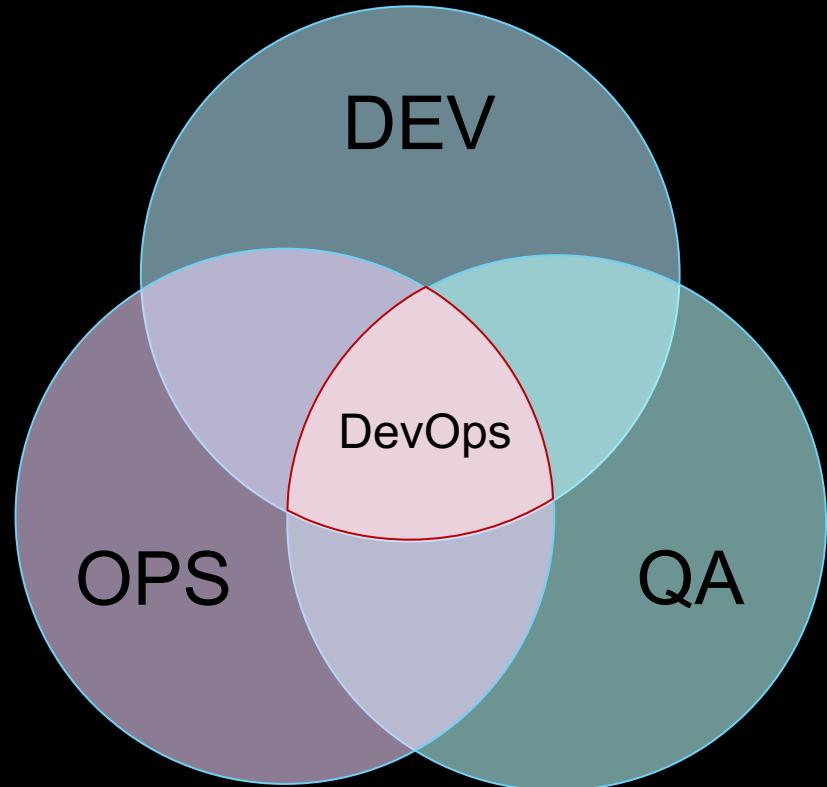
9:48 am - 26 Jun 2018

1,348 Retweets 3,050 Likes



40 1.3K 3.1K

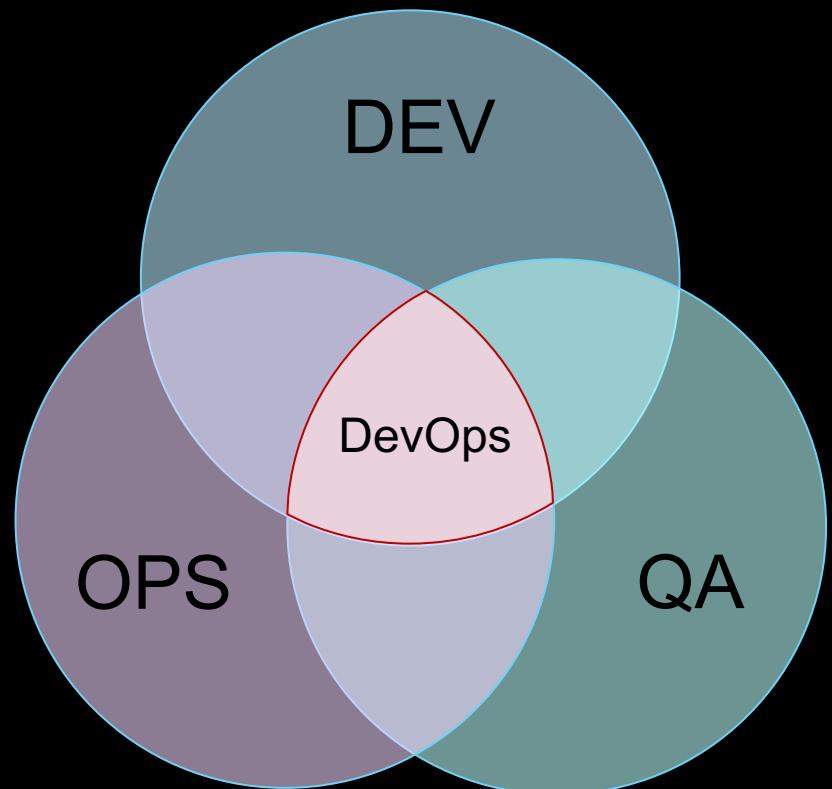
DevOps according to Wikipedia™



DevOps is a software development method that stresses communication, collaboration and integration between software developers and IT professionals.

DevOps is a response to the interdependence of software development and IT operations. It aims to help an organization rapidly produce software products and services.

DevOps according to Wikipedia™



CULTURE

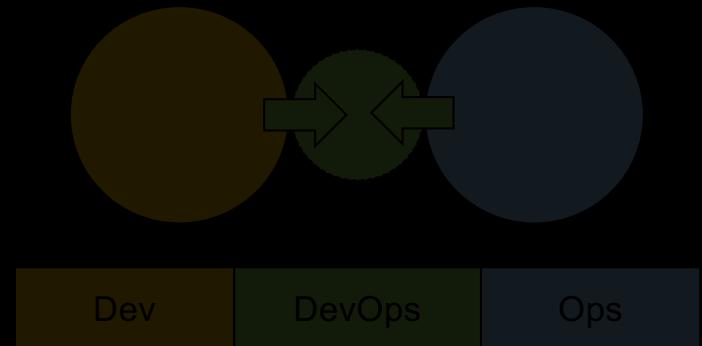
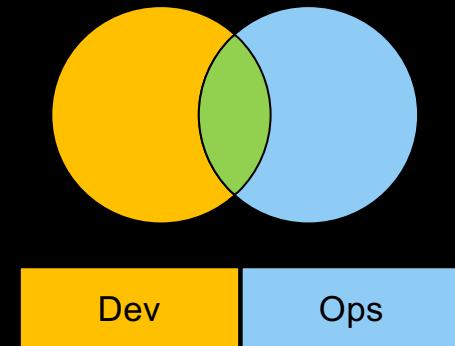
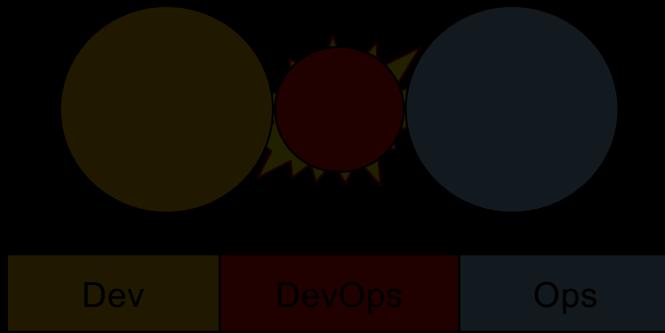
PROCESS

DevOps is a software development **method** that **communication**, **collaboration** and **integration** **developers** and **IT professionals** and **IT professionals**.

PEOPLE

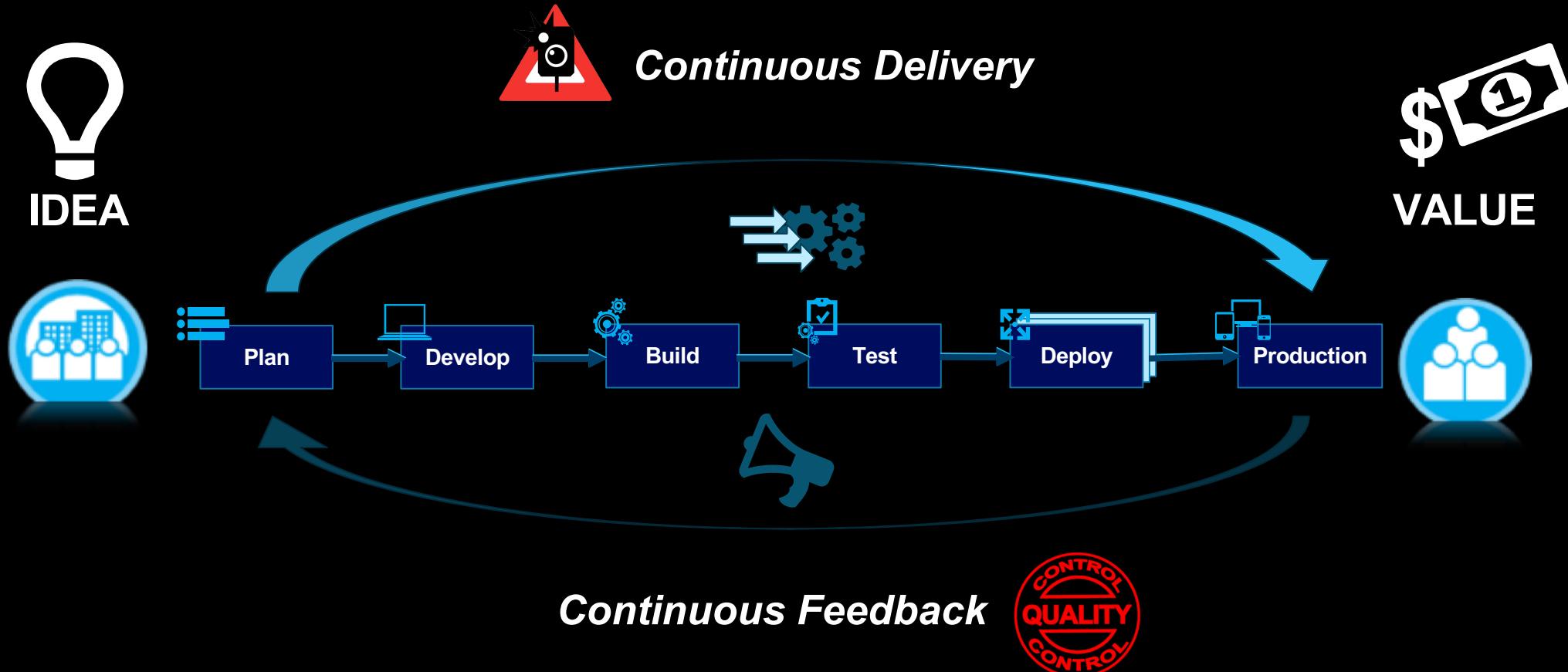
Tools, Processes
AND **People!**

DevOps - Culture



DevOps – Process

Business Success - get capabilities to the customers



DevOps – Adoption

What DevOps is NOT

- DevOps is not simply ***combining Development & Operations teams***
- DevOps is not a ***separate team***
- DevOps is not only a ***tool***
- DevOps is not only ***automation***
- DevOps is not a ***one-size-fits-all strategy***

DevOps – Adoption

Getting started on the DevOps journey

- **Be honest**

- **Assess** your own DevOps maturity and aspirations – where are you and where do you want to be?

- **Don't go nuts**

- **Experiment** on a few smaller, low-risk projects – set up a couple of “**two pizza” teams** comprised of Dev/QA + Ops
 - Consider creating a (temporary!) **DevOps "center of excellence"**

- **Cultural change takes time – take reasonable steps**

- Work on **shifting culture** through teambuilding, cross-training, improved communication, perhaps more collaborative tooling

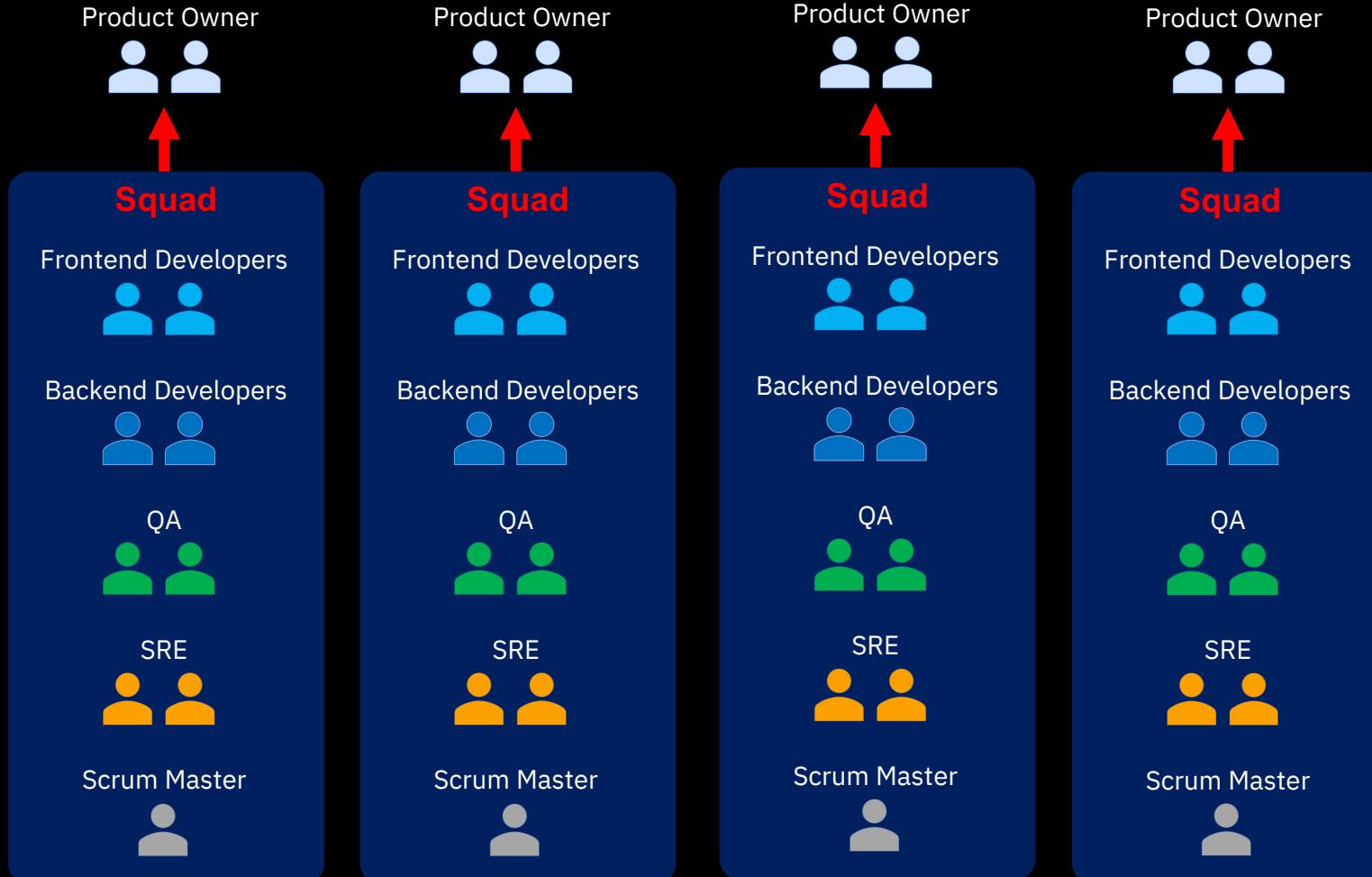
- **Don't try to change the laws of physics**

- Reality is that your “Systems of Record” applications move at a **different pace** than your “Systems of Engagement” apps (e.g. mobile)

DevOps – Adoption

The Five Ideals as of Gene Kim

Agile Team Organisation: Squads



Squads contains more than just developers – remove „stigma“

→ rename „Teams“ to „Squads“ 😊 or Crew, Party, ...

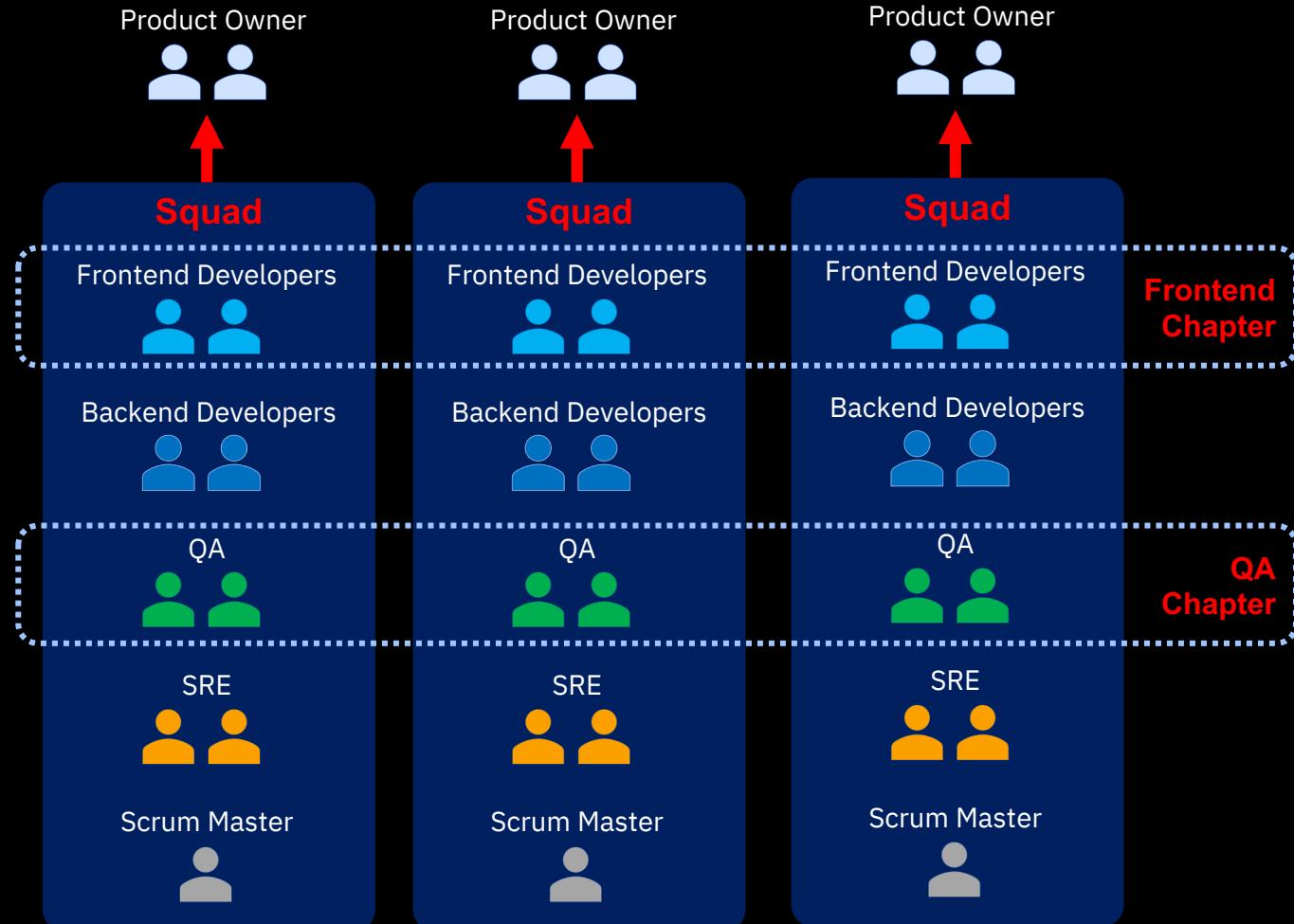
Fully autonomous, cross functional team that has full responsibilities for an application component/functionality/microservice and little to no dependencies on others

5-7 people ideally

DevOps – Adoption

The Five Ideals as of Gene Kim

Agile Team Organisation: Chapters



Chapters are a group or team members working within a special area

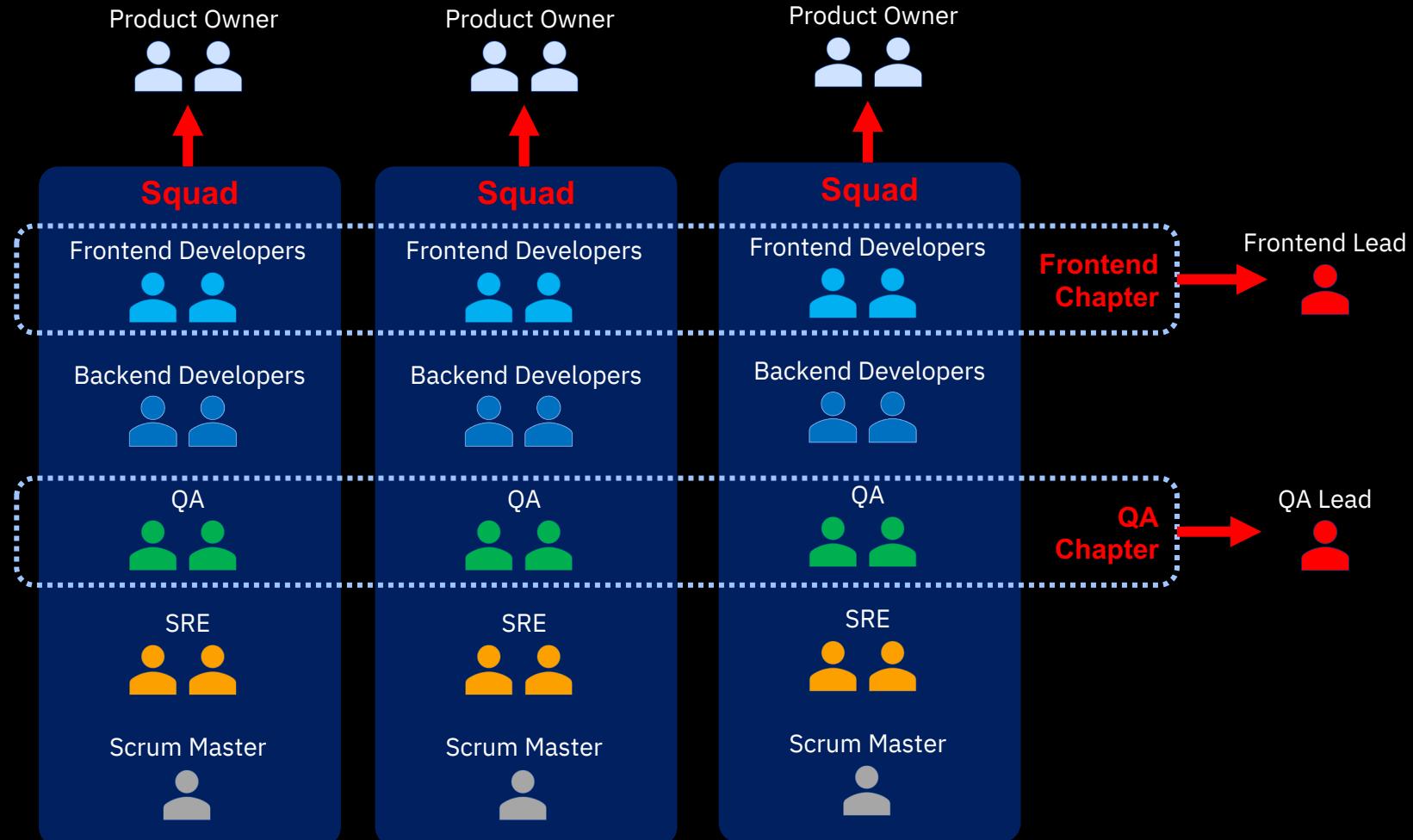
Relay experience to the rest of the chapter and discuss on how other squads can use it.

Promotes team collaboration and innovation.

DevOps – Adoption

The Five Ideals as of Gene Kim

Agile Team Organisation: Chapters



A **Chapter Lead** is the line manager for chapter members.

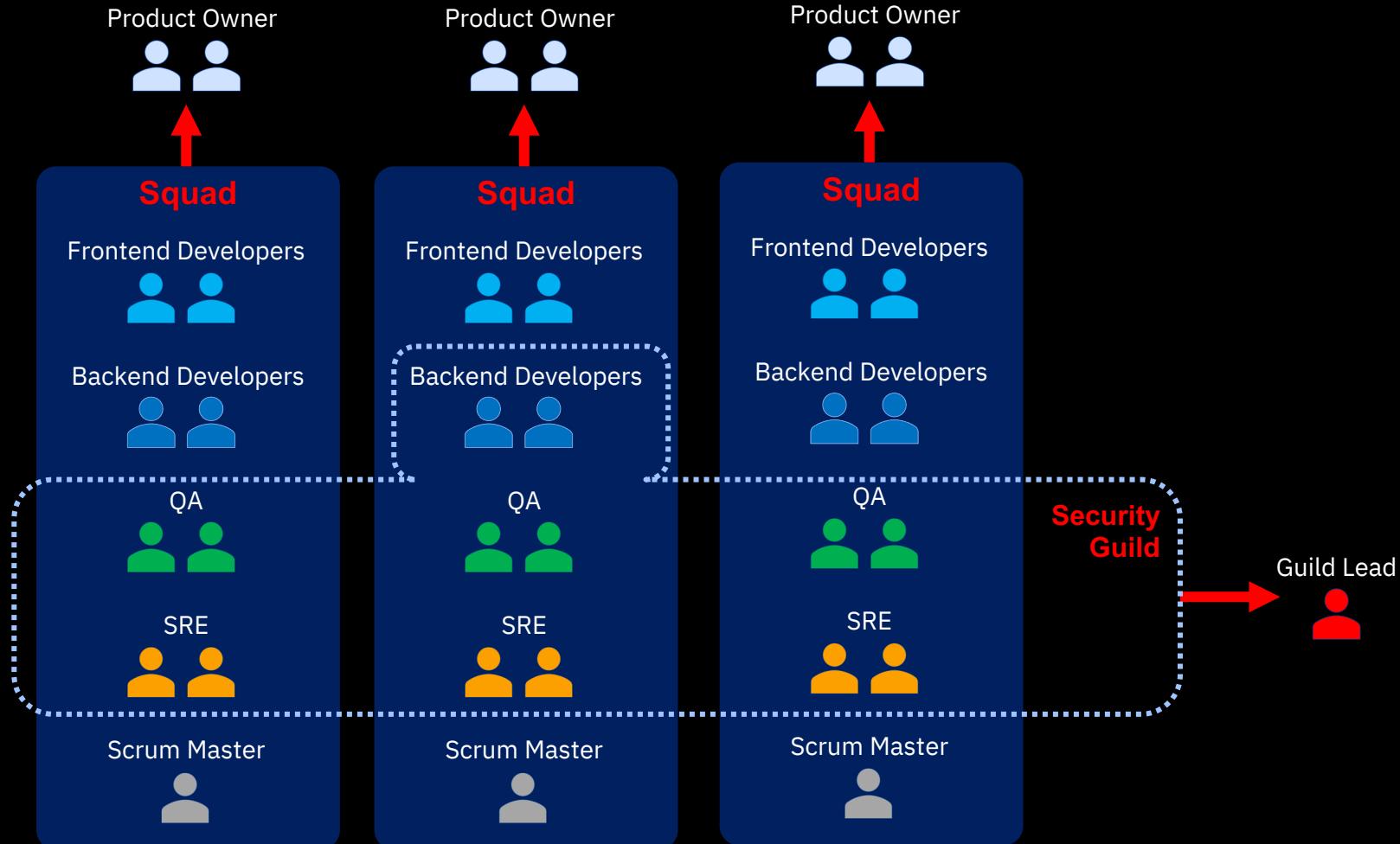
Responsible for developing people .

Remain part of a squad and still do day-to-day work.

DevOps – Adoption

The Five Ideals as of Gene Kim

Agile Team Organisation: Guilds



Guilds cover a problem or something that can be improved and get together a small group of team members together to solve it.

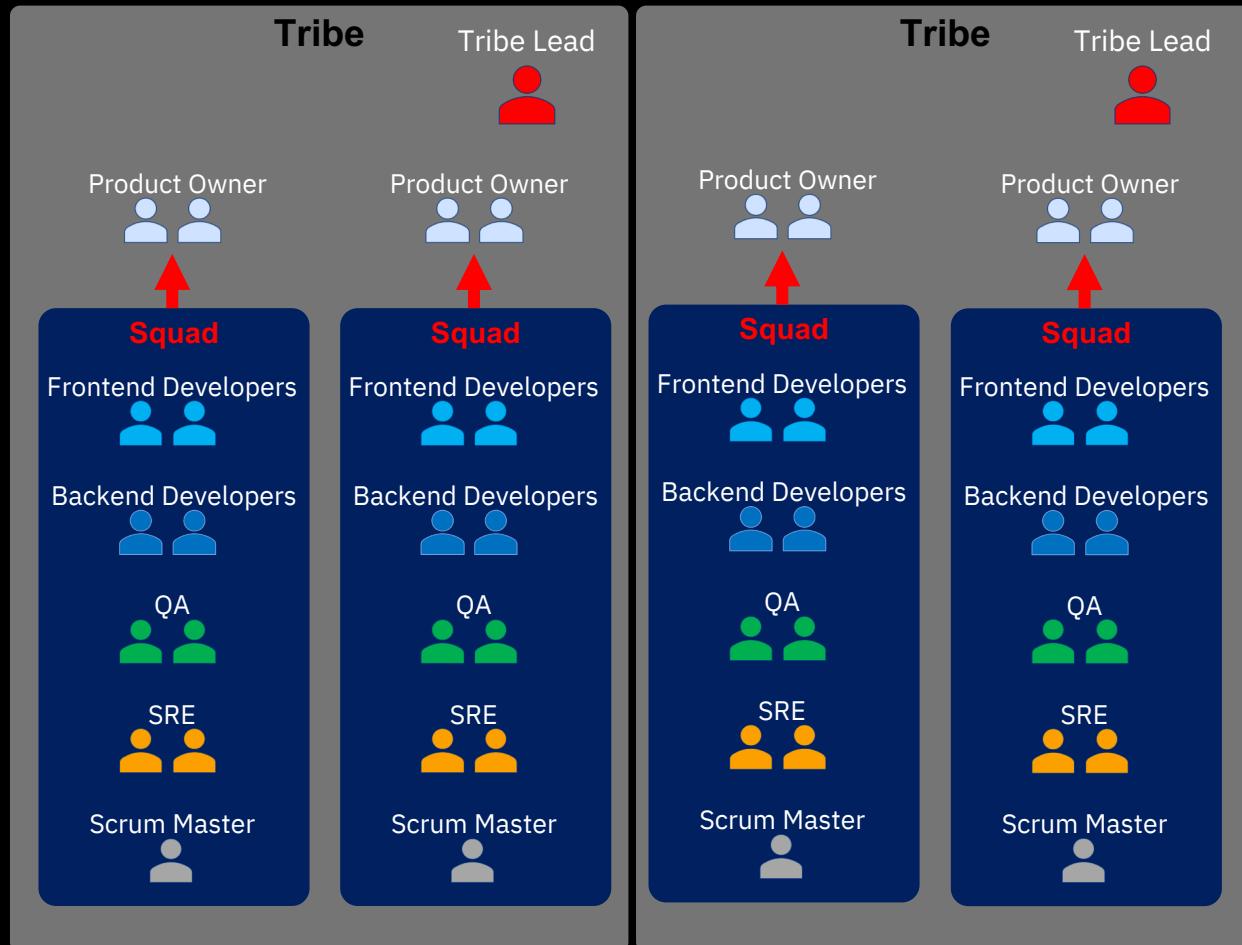
Examples:

- Performance monitoring / optimization
- Testing automation
- Security & vulnerabilities
- Services & Architecture

DevOps – Adoption

The Five Ideals as of Gene Kim

Agile Team Organisation: Tribes



Tribes are a collection of squads that work in related areas like mobile app, backend infrastructure, ...

“Incubator” for the **Squad** mini-startups

Challenging to implement

Only for large complicated infrastructure that needs to be split this way

< 100 people (Dunbar number)

DevOps – Adoption

The Five Ideals as of Gene Kim

«**Technical Debt** is what you feel the next time
you want to make a change.»

Usually refers to things we need to **clean up**, or where we need to
create or restore simplicity, so that we can **quickly, confidently,**
and safely make changes to the system

Ward Cunningham in 2003

DevOps – Culture

Start by understanding your Flow

Pathological Power-oriented	Bureaucratic Rule-oriented	Generative Performance-oriented
Low co-operation	Modest co-operation	High co-operation
Messengers shot	Messengers neglected	Messengers trained
Responsibilities shirked	Narrow responsibilities	Risks are shared
Bridging discouraged	Bridging tolerated	Bridging encouraged
Failure leads to scapegoating	Failure leads to justice	Failure leads to inquiry
Novelty crushed	Novelty leads to problems	Novelty implemented
Characterized by large amounts of fear and threat. People often hoard information or withhold it for political reasons, or distort it to make themselves look better.	Organisations protect departments. Those in the department want to maintain their “turf,” insist on their own rules, and generally do things by the book — their book	Organisations focus on the mission. How do we accomplish our goal? Everything is subordinated to good performance, to doing what we are supposed to do.

DevOps – Adoption

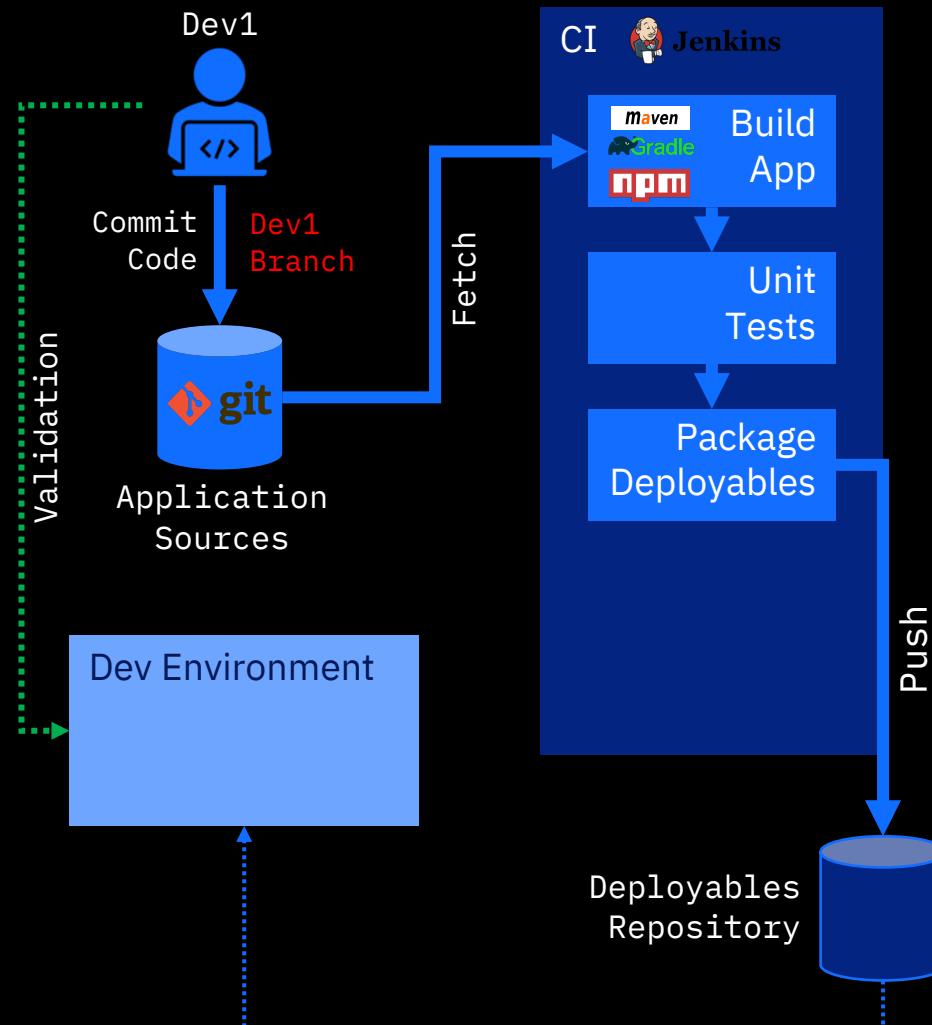
Measuring Adoption

	High performers	Low performers
Lead time for changes		
The time it takes to go from code committed to code successfully	1 hour	1-6 month
Deployment frequency		
frequency of <i>production</i> deployments matters is because it tells you how often you're delivering something of <i>value</i> to end users	1 per day	1 month
Time to restore service		
(MTTR) metric calculates the average time it takes to restore service.	1 hour	>1 week
Change failure rate		
how often deployment failures occur in production that require i	0-15%	46-60%

... predictive of software delivery, operational performance, and organizational performance, and they correlate with burnout, employee engagement, and so much more."

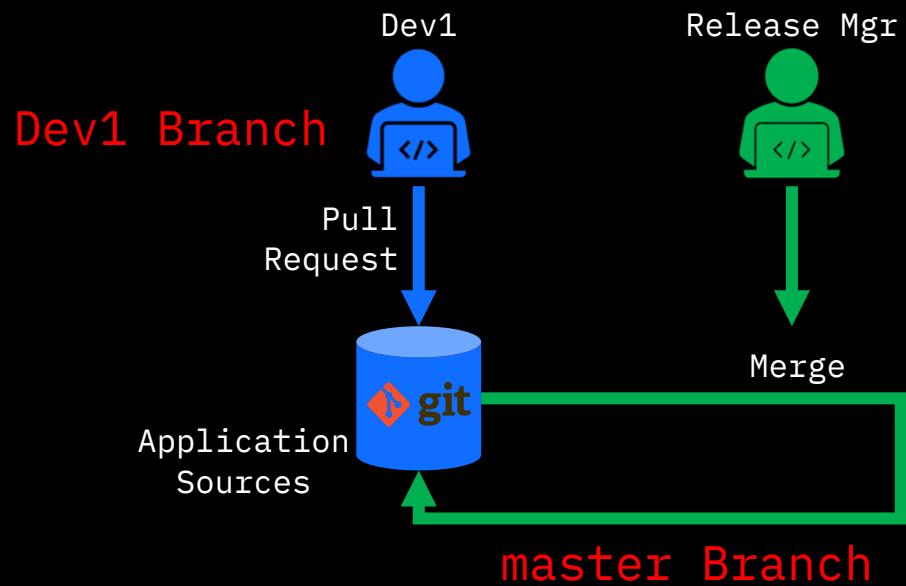
DevOps – CI/CD Pipeline

Development – Dev1 Branch



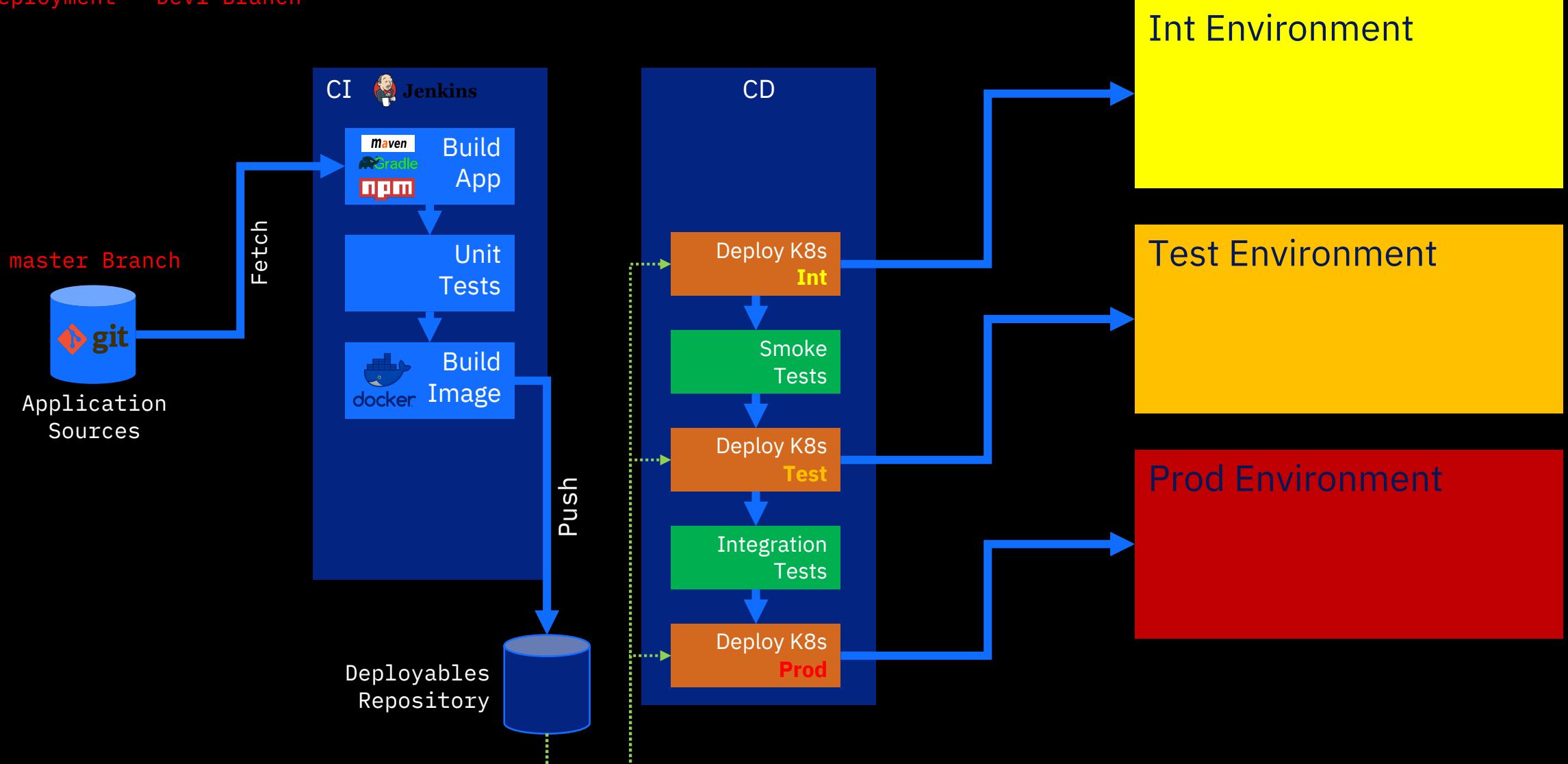
DevOps – CI/CD Pipeline

Merge – Dev1 Branch to master Branch



DevOps – CI/CD Pipeline

Deployment – Dev1 Branch



DevOps – **Git**Ops

GitOps builds on DevOps with **Git as a single source of truth** for declarative infrastructure and applications - the whole system.

1. Having a **completely automated delivery pipeline** that can roll out changes to your infrastructure when changes are made to Git.
2. Operating a fast paced business 24/7 requires **monitoring and observability** baked into the beginning. **Security** is of critical importance.
3. Everything has to be **version controlled** and stored in a single source of truth from which you can recover.

DevOps – **Git**Ops

Full Audit Trail

GitOps is using git, which is an excellent database to persist the changes made to the system.

Everything as a code

In the cloud, both CI and CD should be defined as code first. The desired state of environments should also be set in full as a declarative code.

...

Today, application security is mostly an **afterthought**, that is run in pre-production and often perceived as a **roadblock** to staying ahead of the competition.

Examples

- Getting access to a Git repository (IAM)
- Open firewall ports
- Getting credentials for an API
- Getting certificates

Dev**Sec**Ops

Putting Security in your DevOps transformation

DevSecOps means thinking about application and infrastructure security from the start.

1. The idea is to get **security** back in to the **lifecycle** → shifting security left
2. The goal is to make **security** as **silent** and **seamless** as possible
3. «Everyone is responsible for security» **mindset**
 - Developers write secure code
 - SREs secure the environments

Dev**Sec**Ops

Putting Security in your DevOps transformation

But we still need clear **responsibilities!**

Most organizations have clear **governance of risk**, and out of that we derive **security policies**.

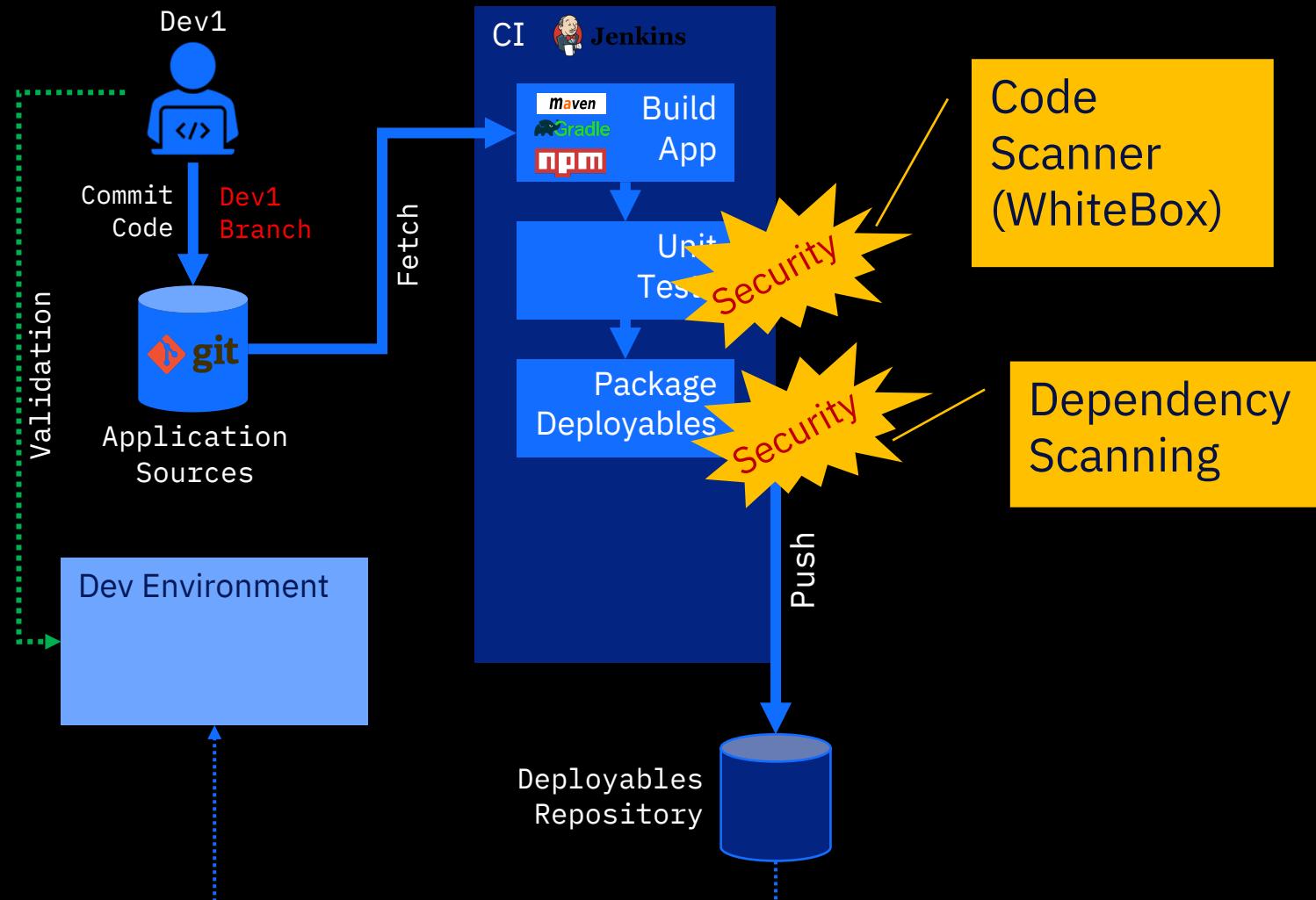
In order to bring Sec into DevOps is to **bake** those **policies** into the processes.

But we still need clear responsibilities!

For example, when you are writing code,
do the developers have a tool that checks for vulnerabilities during the local build
process at **build time**
or do you do this within your **CI process** using Jenkins?
or do you do this within your **CD process** before deployment?

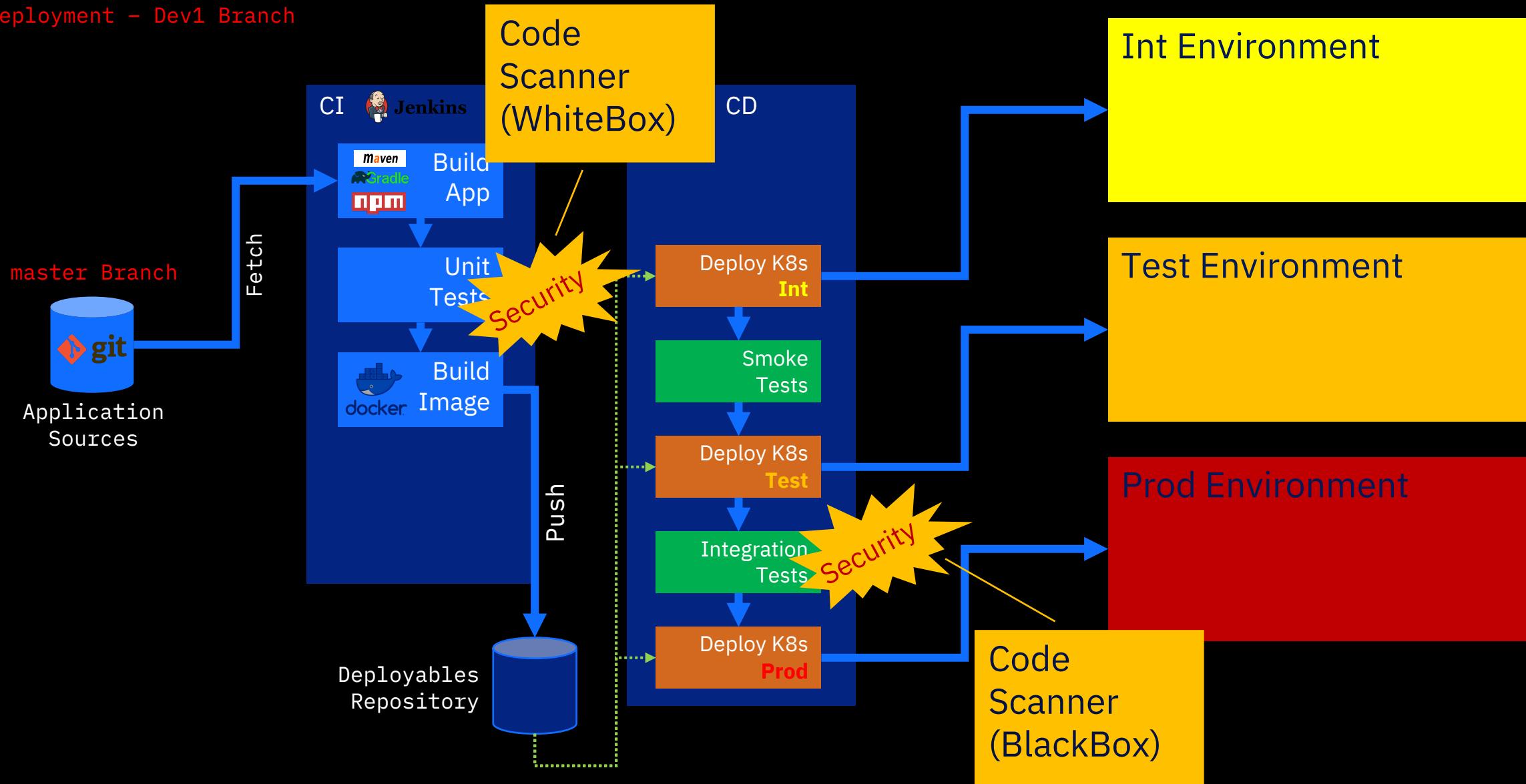
DevSecOps – CI/CD Pipeline

Development – Dev1 Branch



DevSecOps – CI/CD Pipeline

Deployment – Dev1 Branch



Challenges with Traditional Management and DevOps

Disparate monitoring tools

Lower visibility across multicloud

Slow delivery & management processes

Sequential framework, manual

Security and quality an afterthought

Inserted just before release

>60%

of customers state they
don't have the tools and
procedures to manage and
operate in a complex
multicloud environment

Cloud accelerates Digital Transformation

- Businesses are looking **Innovate** with the latest technology from any source
- **Access** more types of data, analytics and AI from anywhere
- **Optimize** on the right model and improve ROI



81%
of companies place the
**personalized
customer
experience**
in top 3 priorities (Accenture)

Digital Transformation - One step further - Hybrid MultiCloud

New challenges and new opportunities

- Drive competitive advantage
- Retain and Delight customers
- Adapt to regulatory governance
- Take care of business, even the parts that aren't transforming (yet)

A real world look at multicloud

94%

of enterprise customers are using multiple cloud environments (public and/or private)

67%

of enterprise customers are using more than one public cloud provider
(expected to remain constant or increase by 2022)

IT is all about better applications, faster

	Movement between clouds	73% priority concern
	Connectivity between clouds	82% priority concern
	Consistency of management	67% priority concern

Journey to cloud requires an open, hybrid approach

Advise

Advise on every step of the journey to cloud

Move

Migrate and modernize workloads, data, and applications

Build

Build innovative applications and experiences

Manage

Manage, govern and optimize hybrid multicloud environments

Portable

Building once, deploying anywhere for open, flexible data and workload placement

Container platform



Hybrid cloud platform

Scalable and efficient

Consistent management services that ensure operational integrity and reduce cost

Common operational services



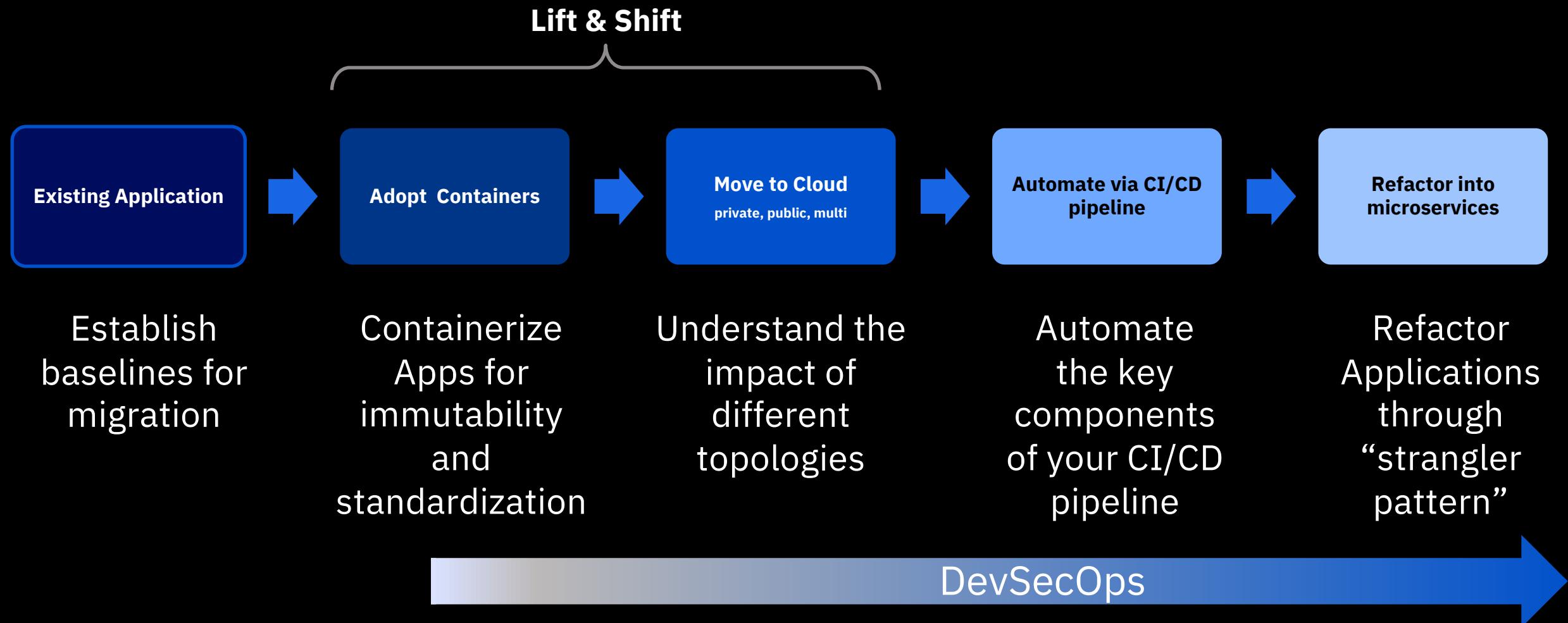
Enterprise-grade

Integrated and secure containerized software for visibility and governance

Trusted and up-to-date software

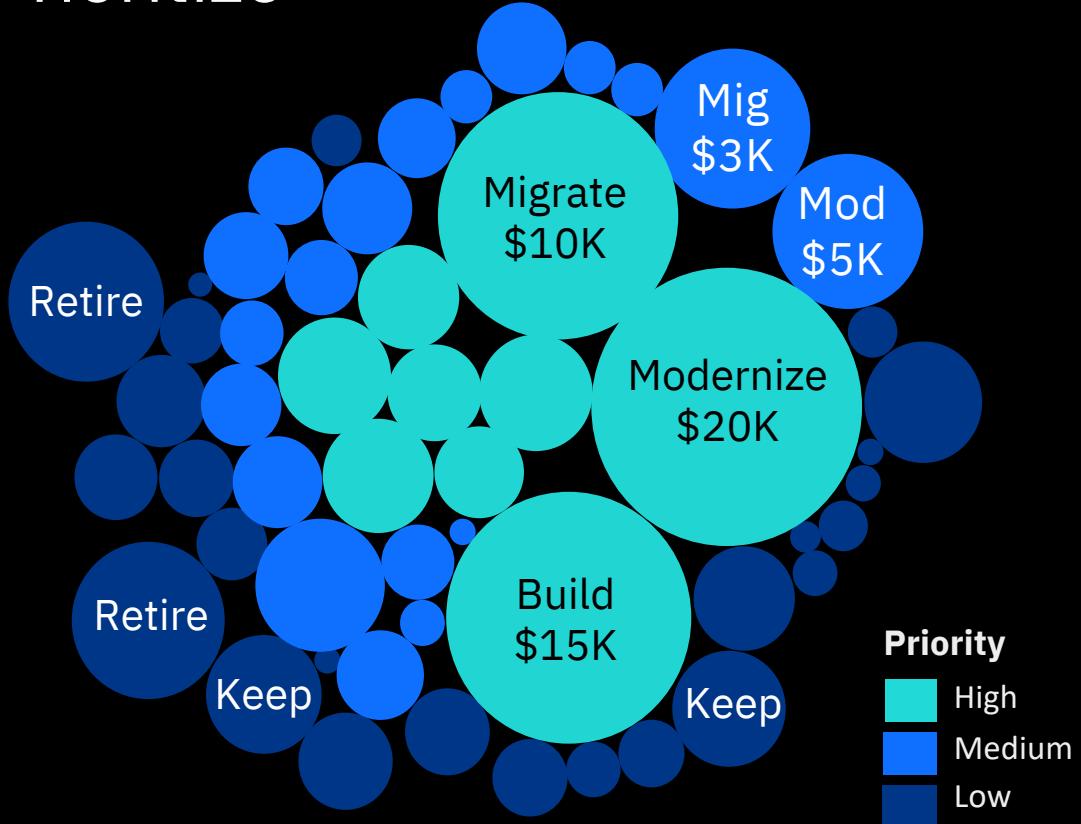


Digital Transformation - Application modernization

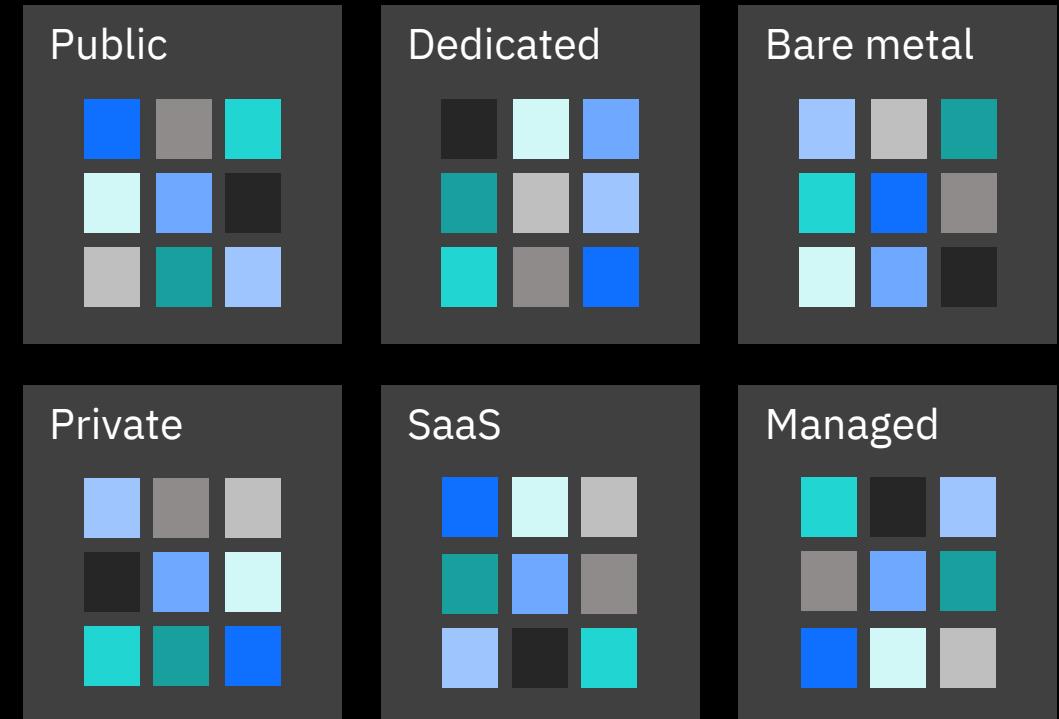


Digital Transformation – Workload positioning

Prioritize



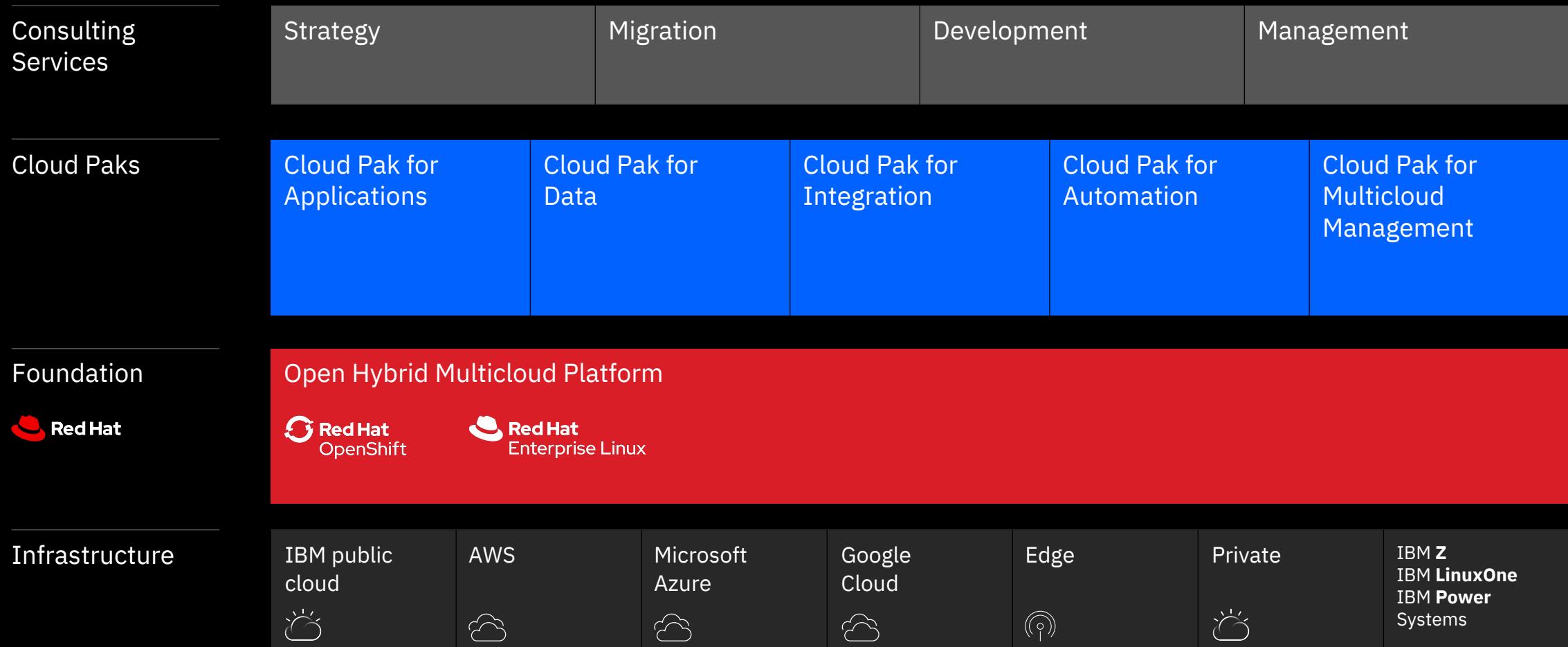
Optimize



Workload affinity

Cost/benefit

IBM Cloud Strategy and Architecture: The Hybrid Multicloud Platform



Kubernetes

Simple Kubernetes is easy



Kubernetes



Kubernetes

*Enterprise grade Kubernetes is **hard***



Kubernetes

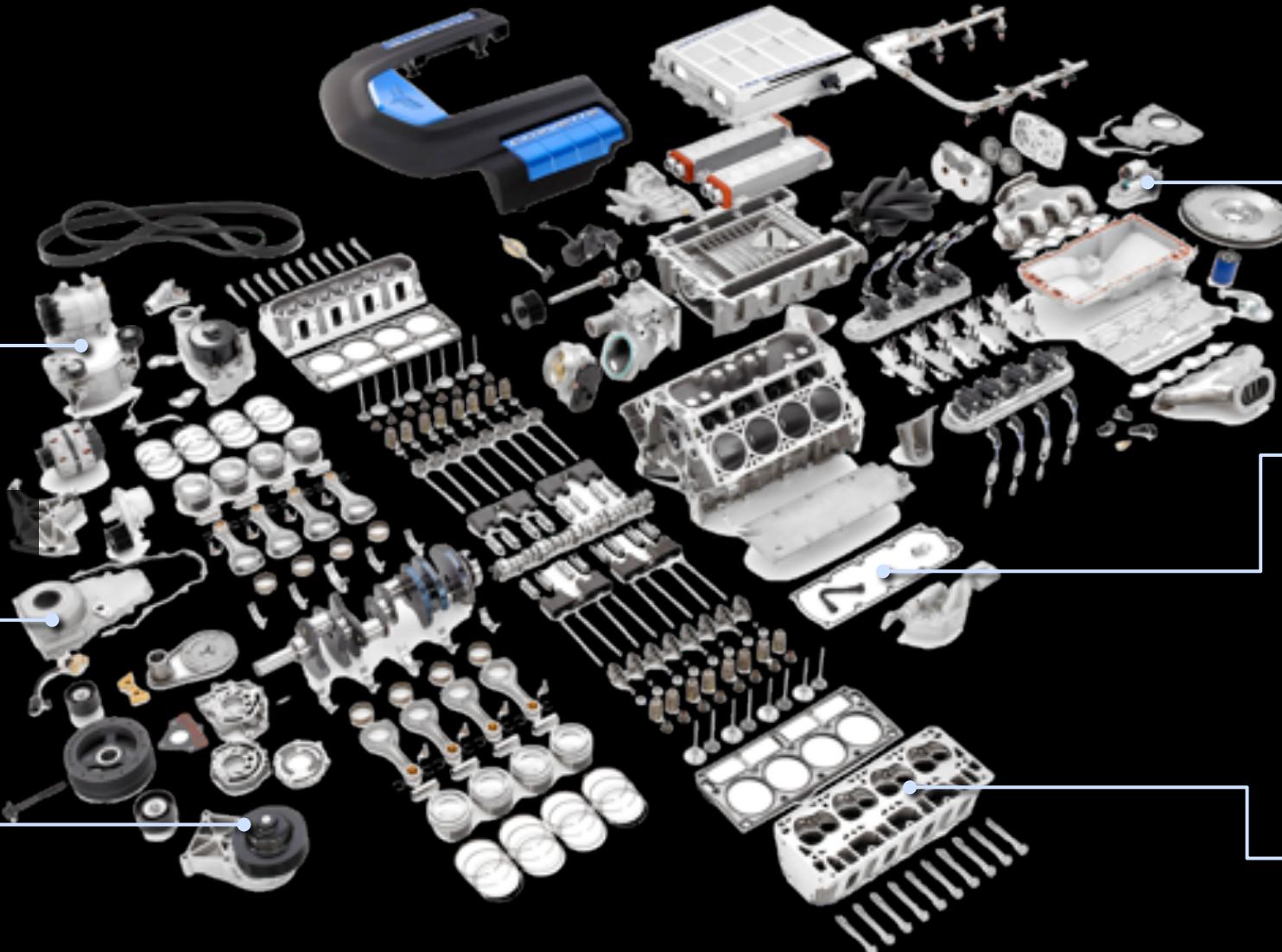
Networking

Deployment Catalog

Logging

Monitoring

Security



Kubernetes Challenges



kubernetes
<https://github.com/kubernetes/>



calico
<https://github.com/projectcalico/>



helm
<https://helm.sh/>

So Much Open Source ... Why Not Build my own then ?

Handle dependencies

- Over 100 OS Services and Technologies to stitch together
- Many to many Interfaces to maintain
- Regular changes to versions and APIs (at least every 2 to 3 months)

Get experts

- Experts are hard to find and very expensive

Get support

- Need to participate in OS communities for support



istio
<https://github.com/istio/istio>



grafana
<https://github.com/grafana/grafana>



knative
<https://github.com/knative/>

Kubernetes

*Enterprise grade Kubernetes is **hard***

Building your full DIY Kubernetes Stack

Testing your DIY Kubernetes Stack

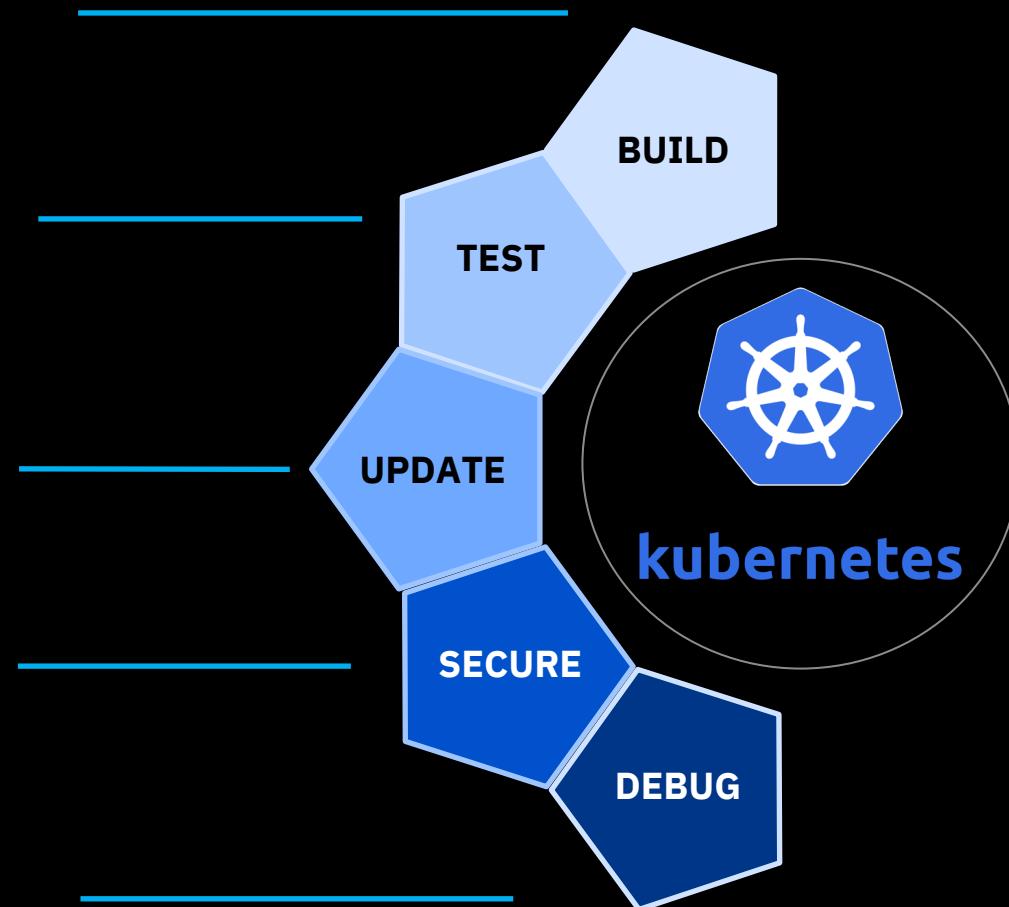
Keeping up with Kubernetes (every 90 days)

Updating add-on projects

Patching OS Security CVEs (~1 per week)

Finding full-stack problems

Fixing full-stack problems



All **without technical support**

Kubernetes

RedHat OpenShift provides turnkey solutions

Building your full DIY Kubernetes Stack

Testing your DIY Kubernetes Stack

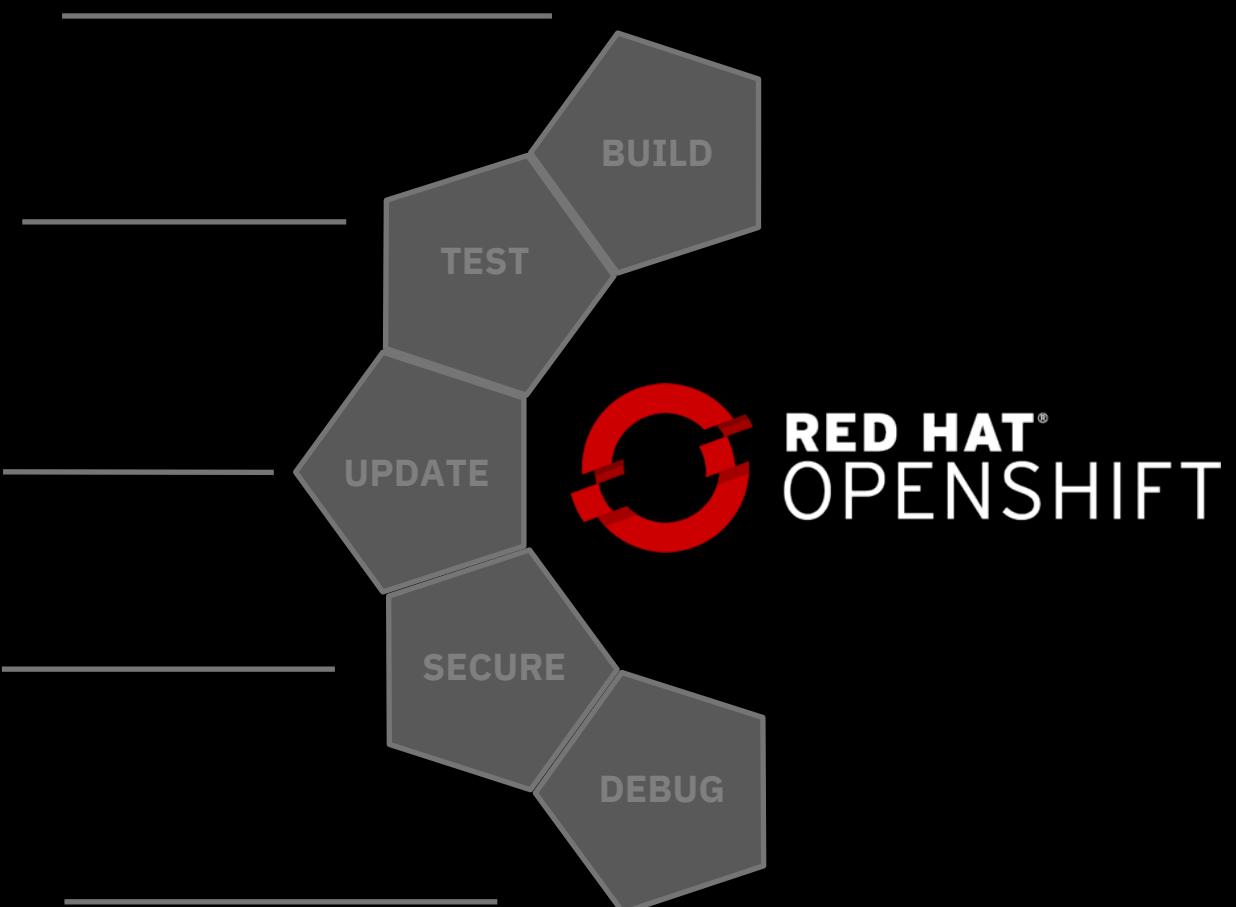
Keeping up with Kubernetes (every 90 days)

Updating add-on projects

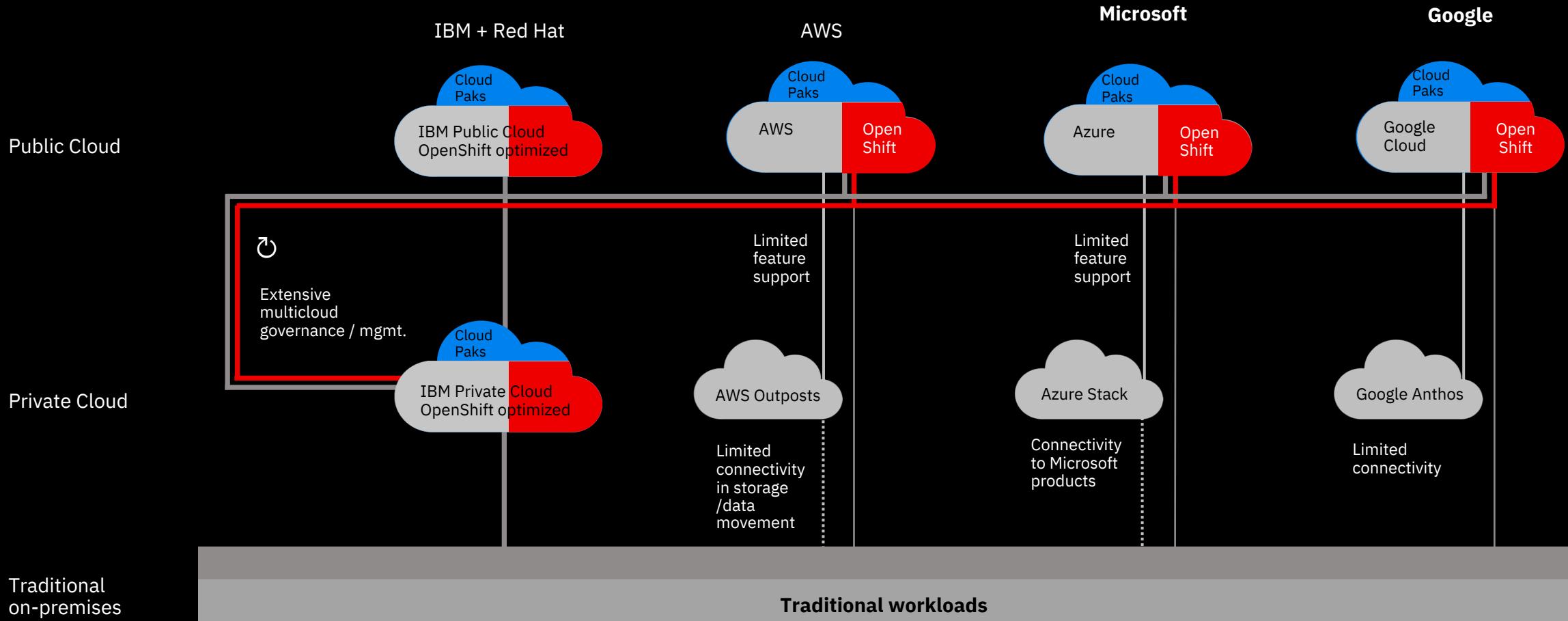
Patching OS Security CVEs (~1 per week)

Finding full-stack problems

Fixing full-stack problems



IBM and Red Hat deliver the industry's true hybrid multicloud platform



IBM Cloud Strategy and Architecture: The Hybrid Multicloud Platform

Consulting
Services

Strategy

Migration

Development

Management

Cloud Paks

Cloud Pak for
Applications

Cloud Pak for
Data

Cloud Pak for
Integration

Cloud Pak for
Automation

Cloud Pak for
Multicloud
Management

Foundation



Open Hybrid Multicloud Platform



Infrastructure

IBM public
cloud



AWS



Microsoft
Azure



Google
Cloud



Edge



Private



IBM Z
IBM LinuxOne
IBM Power
Systems

Making IBM's key software products cloud native for all clouds – middleware anywhere

Reduce dev time up to 84%¹

Make data ready for AI in hours vs. days

Eliminate 33% of integration cost²

Reduce manual processes up to 80%³

Reduce IT op expense by up to 75%⁴

Cloud Pak for Applications
Build, deploy, and run applications

Cloud Pak for Data
Collect, organize, and analyze data

Cloud Pak for Integration
Integrate applications, data, cloud services, and APIs

Cloud Pak for Automation
Transform business processes, decisions, and content

Cloud Pak for Multicloud Management
Multicloud visibility, governance, and automation

IBM containerized software



Open Hybrid Multicloud Platform



IBM public cloud



AWS



Microsoft Azure



Google Cloud



Edge



Private

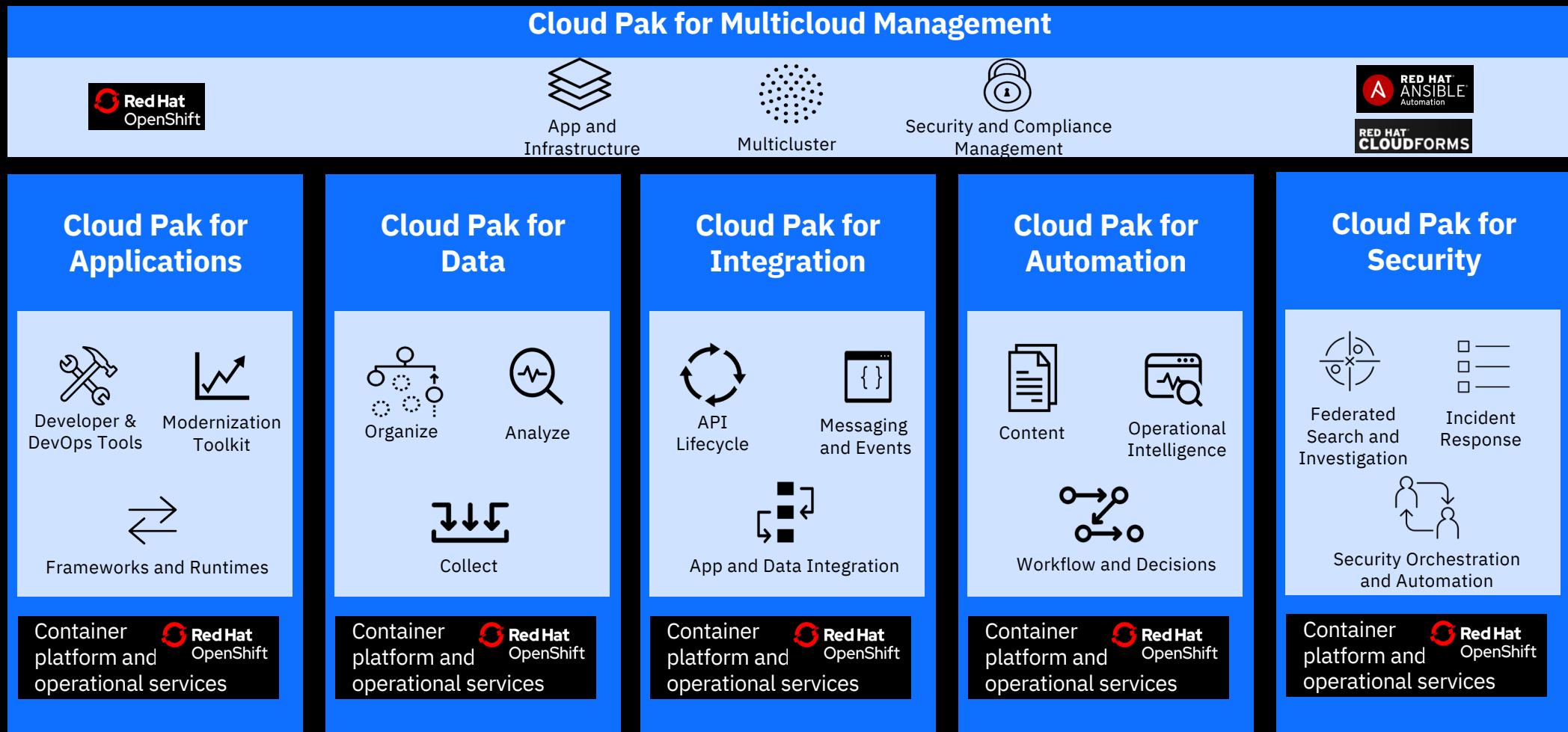


IBM Z
IBM LinuxOne
IBM Power Systems

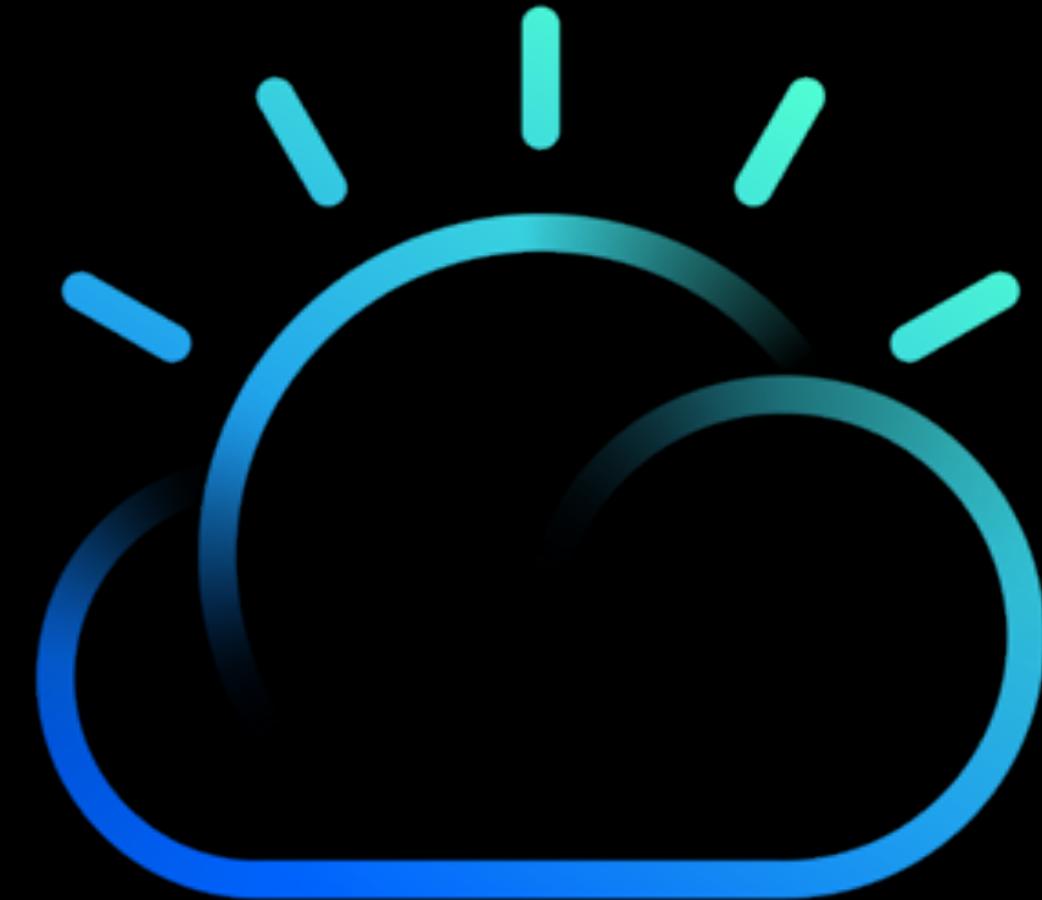
Source: 1,4: Ovum white paper; 2: IBM Solution Brief; 3: Forrester study

Cloud Paks – Pre-integrated for cloud use cases

Today, IBM offers clients *the first six Cloud Paks...*

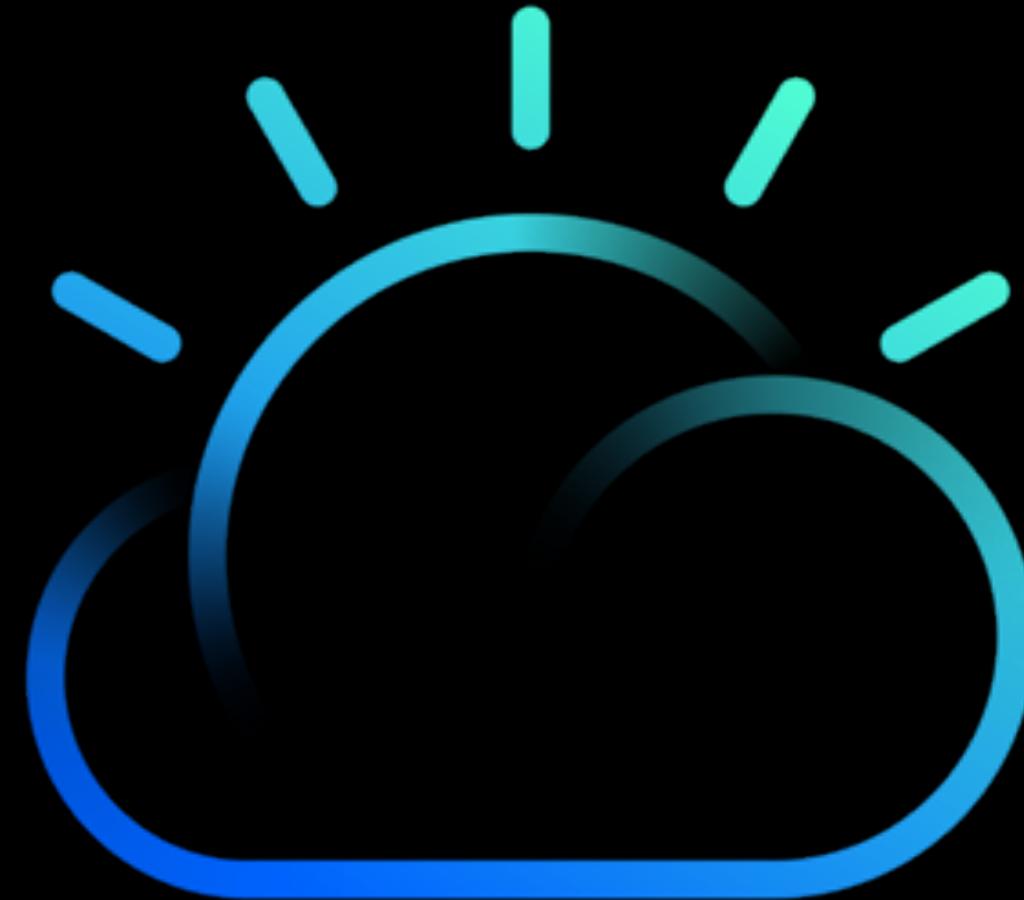


QUESTIONS?



The Journey to Cloud **MultiCloud Management**

02



IBM Cloud

Multicloud Management

>60%

of customers state they
don't have the tools and
procedures to manage and
operate in a complex
multicloud environment

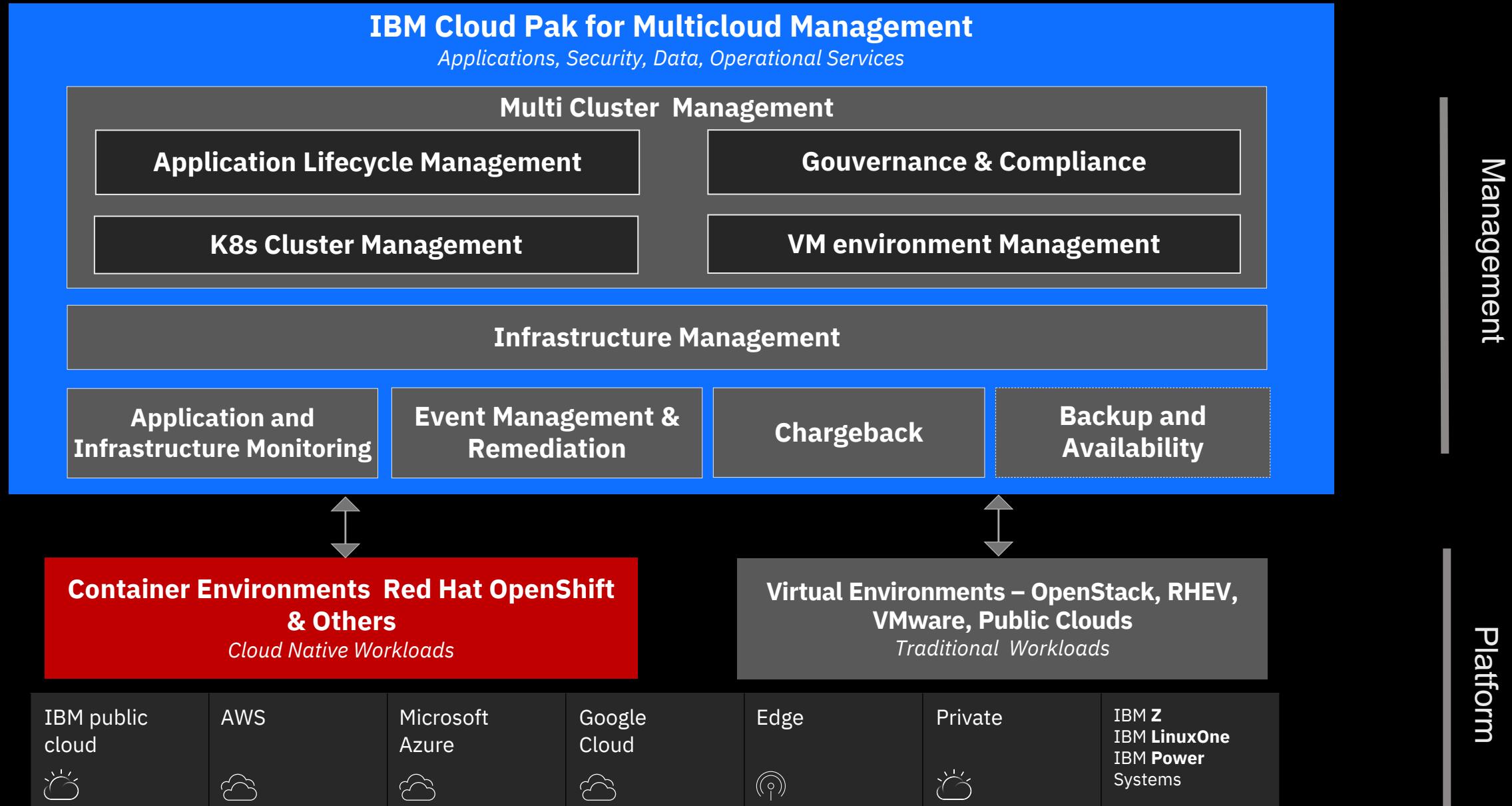
Even if you have just one cloud . . .

You need consistent automation to
manage, secure and optimize
applications and cloud resources.

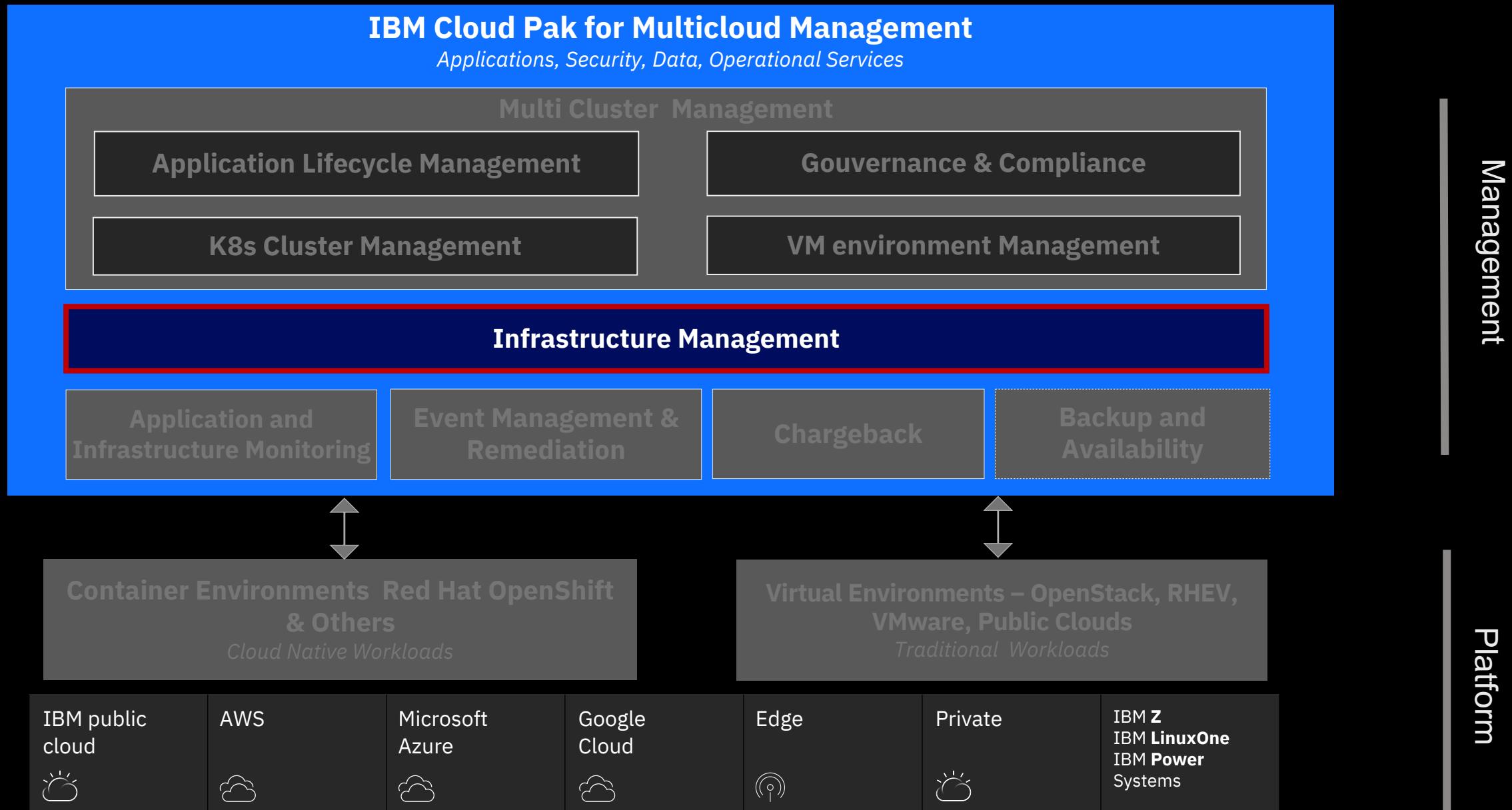
Even if you have just one cluster . .

You need security, automated
operations, & visibility.

Multicloud management technology stack



Multicloud management technology stack



Infrastructure Management

IBM Cloud Automation Manager

Automate infrastructure provisioning with **Terraform**

Automate Infrastructure-as-Code

Compute · Networking · Storage · VMs · Containers · Functions · Public · Private

Run-on & manage-to ...

Intel

Power

Z

VMWare

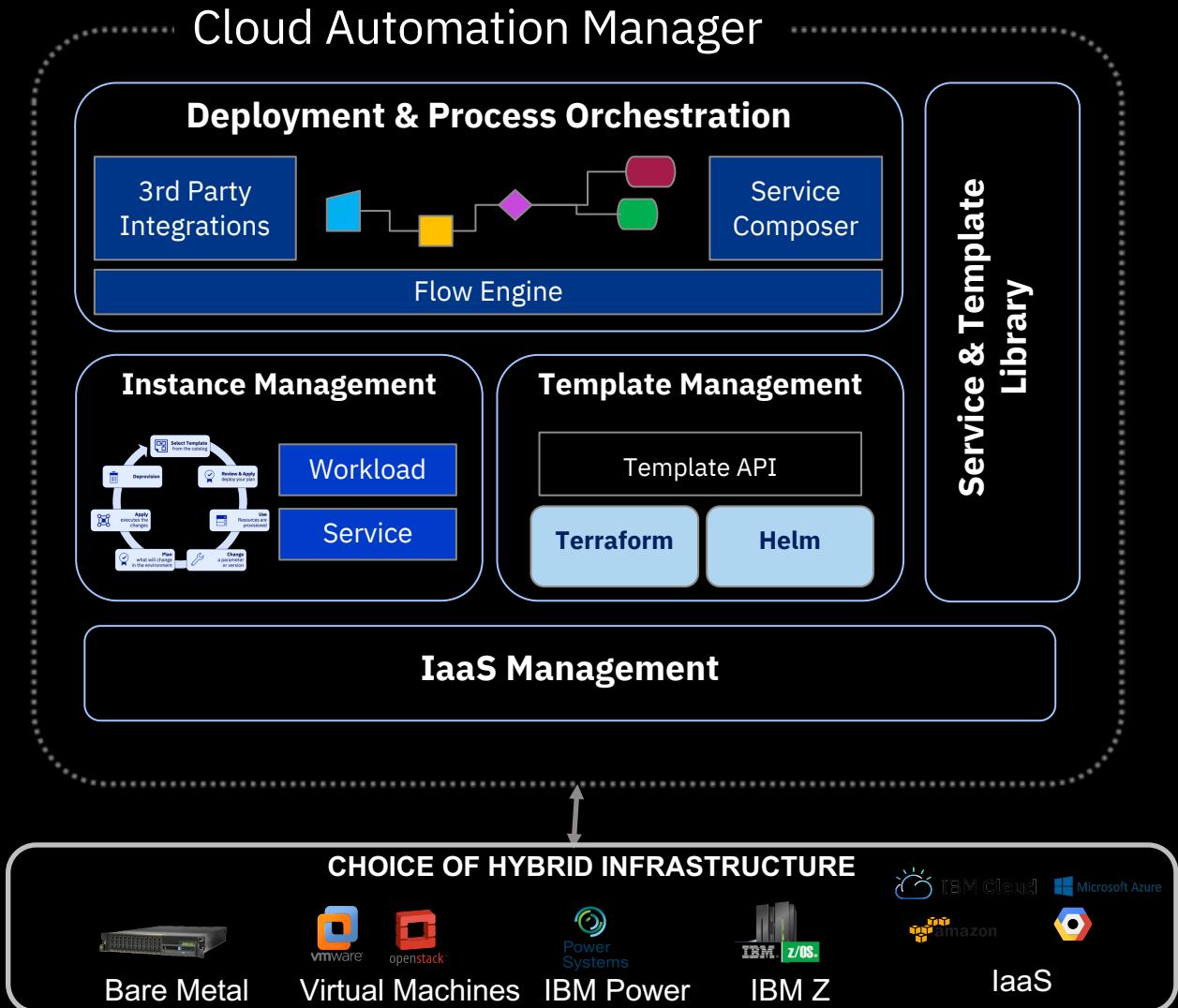
OpenStack



IBM Cloud Automation Manager

Full stack automation and service orchestration

- **Automated provisioning**
Automated provisioning of infrastructure and applications with workflow orchestration
- **Self-service**
Self-service access to cloud infrastructure and application services
- **Manage and govern**
Manage and govern workloads across multiple and hybrid clouds
- **Built with open technology**
Avoid vendor lock-in

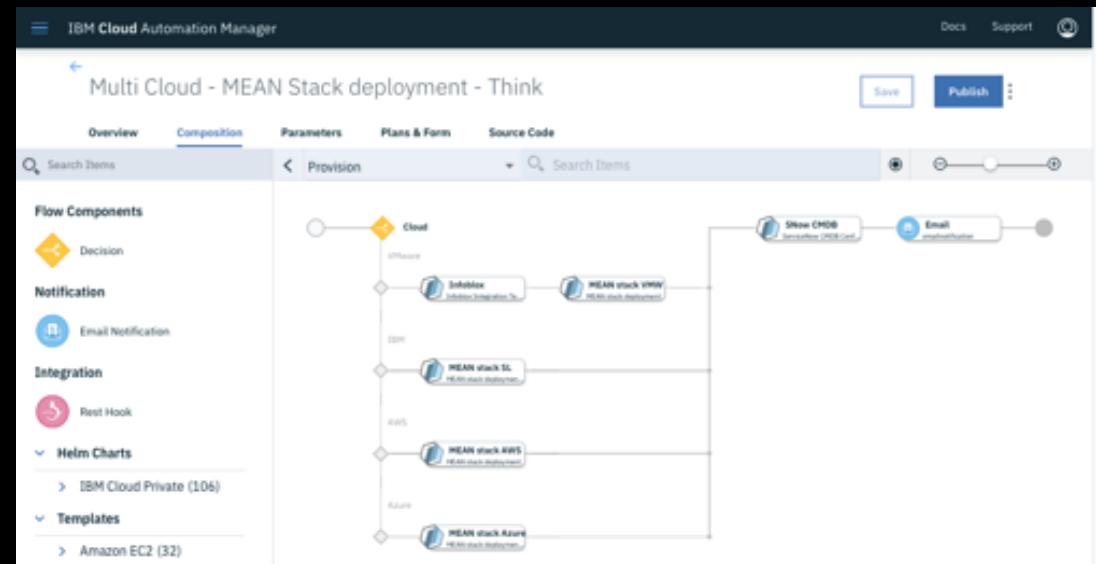


IBM Cloud Pak for Multicloud Management

Infrastructure Automation

Manage infrastructure as code

- Standardize infrastructure provisioning with common, repeatable, scalable processes
- Public, private, & hybrid clouds ⁽¹⁾
- Pair with popular configuration management solutions for full stack provisioning ⁽²⁾



Operationalize

- Combine Terraform, Helm Charts, REST APIs, and ITIL integrations into composite, orchestrated service objects
- Publish curated services into Open Service Broker catalogs
- Scale efficiently to 1000's of VMs

The screenshot shows the IBM Cloud Private Catalog. On the left, a sidebar lists categories: DevOps, Operations, Security, Network, Data Science & Analytics, Data, Integration, Storage, Blockchain, Tools, IoT, Runtimes & Frameworks, Business Automation, and Other. The main area displays a grid of Helm Charts, including 'artifactory-ha', 'audit-logging', 'auth-ansible', 'auth-ldap', 'auth-pgp', 'calico', 'dns-gateway', and 'etcd'. To the right, two specific services are highlighted: 'ubuntu-vm-service-v2' and 'ubuntu-vm-service-v1', both categorized under 'service'. Below them is the 'icp-mongodb' service, described as a 'NoSQL document-oriented database that stores JSON-like documents with'. At the bottom, there is a section for 'mgmt-charts'.

(1) <https://www.terraform.io/docs/providers/index.html>

(2) Ansible, Chef, Puppet, SaltStack, UrbanCode Deploy

Infrastructure Management

IBM Cloud Automation Manager

Operate deployments across all clouds **from one console**

Operations

Plan/Apply · Start/Stop/Restart · Power On/Off · Snapshots

Automate infrastructure provisioning with **Terraform**

Automate Infrastructure-as-Code

Compute · Networking · Storage · VMs · Containers · Functions · Public · Private

Run-on & manage-to ...

Intel

Power

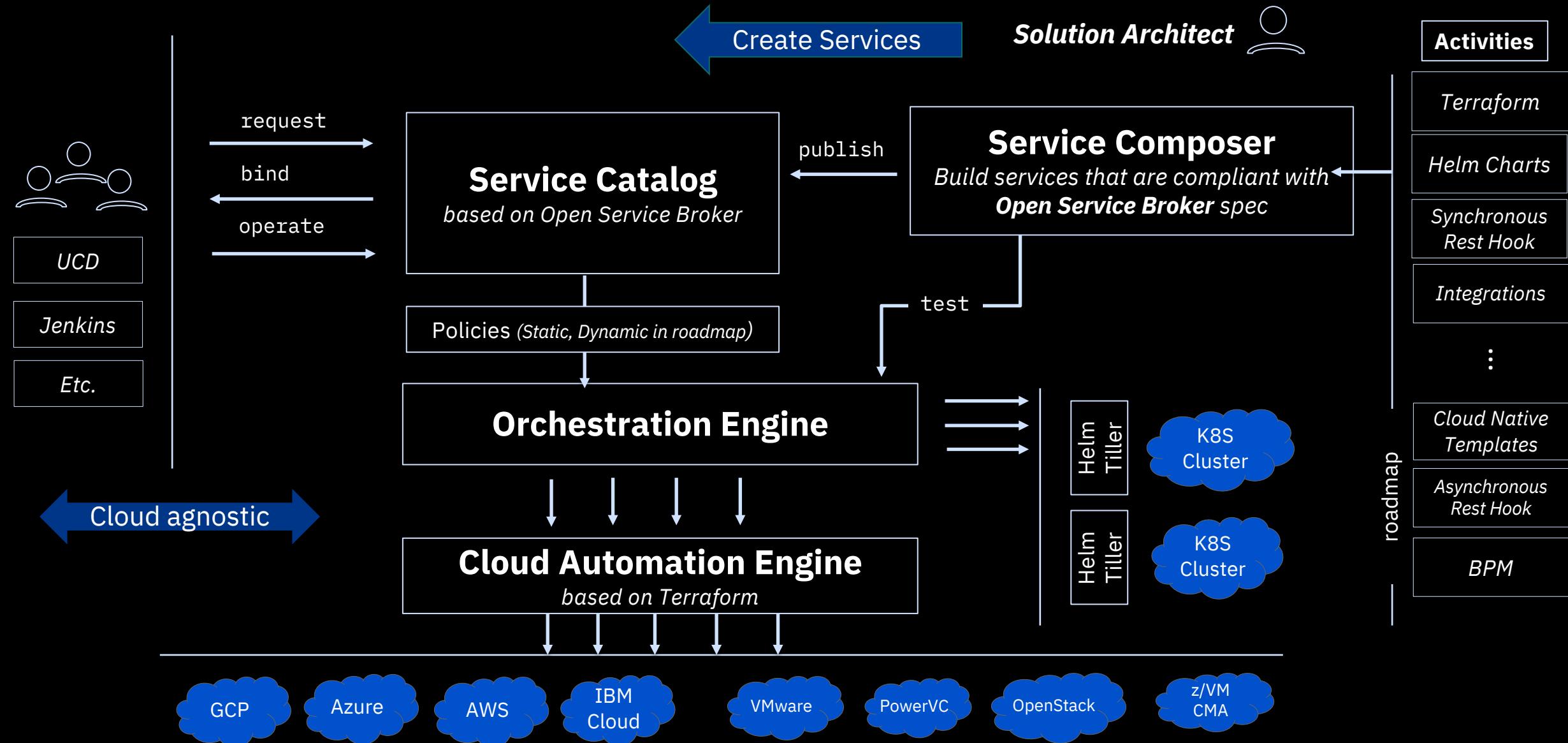
Z

VMWare

OpenStack

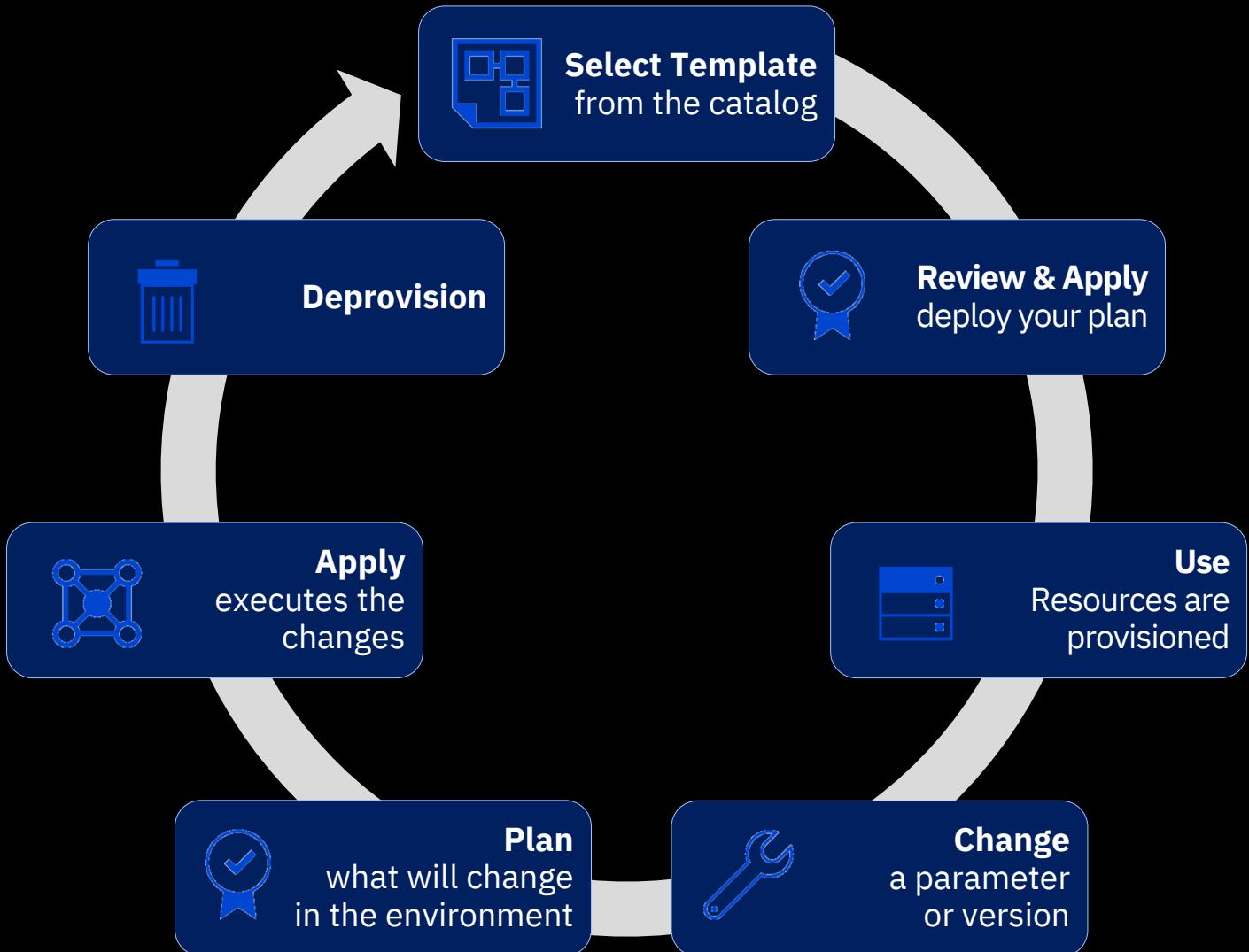


Infrastructure Automation: Deliver Templates as Services



IBM Cloud Automation Manager -Template lifecycle

- Allows user to transparently provision offerings based on VMs and Containers
- Allows user to make changes on running instances
- Two major scenarios
 - User want to change an parameter on an instance
 - Example – change memory/cpu
 - User wants to update to a new version of the template used to create their instance
 - Example – add more VMs or other components



Infrastructure Management

IBM Cloud Automation Manager

Build Services graphically
With ***Service Composer***

Self Service Catalog

Hybrid · Multimodal · Multicloud · Orchestration · Open Service Broker

Operate deployments across all
clouds ***from one console***

Operations

Plan/Apply · Start/Stop/Restart · Power On/Off · Snapshots

Automate infrastructure
provisioning with **Terraform**

Automate Infrastructure-as-Code

Compute · Networking · Storage · VMs · Containers · Functions · Public · Private

Run-on & manage-to ...

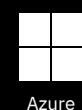
Intel

Power

Z

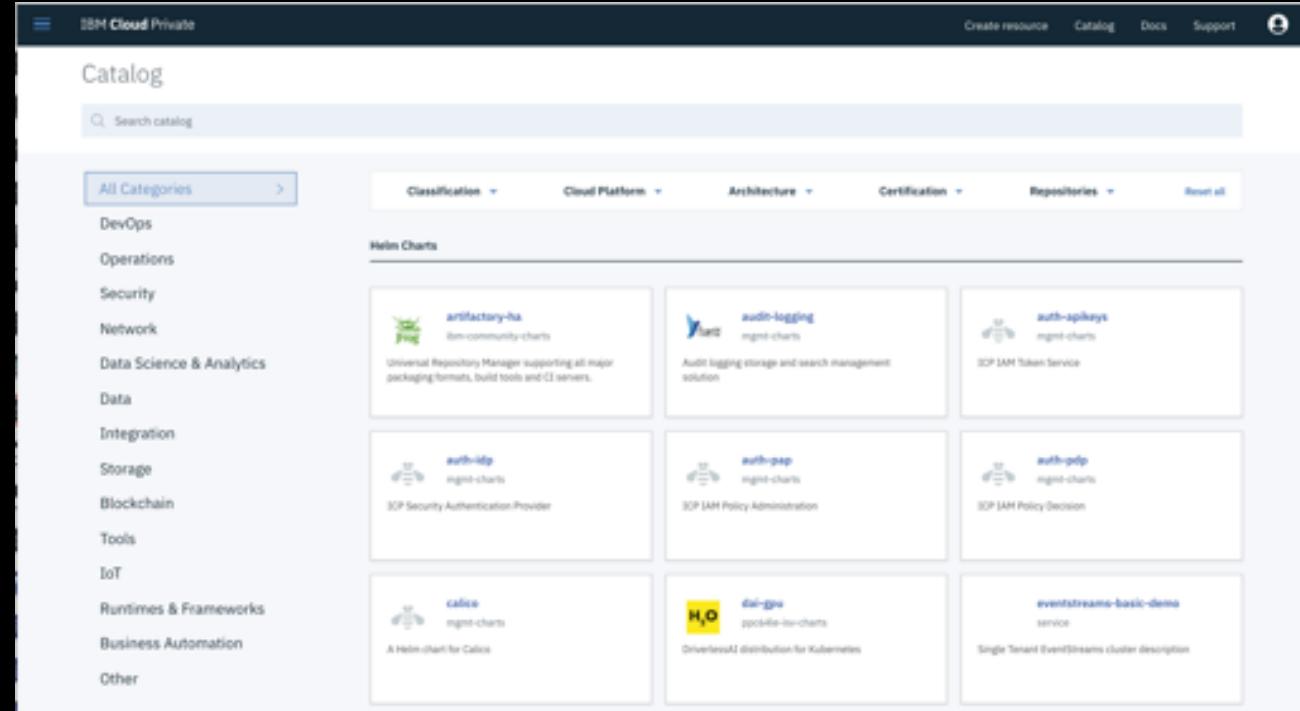
VMWare

OpenStack



Open Service Broker Catalog

Catalog of IBM certified software



The screenshot shows the IBM Cloud Private Catalog interface. On the left, there's a sidebar with 'All Categories' and various service categories like DevOps, Operations, Security, Network, Data Science & Analytics, Data, Integration, Storage, Blockchain, Tools, IoT, Runtimes & Frameworks, Business Automation, and Other. The main area is titled 'Helm Charts' and displays a grid of nine cards, each representing a different Helm chart:

- artifactory-ha (jenkins-community-charts)
- audit-logging (mgmt-charts)
- auth-apikeys (mgmt-charts)
- auth-idp (mgmt-charts)
- auth-pap (mgmt-charts)
- auth-pdp (mgmt-charts)
- calico (mgmt-charts)
- dat-gpu (zodcfile-iso-charts)
- eventstreams-basic-demo (service)

- Certified and verified off-the-shelf Helm and Terraform for IBM, Open Source, and 3rd party software
- Use catalog entries via web portal or via APIs from CI/CD toolchains
- Present different catalog views by teams with role based access

 **icp-mongodb**
NoSQL document-oriented database
that stores JSON-like documents with
mgmt-charts

 **ibm-open-liberty**
Open Liberty, an open source runtime
for Java microservices & cloud-native
ibm-charts

 **ubuntu-vm-service-v2**
ubuntu-vm-service-v2
service

 **ibm-f5bigip-controller**
A Helm chart for integrating f5-bigip
controller with ICP
ibm-charts

 **vulnerability-advisor**
Vulnerability Advisor runs security
checks on running containers and
mgmt-charts

Infrastructure Management



Infrastructure
Management

Manage Infrastructure-as-Code

Automate application provisioning with a consistent, declarative, highly scalable operational process

–

Operationalize with ‘Git’

Fully scriptable REST API interface for use with CI/CD toolchains and scale efficiently to 1000s of VMs

–

Graphical infrastructure automation

Drag-n-drop VMs, networks, scripts and other cloud components onto the canvas to auto generate Terraform infrastructure definitions in seconds

Multicloud Management

Infrastructure Automation



Physical and virtual
infrastructure automation
powered by Terraform and Git

Manage Infrastructure-as-Code - Automate application provisioning with a consistent, declarative, highly scalable operational process that works the same in all of your clouds. The IBM solution for provisioning infrastructure, which leverages Terraform can be paired with popular configuration management solutions like Ansible to provision applications on physical and virtual infrastructure in these clouds.

Infrastructure Management

IBM Cloud Automation Manager

Build Services graphically
With **Service Composer**

Self Service Catalog

Hybrid · Multimodal · Multicloud · Orchestration · Open Service Broker

Operate deployments across all
clouds **from one console**

Operations

Plan/Apply · Start/Stop/Restart · Power On/Off · Snapshots

Automate infrastructure
provisioning with **Terraform**

Automate Infrastructure-as-Code

Compute · Networking · Storage · VMs · Containers · Functions · Public · Private

Run-on & manage-to ...

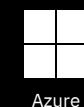
Intel

Power

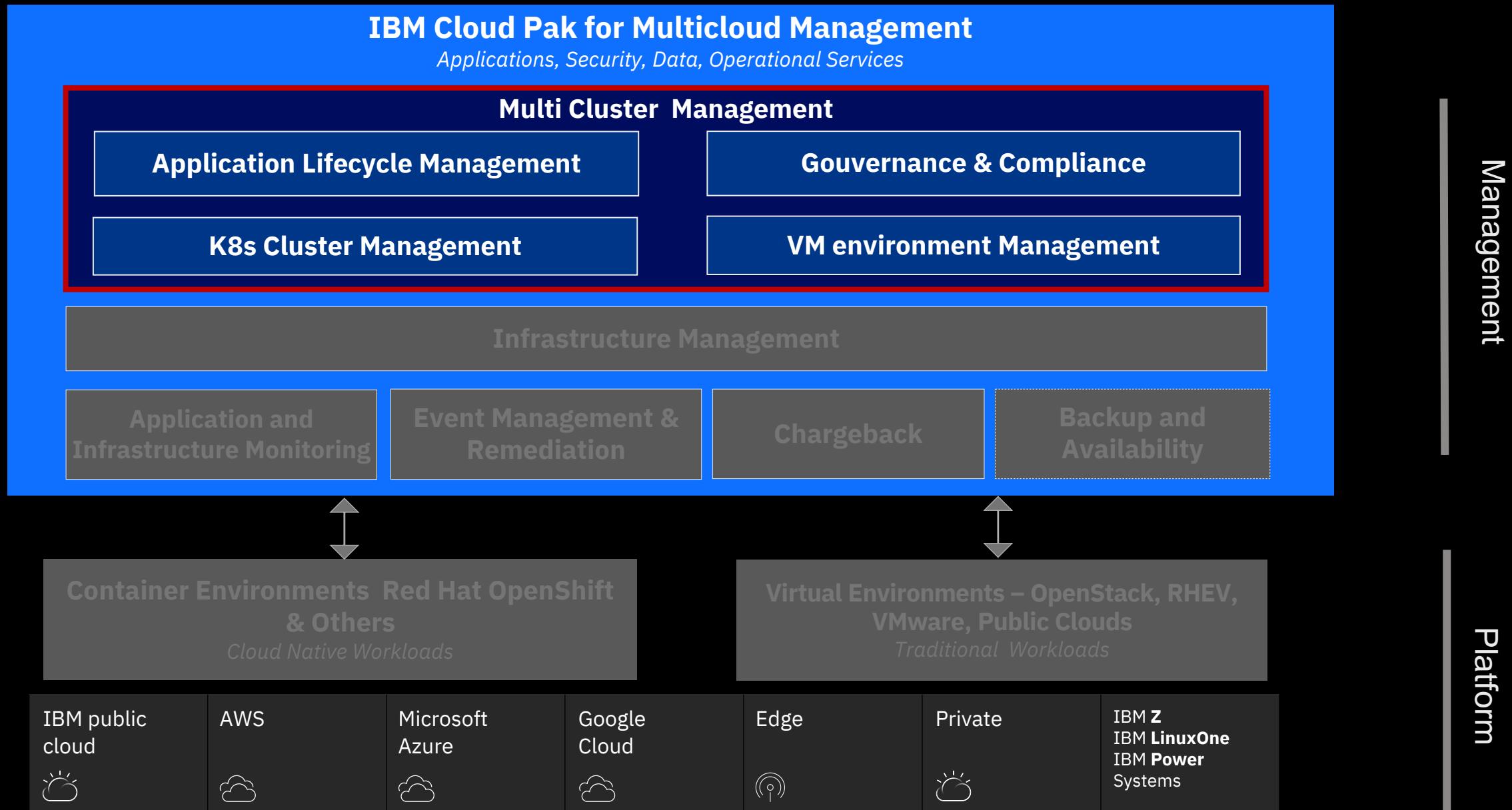
Z

VMWare

OpenStack



Multicloud management technology stack



Hybrid Multi-cluster Management

IBM Multicloud Manager - Application-centric management across clusters

Manage Kubernetes clusters
across multiple clouds

Cluster Management

Single pane of glass · Cluster Health · Topology · Systemwide search

Run-on & manage-to ...

Intel

Power

Z

VMWare

OpenStack

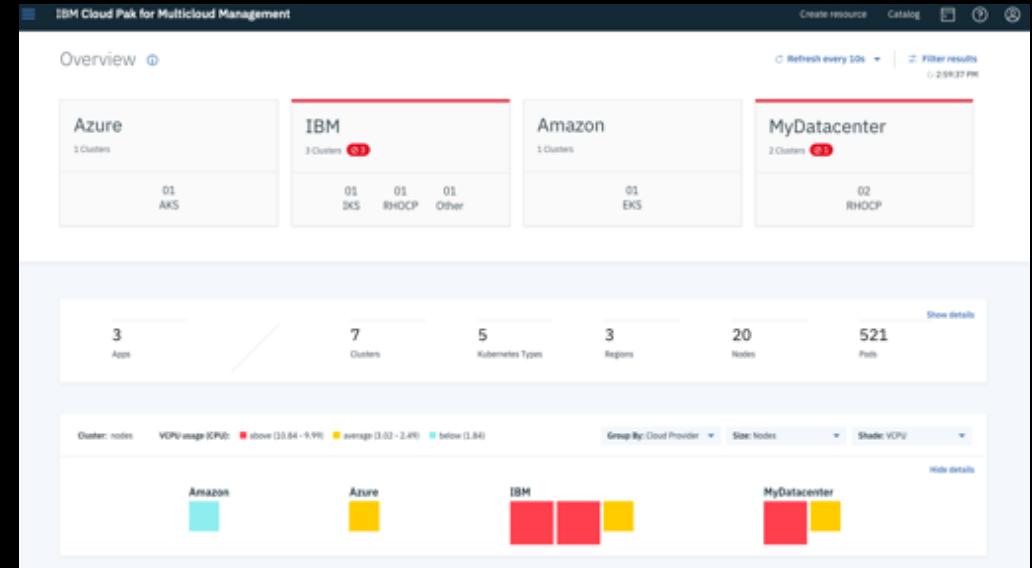


Centralized Cluster Management - Dashboard



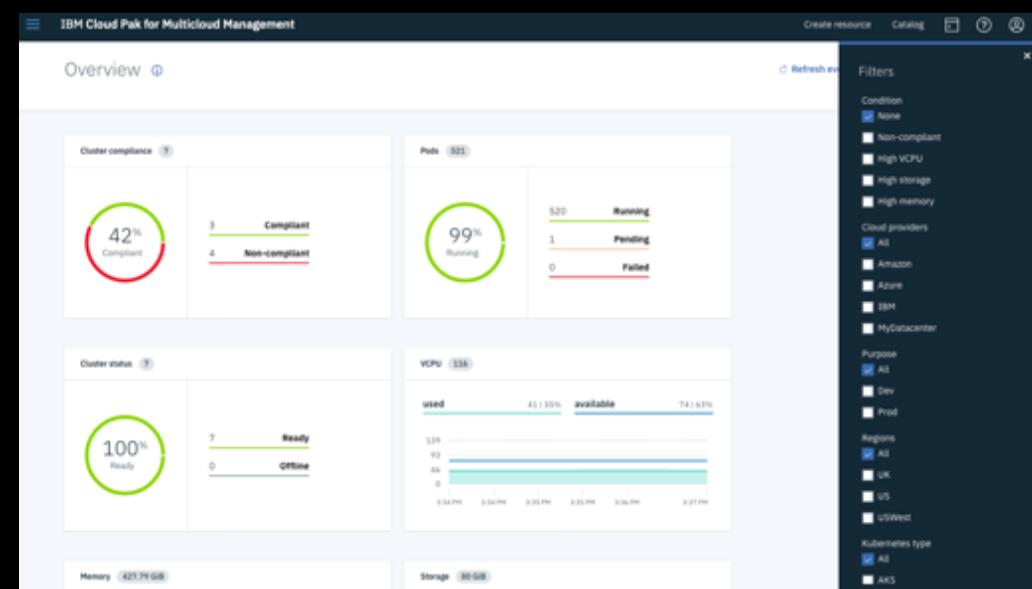
Works across clouds

- Seamlessly manage clusters across multiple Private Datacenters, Public IaaS, Cloud Kubernetes services
- Consistent firewall traversal methodology
- Common management policies across environments



Integrated Views across all clusters and environments

- Look at cluster health, capacity, pods running/failed, app versions across clusters
- Quickly see where the problems are
- Organize clusters in any way – e.g. dev/prod/location/org



Centralized Cluster Management – Global Search

**Searches
across all
clusters**

- Seamlessly search Kubernetes objects across multiple Private Datacenters, Public IaaS, Cloud Kubernetes services
- Saved searches for quick access

The screenshot shows the 'Search' interface with a search bar at the top. Below it, a 'Saved searches' section displays a count of 15 results for 'GBAPP'. A 'New search' button is available. The main area shows three suggested search templates: 'Workloads' (267 results), 'Unhealthy Pods' (6 results), and 'Created Last Hour' (1 result). Each template has a description and a 'Show' button.

The screenshot shows the search results for 'namespace default - gbase'. It includes sections for 'Pod (2) +', 'Applicationrelationship (3) +', 'Deployable (3) +', 'Placementpolicy (2) +', and 'Monitoringdashboard (1) +'. Each section lists items with columns for Name, Namespace, Cluster, Status, Restarts, Host IP, Pod IP, and Created. For example, under 'Pod (2) +', there are two entries for 'nd-gbase-gbase-ica-fyre-small-pod-54794bc771-8l9ew2' and 'nd-gbase-gbase-ica-fyre-small-pod-54794bc771-929'. Under 'Deployable (3) +', there are three entries for 'gbase-gbase-0.1.0.tgz', 'gbase-gbase-revisionmaster', and 'gbase-gbase-revisionlive'. The 'Monitoringdashboard' section shows one entry for 'gbase-gbase-monitoring'.

Name	Namespace	Cluster	Status	Restarts	Host IP	Pod IP	Created
nd-gbase-gbase-ica-fyre-small-pod-54794bc771-8l9ew2	default	ica-fyre-small	Terminating	0	172.16.34.203	-	5 days ago
nd-gbase-gbase-ica-fyre-small-pod-54794bc771-929	default	ica-fyre-small	Terminating	0	172.16.34.203	10.3.91.176	5 days ago

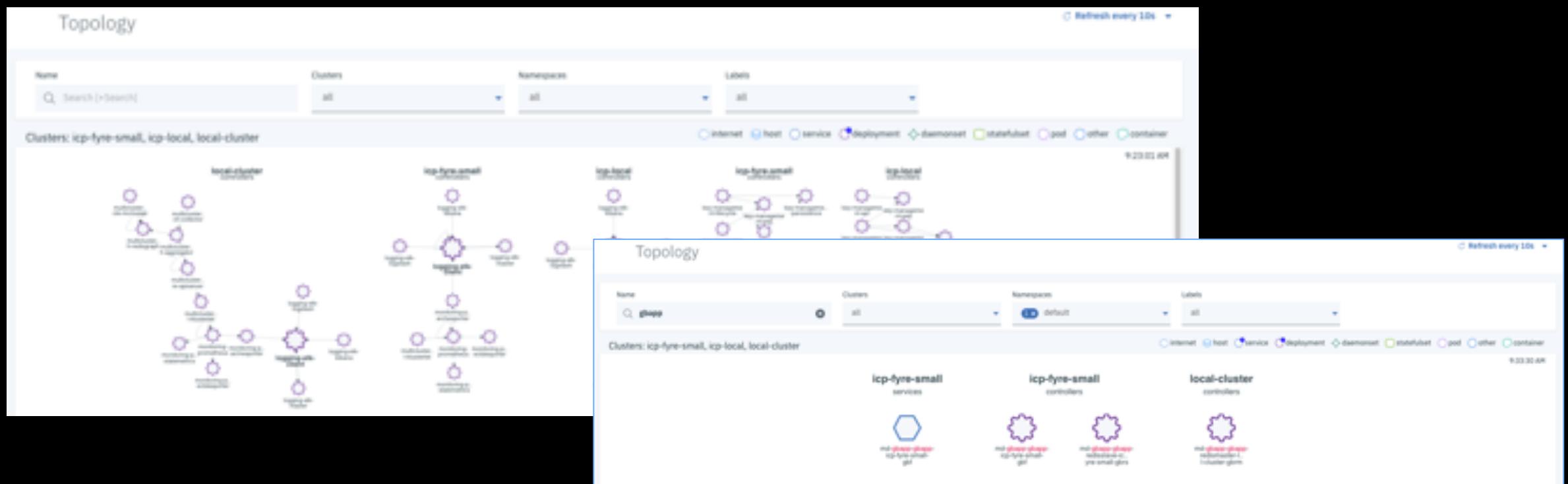
Name	Namespace	Chart URL	Dependencies	Created
gbase-gbase-0.1.0.tgz	default	https://raw.githubusercontent.com/abhishegupta/helm-repo/master/3.1-mcm-guestbook/gbase-0.1.0.tgz	-	5 days ago
gbase-gbase-revisionmaster	default	https://raw.githubusercontent.com/abhishegupta/helm-repo/master/3.1-mcm-guestbook/gbase-0.1.0.tgz	-	5 days ago
gbase-gbase-revisionlive	default	https://raw.githubusercontent.com/abhishegupta/helm-repo/master/3.1-mcm-guestbook/gbase-0.1.0.tgz	-	5 days ago

Name	Namespace	Cluster	Created
gbase-gbase-monitoring	default	local-cluster	5 days ago

Centralized Cluster Management - Topology

Topology view across clouds

- Understand what workloads are running on your clusters across multiple Private Datacenters, Public IaaS, Cloud Kubernetes services
- Powerful filtering mechanism



Hybrid Multi-cluster Management

IBM Multicloud Manager - Application-centric management across clusters

Maintain compliance with enterprise policies

Compliance and Governance

Policy based · Set and enforce compliance · Automatic remediation

Manage Kubernetes clusters across multiple clouds

Cluster Management

Single pane of glass · Cluster Health · Topology · Systemwide search

Run-on & manage-to ...

Intel

Power

Z

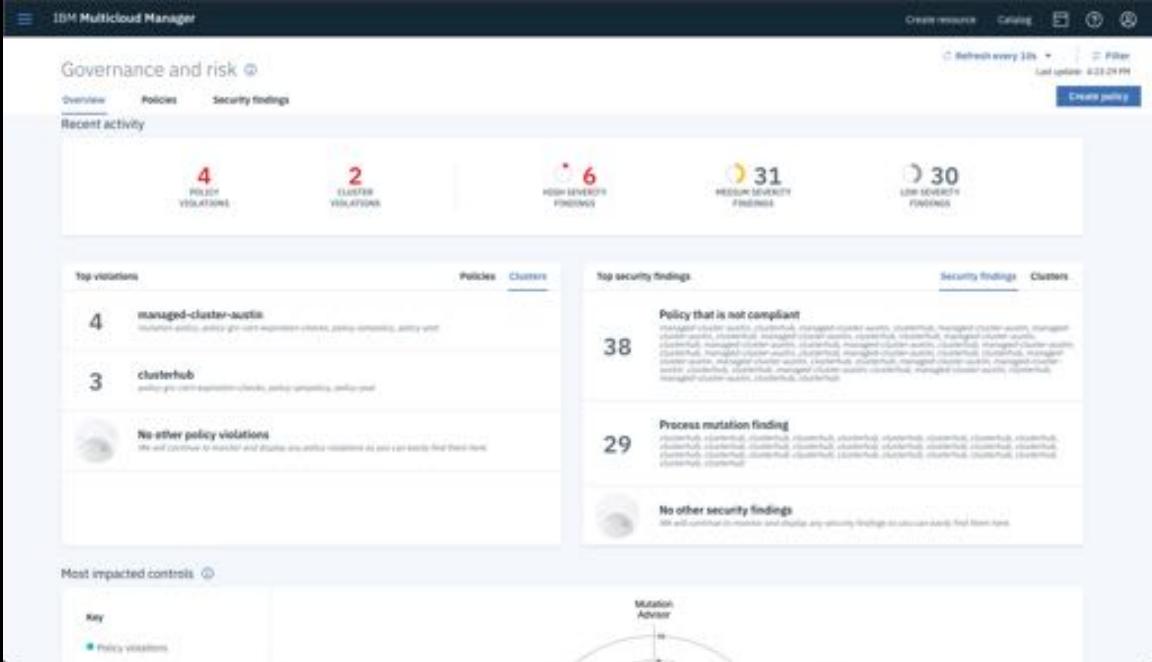
VMWare

OpenStack

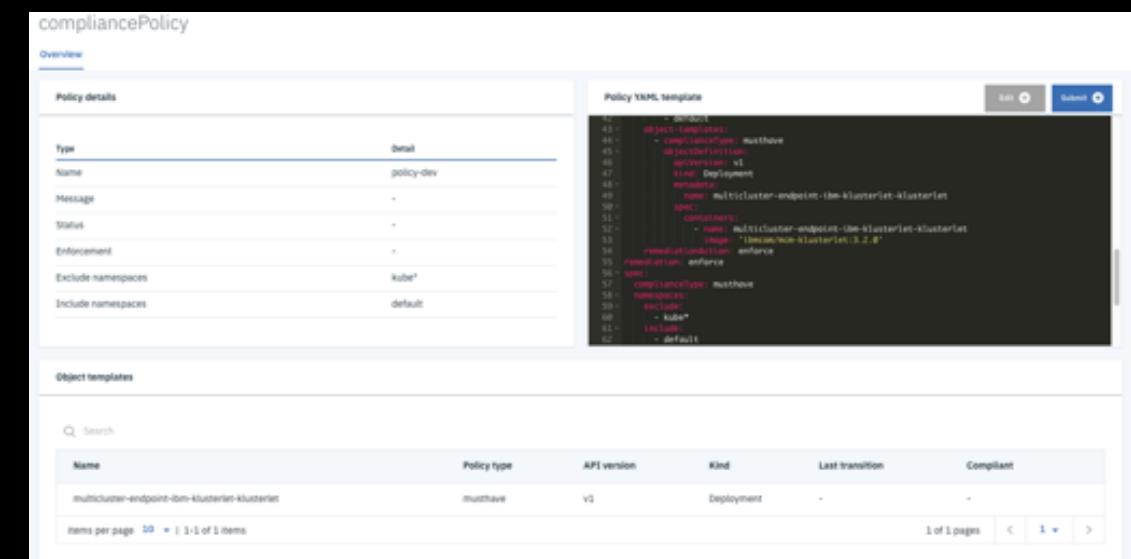


Policy-based Governance

Policy Based Role & Compliance Management



- Set and enforce policies for Security, Applications, infrastructure (Auto enforcement at cluster level)
 - Check compliance against deployment parameters, configuration and policies
 - Automatically remediate violations



Hybrid Multi-cluster Management

IBM Multicloud Manager - Application-centric management across clusters

Define, deploy, and monitor
applications across clusters

Application Management

Deploy across clusters · Placement policies · K8s CRD · Monitoring

Maintain compliance with
enterprise policies

Compliance and Governance

Policy based · Set and enforce compliance · Automatic remediation

Manage Kubernetes clusters
across multiple clouds

Cluster Management

Single pane of glass · Cluster Health · Topology · Systemwide search

Run-on & manage-to ...

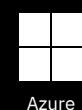
Intel

Power

Z

VMWare

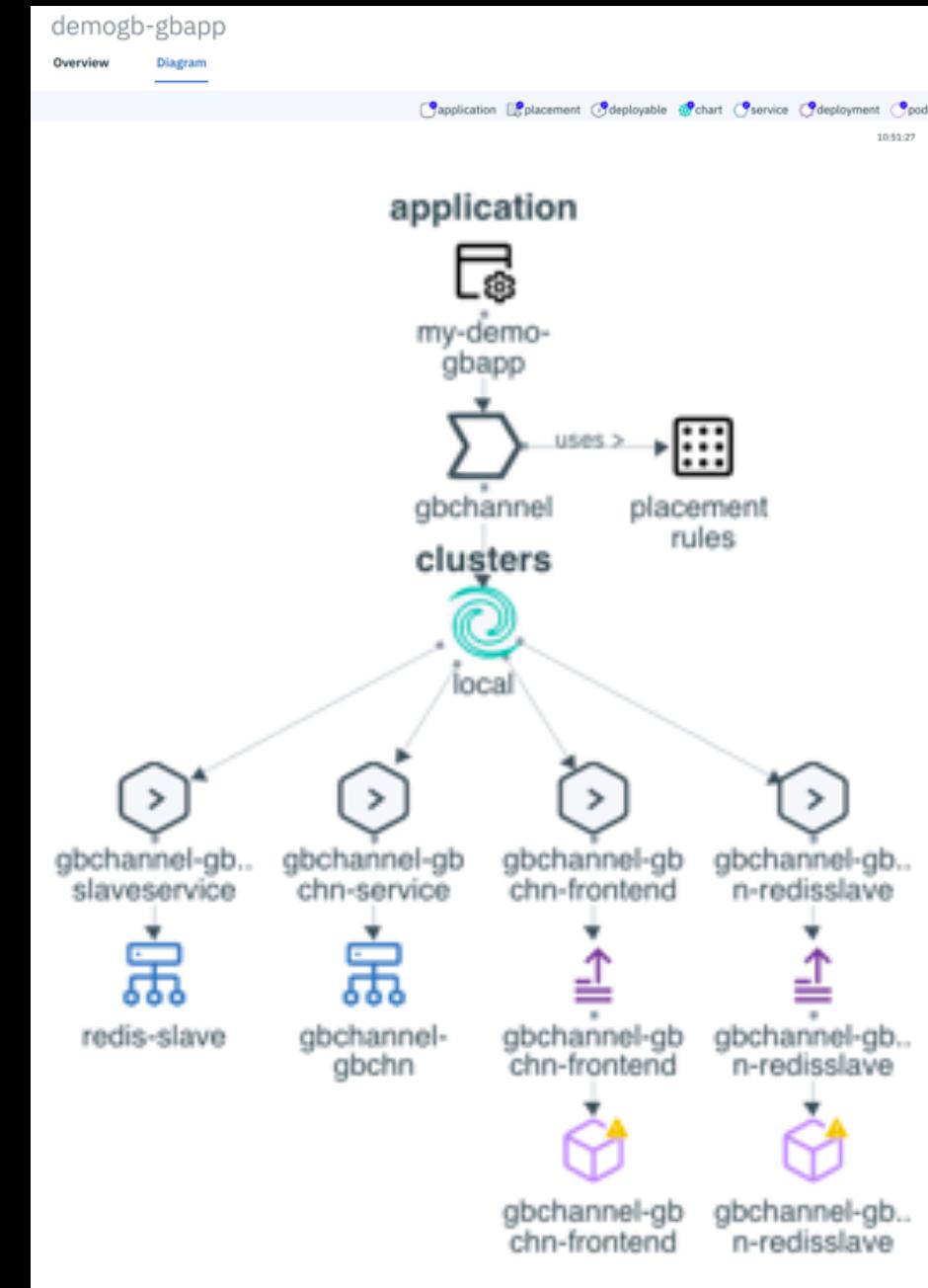
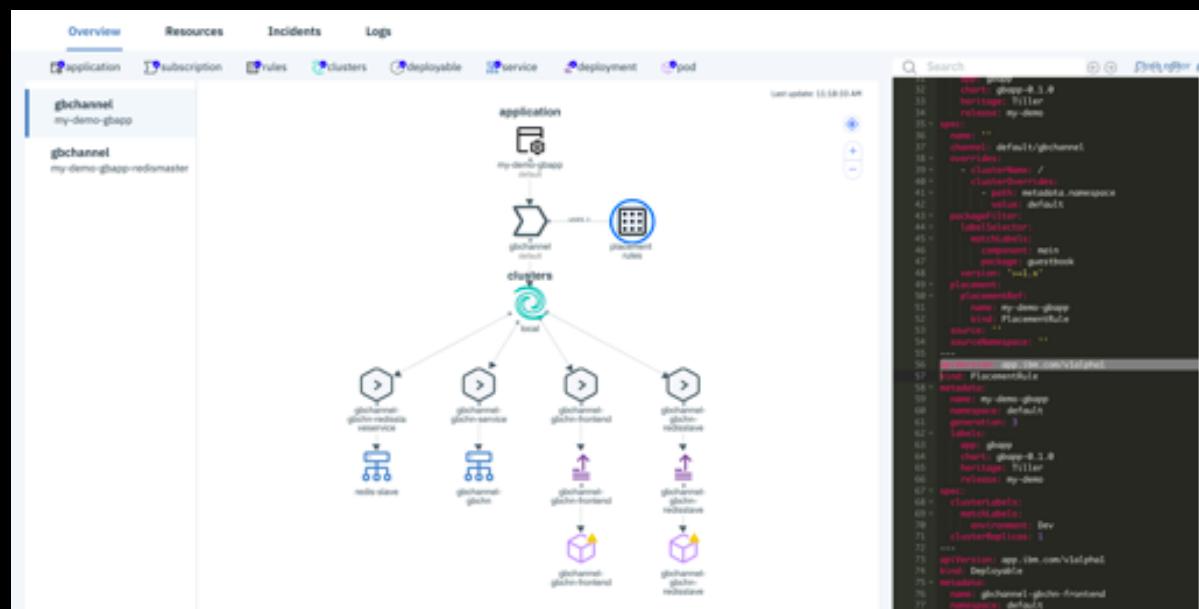
OpenStack



Application Centric Management

Multi-Cluster Application Management

- Deploy applications across clusters based on policy – compliance, DEV vs TEST etc.
- Automatically update monitoring dashboard based on deployment
- Understand failure dependencies – identify system affected if a component (shared) fails.



Application Centric Management

Multi-Cluster Application Management

- Application Defines deployable components and placement policies
- Based on Sig-App from Cloud Native Computing Foundation
- Extended with **policy based placement**

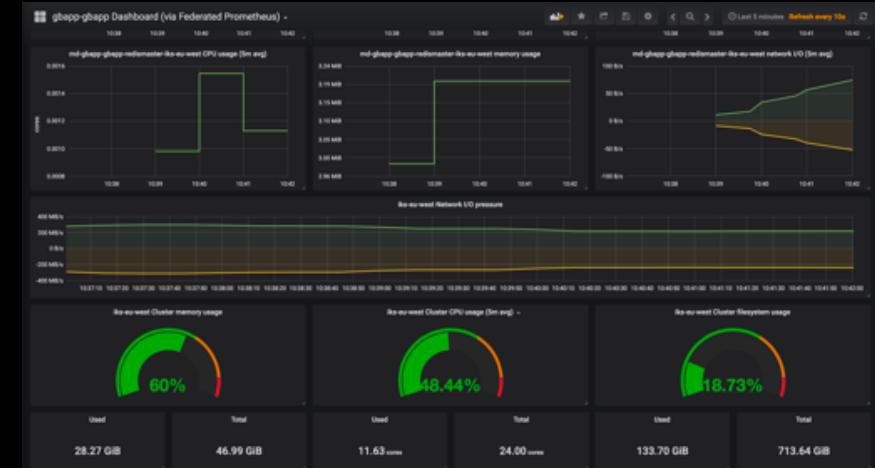
The screenshot shows the IBM Multicloud Manager interface. At the top, it displays an application named 'srch-rls-gbapp' with a status of 'READY'. Below this, the 'Overview' tab is selected, showing a summary of resources: 6 DEPLOYABLES, 13 RESOURCES (Total), 13 COMPLETED RESOURCES, 0 IN PROGRESS RESOURCES, 0 FAILED RESOURCES, and 0 INCIDENTS. The 'Deployment topology' section shows a hierarchical tree of clusters, services, and pods across multiple clouds. The bottom navigation bar includes links for Application, Subscription, Rules, Clusters, Deployable, Service, Deployment, and Pod.

```
apiVersion: app.k8s.io/v1beta1
kind: Application
metadata:
  name: gbapp-gbapp
  labels:
    app: gbapp
    chart: gbapp-0.1.0
    release: gbapp
    heritage: Tiller
    name: gbapp-gbapp
spec:
  selector:
    matchExpressions:
    - key: app
      operator: In
      values:
      - gbapp
      - gbf
      - gbrm
      - gbrs
  componentKinds:
  - group: core
    kind: Pods
```

Application Centric Management

Integrated Operations Management Tools

- Logging, Monitoring and Event across applications and infrastructure
- Integration with Service Management tools
- Automated dashboard creation for applications based on deployment across clusters
- Automatically update monitoring dashboard based on deployment



Hybrid Multi-cluster Management

IBM Multicloud Manager - Application-centric management across clusters

Define, deploy, and monitor
applications across clusters

Application Management

Deploy across clusters · Placement policies · K8s CRD · Monitoring

Maintain compliance with
enterprise policies

Compliance and Governance

Policy based · Set and enforce compliance · Automatic remediation

Manage Kubernetes clusters
across multiple clouds

Cluster Management

Single pane of glass · Cluster Health · Topology · Systemwide search

Run-on & manage-to ...

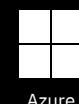
Intel

Power

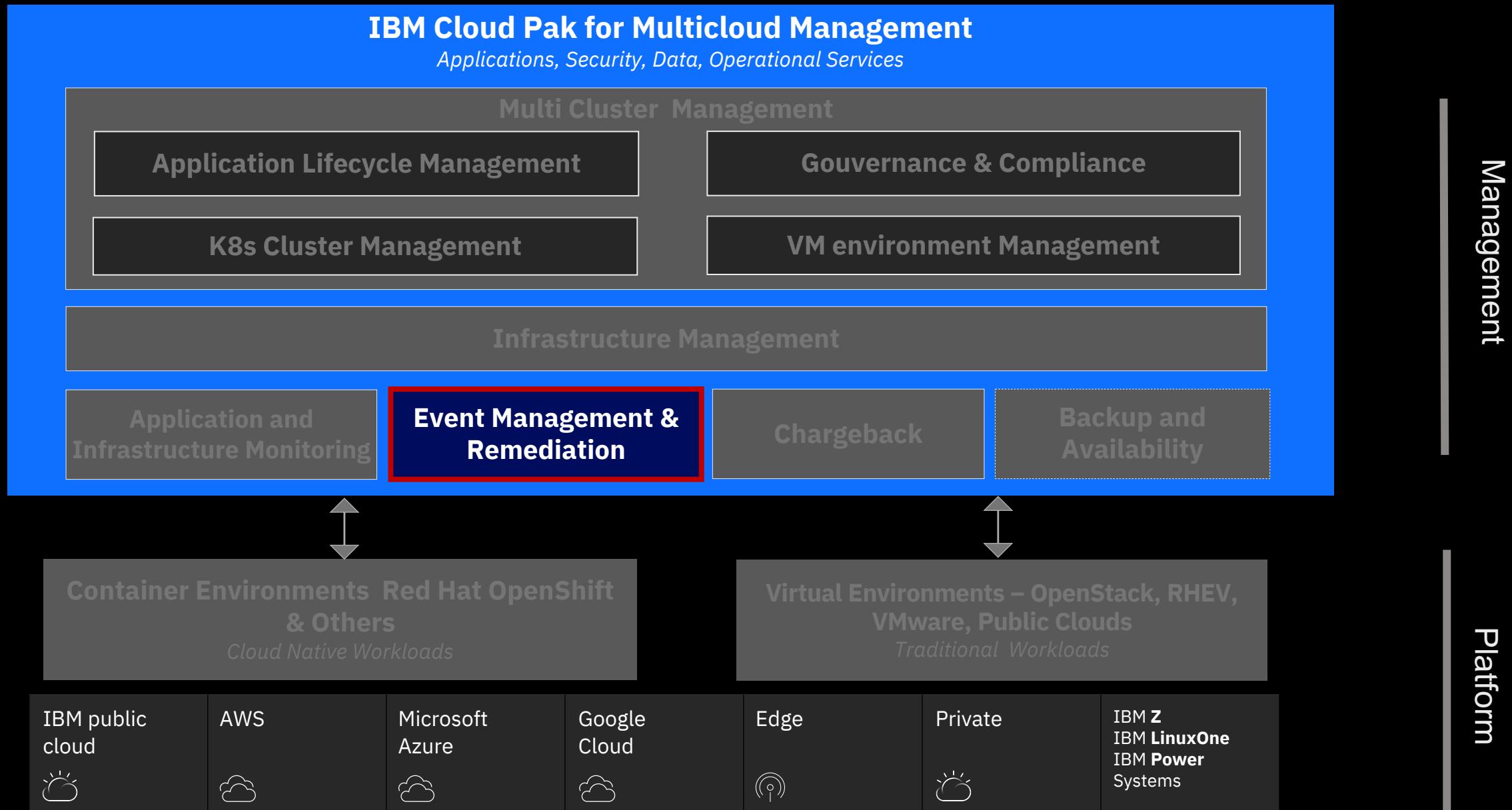
Z

VMWare

OpenStack



Multicloud management technology stack



Events, Incidents and Runbooks

- Analytics *correlate events* into a single incident; greatly reducing the noise
- Incident Management enables *routing of incidents* to the users (and groups) with the right expertise
- *Policy-driven notifications*; not limited to alerting on new incident, e.g. incident escalation
- *Runbooks* provide structured manual and automated steps for *quick resolution* of incidents

The screenshot shows two main sections of the IBM Cloud Incident Management interface.

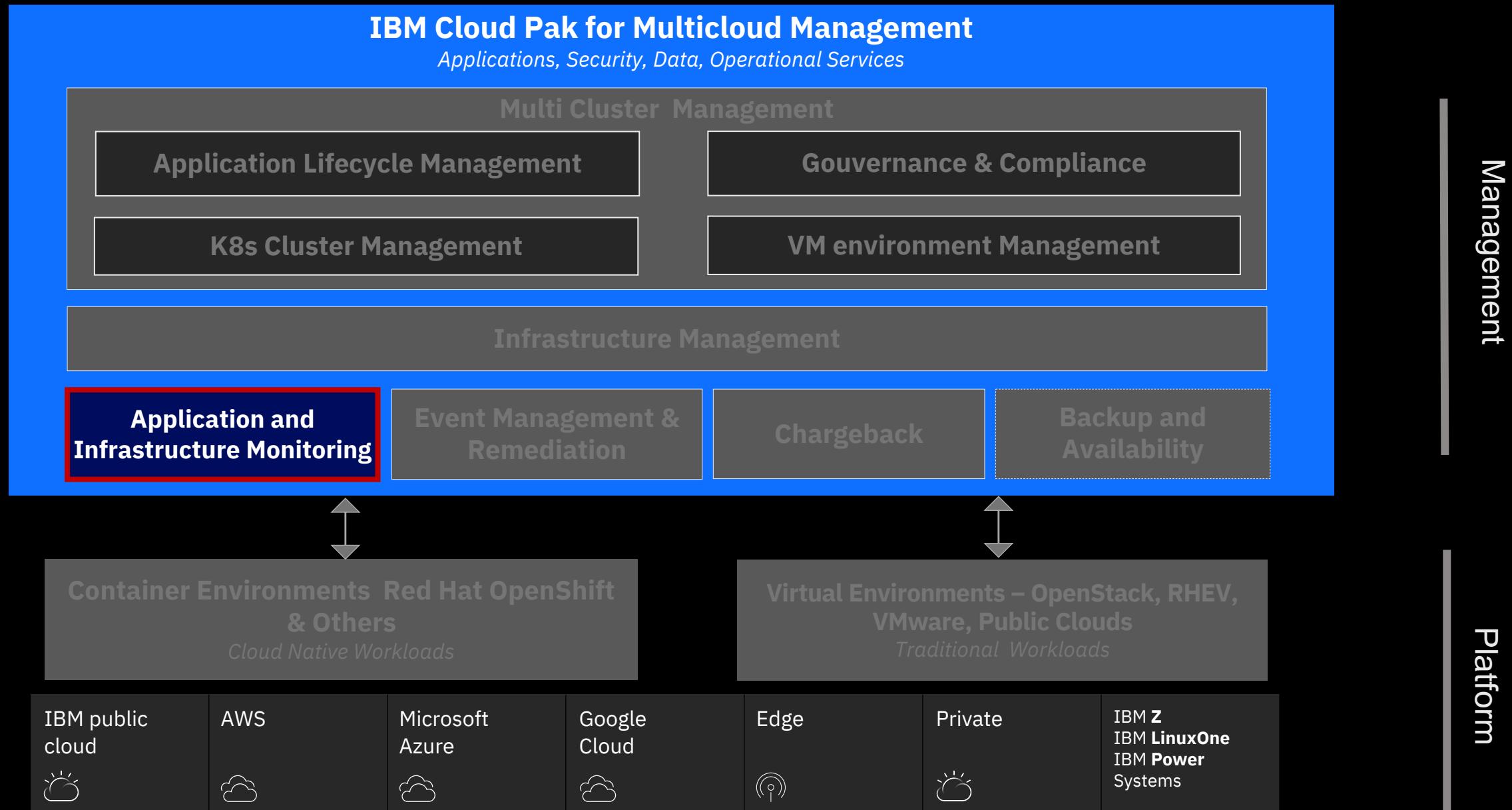
Incident List: The top section displays a list of 9 incidents. Each incident card includes the incident ID (#0187-08-0, #0187-09-0, #0158-41-0), a brief description (Host name: was, Host name: ihs, Host name: was), priority (Priority 1), status (Open), duration (1w 3d), last changed time (Aug 21, 2019 | 05:03:52 PM CDT), and an 'Investigate' button.

Incident ID	Description	Priority	Status	Duration	Last changed	Action
#0187-08-0	Host name: was	Priority 1	Open	1w 3d	Aug 21, 2019 05:03:52 PM CDT	Investigate
#0187-09-0	Host name: ihs	Priority 1	Open	1w 3d	Aug 17, 2019 05:53:10 PM CDT	Investigate
#0158-41-0	Host name: was	Priority 1	Open	1w 3d	Aug 17, 2019 05:53:10 PM CDT	Investigate

Runbook Catalog: The bottom section displays a catalog of 18 available runbooks. Each runbook entry includes a title, description, rating (4 stars), and success rate (86%).

Name	Description	Rating	Success rate
Drain_Queue	Drain the Queue that is full	★★★★★	86%
Example: Add node to SA MP cluster	Add another node to an existing SA MP high ava...		
Example: Analyze application failure in SA M...	Procedure to find out why a System Automatio...		
Example: Cleanup a file system on Linux	Steps to clean up a Linux file system		
Example: Cleanup a file system on Windows	Safely clean a file system on a Windows system...		
Example: Collect troubleshooting data for We...	Describes how to gather troubleshooting data p...		
Example: Connectivity check for a server	Procedure describing how to check whether co...		
Example: Find the right team to assign BCCH L...	Steps to determine the responsible team if a Br...		
Example: Handle full DB2 tablespace on Linux	Procedure describing how to handle a full DB2 ...		
Example: Handle High CPU usage on Windows	Procedure describing how to react on a high CP...		

Multicloud management technology stack



Learn from the SRE discipline – What are the golden signals?



Latency



Errors



Traffic



Saturation



Why monitor
golden signals?

- Golden signals are a common language to monitor across **technologies** and **clouds**, simplifying communication and troubleshooting without having to be an SME
- Golden signals are a direct measure of signals that impact the end user, making it clear when an **important** issue occurs
- **Waste less time** reacting to unclear or unnecessary alerts

Learn from the SRE discipline – What are the golden signals?



Latency

The time it takes to service a request



Errors

Trend view of request error rate



Traffic

Demand being placed on the system



Saturation

View of utilization against max capacity

Symptoms

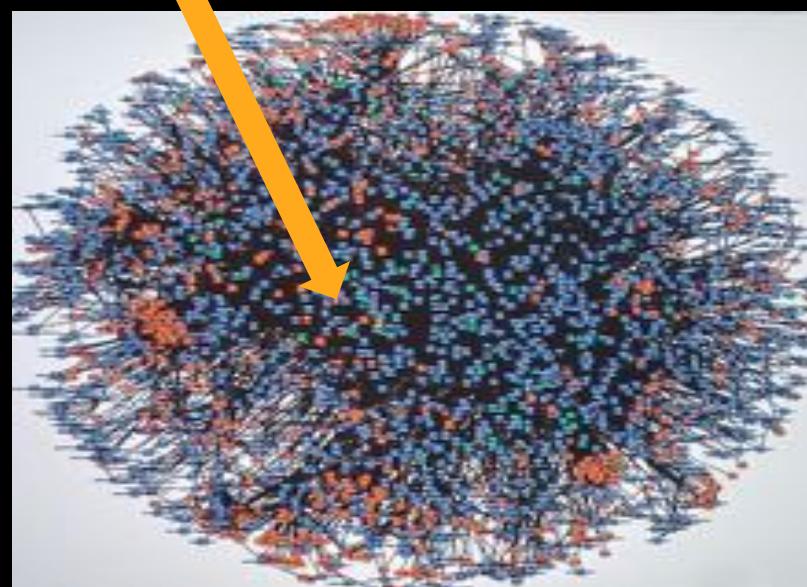
Causes

Across all technologies and clouds,
Latency and Errors indicate symptoms,
Traffic and Saturation indicate causes.

Simplify complex environments with one-hop dependency views

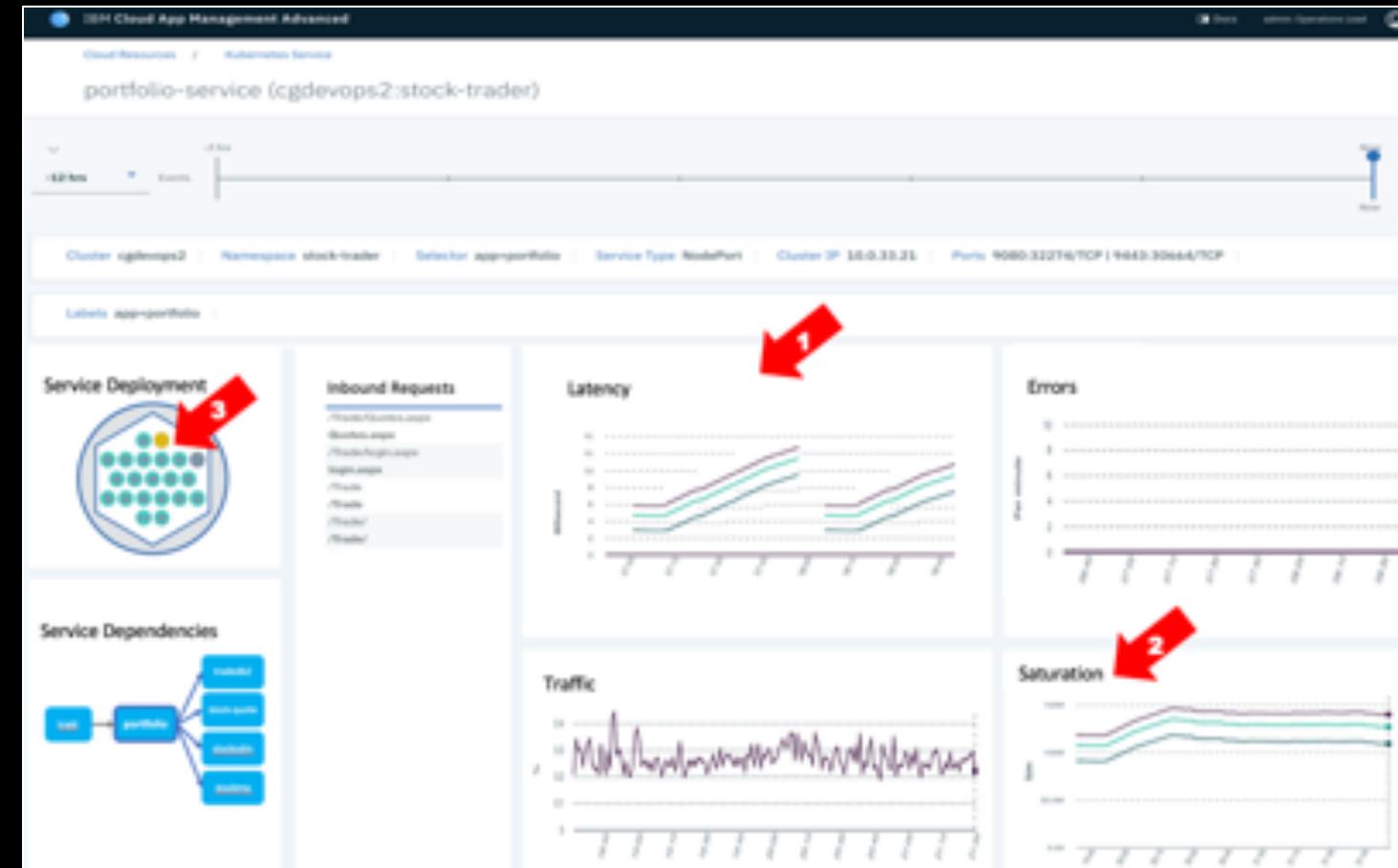
Instead of deciphering this...

Service A has a slow response time



Now what???

...Look at this instead



Cloud Resources

*Manage dynamic workloads
simplified with Golden Signals*



Platform

- Kubernetes
- Red Hat OpenShift
- IBM Cloud Private

Runtime

- Liberty DC
- Node.js DC

Others

- NGINX
- Spring Boot (via Unified Agent)
- Telegraf plugin framework
- Zipkin & Jaeger

Traditional Resources

Modernize with Golden Signals



IBM Middleware

- IBM DataPower
- IBM DB2
- IBM Infosphere DataStage
- IBM Integration Bus (IIB)
- IBM MQ
- IBM WAS (including Liberty)
- WebSphere Infra Manager

OS

- Linux OS
- Windows OS
- Unix OS

Hypervisors

- Linux KVM
- MS Hyper-V
- VMware

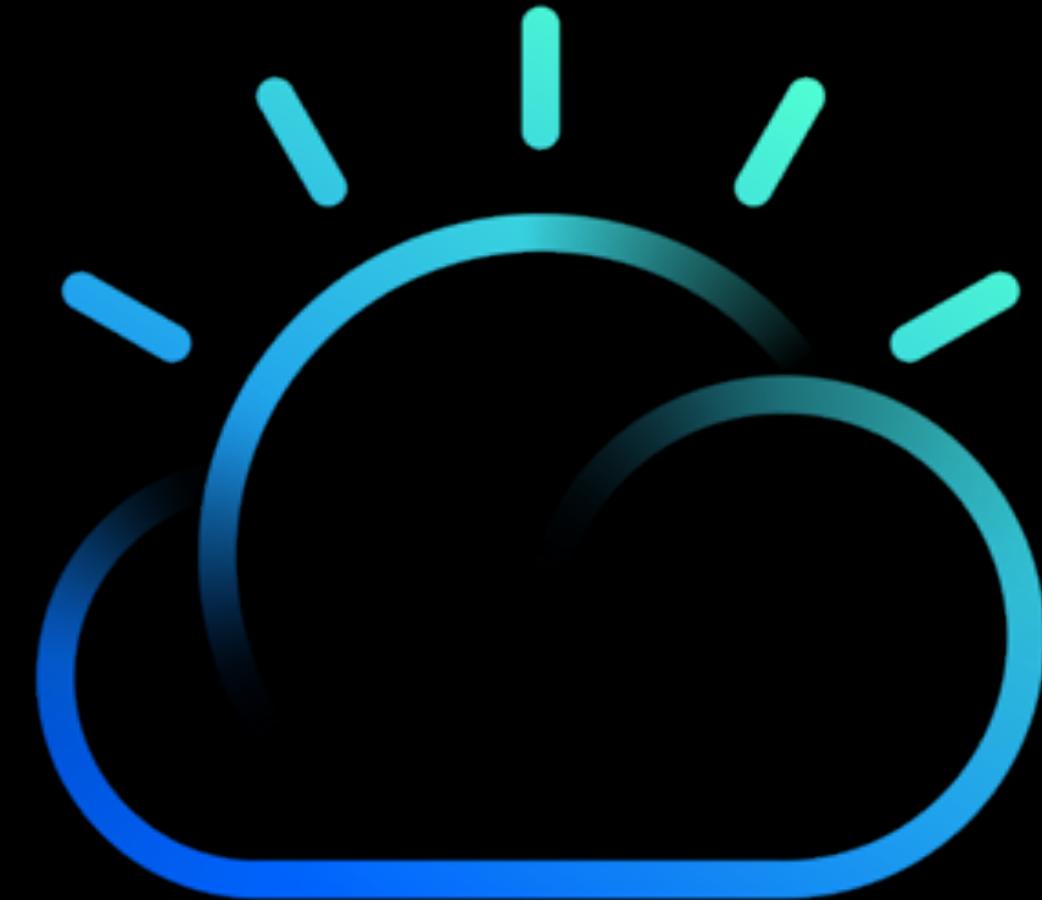
Microsoft

- .NET
- Exchange server
- IIS
- Lync server
- Office 365
- Skype
- SQL Server

Others

- | | |
|----------|---------------|
| • Hadoop | • Oracle DB |
| • HTTP | • Python |
| server | • SAP |
| • J2SE | • SAP HANA |
| • NetApp | • Spring Boot |

QUESTIONS?



The Journey to Cloud **Microservices**

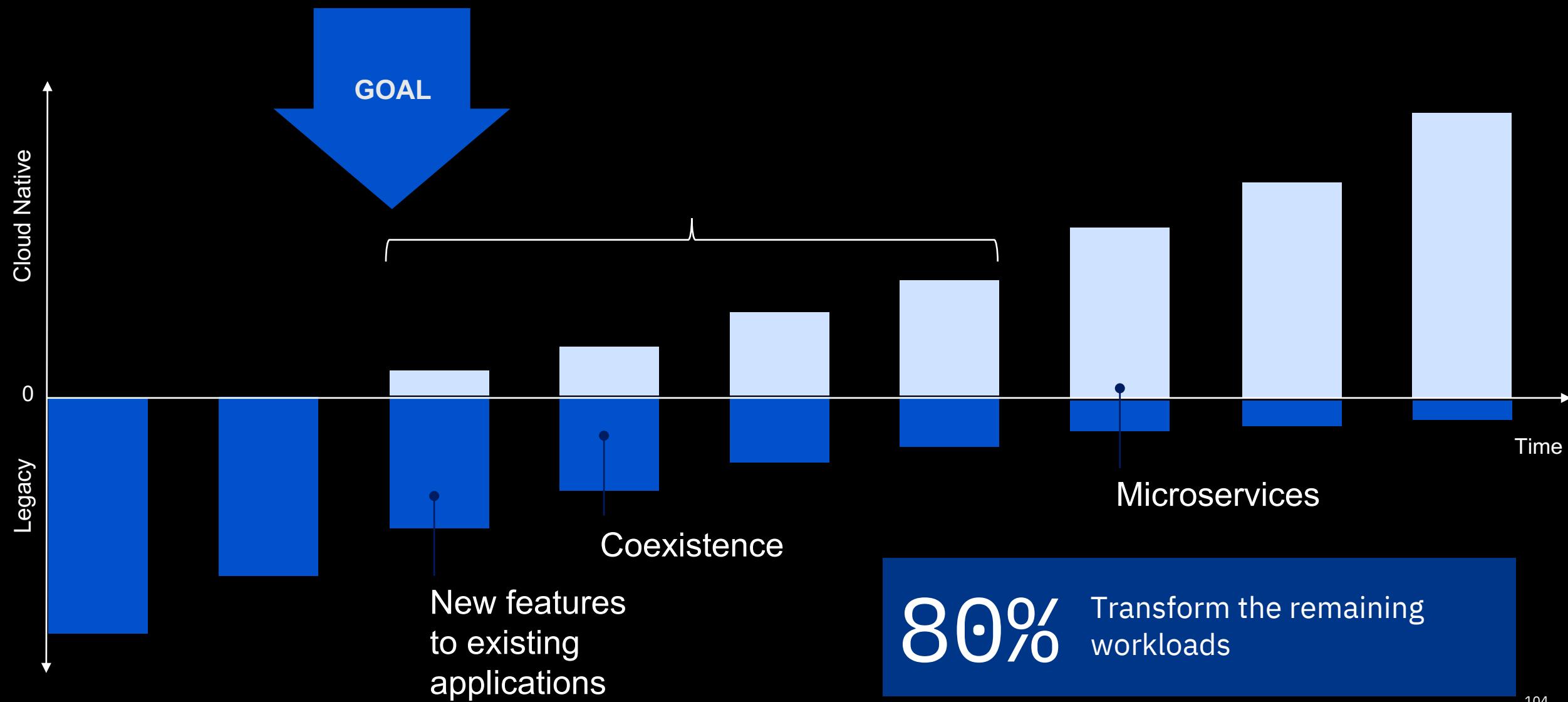
03



IBM Cloud

Digital Transformation

Cloud native and legacy apps will co-exist for the next 10+ years



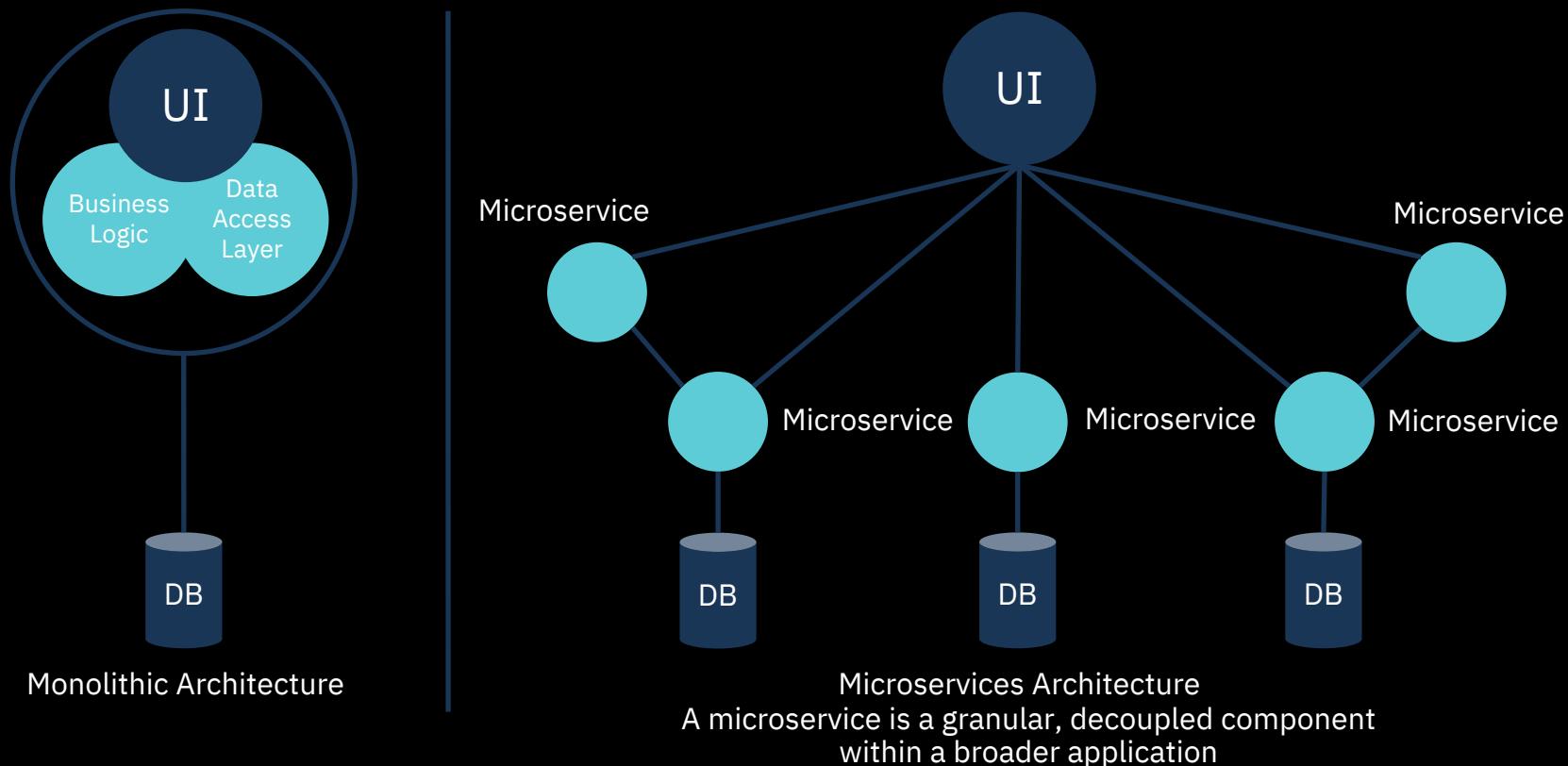
Microservices

Decomposing an application into **single function modules** which are **independently deployed and operated**

Accelerate delivery by minimizing communication and coordination between people

Microservices architecture

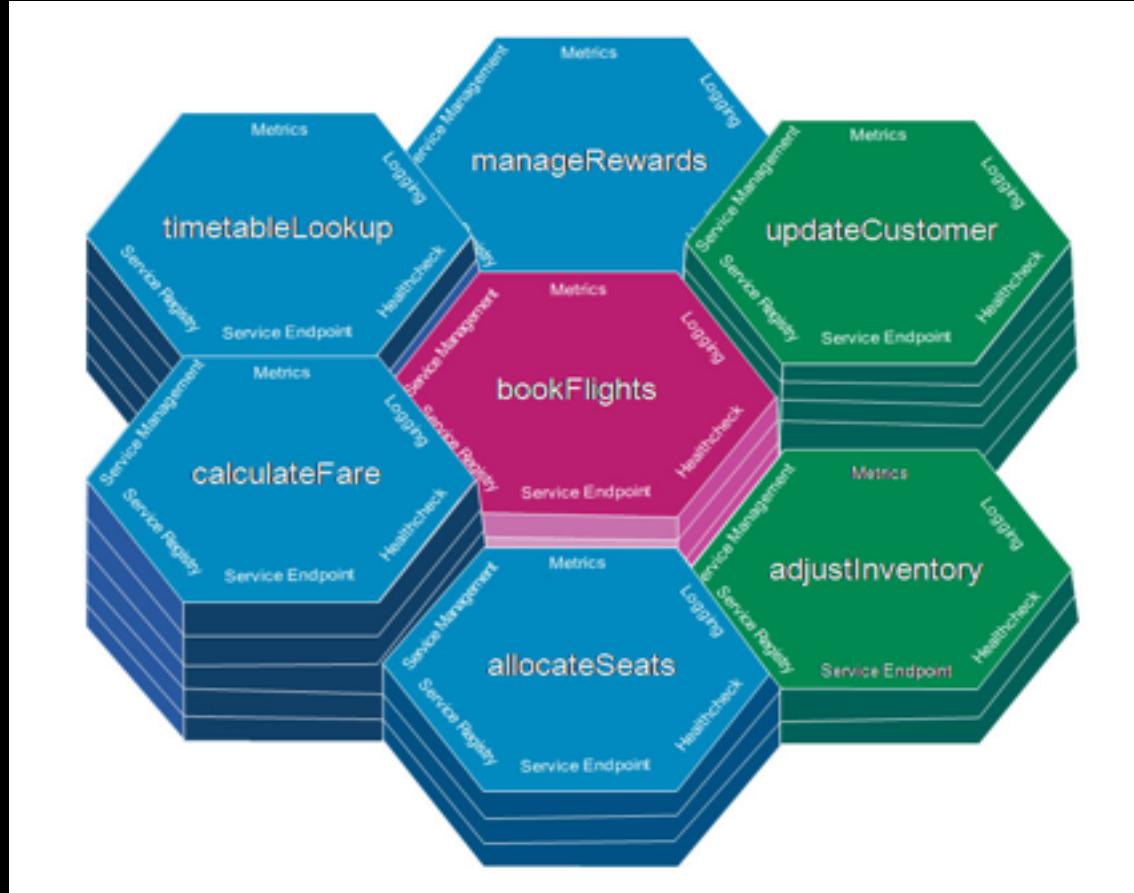
Simplistically, microservices architecture is about breaking down large silo applications into more manageable, fully decoupled pieces



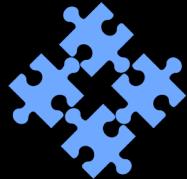
Example application using microservices

Airline reservation application

- Book flights
- Timetable lookup
- Calculate fare
- Allocate seats
- Manage rewards
- Update customer
- Adjust inventory



Microservices – key tenets



Large monoliths are **broken down into many small services**

- Each service runs its own process
- There is one service per container



Services are **optimized for a single function**

- There is only one business function per service
- The Single-responsibility Principle: A microservice should have one, and only one, reason to change



Communication via **REST API** and **message brokers**

- Avoid tight coupling introduced by communication through a database



Per-service continuous delivery (CI/CD)

- Services evolve at different rates
- Let the system evolve, but set architectural principles to guide that evolution



Per-service high availability and clustering decisions

- One size or scaling policy is not appropriate for all
- Not all services need to scale; others require autoscaling up to large numbers

Microservices – advantages

In a microservices architecture each component:

- Is **developed independently** and has **limited, explicit dependencies** on other services
- Is developed by a **single, small team** in which all team members can understand the entire code base
- Is developed on its **own timetable** so new versions are delivered independently of other services
- **Scales and fails independently** which isolates any problems
- Can be developed in a different **language**
- **Manages its own data** to select the best technology and schema

Obvious rules for developing cloud applications

1. Don't code your application directly to a **specific topology**
2. Don't assume the **local file system** is permanent
3. Don't keep **session state** in your application
4. Don't assume any specific **infrastructure dependency**
5. Don't use **infrastructure APIs** from within your application
6. Don't rely on **OS-specific features**
7. Don't **manually install** your application

Read the article : http://www.ibm.com/developerworks/websphere/techjournal/1404_brown/1404_brown.html

The 12 Factor App

1. Codebase - One codebase tracked in revision control, many deploys
2. Dependencies - Explicitly declare and isolate dependencies
3. Config - Store config in the environment
4. Backing services - Treat backing services as attached resources
5. Build, release, run - Strictly separate build and run stages
6. Processes - Execute the app as one or more stateless processes
7. Port binding - Export services via port binding
8. Concurrency - Scale out via the process model
9. Disposability - Maximize robustness with fast startup and graceful shutdown
10. Dev/prod parity - Keep development, staging, and production as similar as possible
11. Logs - Treat logs as event streams
12. Admin processes - Run admin/management tasks as one-off processes

Not a methodology to do cloud-native design!

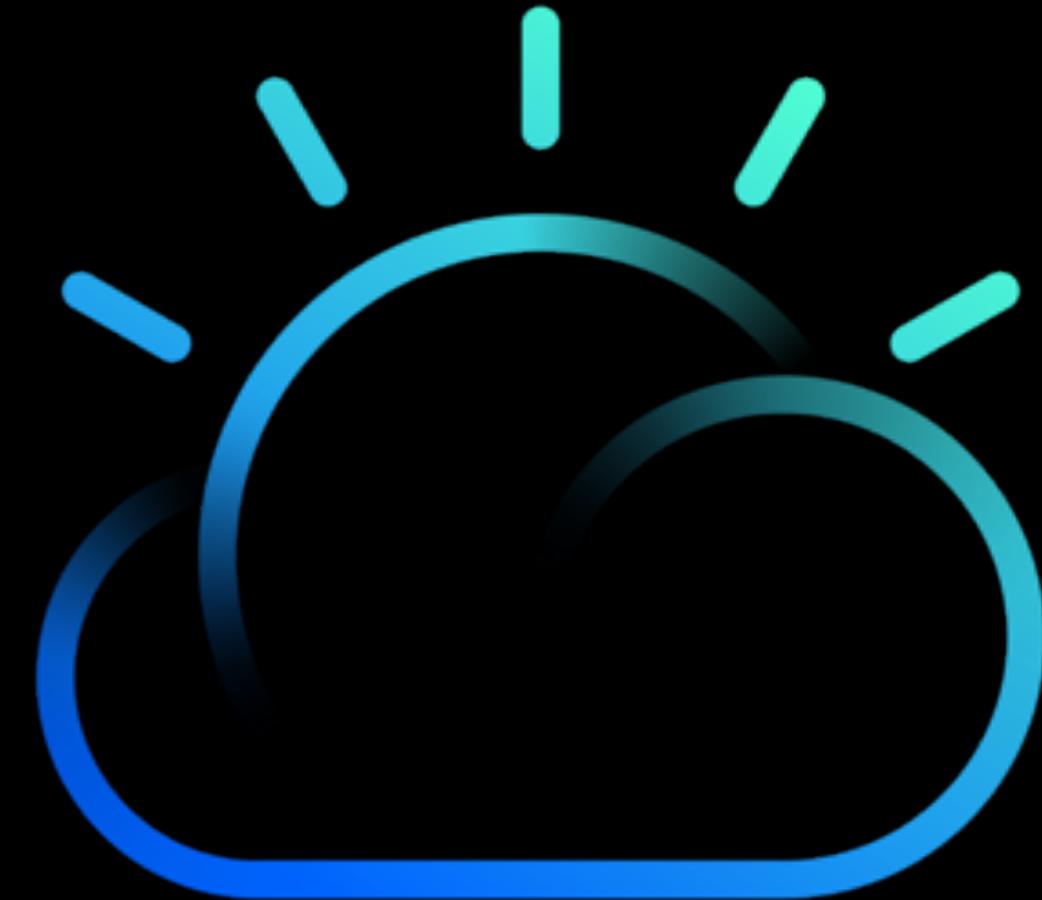
<https://12factor.net/>

That said....

Microservices
are
hard

..but we're
getting ahead
of ourselves

QUESTIONS?



The Journey to Cloud **Docker**

04



IBM Cloud

Everybody Loves Containers



A **standard way to package an application and all its dependencies** so that it can be moved between environments and run without changes

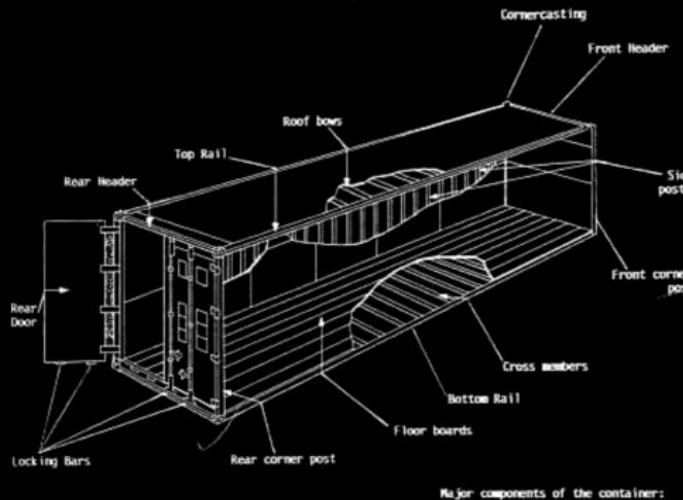
Containers work by **isolating the differences between applications** inside the container so that everything outside the container can be **standardized**

Microservices implementation with Containers

Why it works – separation of concerns

Development

- Worries about what's “**inside**” the container
 - Code
 - Libraries
 - Package Manager
 - Apps
 - Data
- All Linux servers look the same

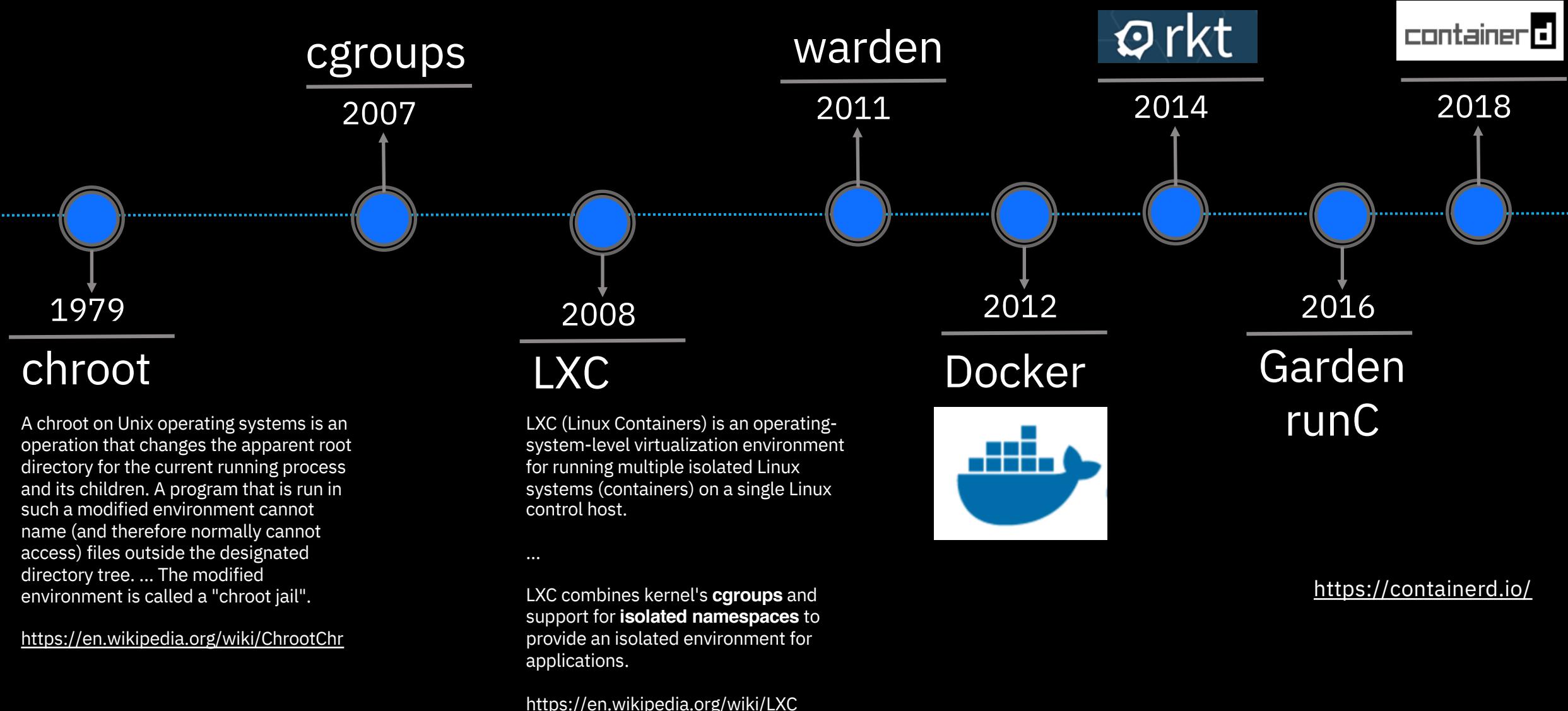


Operations

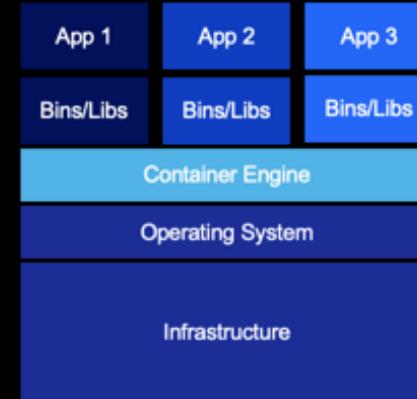
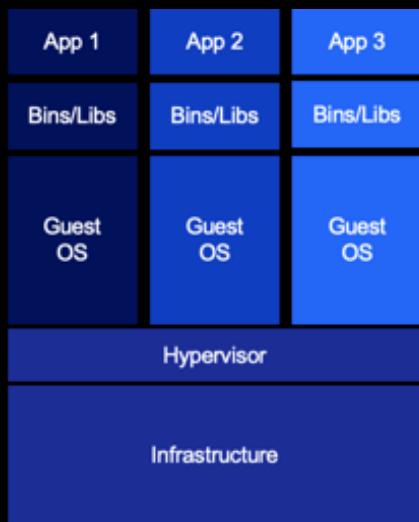
- Worries about what's “**outside**” the container
 - Logging
 - Remote Access
 - Monitoring
 - Network Config
- All containers start, stop, copy, attach, migrate, etc... the same way

Clear ownership boundary between
Dev and IT Ops drives DevOps adoption and fosters agility

Container History



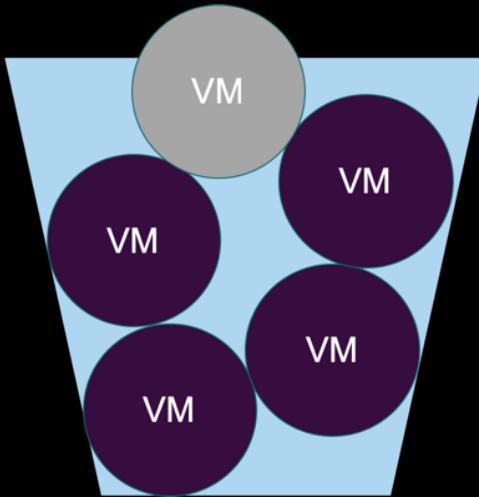
VMs vs. Containers



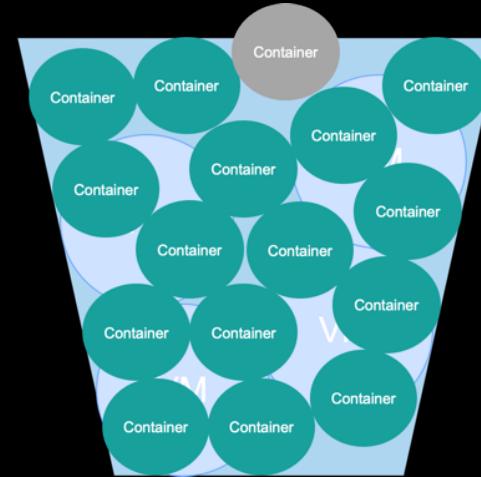
- + VM Isolation
- Complete OS
- Static Compute
- Static Memory

- + Container Isolation
- + Shared Kernel
- + Burstable Compute
- + Burstable Memory

VMs vs. Containers



- + VM Isolation
- Complete OS
- Static Compute
- Static Memory
- Low Resource Utilization



- + Container Isolation
- + Shared Kernel
- + Burstable Compute
- + Burstable Memory
- + High Resource Utilization

Advantages of Containers



Containers are **portable**



Containers are **easy to manage**



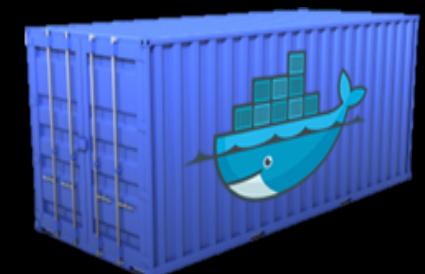
Containers provide “**just enough**” isolation



Containers use hardware **more efficiently**



Containers are **immutable**



Docker Components

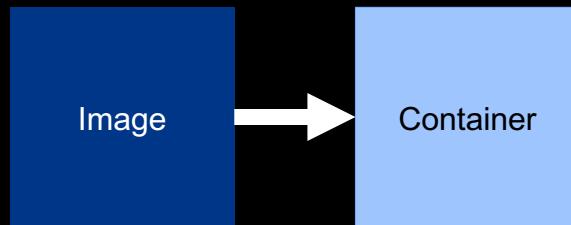
Container

Smallest compute unit



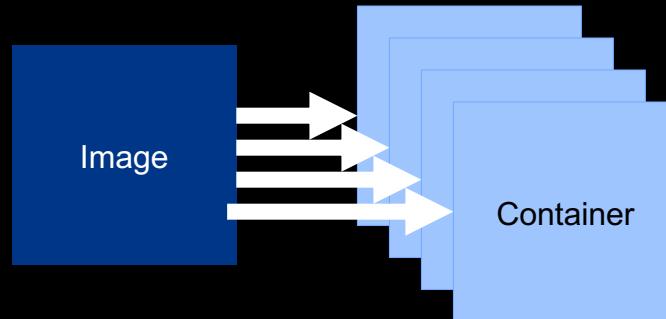
Docker Components

Containers
are created from
Images



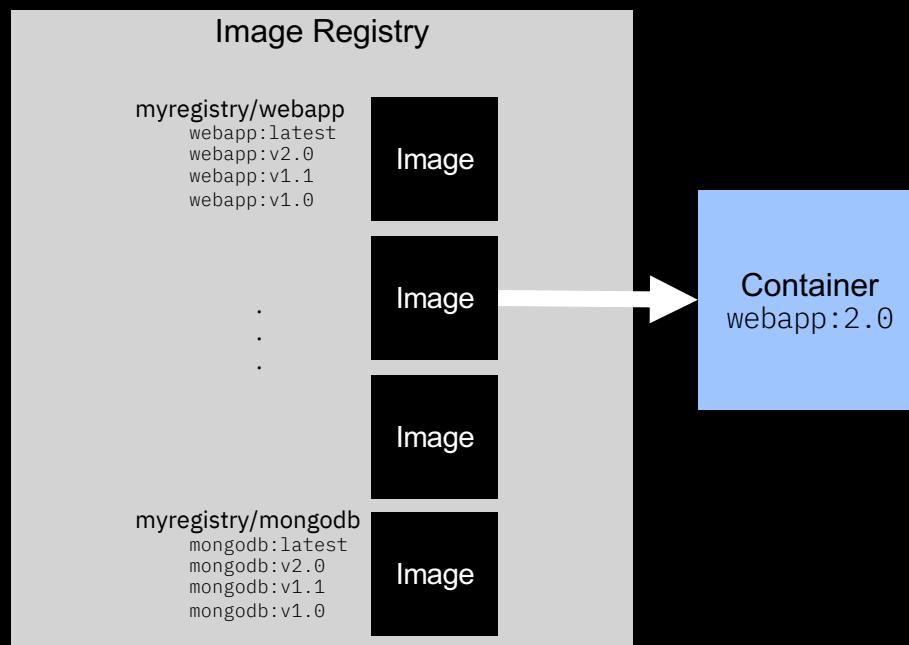
Docker Components

As **many Containers**
as needed can be created from
Images



Docker Components

The **Image Registry**
stores the versioned
Images
to create
Containers



Docker Components



Image

A read-only snapshot of a container stored in Docker Hub to be used as a template for building containers



Container

The standard unit in which the application service resides or transported



Docker Hub/Registry

Available in SaaS or Enterprise to deploy anywhere you choose
Stores, distributes, and shares container images



Docker Engine

A program that creates, ships, and runs application containers
Runs on any physical and virtual machine or server locally, in private or public cloud. Client communicates with Engine to execute commands

Why Containers

Docker Layers

Inheritance

- Build on the work of those who came before you

```
# Pull base image
FROM tomcat:8-jre8

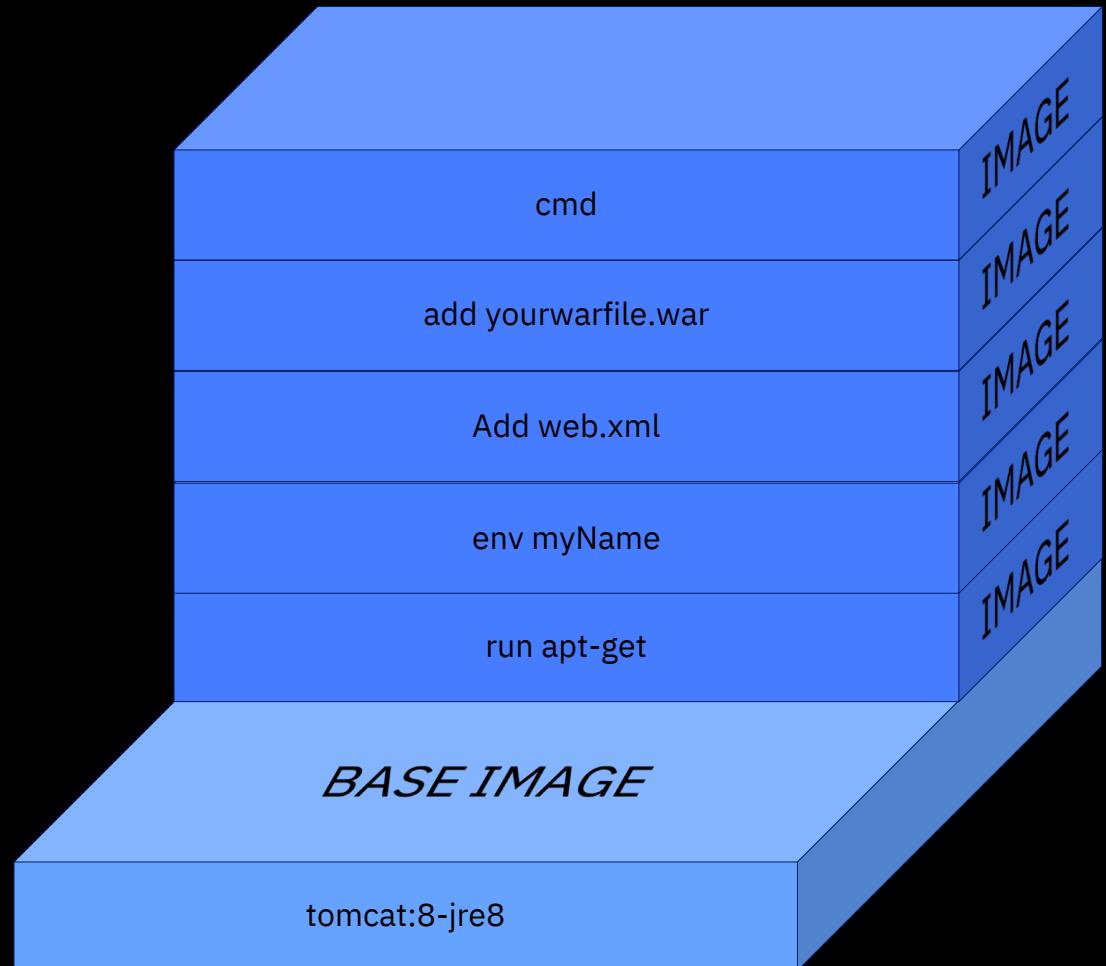
# Maintainer
MAINTAINER youremailaddress"

# Run command
RUN apt-get update && apt-get -y upgrade

# Set variables
ENV myName John Doe

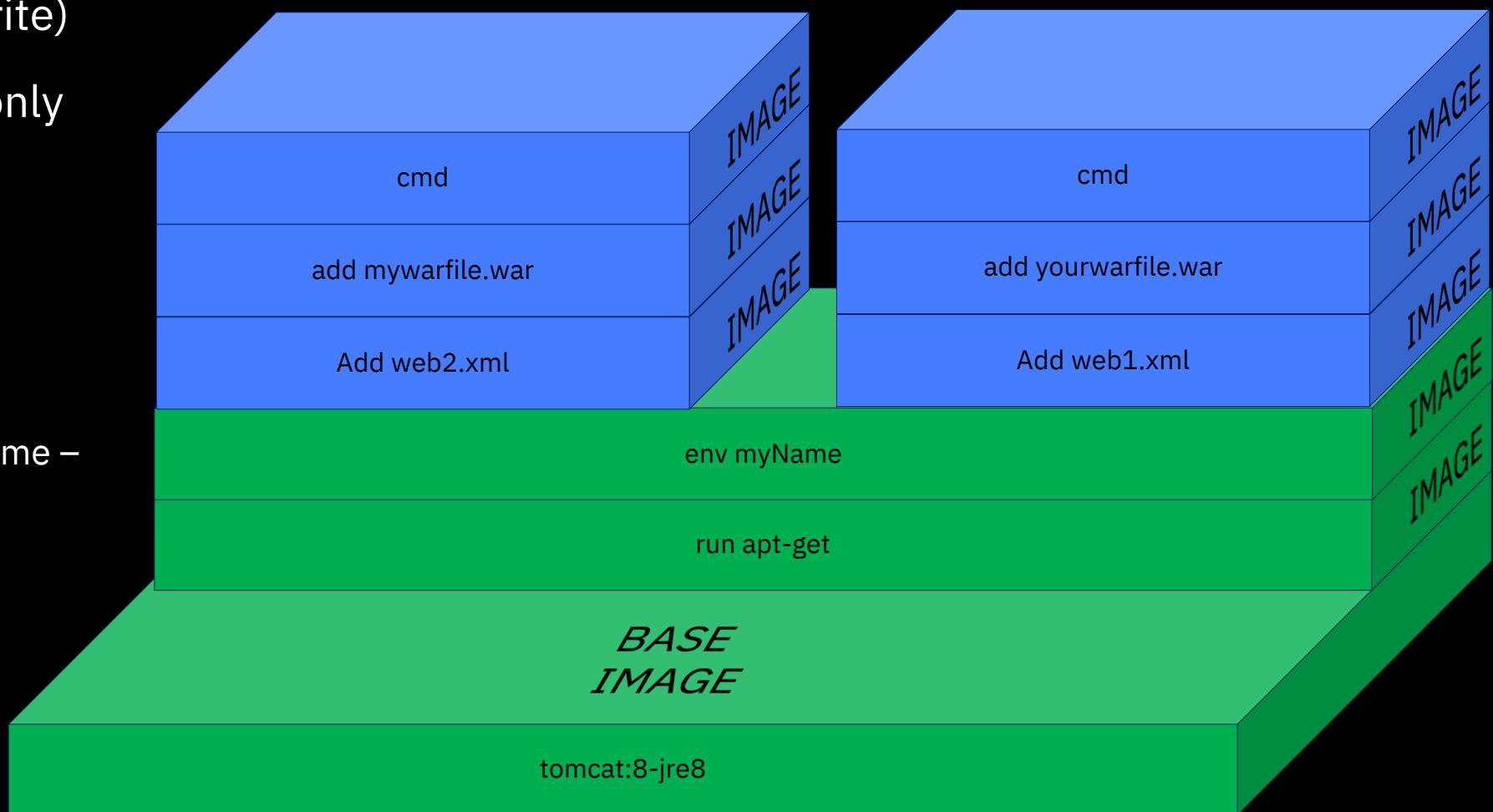
# Copy to images tomcat path
ADD web.xml /usr/local/tomcat/conf/
ADD yourwarfile.war /usr/local/tomcat/webapps/

# Run server
CMD ["catalina.sh", "run"]
```

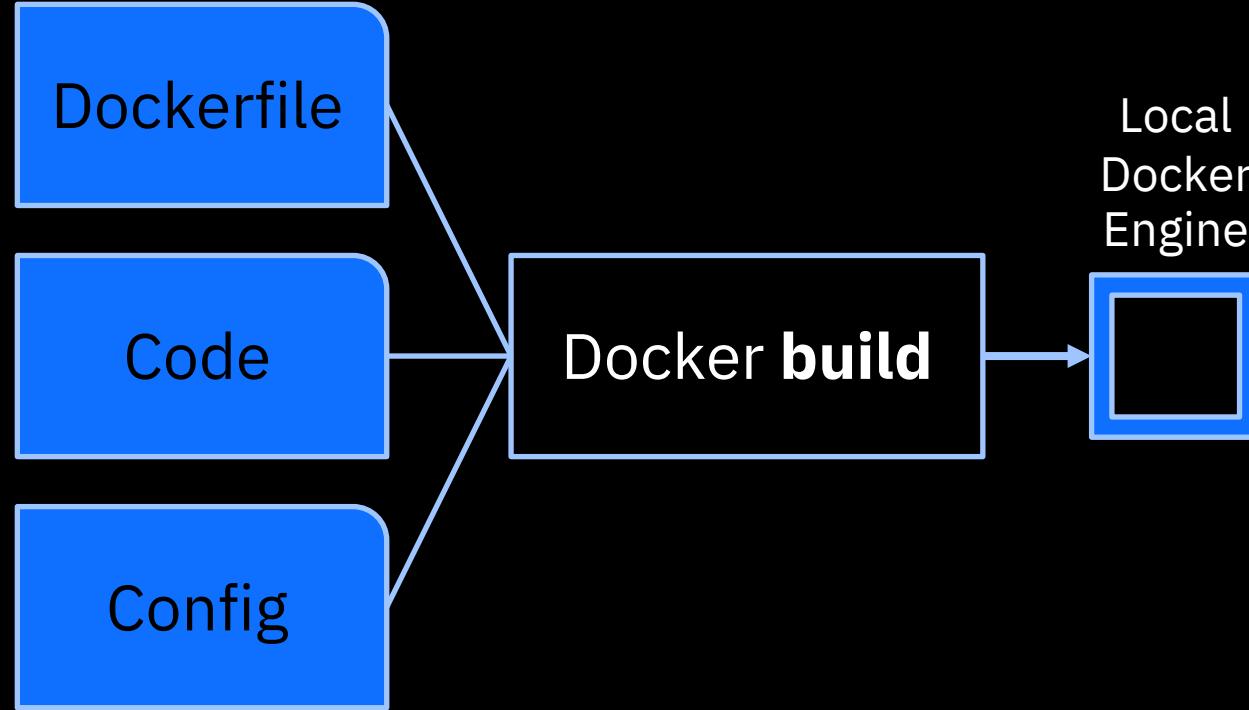


Why Containers

- Docker uses a **layered filesystem** (copy-on-write)
- New files (& edits) are only visible to current/above layers
- Layers allow for **reuse**
 - More containers per host
 - Faster start-up/download time – base layers are "cached"



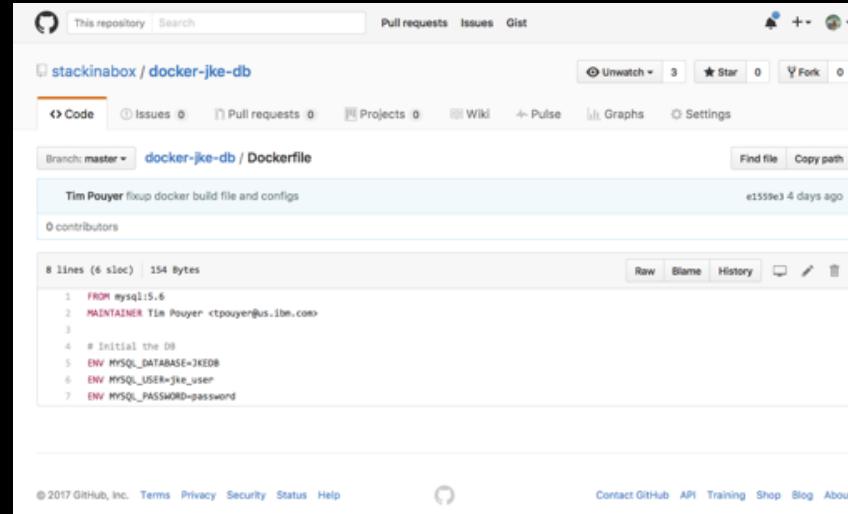
Docker Basics – Build



Docker Basics - Build

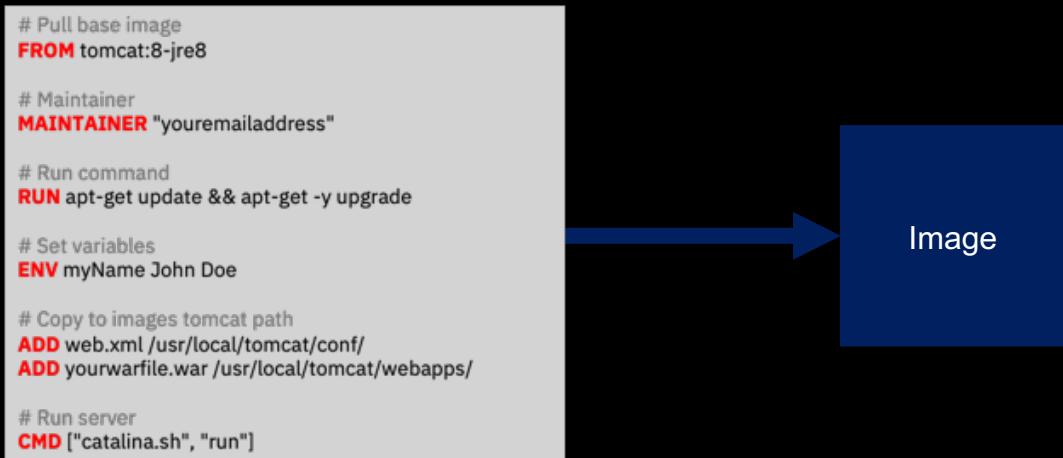
Dockerfile

- Text based file describing:
 - Previous Layer
 - Environment Variables
 - Commands used to populate data/software/frameworks/etc...
 - Command to run when executed



A screenshot of a GitHub repository page for 'stackinabox / docker-jke-db'. The 'Dockerfile' tab is selected, showing the following code:

```
FROM mysql:5.6
MAINTAINER Tim Poyer <tpoyer@us.ibm.com>
# Initial the db
ENV MYSQL_DATABASE=jKEdb
ENV MYSQL_USER=jke_user
ENV MYSQL_PASSWORD=password
```



A diagram illustrating the build process. On the left, a white rectangular box contains a Dockerfile with various commands. A large blue arrow points from this box to a solid blue square labeled 'Image' on its right side.

```
# Pull base image
FROM tomcat:8-jre8

# Maintainer
MAINTAINER "youremailaddress"

# Run command
RUN apt-get update && apt-get -y upgrade

# Set variables
ENV myName John Doe

# Copy to images tomcat path
ADD web.xml /usr/local/tomcat/conf/
ADD yourwarfile.war /usr/local/tomcat/webapps/

# Run server
CMD ["catalina.sh", "run"]
```

Docker Basics – Build -Dockerfile

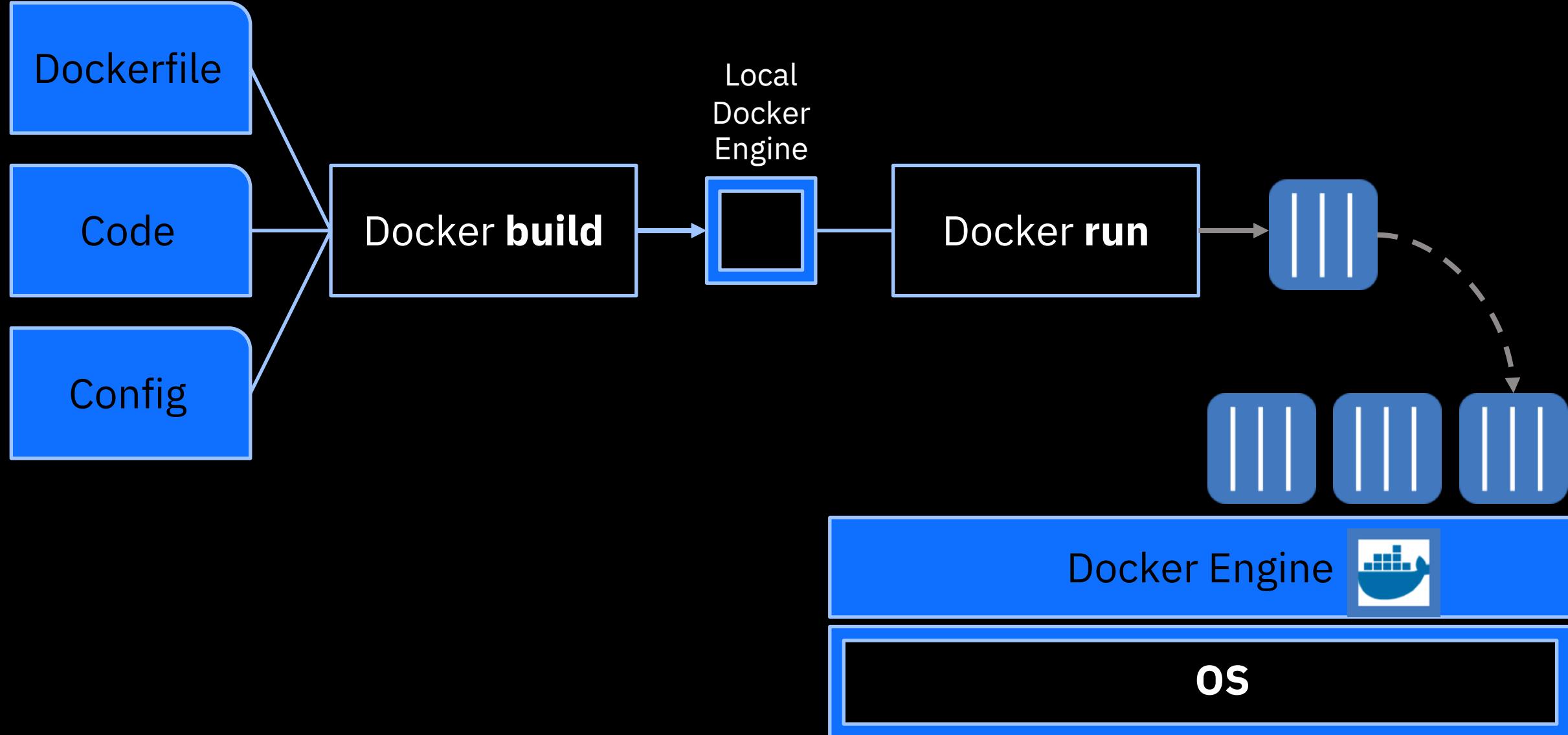
one process per container

- A text file that builds an image using Docker directives

- FROM # Pull base image
FROM tomcat:8-jre8
- RUN # Maintainer
MAINTAINER "youremailaddress"
- COPY # Run command
RUN apt-get update && apt-get -y upgrade
- ENTRYPOINT # Set variables
ENV myName John Doe
- LABEL # Copy to images tomcat path
ADD web.xml /usr/local/tomcat/conf/
ADD yourwarfile.war /usr/local/tomcat/webapps/
- ENV # Run server
CMD ["catalina.sh", "run"]

```
docker build -t myimage ./Dockerfile
```

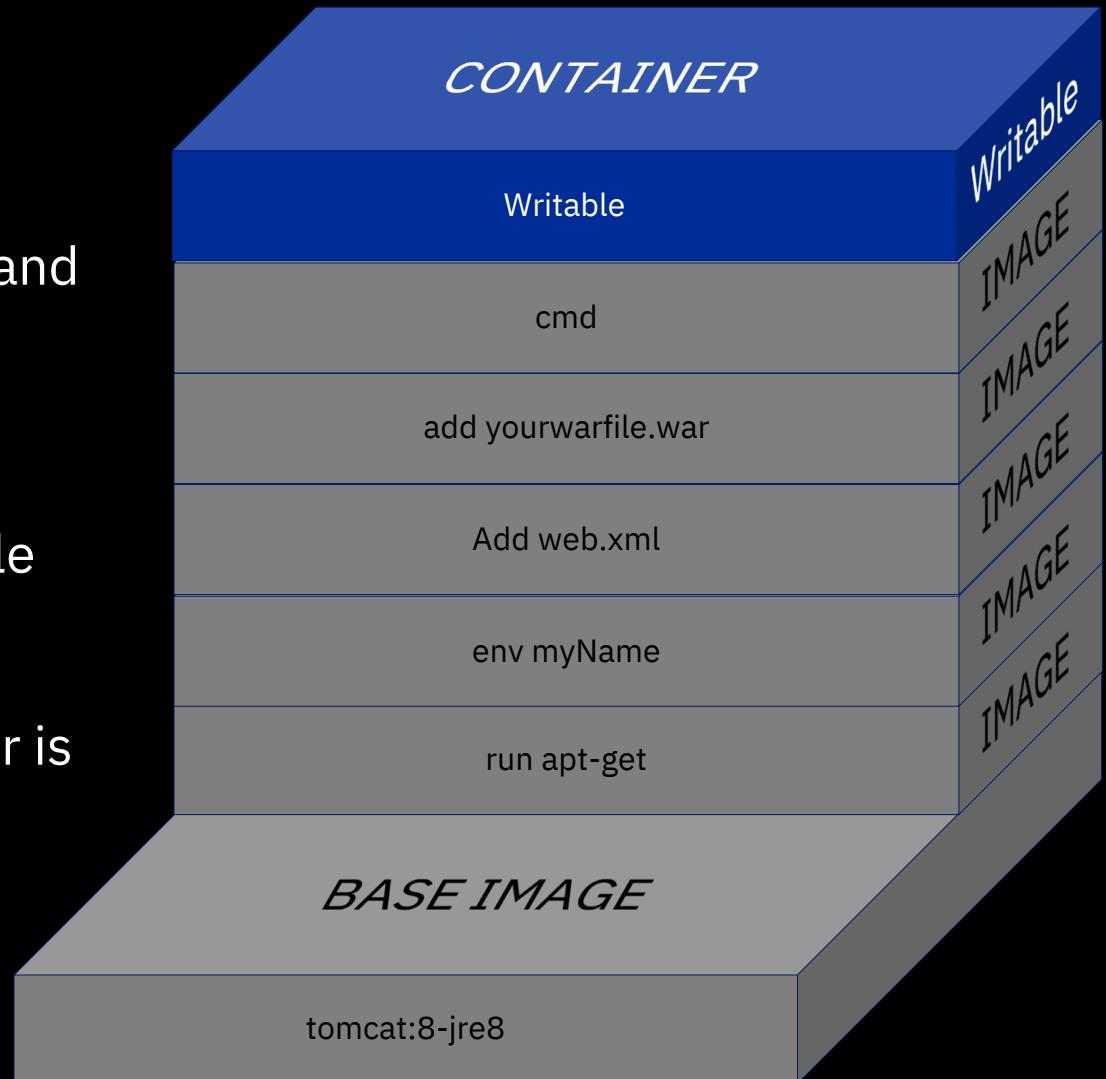
Docker Basics – Run



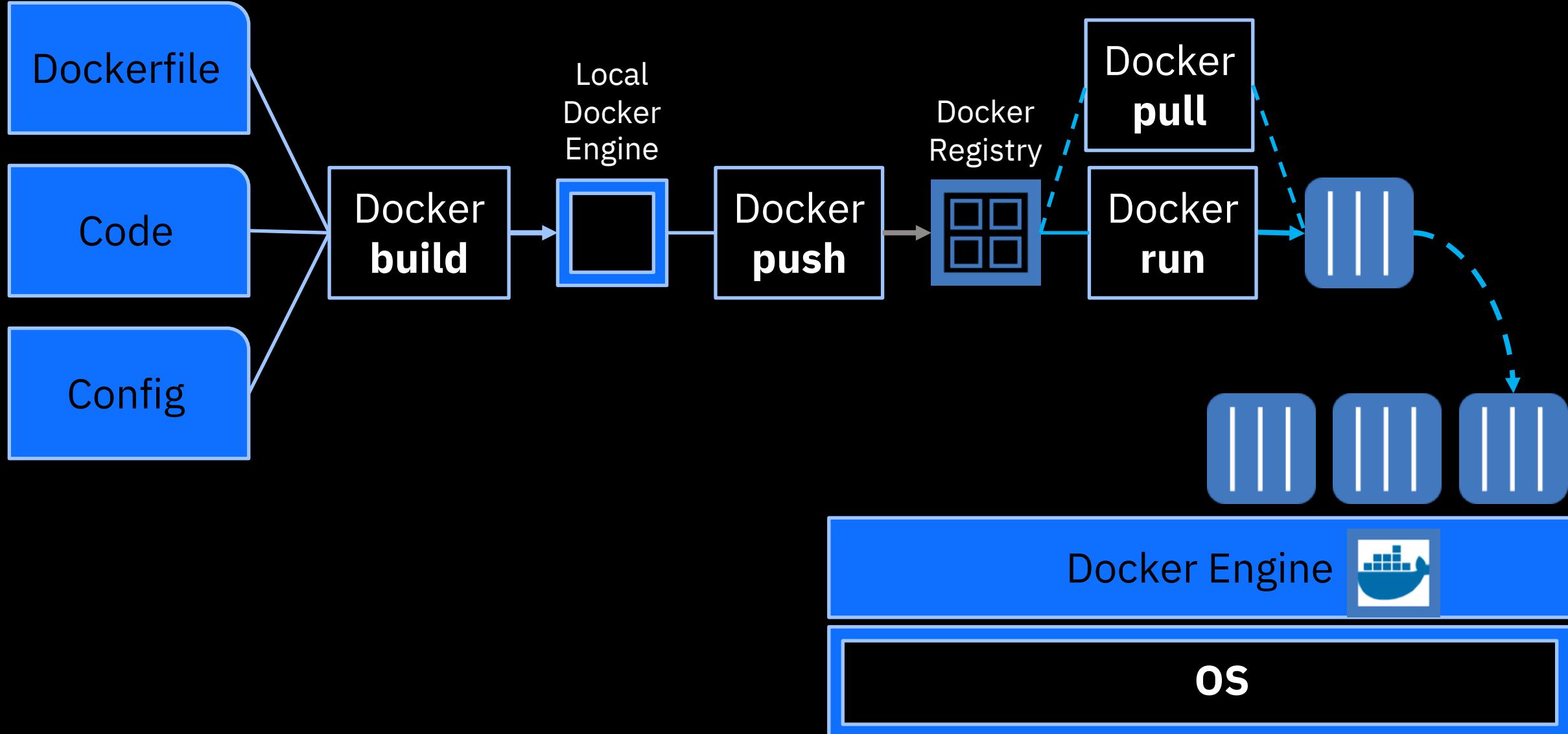
Docker Basics – Run

Docker Layers

- The biggest difference between a **container** and an **image** is the **top writable layer**.
- All writes to the container that add new or modify existing data are stored in this writable layer.
- When a container is deleted, its writable layer is also deleted. The underlying image is unchanged.



Docker Basics – Store, Retrieve & Run with registry



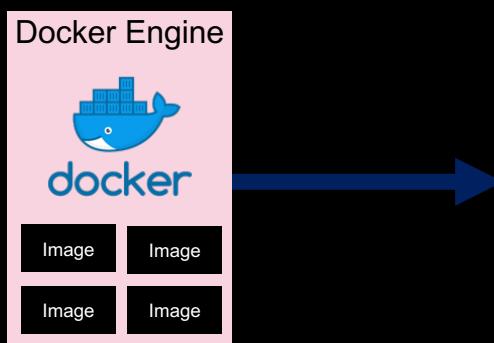
Docker Basics – Store & Retrieve

Docker Registry

- Private Local
- Public Docker Hub
- Private Shared

docker images

```
tpouyer at Laptop in ~/Development/workspace/docker/docker-jke-db on master
$ docker images
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
stackinabox/demo-docker    dev      4e2a1f6cc1a4   25 hours ago  528.3 MB
stackinabox/demo-jke        1.1      a596fbf2c3b7   3 days ago   672 MB
stackinabox/demo-jke        dev      a596fbf2c3b7   3 days ago   672 MB
stackinabox/demo-jke        latest   a596fbf2c3b7   3 days ago   672 MB
stackinabox/demo-jke        1.1      a596fbf2c3b7   3 days ago   672 MB
stackinabox/demo-jke        latest   a596fbf2c3b7   3 days ago   672 MB
stackinabox/jke-web         latest   a4be0cdad8bc   4 days ago   462.3 MB
stackinabox/demo-jke-db     dev      54bdbf42b999   4 days ago   327.5 MB
stackinabox/jke-db          dev      54bdbf42b999   4 days ago   327.5 MB
stackinabox/jke-db          latest   53c878fbf034   7 days ago   477 MB
websphere-liberty           6.2.2.0  7e27c3fb9c96   10 days ago  445.7 MB
mysql                       5.6      a896fd82dcfd5  13 days ago  327.5 MB
mysql                       latest   f3694c67abdb   13 days ago  400.1 MB
$
```



myregistry/webapp

webapp:latest
webapp:v2.0
webapp:v1.1
webapp:v1.0

myregistry/mongodb

mongodb:latest
mongodb:v2.0
mongodb:v1.1
mongodb:v1.0

Docker Basics – Run

Docker run

- Local (containerd)

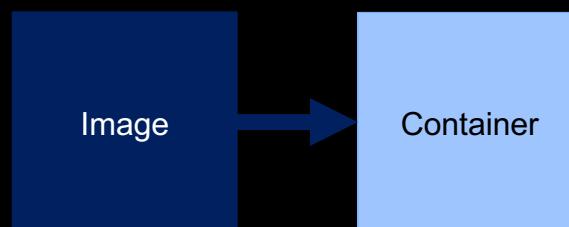
```
tpouyer at Laptop in ~/Development/workspace/docker/docker-jke-db on master
$ docker run -d postgres:latest
4eec29ae5eaa20798b1af8fa37873f7fc28cbb7cb789986b44f66cd94a784e0a

tpouyer at Laptop in ~/Development/workspace/docker/docker-jke-db on master
$ docker ps
CONTAINER ID        IMAGE               COMMAND      CREATED          STATUS          PORTS
ORTS
4eec29ae5eaa        postgres:latest   "/docker-entrypoint.s"   5 seconds ago   Up 4 seconds   5
432/tcp

tpouyer at Laptop in ~/Development/workspace/docker/docker-jke-db on master
$
```

```
docker run -d -e MYVAR=foo myimage:1.0.0
docker run -ti myimage:1.0.0 /bin/bash
```

...



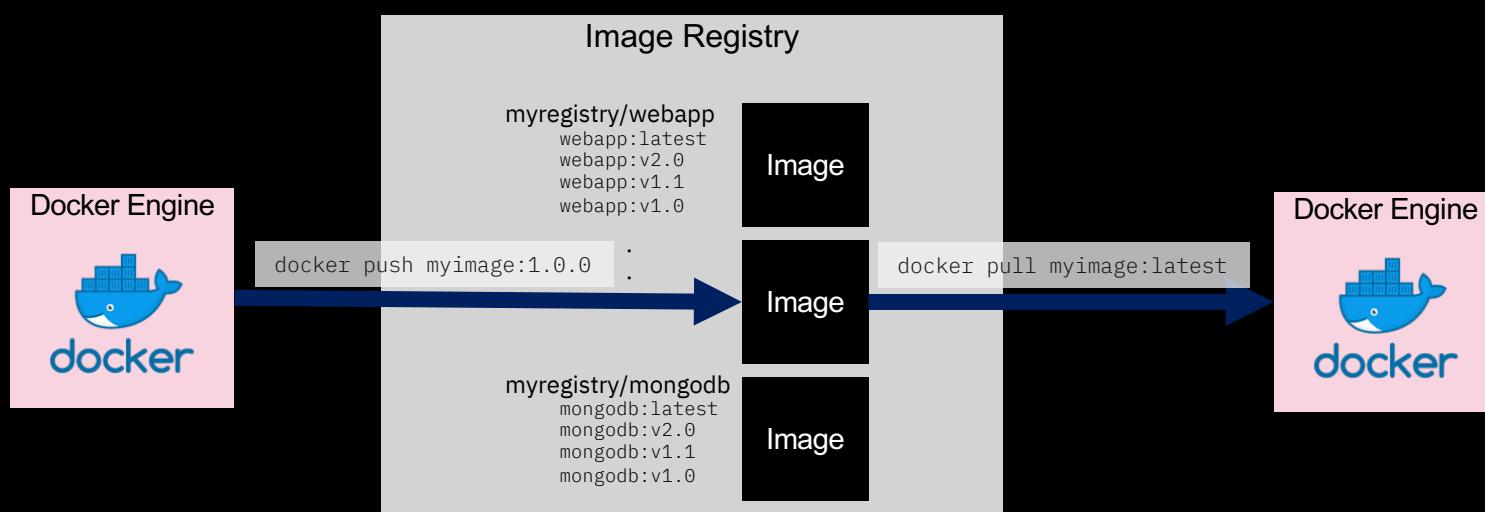
Docker Basics – Store & Retrieve with registry

Docker Registry

- Private Local
- Public Docker Hub
- Private Shared

```
docker pull myimage:latest  
docker push myimage:1.0.0
```

```
touyer at Laptop in ~/Development/workspace/docker/docker-jke-db on master  
$ docker pull postgres:latest  
latest: Pulling from library/postgres  
5040bd298390: Already exists  
f08454c3c700: Pull complete  
4db038cdfe03: Pull complete  
e1d9ba315f03: Pull complete  
25e8ee93170e: Pull complete  
3f28084c3f51: Pull complete  
78c91f0aedcd: Pull complete  
93ab52dbcbb8: Pull complete  
27ec75825613: Pull complete  
28ef691a9920: Pull complete  
0f0dd28755c9: Pull complete  
2a4a824861f7: Pull complete  
Digest: sha256:0842a7ef786aa26586238085160cb38451eb3d40856e7d222ae0069b6e6296877  
Status: Downloaded newer image for postgres:latest  
  
touyer at Laptop in ~/Development/workspace/docker/docker-jke-db on master  
$
```



Docker Basics – Registries

Hosting image repositories

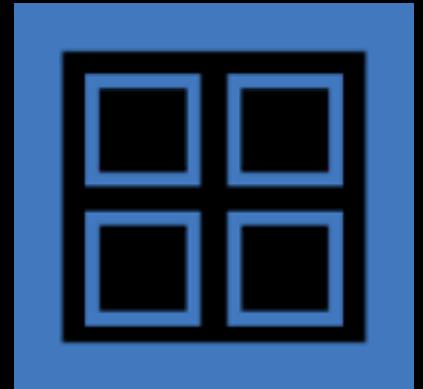
- You can define your own registry
- A registry is managed by a registry container

Public and Private registries

- Public Registry like **Docker Hub**
- <https://hub.docker.com>

Login into the registry

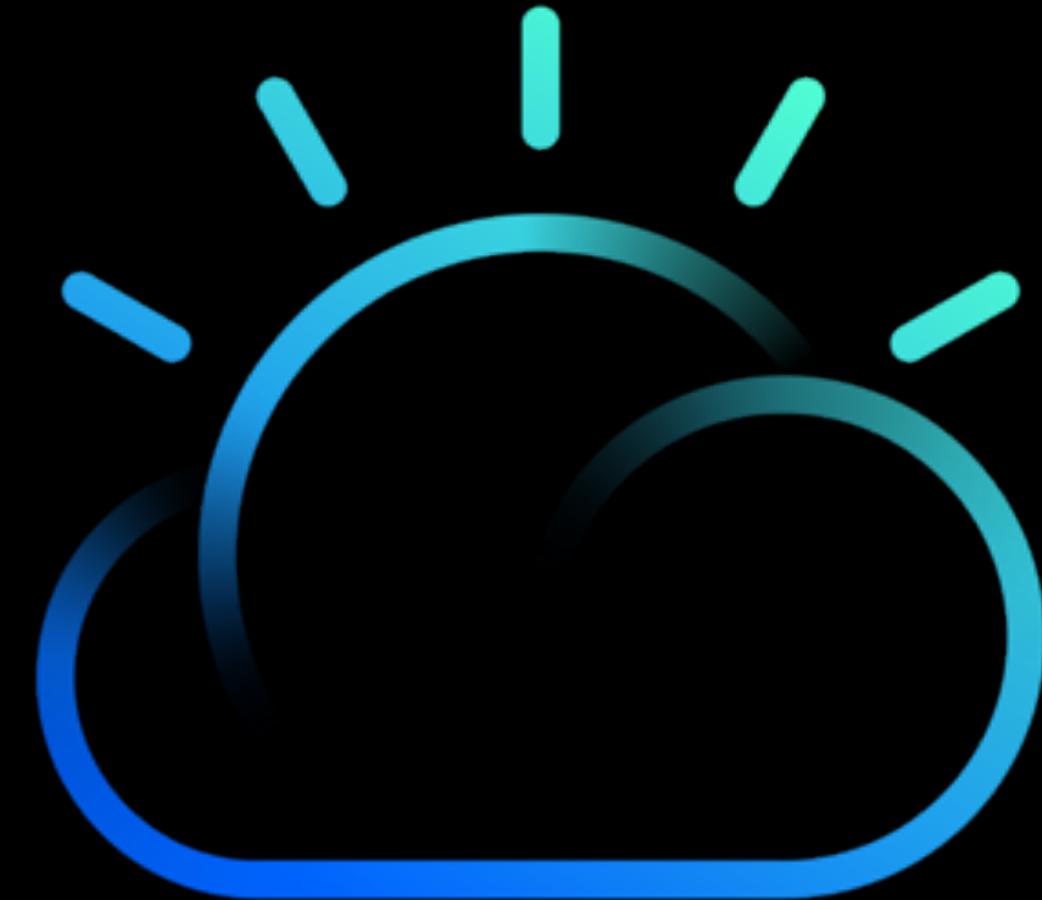
- Docker login domain:port



Docker Recap

- **Containers are not VMs**
- **Containers provide many benefits:**
 - Efficiency
 - Portability
 - Consistency
- **New challenges with containers:**
 - Production apps dependent on open-source projects
 - Existing tools may not be sufficient for container
 - Need to focus on business objectives

QUESTIONS?



The Journey to Cloud **Kubernetes**

05



IBM Cloud

Everybody Loves Containers

But when you go



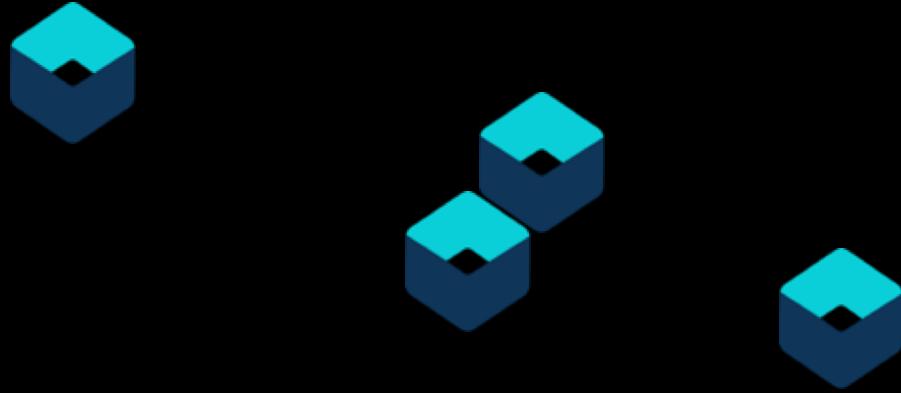
From this....



To this....



Everyone's container journey starts with one container....



At first the growth is easy to handle....

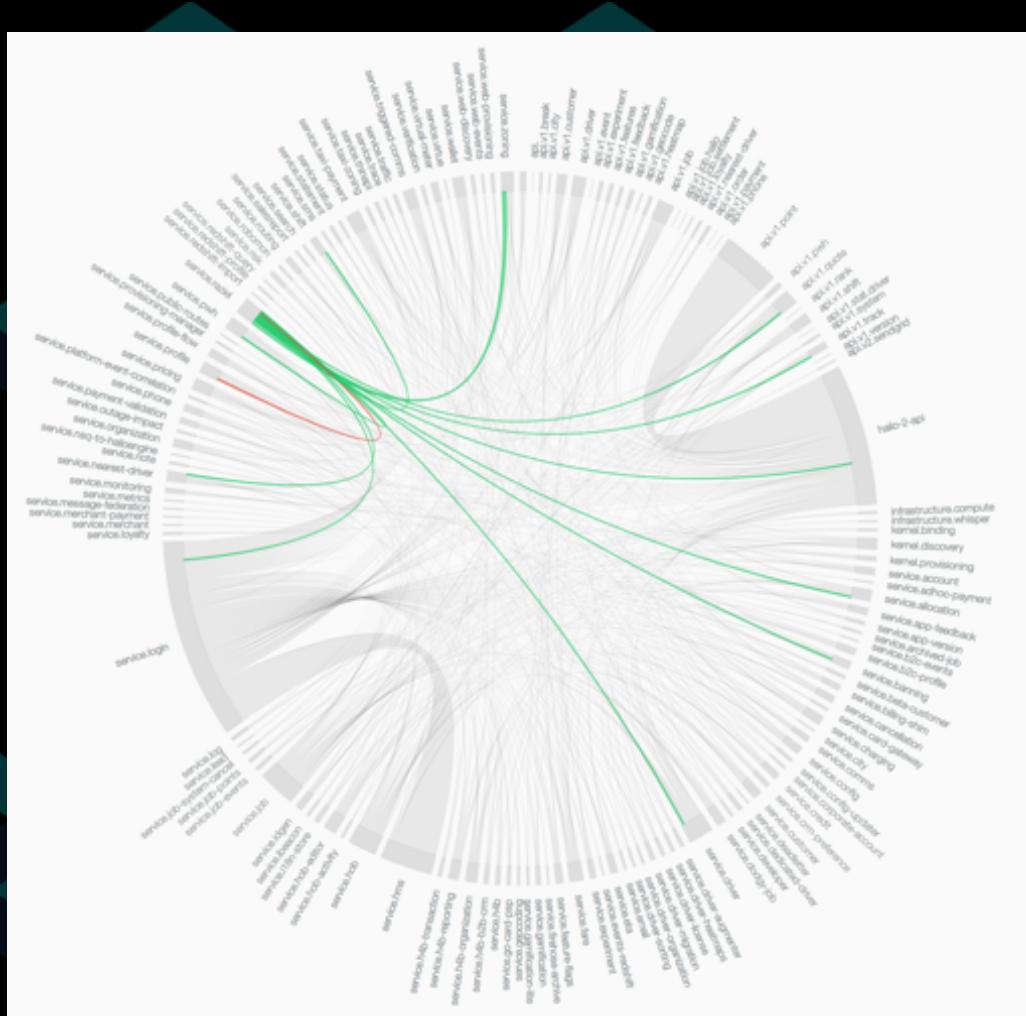


Pets vs Cattle

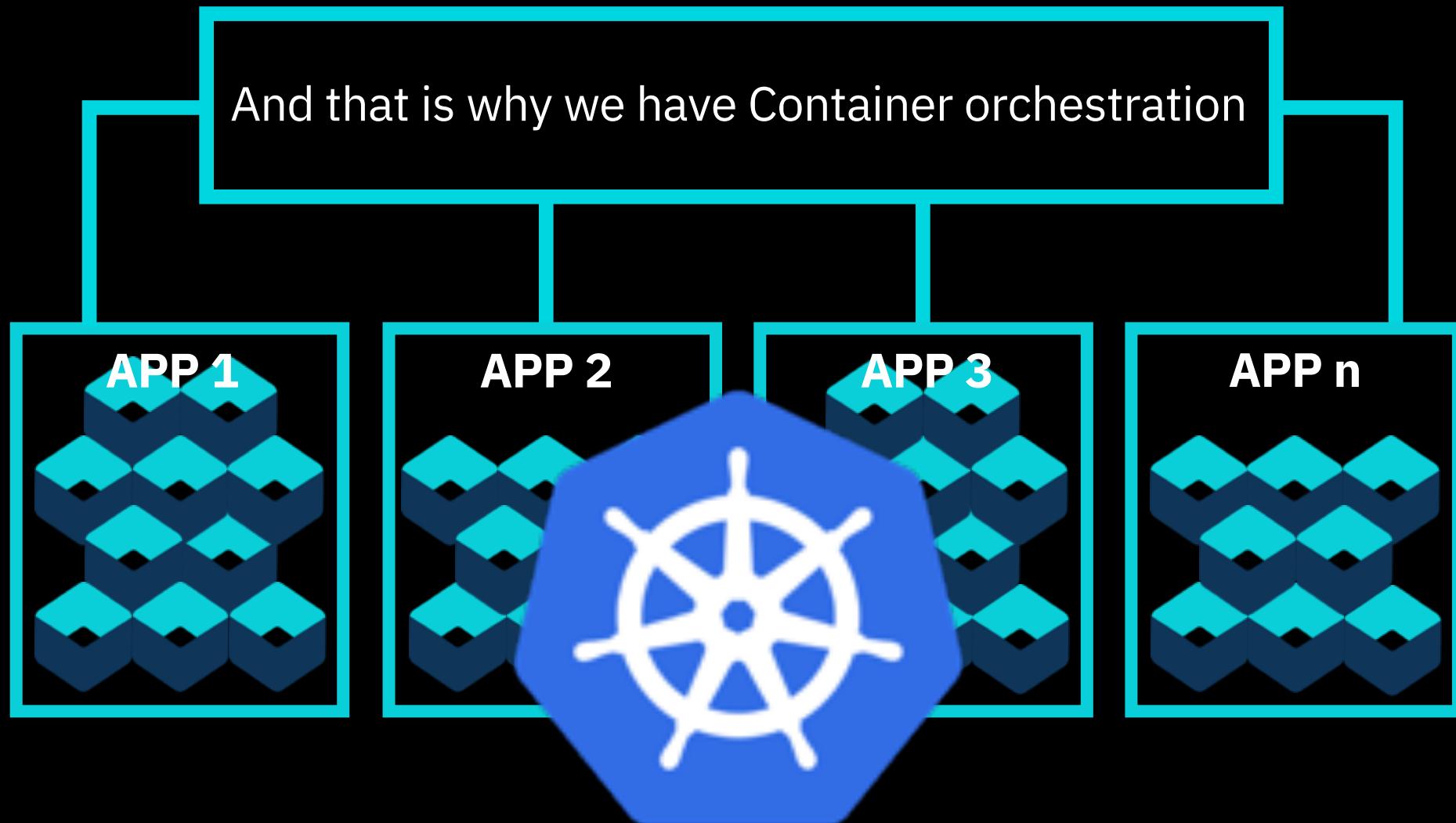
But soon you have many applications, many instances...

The trade off

Improved delivery velocity
in exchange for
increased operational complexity



Enter Kubernetes (K8s)





Imperative Systems

In an imperative system, **the user** knows the desired state and **the user** determines the sequence of commands to transition the system to the desired state.

Declarative Systems

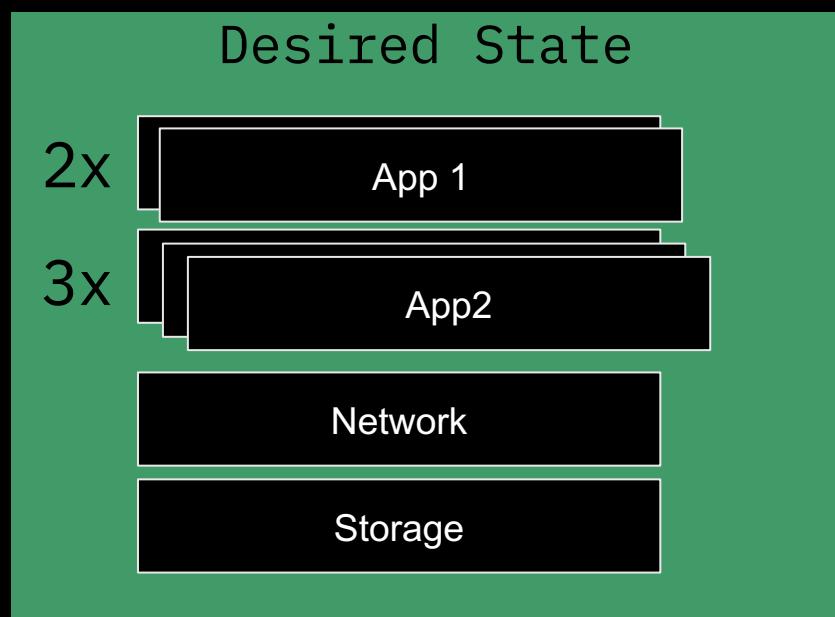
By contrast, in a declarative system, **the user** knows the desired state, supplies a representation of the desired state to the system, then **the system** reads the current state and determines the sequence of commands to transition the system to the desired state.

Kubernetes – Declarative System

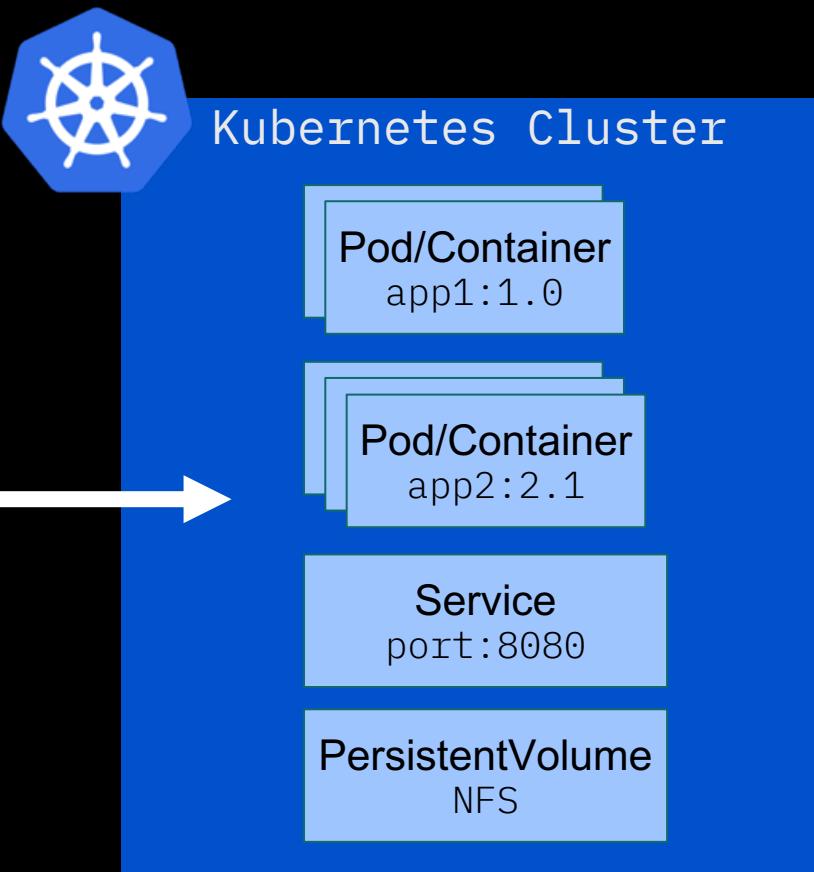


The Desired State

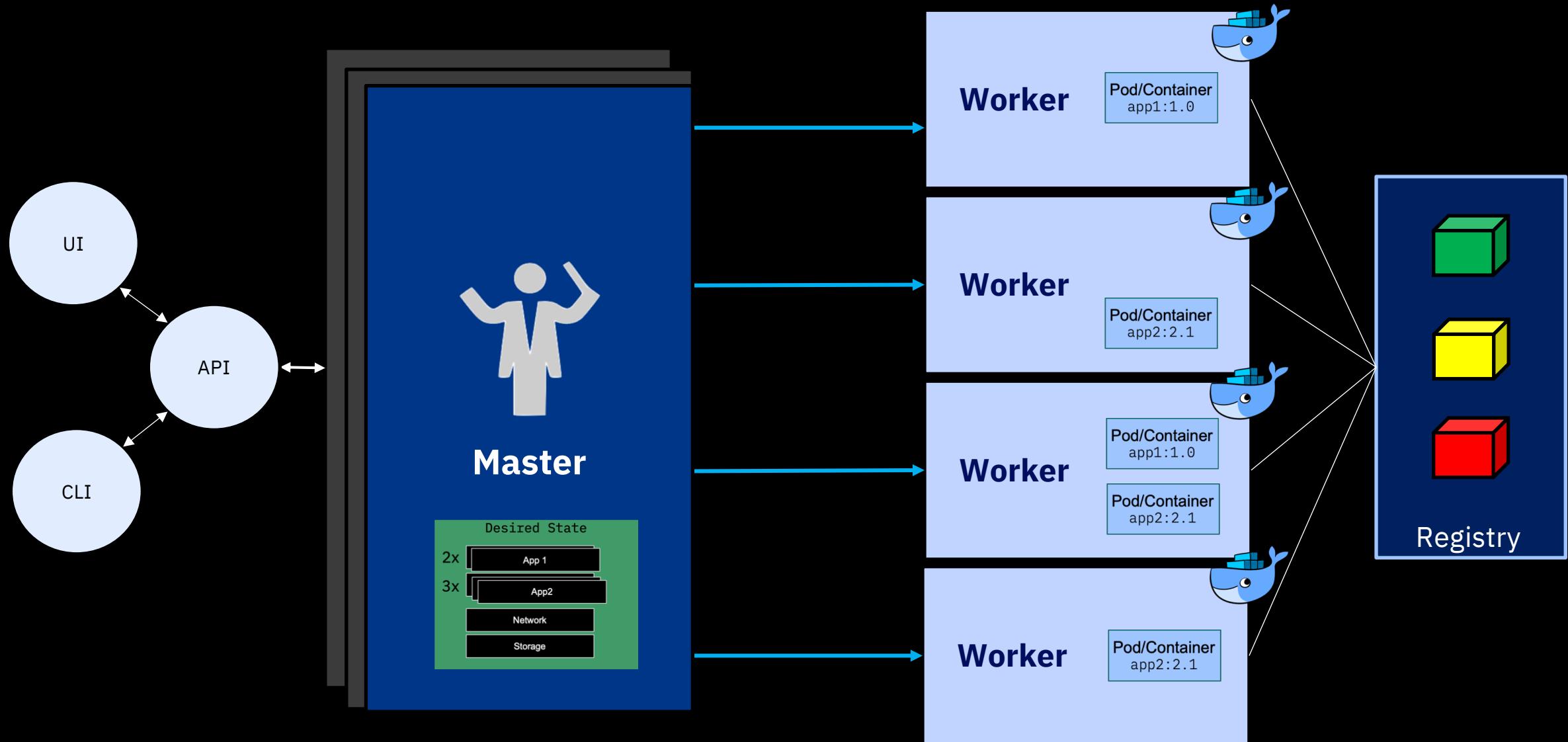
Kubernetes ensures that all the containers running across the cluster are in the desired state at any moment.



```
kind: Deployment
apiVersion: extensions/v1beta1
metadata:
  name: mcmk-ibm-mcmk-prod-klusterlet
  labels:
    app: ibm-mcmk-prod
    chart: ibm-mcmk-prod-3.1.2
    component: "klusterlet"
    release: mcmk
    heritage: Tiller
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ibm-mcmk-prod
      component: klusterlet
      release: mcmk
  template:
    metadata:
      labels:
        app: ibm-mcmk-prod
        component: "klusterlet"
        release: mcmk
        heritage: Tiller
        chart: ibm-mcmk-prod-3.1.2
      annotations:
        productName: "IBM Multi-cloud Manager - Klusterlet"
        productID: "354b8990aab44c9988a0edfd101b128"
        productVersion: "3.1.2"
    spec:
```



Kubernetes Management Architecture





What is Kubernetes?

Container orchestrator

- Runs and manages containers
- Unified API for deploying web applications, batch jobs, and databases
- Maintains and tracks the global view of the cluster
- Supports multiple cloud and bare-metal environments

Manage applications, not machines

- Rolling updates, canary deploys, and blue-green deployments

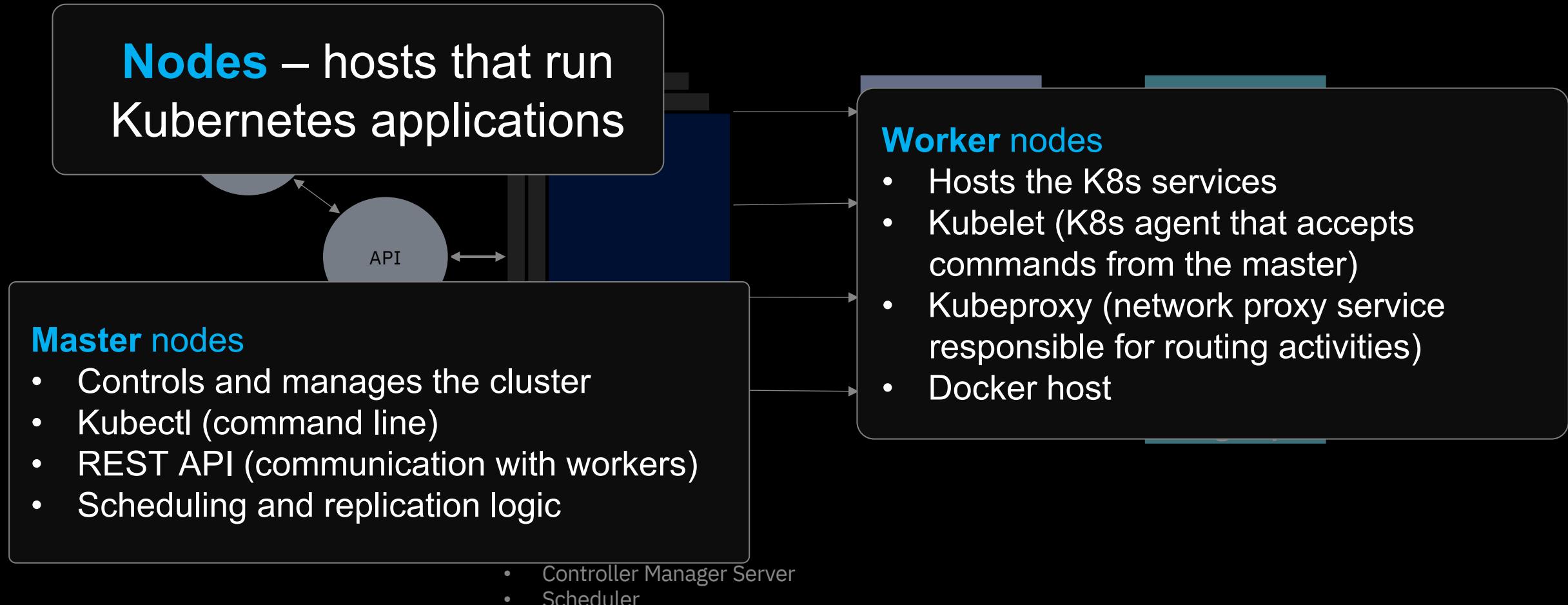
Designed for extensibility

- Rich ecosystem of plug-ins for scheduling, storage, networking

Open source project managed by the Linux Foundation

- Inspired and informed by Google's experiences and internal systems
- 100% open source, written in Go

Kubernetes Management Architecture

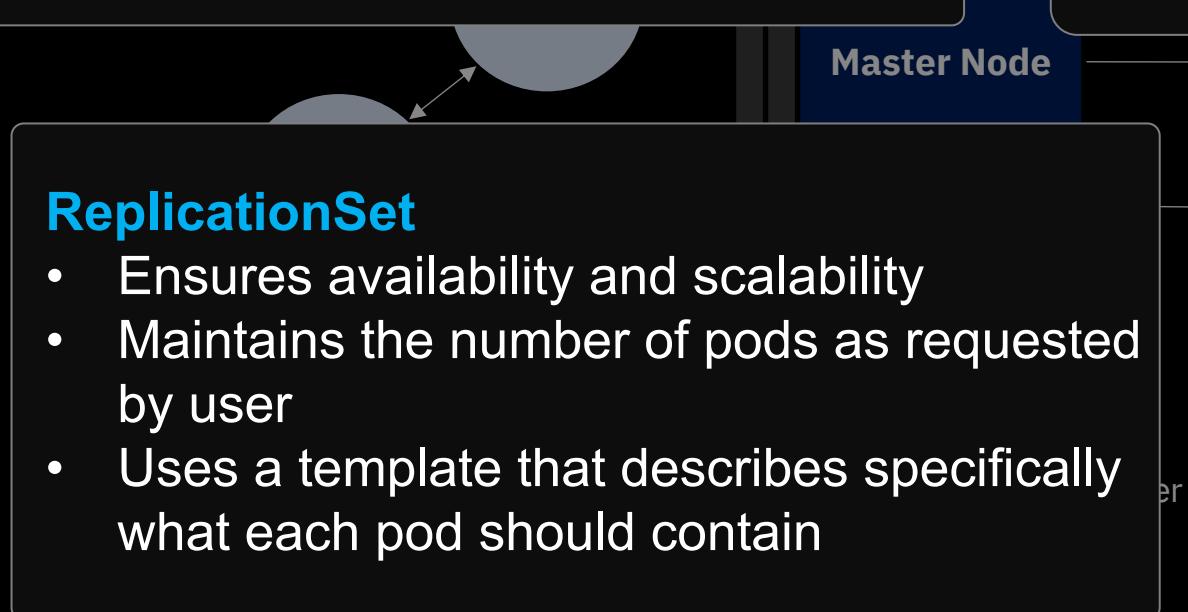


Kubernetes Management Architecture



Pods

- Smallest deployment unit in K8s
- Collection of containers that run on a worker node
- Each has its own IP
- Pod shares a PID namespace, network, and hostname



Deployment

- Smallest deployment unit in K8s
- Collection of containers that run on a worker node
- Each has its own IP
- Pod shares a PID namespace, network, and hostname

Service

- Collections of pods exposed as an endpoint
- A service provides a way to refer to a set of Pods (selected by labels) with a single static IP address

Kubernetes Management Architecture



ConfigMap

- Configuration values to be used by containers in a pod
- Stores configuration outside of the container image, making containers more reusable

Secrets

- Sensitive info that containers need to consume
- Encrypted in special volumes mounted automatically

- Etcd
- API Server
- Controller Manager Server
- Scheduler

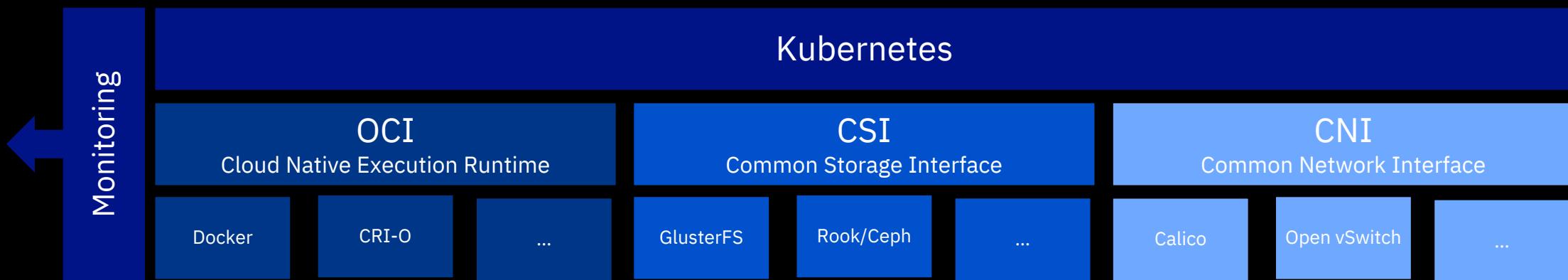
Worker Node 1

Labels

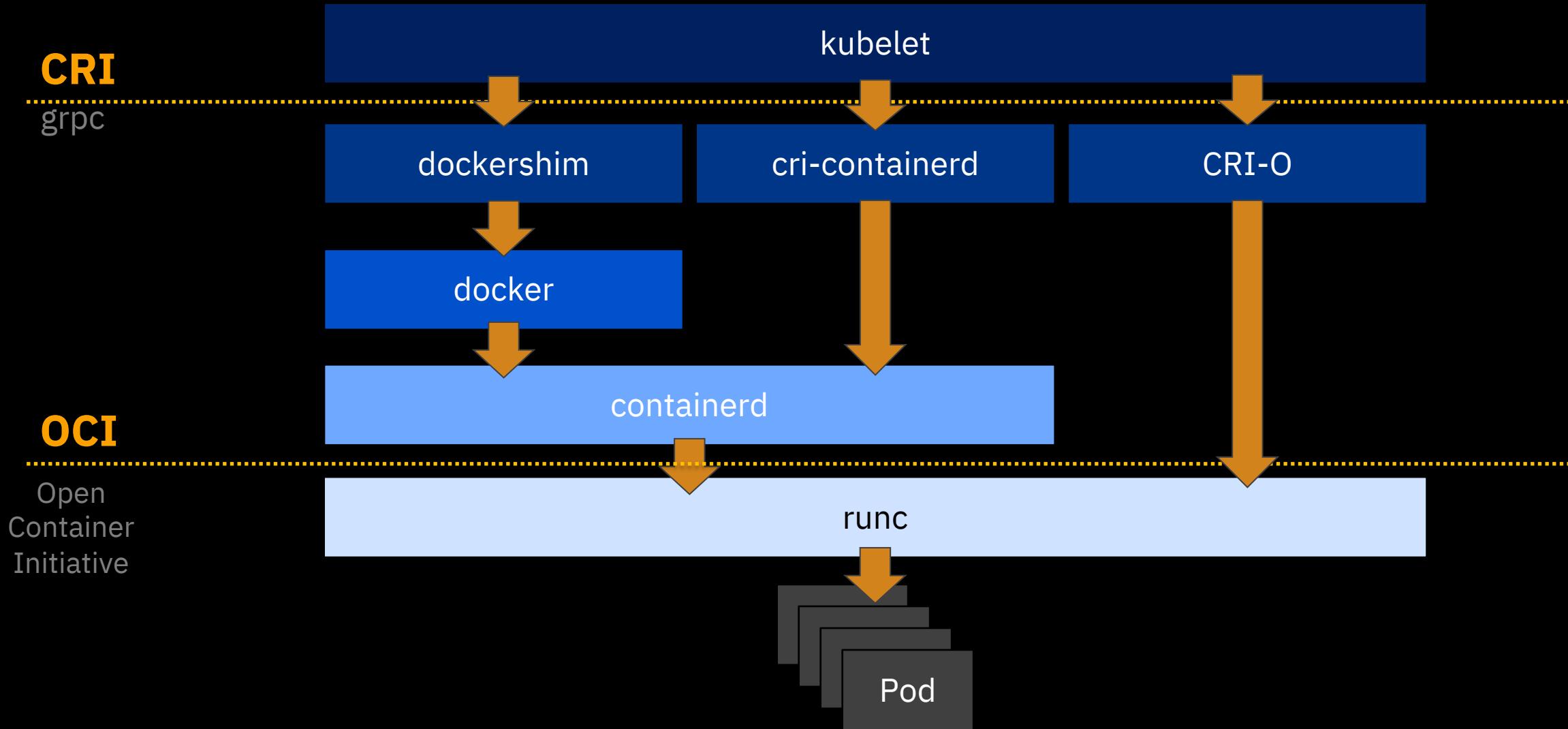
- Metadata assigned to K8s resources
- Key-value pairs for identification
- Critical to K8s as it relies on querying the cluster for resources that have certain labels

Registry

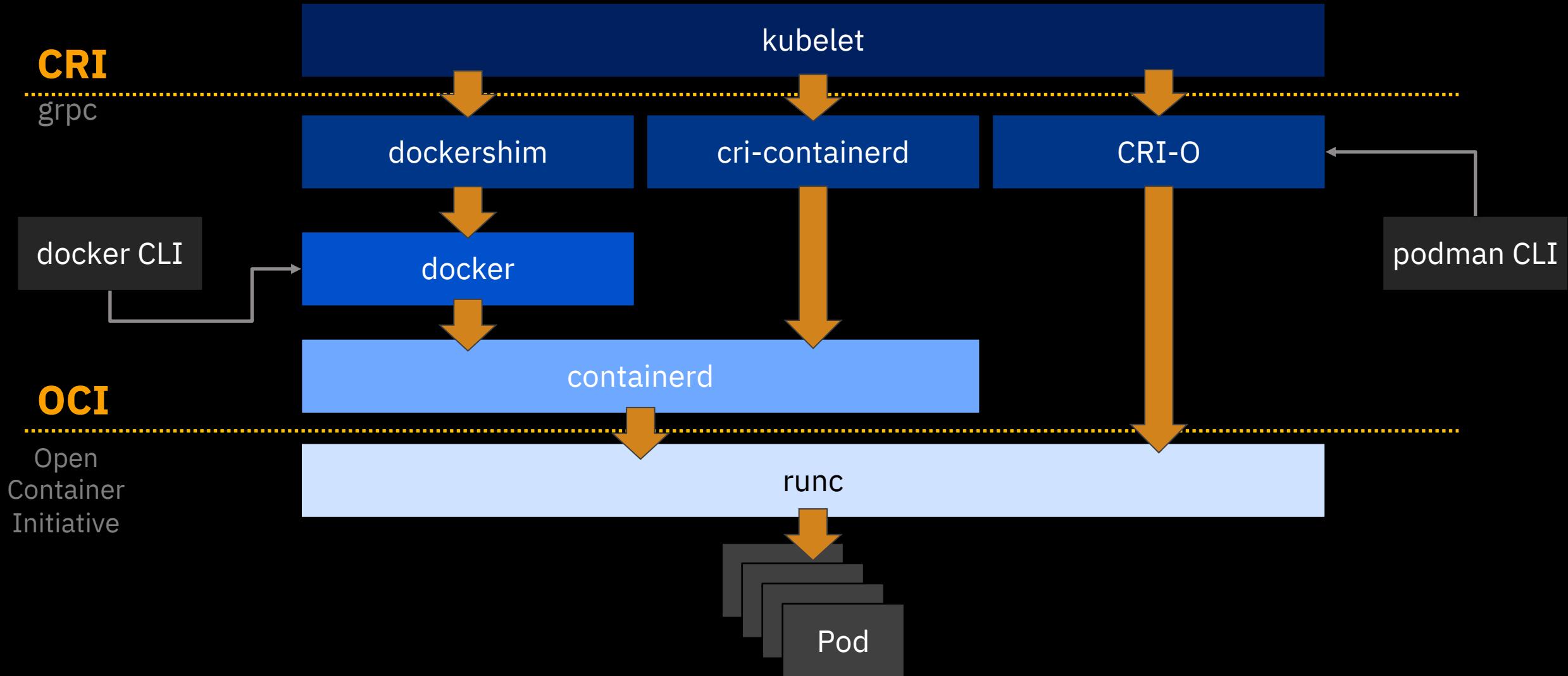
Kubernetes Cluster Architecture



Kubernetes – Common Runtime Interface



Kubernetes – Common Runtime Interface



Kubernetes Cluster Architecture

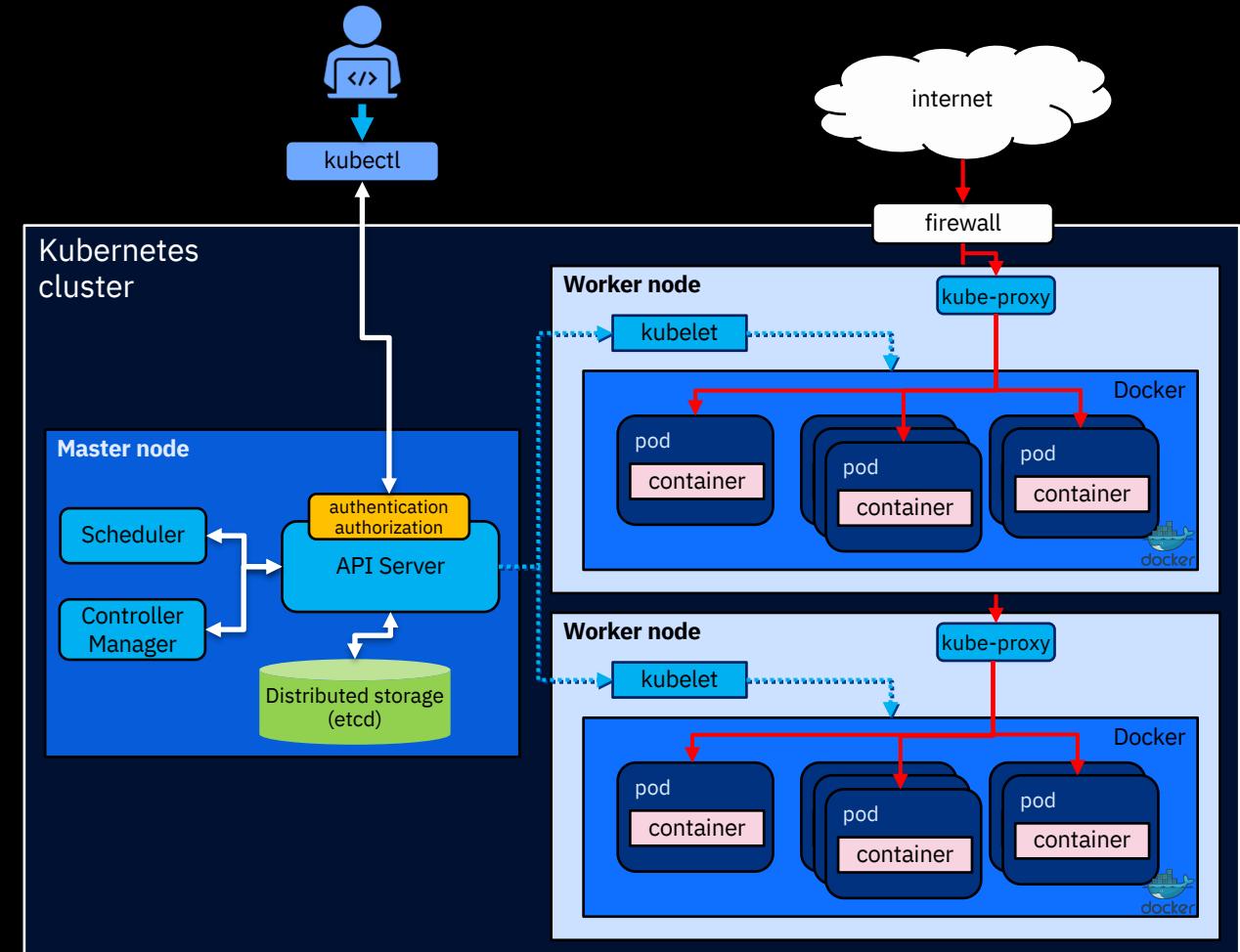


Master node

- Node that manages the cluster
- Scheduling, replication & control
- Multiple nodes for HA

Worker nodes

- Node where pods are run
- Docker engine
- kubelet agent accepts & executes commands from the master to manage pods
- kube-proxy – routes inbound or ingress traffic



kubectl – talking to the Cluster

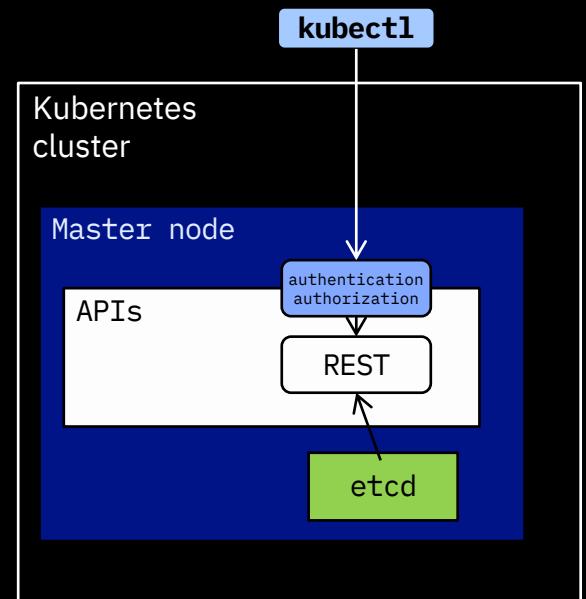


kubectl is a command line interface for running commands against Kubernetes clusters (read state, create objects, ...).

kubectl looks for a file named config in the \$HOME/.kube directory.

Kubernetes uses etcd as a key-value database store.
It stores the configuration of the Kubernetes cluster in etcd.
It also stores the *actual* state of the system and the *desired* state of the system in etcd.

Anything you might read from a `kubectl get xyz` command is stored in etcd.
Any change you make via `kubectl create` will cause an entry in etcd to be updated.



kubectl – talking to the Cluster



kubectl is a command line interface for running commands against Kubernetes clusters.

kubectl looks for a file named config in the \$HOME/.kube directory.

`kubectl [command] [TYPE] [NAME]`

`kubectl create -f example.yaml`

Create objects in yaml file

`kubectl apply -f example.yaml`

Modify objects in yaml file

`kubectl delete -f example.yaml`

Delete objects in yaml file

`kubectl get pods`

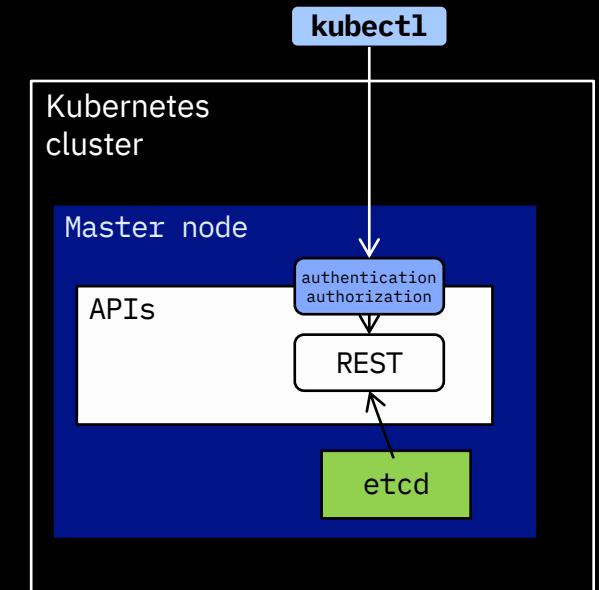
List Pods in Namespace

`kubectl describe nodes <node-name>`

Details about K8s object

`kubectl logs <pod-name>`

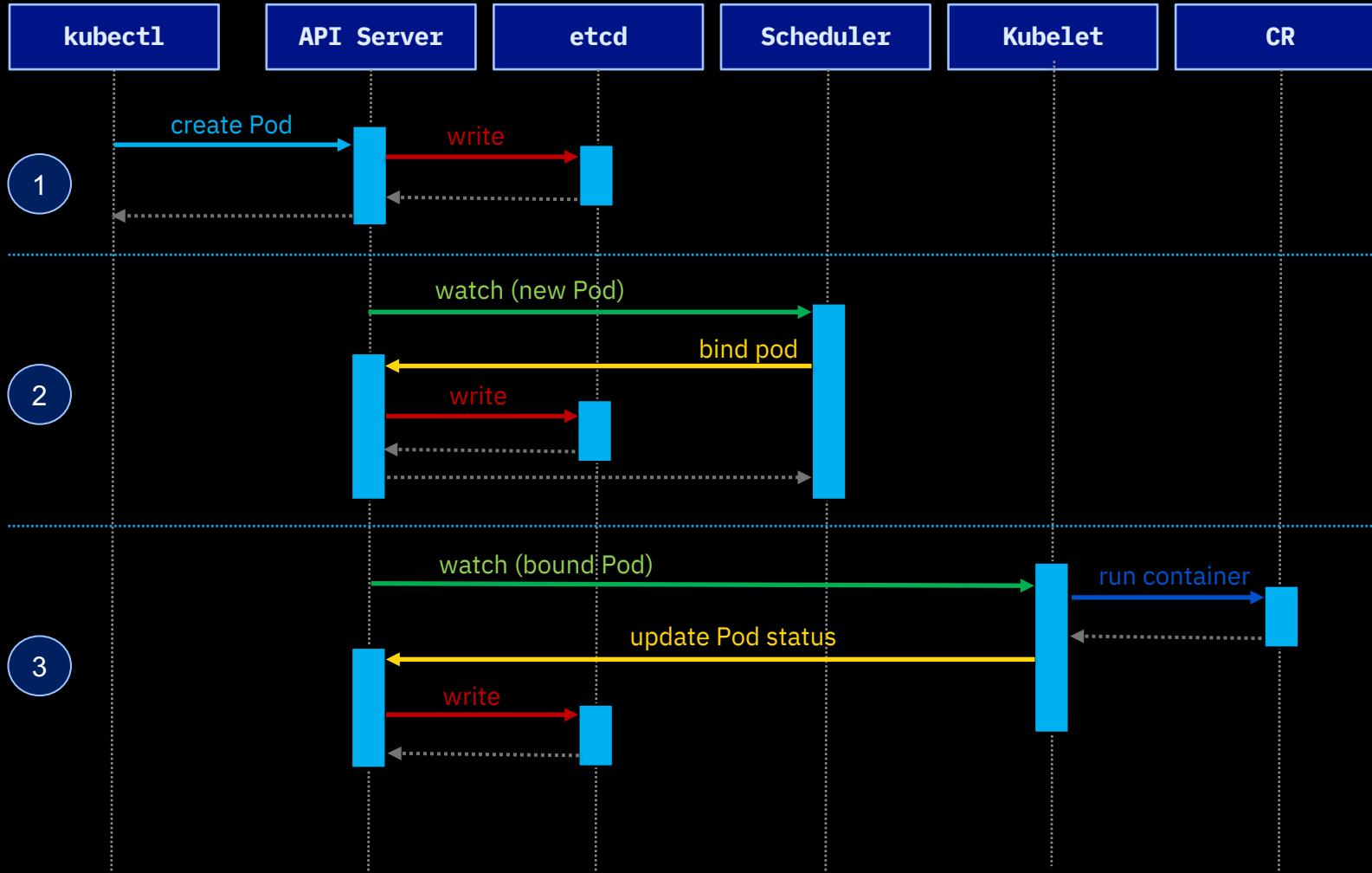
Get the logs for a Pod



kubectl – talking to the Cluster



What does this actually do: `kubectl run nginx --image= nginx:1.7.9`



1. Create a Pod via the API Server and the API server writes it to etcd
2. The scheduler notices an “unbound” Pod and decides which node to run that Pod on. It writes that binding back to the API Server.
3. The Kubelet notices a change in the set of Pods that are bound to its node. It, in turn, runs the container via the container runtime (i.e. Docker).

The Kubelet monitors the status of the Pod via the container runtime. As things change, the Kubelet will reflect the current status back to the API Server

Pod



- A group of one or more containers is called a pod.
- Containers in a pod are deployed together, and are started, stopped, and replicated as a group.
- **When designing pods ask, “Will these containers work correctly if they land on different machines?”**

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
    - name: nginx
      image: nginx:1.7.9
      ports:
        - containerPort: 80
```



10.0.0.1

10.0.0.2

Creating a Pod



```
kubectl run nginx --image=nginx:1.7.9  
or  
kubectl create -f example.yaml
```

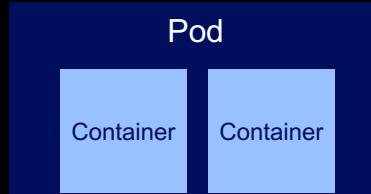
```
kubectl get pods  
NAME          READY  STATUS    RESTARTS  AGE  
nginx-5bd87f76c-vxc79  1/1    Running   0          29s
```

example.yaml

```
apiVersion: v1  
kind: Pod  
metadata:  
  name: nginx  
spec:  
  containers:  
  - name: nginx  
    image: nginx:1.7.9  
    ports:  
    - containerPort: 80
```



10.0.0.1

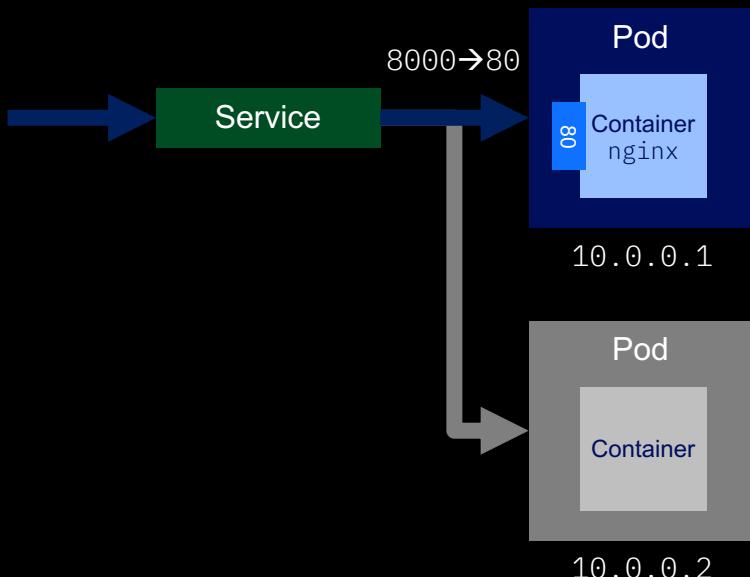


10.0.0.2

Service



- A service provides a way to refer to a set of Pods (selected by labels) with a **single static IP address**.
- Also provide load balancing

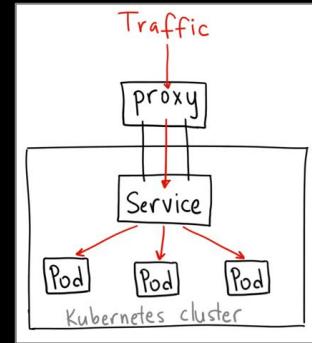


```
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  ports:
    - port: 8000
      targetPort: 80
      protocol: TCP
  selector:
    app: nginx
```

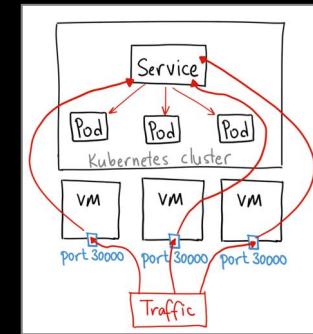
Service



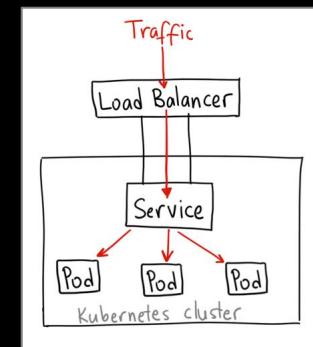
ClusterIP: This type exposes the service on the cluster internal IP. This means that the service is only reachable from within the cluster.



NodePort: This type exposes the service on each Node's static IP address.



LoadBalancer: This service type exposes a service using the cloud provider load balancer.

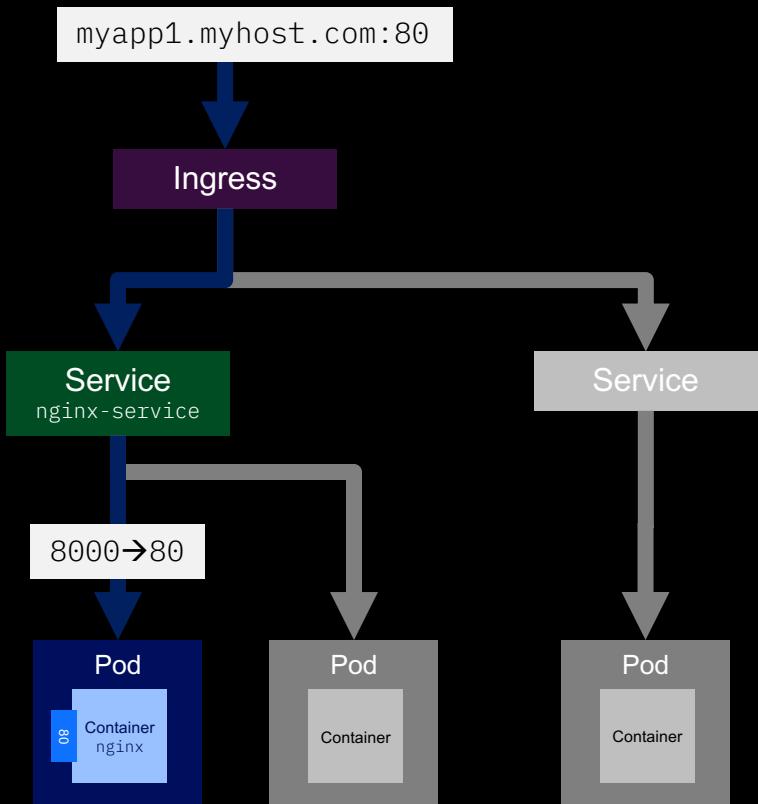


Ingress



An ingress can be configured to give services externally-reachable URLs, load balance traffic, terminate SSL, and offer name based virtual hosting. An ingress controller is responsible for fulfilling the ingress, usually with a loadbalancer.

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: nginx-ingress
  annotations:
    ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - host: myapp1.myhost.com
    http:
      paths:
      - backend:
          serviceName: nginx-service
          servicePort: 80
```

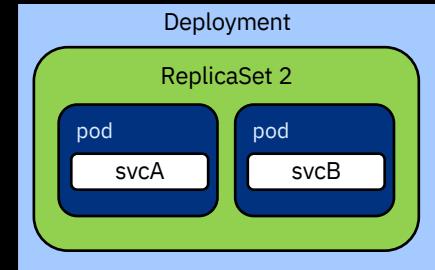


Deployments & ReplicaSets



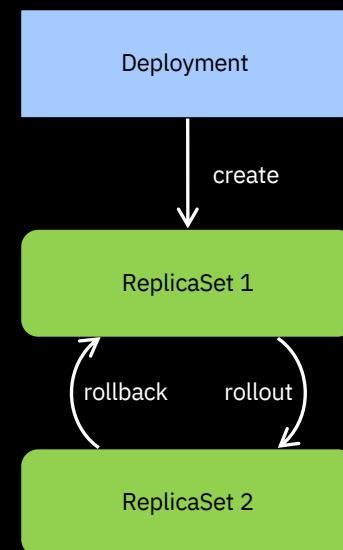
Deployment

- A set of **pods to be deployed together**, such as an application
- **Declarative**: Revising a Deployment **creates a ReplicaSet** describing the desired state
- **Rollout**: Deployment controller changes the actual state to the desired state at a controlled rate
- **Rollback**: Each Deployment revision can be rolled back
- **Scale** and autoscale: A Deployment can be scaled



ReplicaSet

- Cluster-wide pod manager that **ensures the proper number of pods are running** at all times.
- A set of pod templates that describe a set of pod replicas
- Uses a template that describes specifically what each pod should contain
- Ensures that a specified number of pod replicas are running at any given time



Deployment



- A Deployment object defines a Pod creation template and **desired replica count**.
- Create or delete Pods as needed to meet the replica count.
- Manage safely **rolling out changes** to your running Pods.

```
kubectl create -f example.yaml
```

example.yaml

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 2
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
      ports:
      - containerPort: 80
```

Naming



Name

- Each resource object by type has a unique name

Namespace

- **Resource isolation:** Each namespace is a virtual cluster within the physical cluster
 - Resource objects are scoped within namespaces
 - Low-level resources are not in namespaces: nodes, persistent volumes, and namespaces themselves
 - Names of resources need to be unique within a namespace, but not across namespaces
- **Resource quotas:** Namespaces can divide cluster resources
- Initial namespaces
 - **default** – The default namespace for objects with no other namespace
 - **kube-system** – The namespace for objects created by the Kubernetes system

Resource Quota

- Limits resource consumption per namespace
- Limit can be number of resource objects by type (pods, services, etc.)
- Limit can be total amount of compute resources (CPU, memory, etc.)
- Overcommit is allowed; contention is handled on a first-come, first-served basis

StatefulSet, DaemonSet, Job...



StatefulSet

- Intended to be used with stateful applications and distributed systems.
 - Pods are created sequentially
 - Ordinal index and stable network identity

DaemonSet

- Ensures that all (or some) nodes run a copy of a pod
 - running a cluster storage daemon
 - running a logs collection daemon
 - running a node monitoring daemon

Job

- Creates one or more pods and ensures that a specified number of them successfully terminate

Cron Job

- Manage time based jobs, once at a specified in time, or repeatedly at a specified time point

Configuring Resources and Containers



ConfigMap

- Configuration values to be used by containers in a pod
- Stores configuration outside of the container image, making containers more reusable

Secret

- Sensitive info that containers need to read or consume
- Encrypted in special volumes mounted automatically

Configuring Resources and Containers



```
apiVersion: extensions/v1beta1
kind: ConfigMap
apiVersion: v1
metadata:
  name: example-config
data:
  allowed: '"true"'
  enemies: aliens
  lives: "3"
```

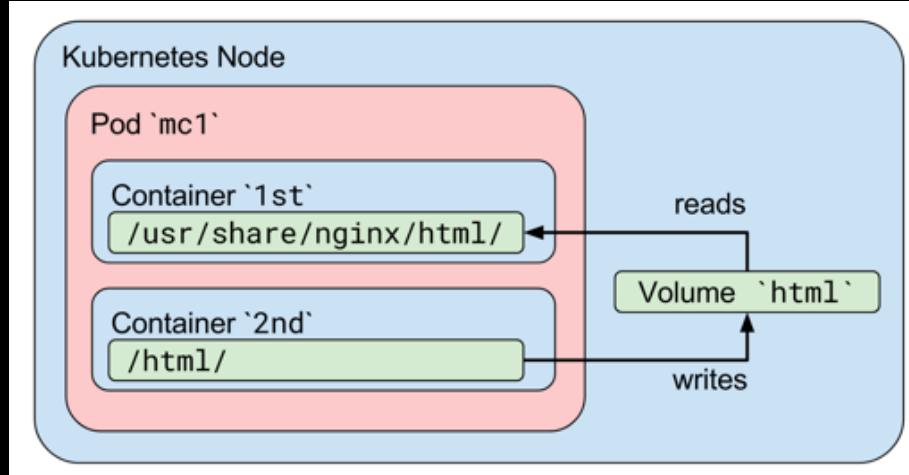
```
kind: Deployment
spec:
  containers:
    - name: test-container
      image: xxx
      ...
  env:
    - name: SPECIAL_LEVEL_KEY
      valueFrom:
        configMapKeyRef:
          name: example-config
          key: enemies
```

Can be used as normal environment variable

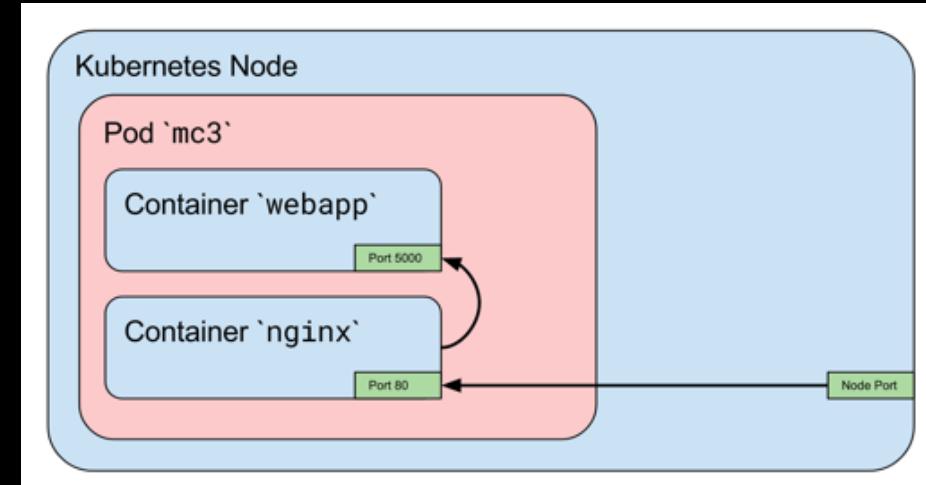
Multi-Container Pod Design Patterns



Shared Volumes

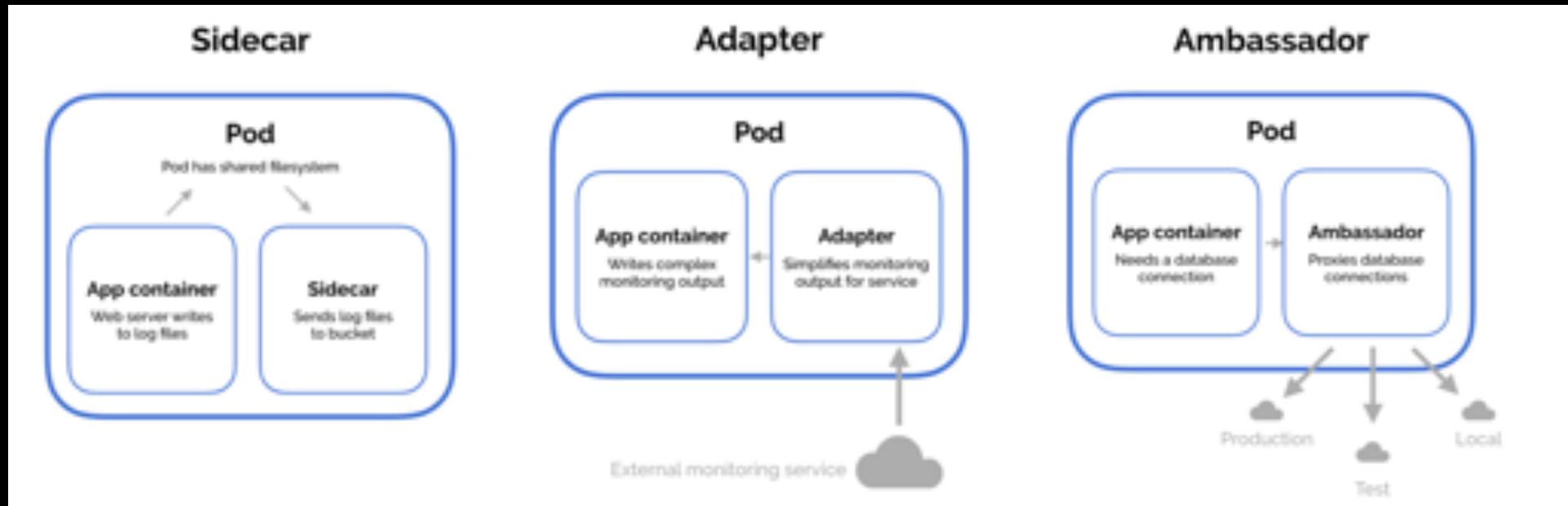


No Network isolation



In what order containers are being started in a Pod?

Multi-Container Pod Design Patterns



Sidecar pattern

The sidecar pattern consists of a main application—i.e. your web application—plus a helper container with a **responsibility that is essential to your application**, but is not necessarily part of the application itself.

Adapter pattern

The adapter pattern is used to standardize and normalize application output or **monitoring data** for aggregation.

Ambassador pattern

The ambassador pattern is a useful way to connect containers with the outside world (**proxy**).

Resources



Linux Foundation – Introduction to Kubernetes

- <https://courses.edx.org/courses/course-v1:LinuxFoundationX+LFS158x+2T2017/course/>

Kubernetes tutorial

- <https://kubernetes.io/docs/tutorials/kubernetes-basics/>

Introduction to container orchestration

- <https://www.exoscale.ch/syslog/2016/07/26/container-orch/>

TNS Research: The Present State of Container Orchestration

- <https://thenewstack.io/tns-research-present-state-container-orchestration/>

Large-scale cluster management at Google with Borg

- <https://research.google.com/pubs/pub43438.html>

Benefits of using a Kubernetes Based Platform

Speed

- Fast boot-time
- Easy scalability
- Rapid deployment

Portability

- Between different environments
- Between private and public cloud

Efficiency

- Better usage of computer resources

Automation

- Next level standardization and automation

13x

More software releases

Eliminate

“works on my machine” syndrome

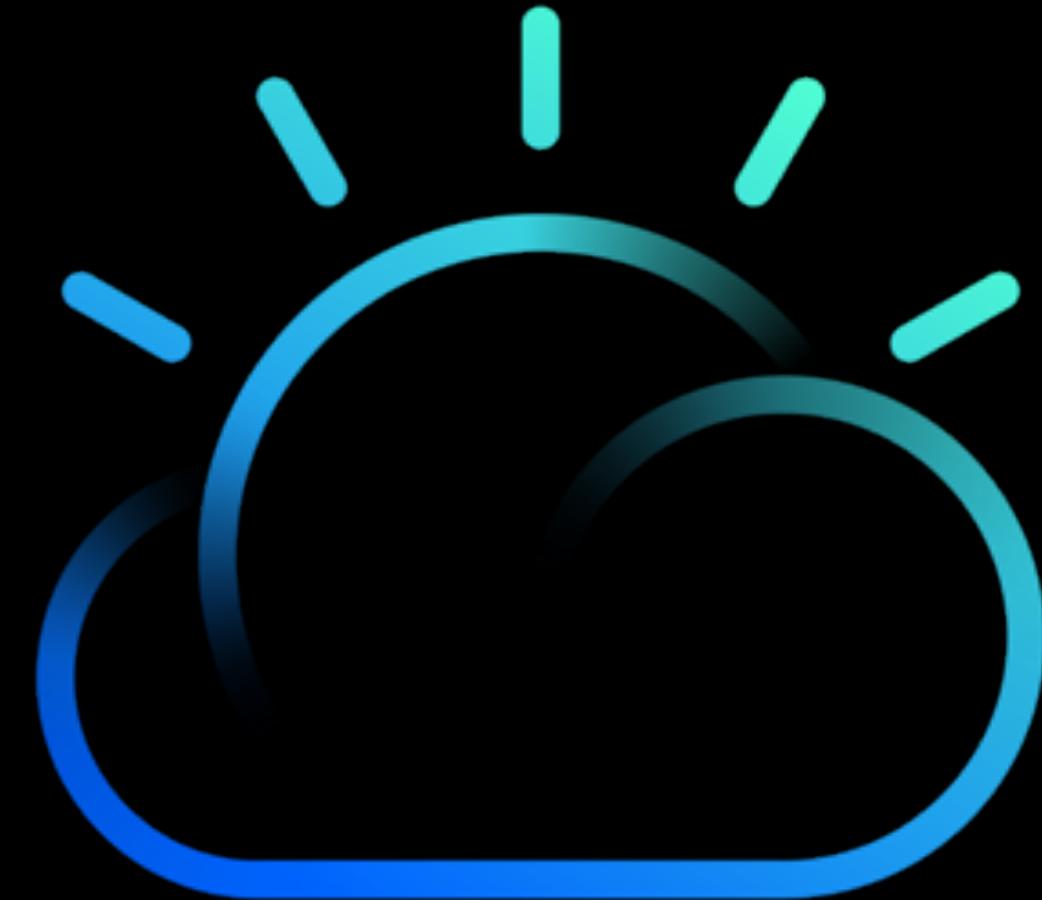
~47%

Reduction of VMs,
OS licensing and
server cost

Reduce

operation cost

QUESTIONS?



The Journey to Cloud
Let's get real



IBM Cloud

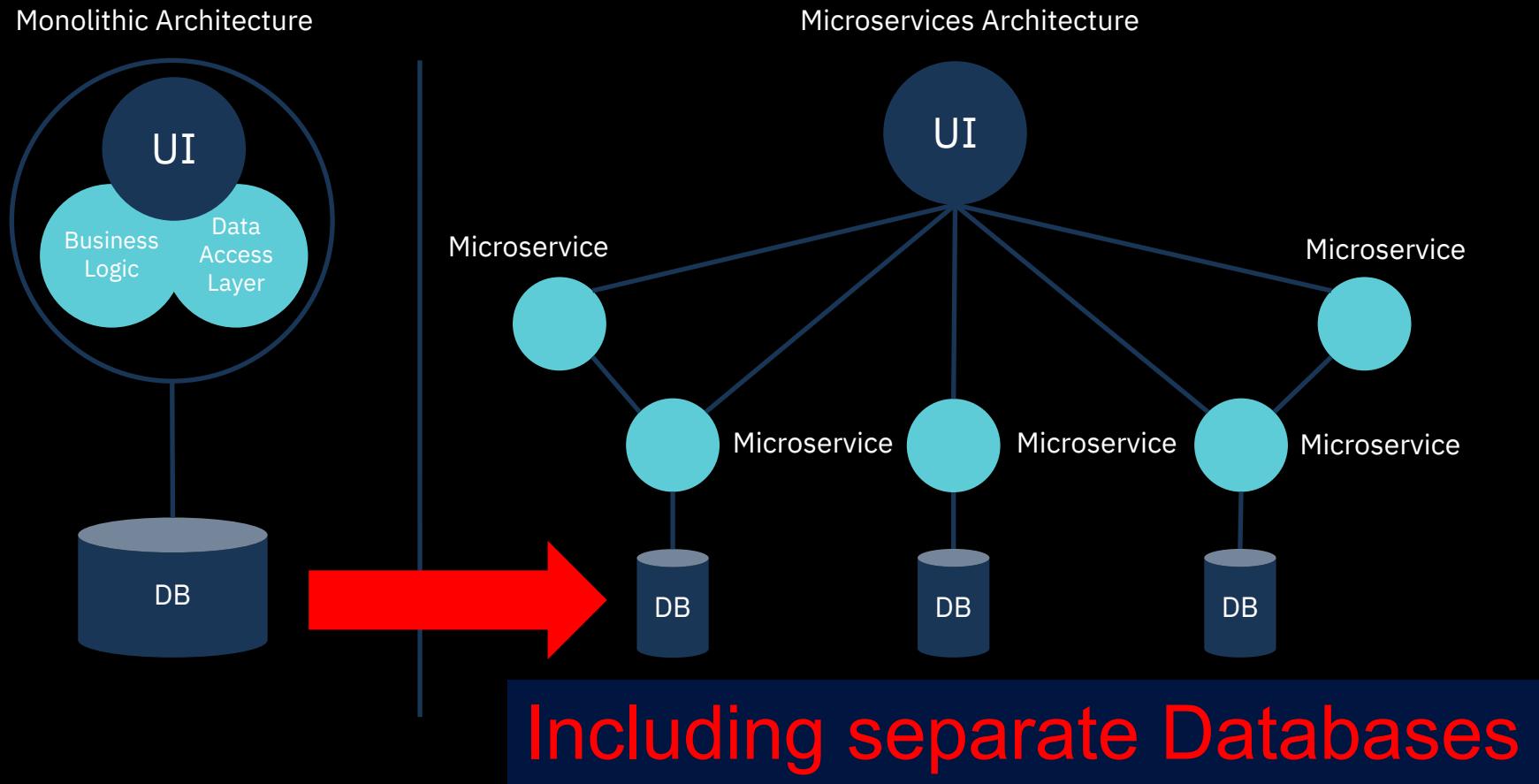
Microservices – breaking down the monolith beast

How to identify candidates

- Is there anything that is **scaling differently** than the rest of the system?
- Is there anything that feels “**tacked-on**”?
- Is there anything **changing much faster** than the rest of the system?
- Is there anything requiring more **frequent deployments** than the rest of the system?
- Is there a part of the system that a small team, **operates independently**?
- Is there a **subset of tables** in your datastore that isn’t connected to the rest of the system?

Microservices – Database refactoring

Simplistically, microservices architecture is about breaking down large silo applications into more manageable, fully decoupled pieces



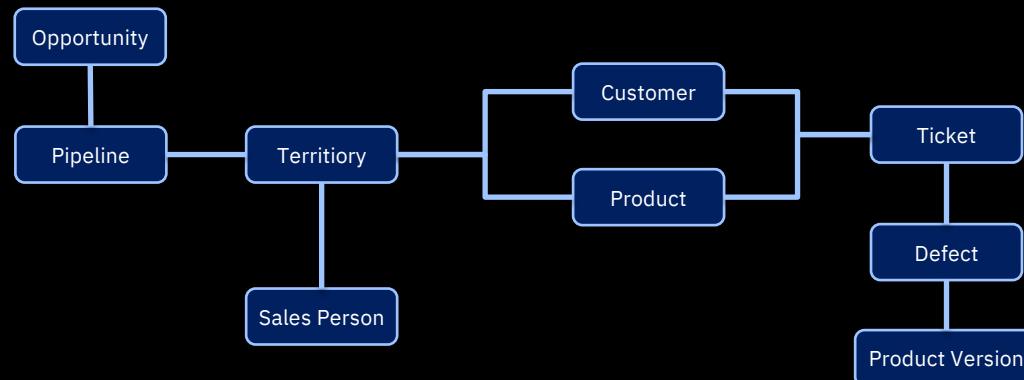
Containerizing your WAR file
doesn't mean you're doing
microservices.

- What is the domain? What is reality?
- Where are the transactional boundaries?
- How should microservices communicate across boundaries?
- What if we just turn the database inside out?

Microservices – Bounded Context

Bounded Context is a central pattern in Domain-Driven Design

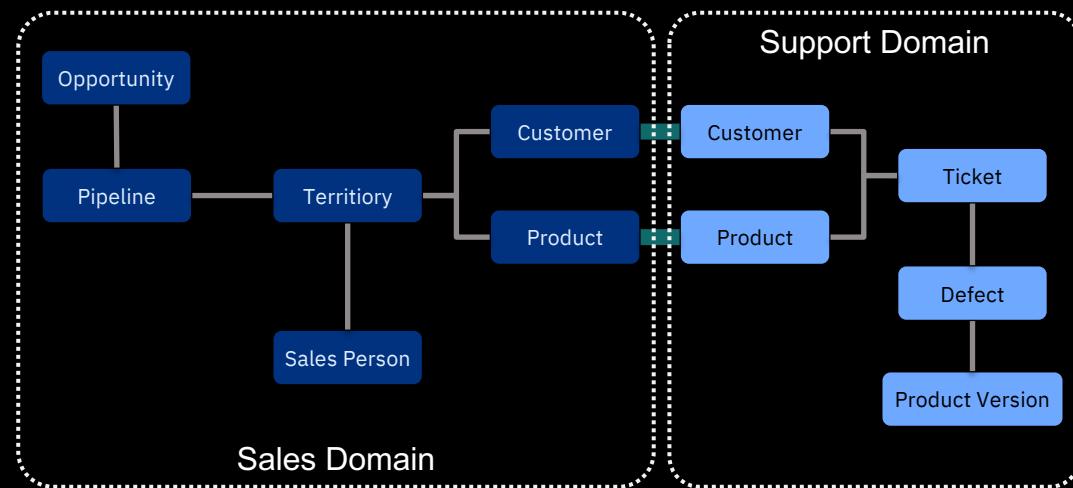
- DDD deals with large models by dividing them into different Bounded Contexts



Microservices – Bounded Context

Bounded Context is a central pattern in Domain-Driven Design

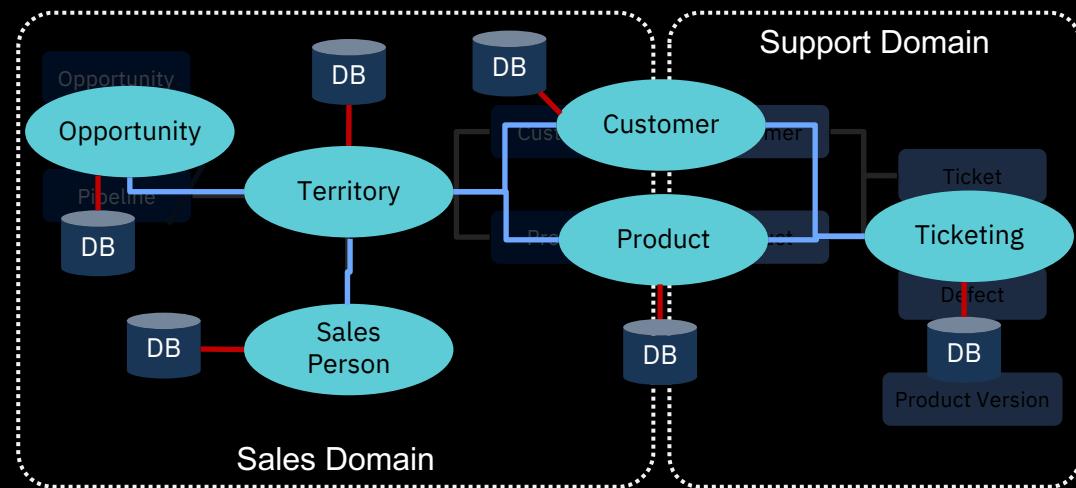
- DDD deals with large models by dividing them into different Bounded Contexts
- Helps to build and refine a model that represents our domains, contained within a boundary that defines our context.



Microservices – Bounded Context

Bounded Context is a central pattern in Domain-Driven Design

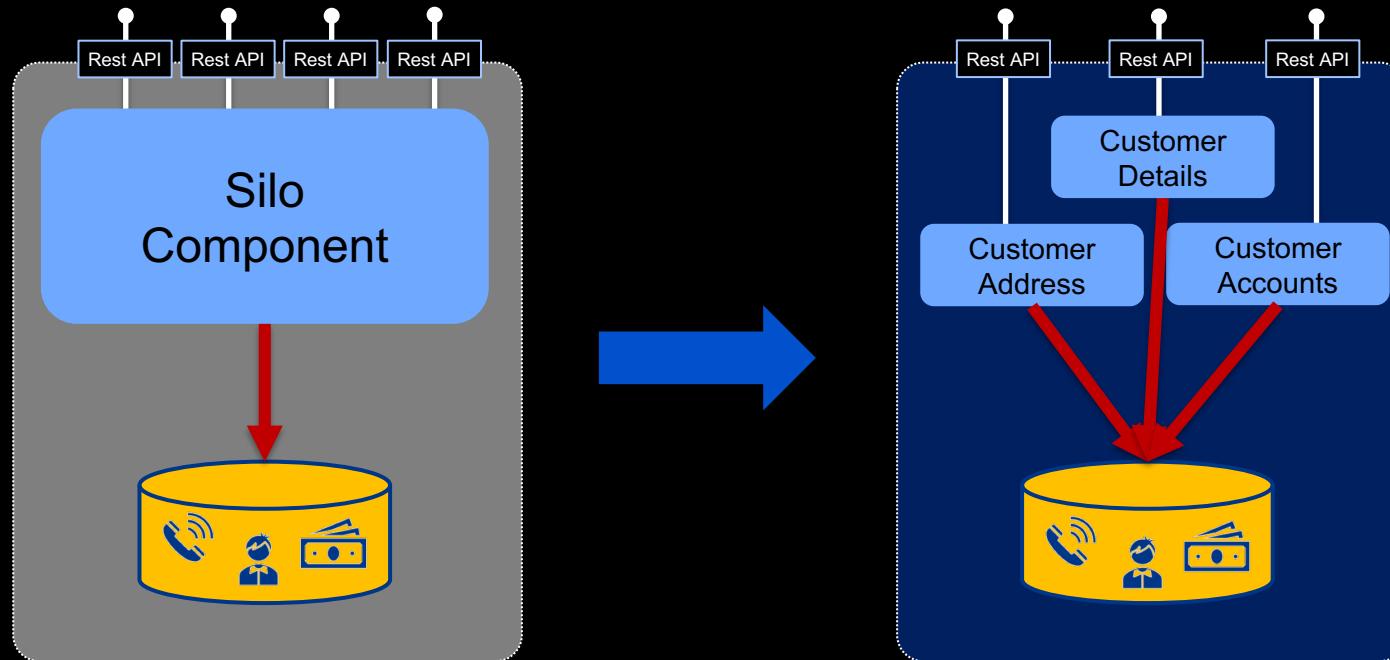
- DDD deals with large models by dividing them into different Bounded Contexts
- Helps to build and refine a model that represents our domains, contained within a boundary that defines our context.
- These boundaries end up being our microservices



Microservices – Database refactoring

Data Isolation

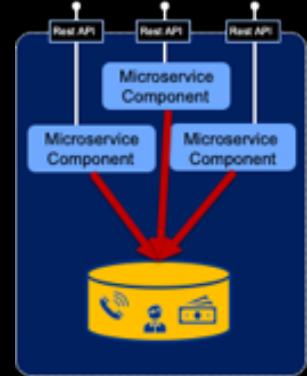
Each microservice must keep its own data. One typical error that happens in the beginning of a microservice transformation is the use of a centralized database, the same way it was used in the monolithic application. This creates a single point of failure and dependency between the microservices.



Microservices – monolithic database

Challenges

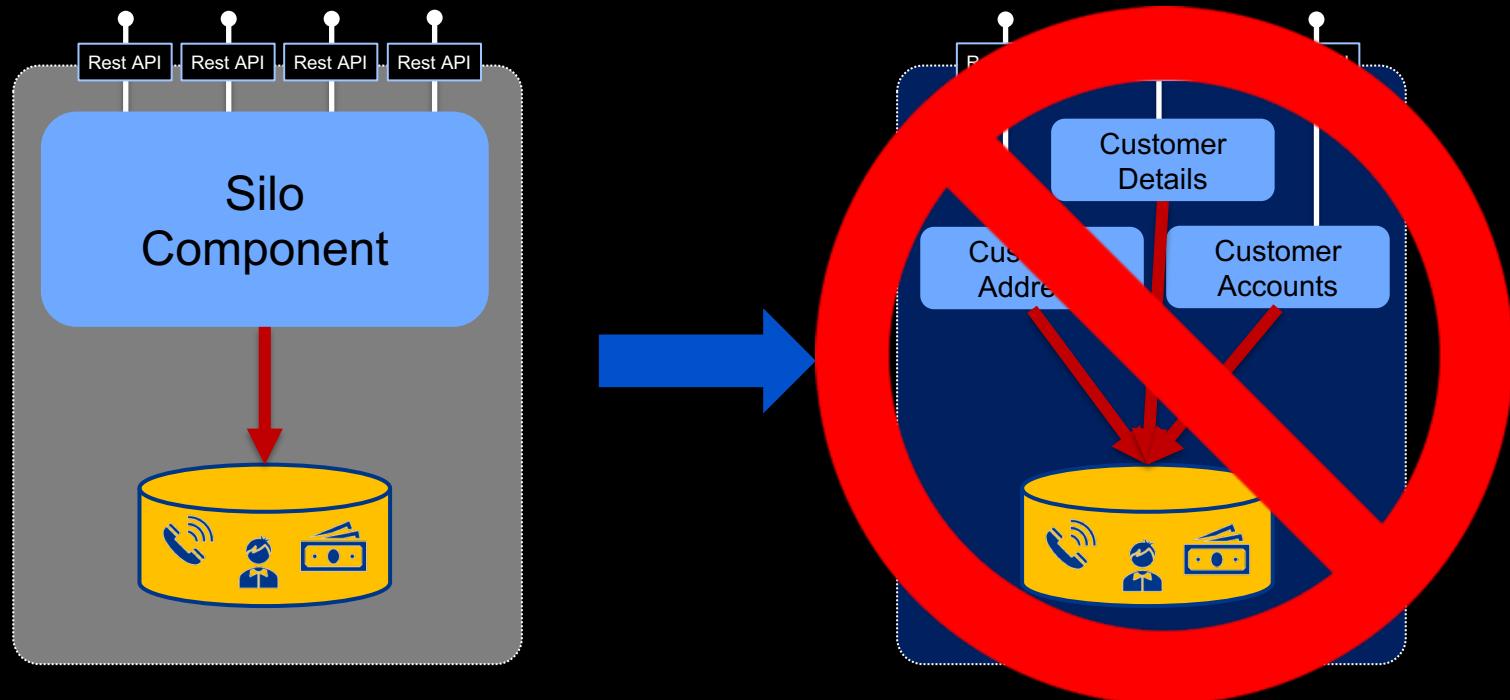
- Inability to deploy your service changes independently through **tight coupling**.
- Schema **changes need to be coordinated** amongst all the services
- **Difficult to scale** individual services
- Difficult to improve application performance (DB/Table size)
- All your services have to use a **relational database** even when a no-SQL datastore would bring benefits



Microservices – Database refactoring

Data Isolation

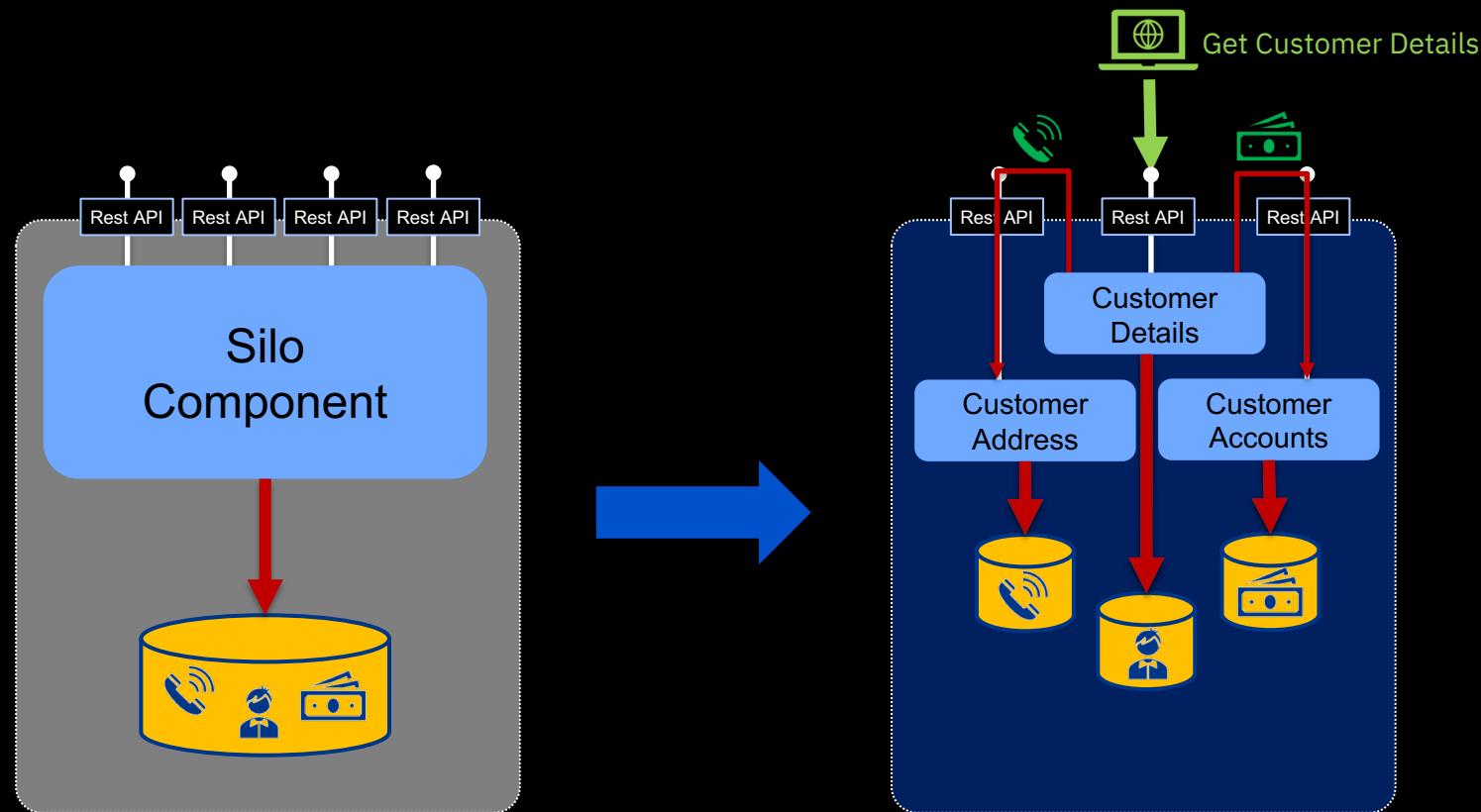
Each microservice must keep its own data. One typical error that happens in the beginning of a microservice transformation is the use of a centralized database, the same way it was used in the monolithic application. This creates a single point of failure and dependency between the microservices.



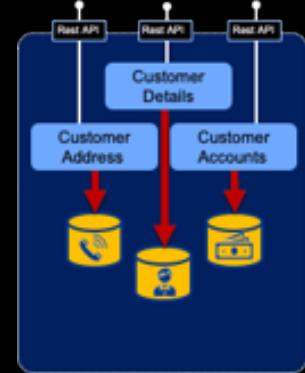
Microservices – Database refactoring

Data Isolation

Each microservice must keep its own data. One typical error that happens in the beginning of a microservice transformation is the use of a centralized database, the same way it was used in the monolithic application. This creates a single point of failure and dependency between the microservices.



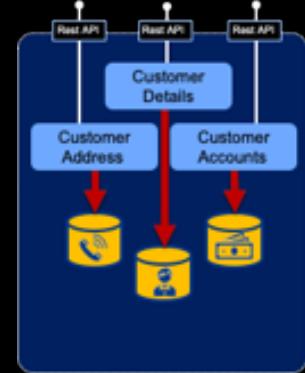
Microservices – per-service database



Benefits

- Avoid unexpected data modification – the API enforces consistency
- Make changes to our system without blocking progress of other parts of the system
- Store the data in a project-owned databases, using the appropriate technology
- Make changes to the schema/databases at our leisure
- Become much more scalable, fault tolerant, and flexible

Microservices – per-service database



Drawbacks

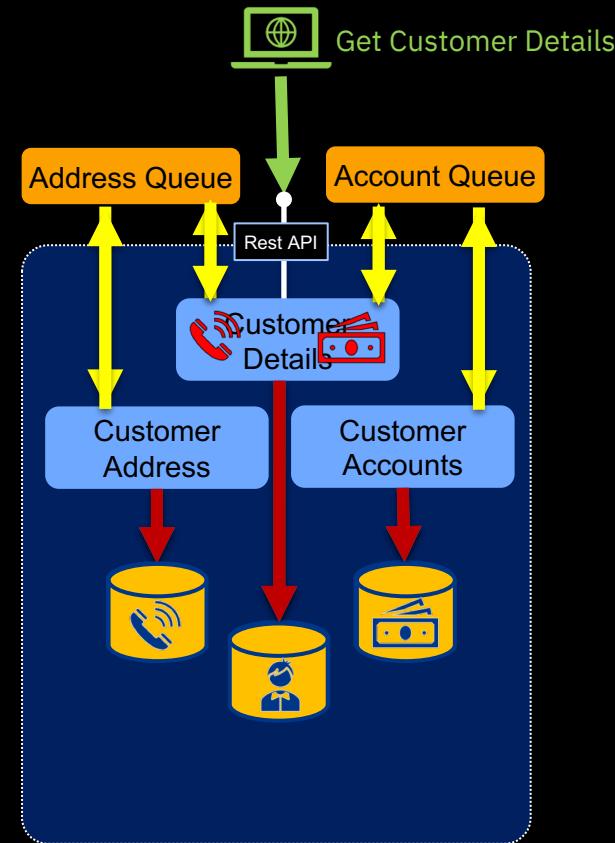
- Costly to refactor
- Needs robust transaction handling by the service (cost!)
- More difficult to debug
- More difficult to operationalize
- More difficult to test (needs well designed API tests)

Event Driven Architecture

Microservices – one step further

Event-Driven Architecture

- Common pattern to maintain data consistency across different services.
- Avoid waiting for ACID transactions to complete
- Makes your application more available and performant
- Provides loose coupling between services



Microservices Good or bad idea?

Microservices – reasons not to use them

Some thoughts

- If speed is your goal, microservices aren't the solution (MVP, doesn't have to scale)
- Not all applications are large enough to break down into microservices
- Applications that require tight integration between individual components and services (real-time processing, ...)
- A moderately large, moderately complex application being maintained by a relatively small development and operations team
- At what point is it in its lifecycle? Mission-critical?

Microservices – reasons not to use them

How to detect antipatterns

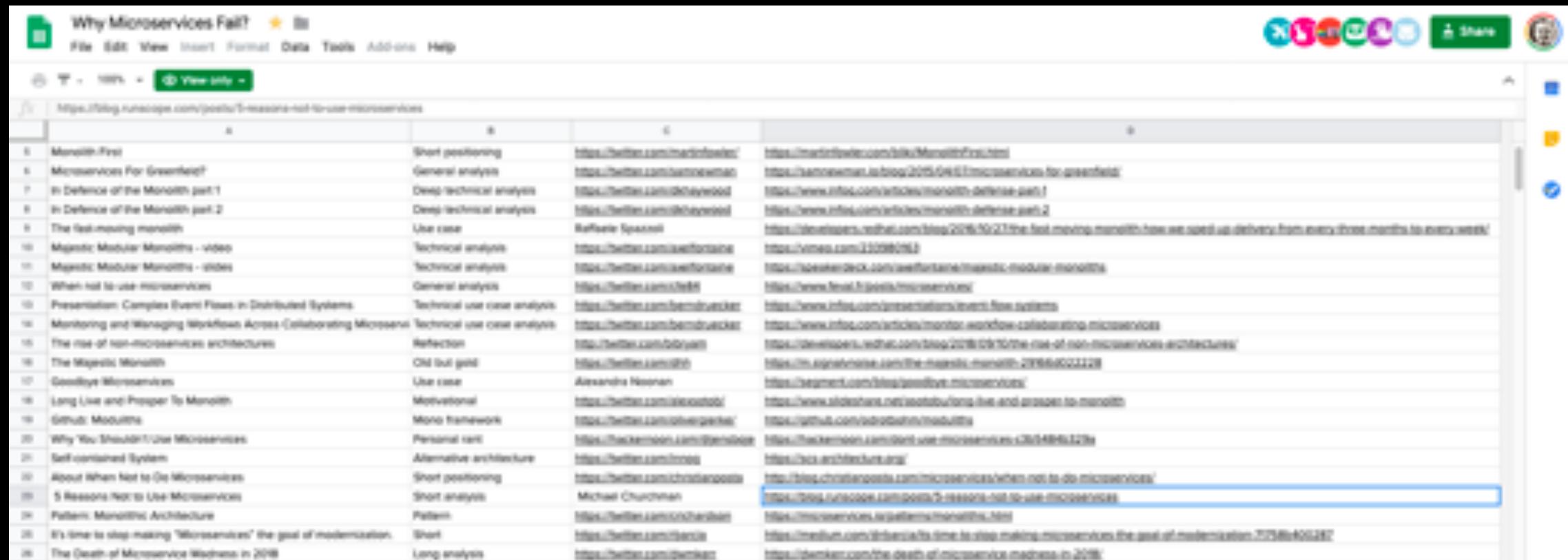
- A change to one microservice often requires **changes to other microservices**
- Many of your microservices **share a datastore**
- Deploying one microservice requires other microservices to be **deployed at the same time**
- Your microservices are **overly chatty**
- The same developers **work across a large number of microservices**
- Your microservices **share a lot of the same code or models**

Microservices – reasons not to use them

What works for **large and complex** applications
does not always work at a **smaller scale**

What makes sense for a **new application**
does not always make sense when maintaining or
updating **existing applications**

Microservices – reasons not to use them



The screenshot shows a Google Sheets document with a title bar 'Why Microservices Fail?' and a toolbar with various icons. The main content is a table with four columns: 'A' (Index), 'B' (Category), 'C' (Title), and 'D' (URL). The table contains 26 rows of data. Row 25 is highlighted with a blue background.

A	B	C	D
1	Monolith First	Short positioning	https://twitter.com/michaelfowler/
2	Microservices For Greenfield?	General analysis	https://twitter.com/tarceveman
3	In Defence of the Monolith part 1	Deep technical analysis	https://twitter.com/dchaywood
4	In Defence of the Monolith part 2	Deep technical analysis	https://twitter.com/dchaywood
5	The fast-moving monolith	User case	https://medium.com/@kyle_brown
6	Magnetic Modular Monoliths - video	Technical analysis	https://twitter.com/aventurinane
7	Magnetic Modular Monoliths - video	Technical analysis	https://twitter.com/aventurinane
8	When not to use microservices	General analysis	@steve
9	Presentation: Complex Event Flows in Distributed Systems	Technical use case analysis	@benjibuck
10	Monitoring and Managing Workflows Across Collaborating Microservices	Technical use case analysis	@benjibuck
11	The rise of non-microservices architectures	Reflection	@dbrumley
12	The Magnetic Monolith	Old school	@steve
13	Goodbye Microservices	User case	@alexanderheine
14	Long Live and Prosper To Monolith	Motivational	@alexisstotz
15	Github: Modular	Mono framework	@olivergarrett
16	Why You Shouldn't Use Microservices	Personal rant	https://michaelfowler.ca/why-you-shouldnt-use-microservices-120344846327#h
17	Self-contained System	Alternative architecture	@lnunes
18	About When Not to Use Microservices	Short positioning	http://blog.christianposta.com/microservices/when-not-to-use-microservices/
19	5 Reasons Not to Use Microservices	Short analysis	https://www.usenix.org/sites/default/files/2016-09/5-reasons-not-use-microservices.pdf
20	Pattern: Monolithic Architecture	Pattern	@michaelfowler
21	It's time to stop making "Microservices" the goal of modernization	Short	@charles
22	The Death of Microservice Madness in 2018	Long analysis	https://medium.com/@kyle_brown/the-death-of-microservice-madness-in-2018-77586406287

https://docs.google.com/spreadsheets/d/1vjnjAII_8TZBv2XhFHra7kEQzQpOHSZpFIWDjynYYf0/edit?pli=1#gid=0

Microservices, when
implemented incorrectly,
can make poorly written
applications even more
dysfunctional

QUESTIONS?





°° The Journey to Cloud
Docker - Hands-On



IBM Cloud

Remember your Team Color

black 31701

olive 31711

peru 31715

white 31702

brown 31712

chocolate 31716

red 31703

lightblue 31713

orchid 31717

blue 31704

orange 31708

gold 31718

yellow 31705

purple 31709

pink 31719

lime 31706

maroon 31710

violet 31720

cyan 31707

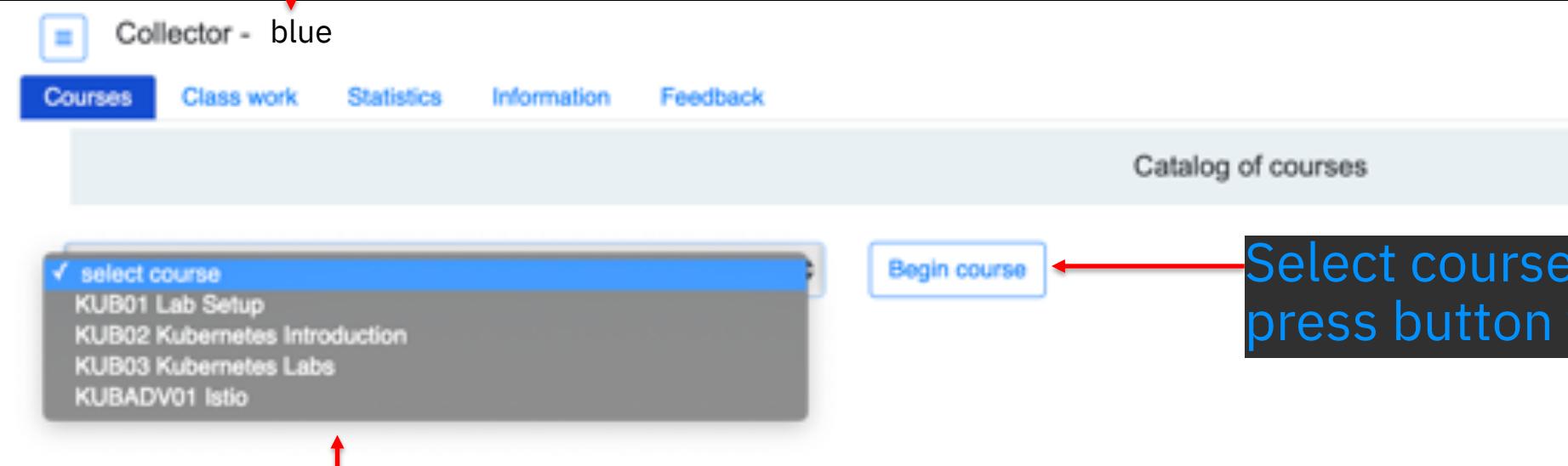
firebrick 31714

Collector - Accessing team web site

`http://158.177.137.195:{port#}`

Team name / color will be shown

blue **31704**



The screenshot shows a web browser displaying the 'Collector - blue' website. At the top, there is a navigation bar with tabs: Courses (selected), Class work, Statistics, Information, and Feedback. Below the navigation bar, a section titled 'Catalog of courses' contains a list of courses under the heading 'select course'. The courses listed are: KUB01 Lab Setup, KUB02 Kubernetes Introduction, KUB03 Kubernetes Labs, and KUBADV01 Istio. To the right of this list is a button labeled 'Begin course'. A large callout box with a red border and white text is positioned over the 'Begin course' button, containing the instruction 'Select course and press button to begin'. A red arrow points from the text 'Current course catalog' at the bottom left to the 'select course' list. Another red arrow points from the text 'Team name / color will be shown' at the top left to the 'blue' text in the top right corner of the screenshot.

Current course catalog

Team name / color will be shown

blue 31704

Collector - blue

Courses Class work Statistics Information Feedback

Catalog of courses

select course

- KUB01 Lab Setup
- KUB02 Kubernetes Introduction
- KUB03 Kubernetes Labs
- KUBADV01 Istio

Begin course

Select course and press button to begin



JTC01 – Docker Labs

Lab 0 : Docker basics

Lab 1 : Build a Docker image

Lab 2 : Run a Docker image

Lab 3 : Use the Portainer tool

Lab 4 : Deploy a more complex Docker application

Lab 5 : Push a Docker image into a registry

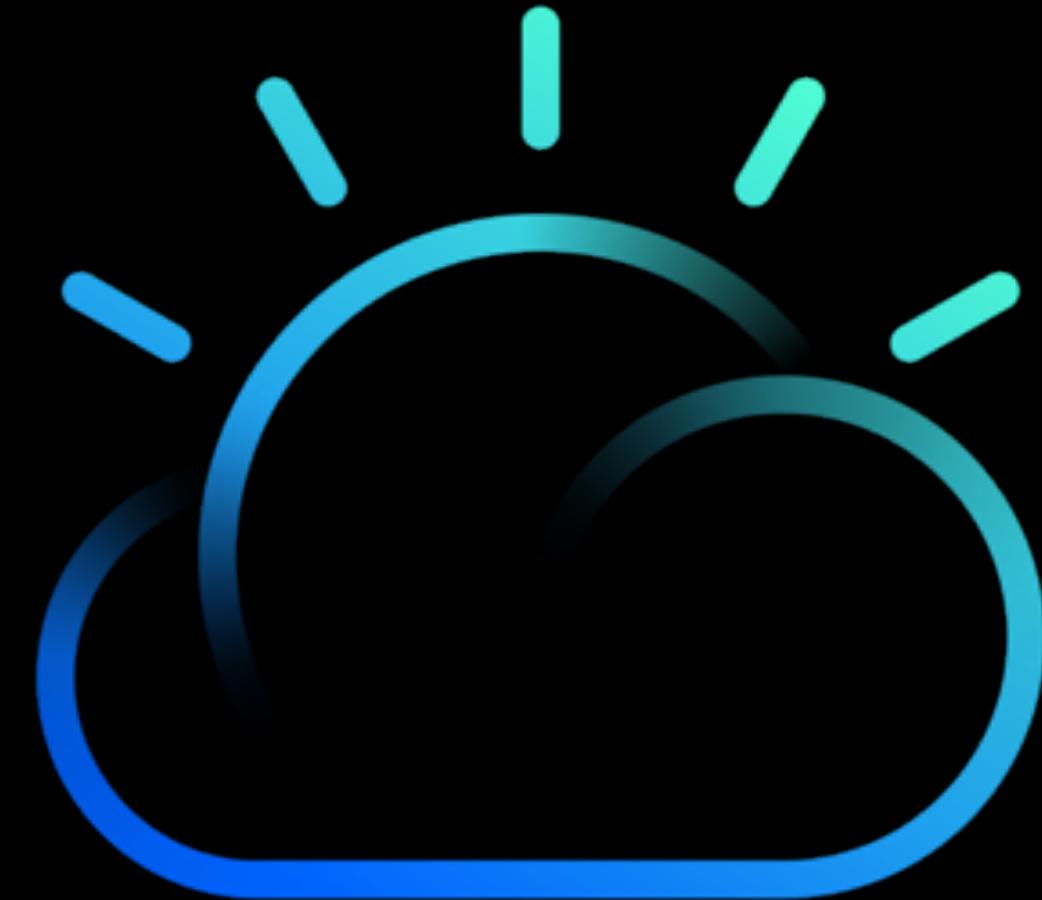


READY
SET
GO!!!!

<http://158.177.137.195:{port#}>

Duration: 60 mins

QUESTIONS?





°° The Journey to Cloud
Kubernetes - Hands-On

08



IBM Cloud



JTC02 – Kubernetes Labs

Lab 1: Refresher/overview over Kubernetes.

Lab 2: Running your first Pod on Kubernetes.

Lab 3: Deploy a Web Application on Kubernetes

Lab 4: Scale an application on Kubernetes

Lab 5: Persisting data in Kubernetes with Volumes



READY
SET
GO!!!!

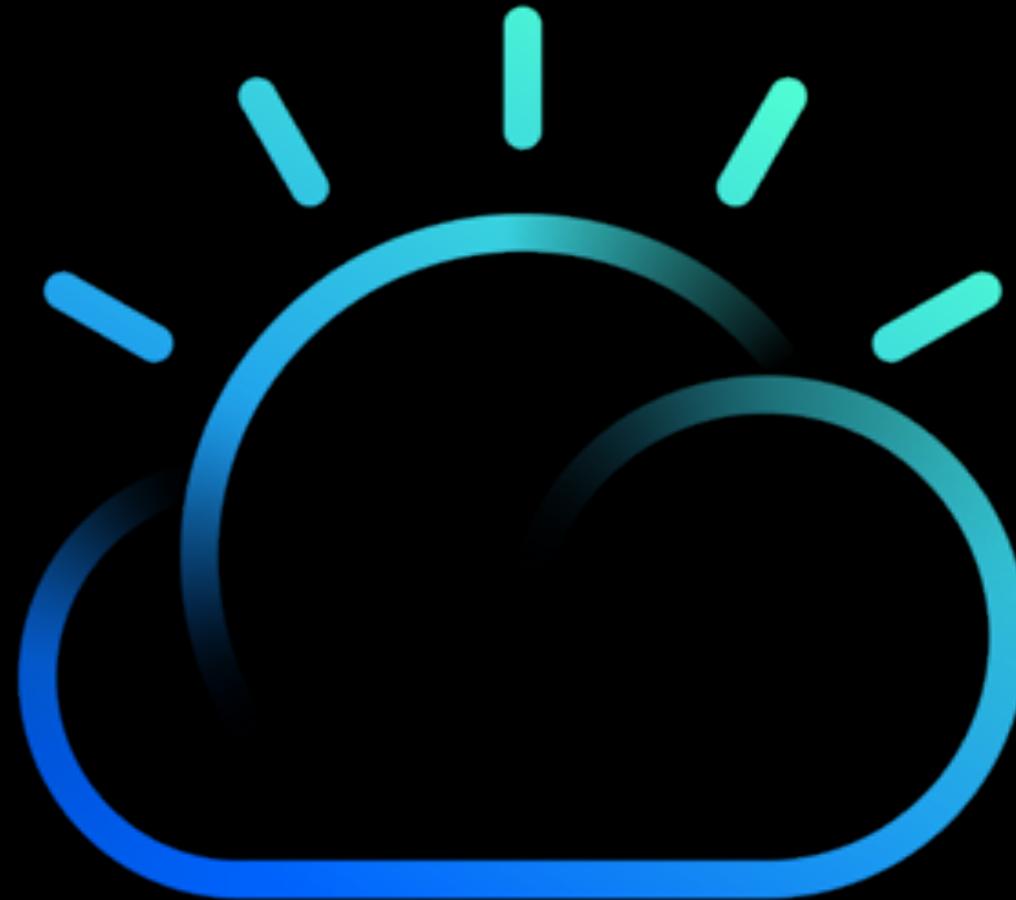
<http://158.177.137.195:{port#}>

Duration: 120 mins

QUESTIONS?



The Journey to Cloud **Kubernetes Security**



IBM Cloud

Kubernetes – Security – Back to basics

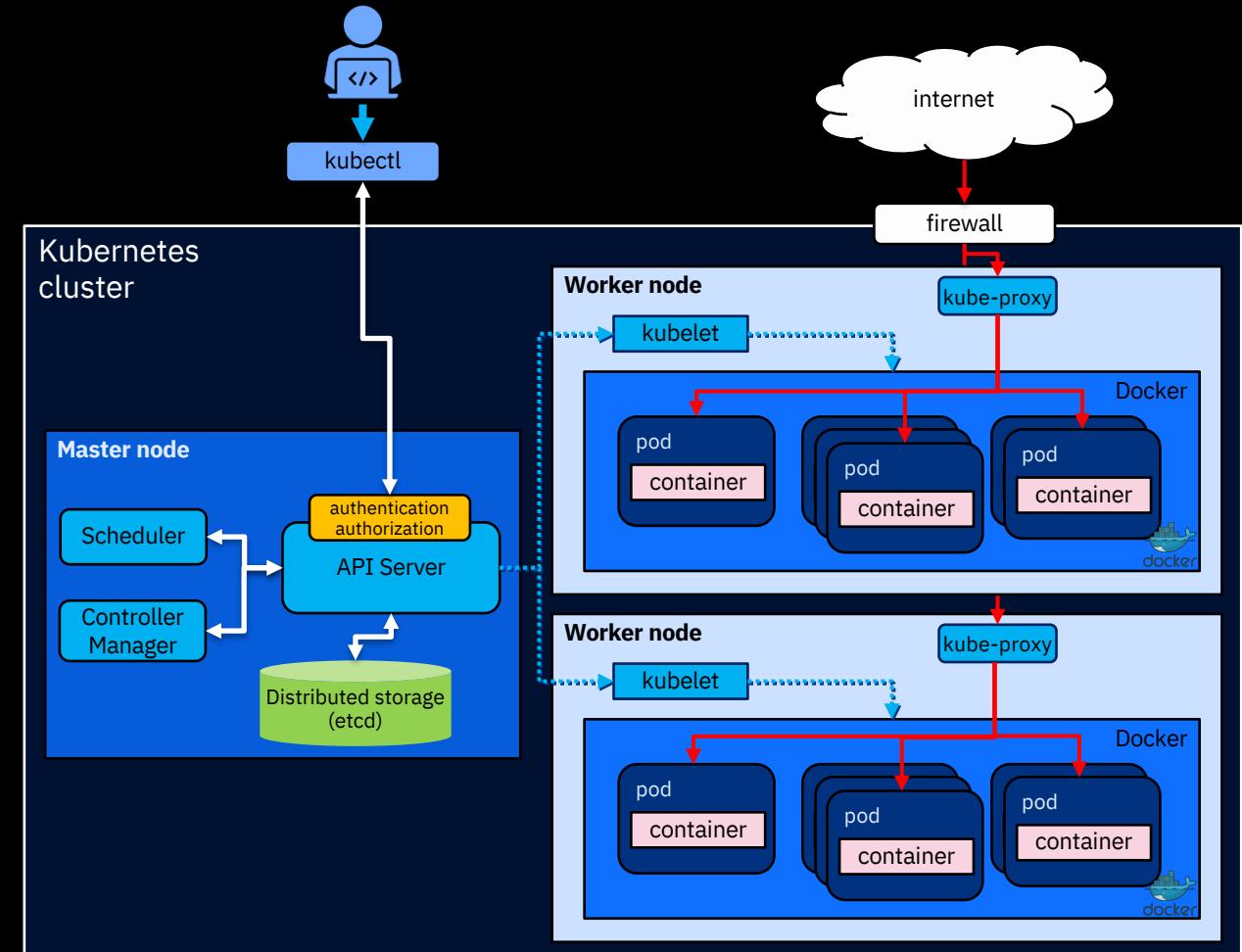


Master node

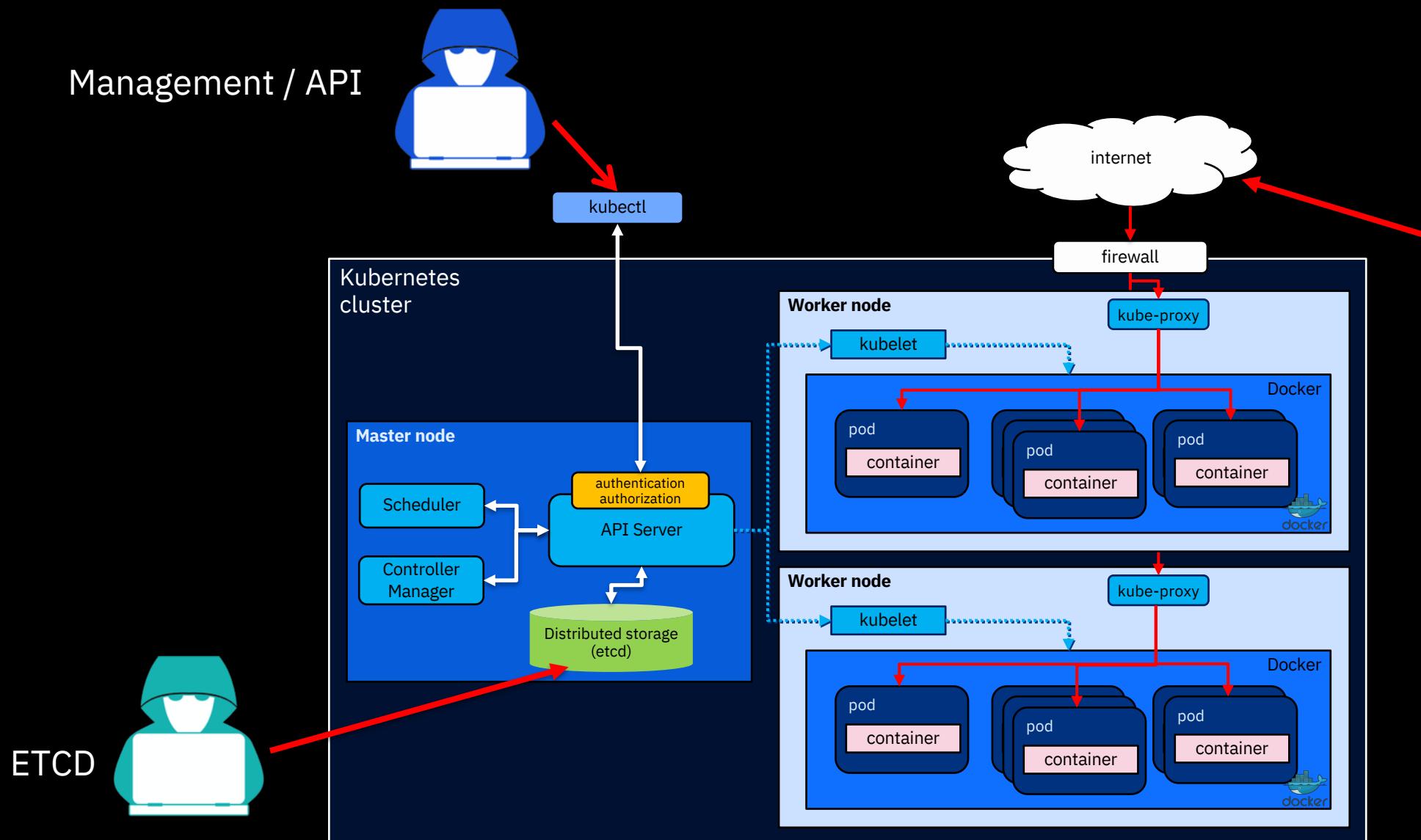
- Node that manages the cluster
- Scheduling, replication & control
- Multiple nodes for HA

Worker nodes

- Node where pods are run
- Docker engine
- kubelet agent accepts & executes commands from the master to manage pods
- kube-proxy – routes inbound or ingress traffic



Kubernetes – Security – Attack surface



Kubernetes – Security Basic Topics

The **Billion Laughs** attack is a particular type of denial of service (DoS) attack which is aimed specifically at XML document parsers. This attack is also referred to as an XML bomb or an exponential entity expansion attack.

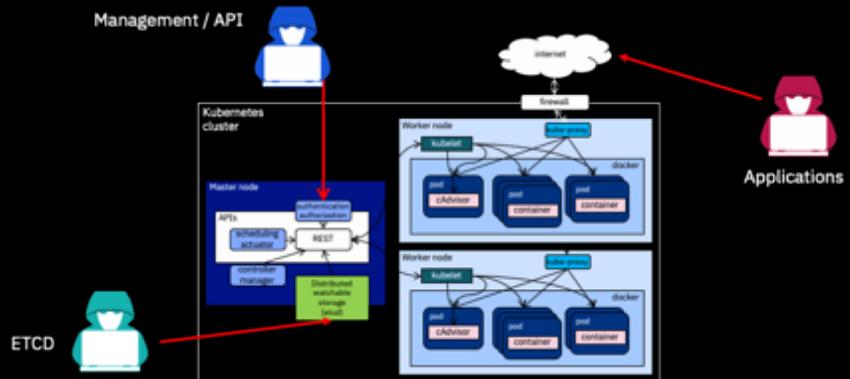
It exploits the fact that nested references to nodes can grow very large when expanded. Because the **kube-apiserver** doesn't perform validation on the manifest, it doesn't detect if those nested references will cause a problem. If the nesting references grow too large, excessive CPU and RAM usage can render the apiserver unresponsive to connections ... hence the Denial of Service.



Kubernetes – Security Basic Topics

Reduce Kubernetes Attack Surfaces

- Secure access to etcd
- Controlling access to the Kubernetes API
- Controlling access to the Kubelet
- Enforce resource usage limits for workloads
- Rotate infrastructure credentials frequently
- Enable audit logging
- Use Linux security features
- Controlling what privileges containers run with
- Enforcing Network Policies
- Image Scanning of your containers
- Use Kubernetes secrets



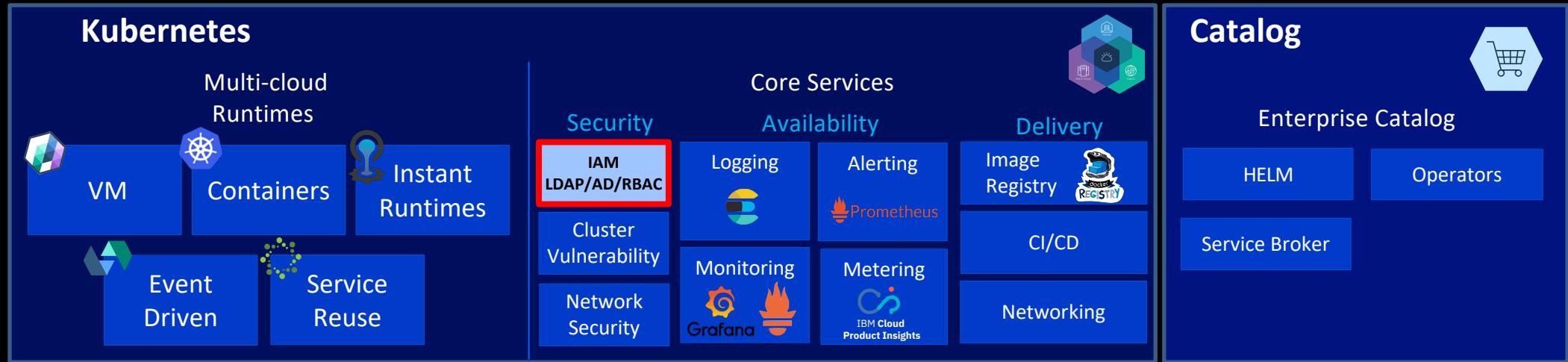
Source: <https://kubernetes.io/docs/tasks/administer-cluster/securing-a-cluster>
<https://kubernetes.io/blog/2018/07/18/11-ways-not-to-get-hacked/>

Management/API

Applications

etcd

Kubernetes – Core Services - Controlling K8s Access



Authentication – WHO am I - (token, certs, OpenID Connect Provider (OIDC)...)

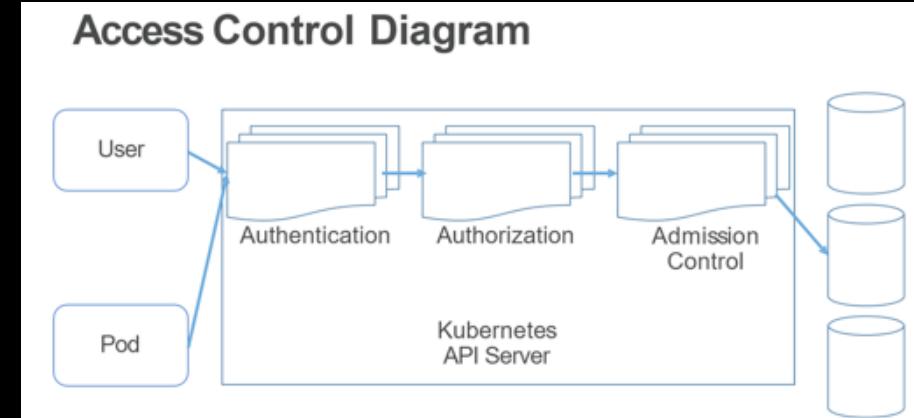
Developer – create, view, get permission.

Tester – create, delete, update, get permission.

Administrator – All permission

Authorization – CAN I - (RBAC)

Is the user or application authorized or have permissions to access a K8s object?



Admission Control

Intercepts requests to the Kubernetes API server prior to persistence of the object, but after the request is authenticated and authorized



Integrating with LDAP/AD

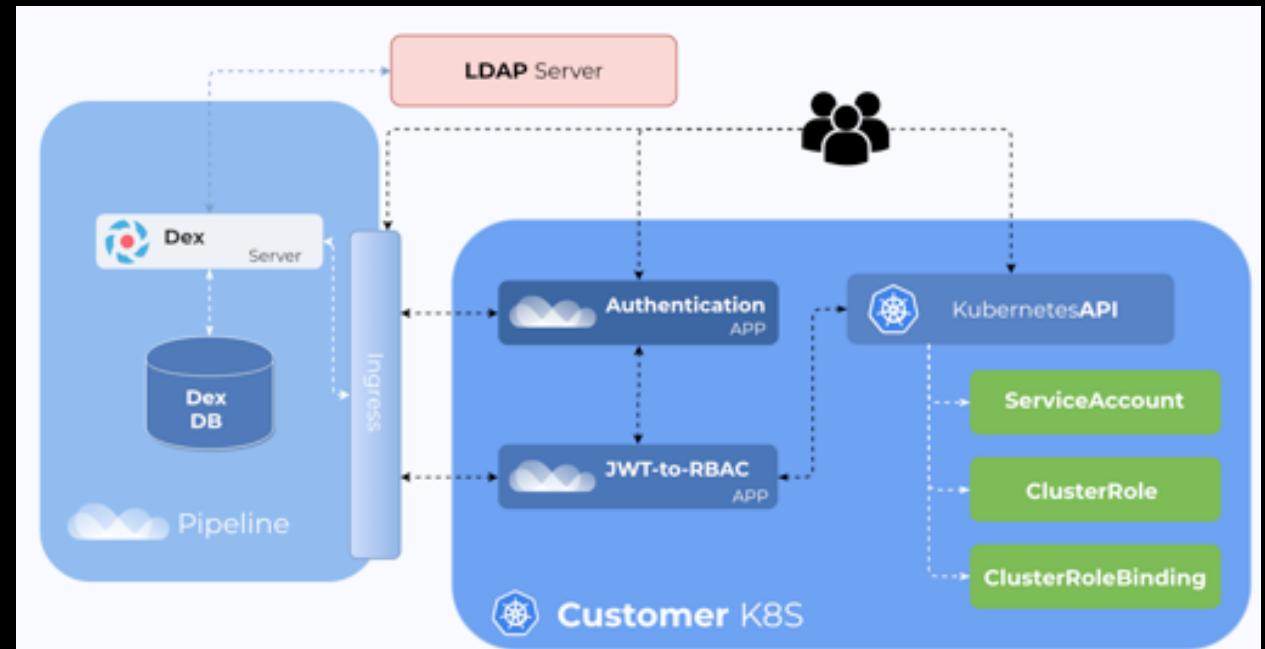
OpenID Connect

ID Tokens are JSON Web Tokens (JWTs)

dex

Dex acts as a portal to other identity providers through connectors

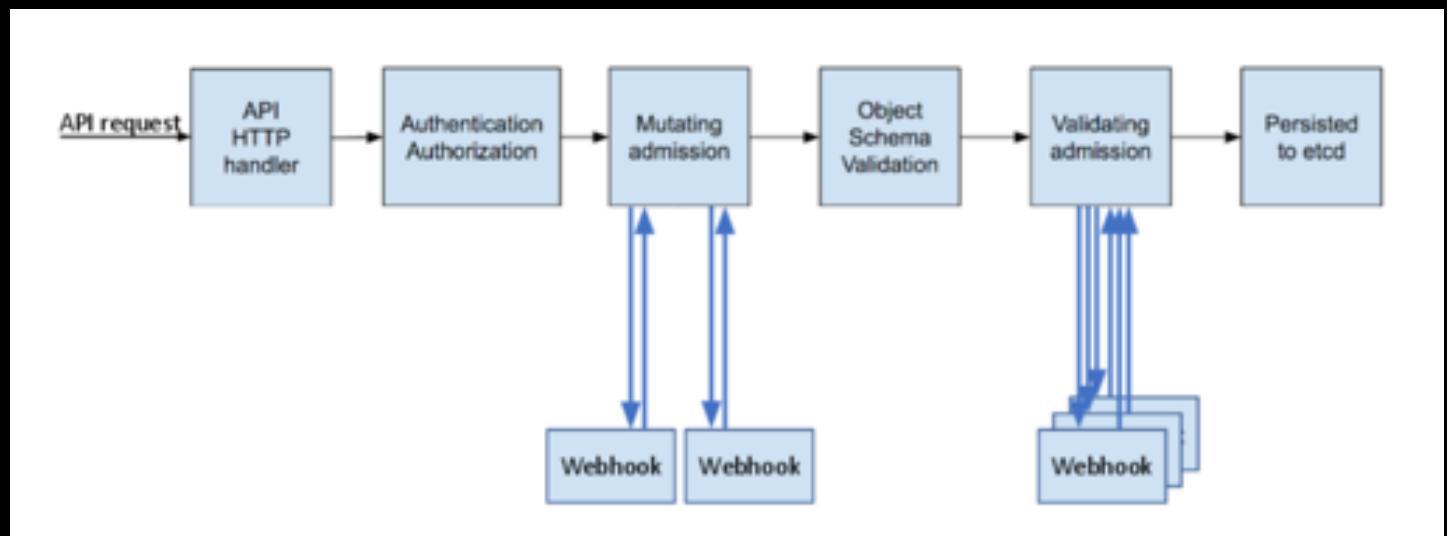
dex defer authentication to LDAP servers, SAML providers, or established identity providers like GitHub, Google, and Active Directory.



Kubernetes WebHooks

WebHooks Type

- **mutating**: to dynamically change incoming deployments on the fly (think automatic Istio sidecar injection), and
- **validating**: to accept or reject those same deployments based on the rules in your callback.
- → Image Scanner



RBAC Basic Elements

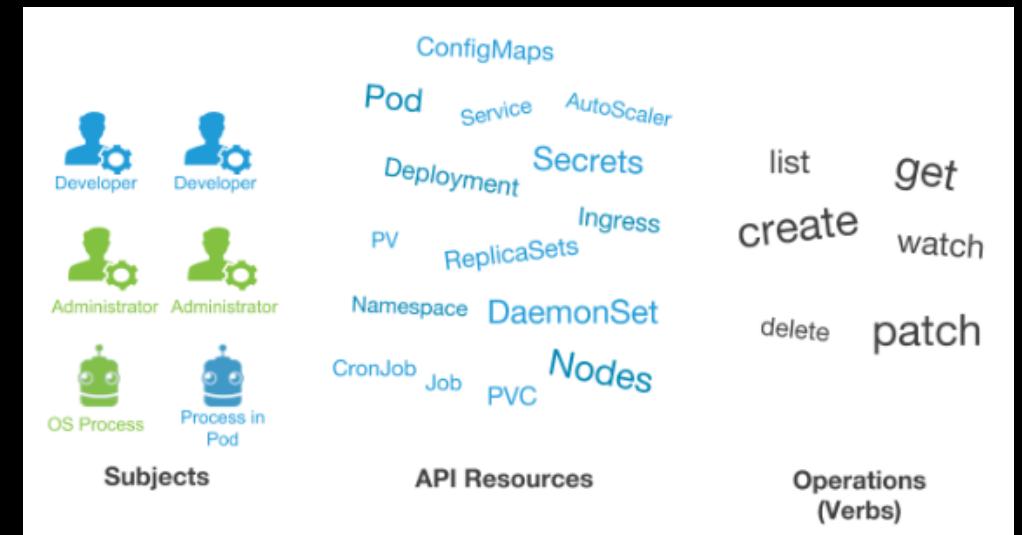
RBAC Building Blocks

Objects

- Pods
- PersistentVolumes
- ConfigMaps
- Deployments
- Nodes
- Secrets
- Namespaces

Verbs

- create
- get
- delete
- list
- update
- edit
- watch
- exec



RBAC Basic Elements

Rules

Operations (verbs) that can be carried out on a group of resources which belong to different API Groups.

Roles and ClusterRoles

Both consist of rules.

- Role: applicable to a single namespace
- ClusterRole: is cluster-wide

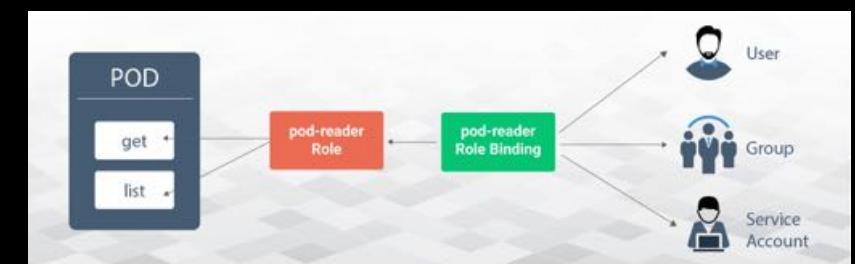
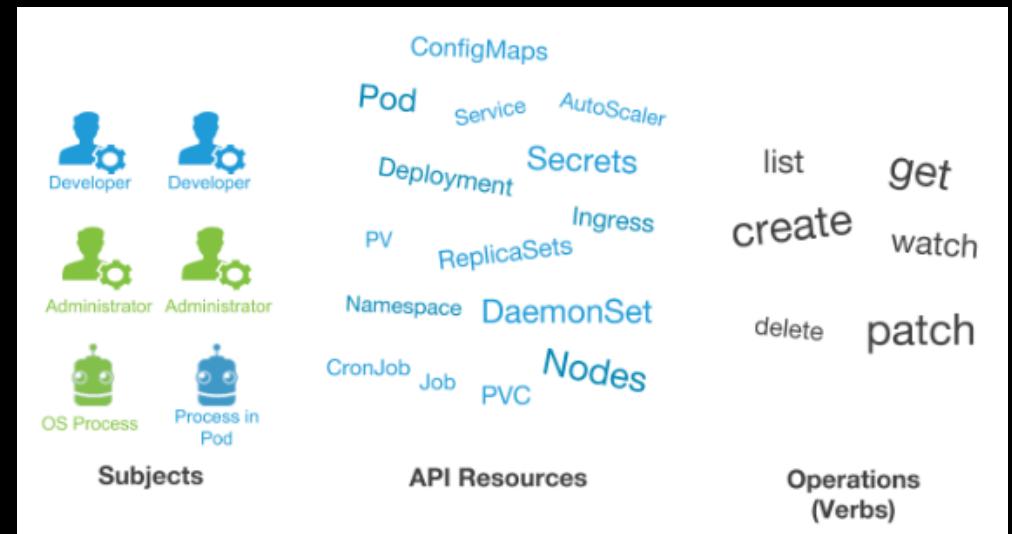
Subjects

Entity that attempts an operation in the cluster

- User Accounts: Humans or processes living outside the cluster.
- Service Accounts: Namespaced account.
- Groups: Reference multiple accounts. (default groups like cluster-admin)

RoleBindings and ClusterRoleBindings

Bind subjects to roles



RBAC Example



```
kind: Role
metadata:
  name: pod-reader
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["get", "list"]
```

```
kind: RoleBinding
metadata:
  name: pod-reader
subjects:
- kind: User
  name: John
roleRef:
  kind: Role
  name: pod-reader
```

Common RBAC Pitfalls

- **Cluster Administrator Role Granted Unnecessarily**
The built-in **cluster-admin** role grants effectively unlimited access to the cluster.
Should be granted only to the specific users that need it.
- **Duplicated Role Grant**
Role definitions may overlap with each other, making access revocation more difficult.
- **Unused Role**
Roles that are created but not granted to any subject can increase the complexity of RBAC management.
- **Grant of Missing Roles**
Role bindings can reference roles that do not exist.
If the role name is reused those role bindings can unexpectedly become active.

Always adapt
least privileged access
practices

Audit RBAC – rbac-lookup

Some tools to help you see more clearly

SUBJECT	SCOPE	ROLE
system:serviceaccounts:openshift	openshift	ClusterRole/system:image-puller
system:serviceaccounts:openshift-console	openshift-console	ClusterRole/system:image-puller
system:serviceaccounts:openshift-console	openshift-console	ClusterRole/system:image-puller
system:serviceaccounts:openshift-infra	openshift-infra	ClusterRole/system:image-puller
system:serviceaccounts:openshift-logging	openshift-logging	ClusterRole/system:image-puller
system:serviceaccounts:openshift-logging	openshift-logging	ClusterRole/system:image-puller
system:serviceaccounts:openshift-metrics-server	openshift-metrics-server	ClusterRole/system:image-puller
system:serviceaccounts:openshift-metrics-server	openshift-metrics-server	ClusterRole/system:image-puller
system:serviceaccounts:openshift-monitoring	openshift-monitoring	ClusterRole/system:image-puller
system:serviceaccounts:openshift-monitoring	openshift-monitoring	ClusterRole/system:image-puller
system:serviceaccounts:openshift-node	openshift-node	ClusterRole/system:image-puller
system:serviceaccounts:openshift-sdn	openshift-sdn	ClusterRole/system:image-puller
system:serviceaccounts:openshift-sdn	openshift-sdn	ClusterRole/system:image-puller
system:serviceaccounts:openshift-template-service-broker	openshift-template-service-broker	ClusterRole/system:image-puller
system:serviceaccounts:openshift-template-service-broker	openshift-template-service-broker	ClusterRole/system:image-puller
system:serviceaccounts:openshift-web-console	openshift-web-console	ClusterRole/system:image-puller
system:serviceaccounts:openshift-web-console	openshift-web-console	ClusterRole/system:image-puller

SUBJECT	SCOPE	ROLE
root	cluster-wide	ClusterRole/cluster-admin

Example Usage

- rbac-lookup root
- rbac-lookup openshift

Audit RBAC - rakkess

Some tools to help you see more clearly

NAME	LIST	CREATE	UPDATE	DELETE
bindings	✓	✗	✗	✗
configmaps	✓	✓	✓	✓
controllerrevisions.apps	✓	✗	✗	✗
cronjobs.batch	✓	✓	✓	✓
daemonsets.apps	✓	✓	✓	✓
daemonsets.extensions	✓	✓	✓	✓
deployments.apps	✓	✓	✓	✓
deployments.extensions	✓	✓	✓	✓
endpoints	✓	✓	✓	✓
events	✓	✗	✗	✗
events.events.k8s.io	✓	✗	✗	✗
horizontalpodautoscalers.autoscaling	✓	✓	✓	✓
ingresses.extensions	✓	✓	✓	✓
jobs.batch	✓	✓	✓	✓
leases.coordination.k8s.io	✗	✗	✗	✗
limitranges	✓	✗	✗	✗
localsubjectaccessreviews.authorization.k8s.io	✓	✗	✗	✗
networkpolicies.extensions	✓	✓	✓	✓

Example Usage

- rakkess --namespace default
- rakkess --verbs get,delete,watch,patch

Audit RBAC – rbac-view

Some tools to help you see more clearly

The screenshot shows a web-based interface for the rbac-view tool. At the top left is a logo with the text "RBAC" and a circular icon. Below the logo are two tabs: "Cluster Roles" (which is selected) and "Roles". A legend at the top provides color-coding for various verbs: create (green), delete (red), get (yellow), list (blue), watch (purple), patch (pink), update (orange), and deletecollection (grey). The main area is titled "Roles" and contains a table with several rows. Each row represents a role and its details. The columns represent different verbs. The "Subjects" column lists the subjects associated with each role. The "verbs" column lists the specific actions allowed by each role. The "Cluster Roles" tab shows a similar grid structure for cluster-level roles.

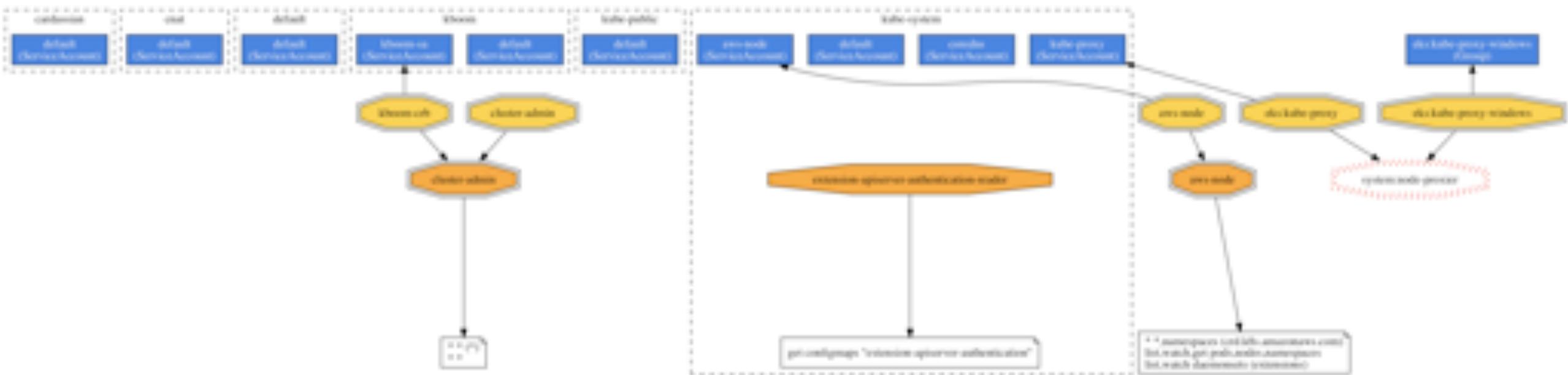
RoleName	Namespace	Subjects	verbs
system:controller:token-cleaner	kube-system	[Subjects]	[verbs]
system:admin:bootstrap-signer-clusterrole	kube-public	[Subjects]	[verbs]
system:controller:bootstrap-signer	kube-public	[Subjects]	[verbs]
system:controller:bootstrap-signer	kube-system	[Subjects]	[verbs]
system:leader-locking:kube-scheduler	kube-system	[Subjects]	[verbs]
system:controller:cloud-provider	kube-system	[Subjects]	[verbs]

Example Usage

```
./rbac-view --render html (default)  
./rbac-view --render json
```

Audit RBAC – rback

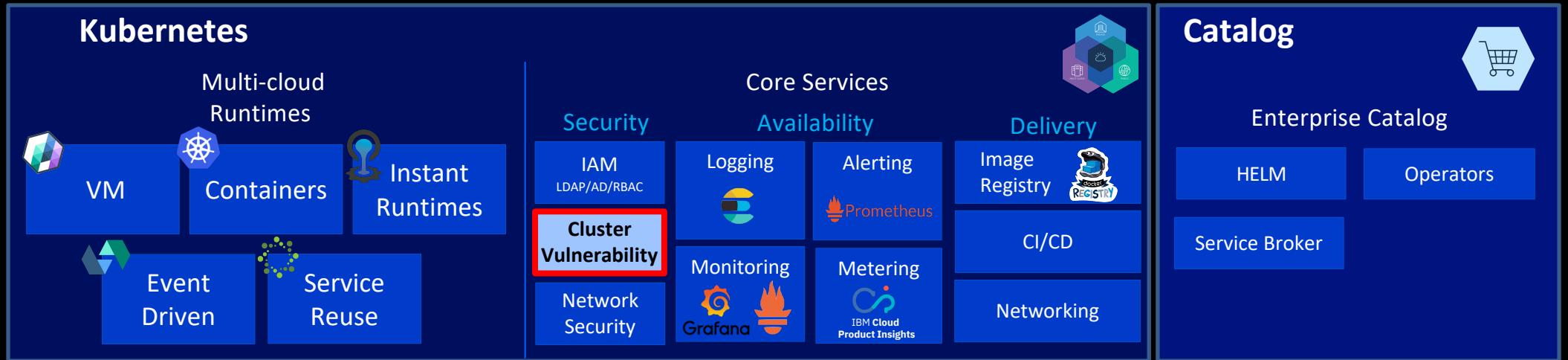
Some tools to help you see more clearly



Example Usage

```
./rbac-view --render html (default)  
./rbac-view --render json
```

Kubernetes – Core Services – Cluster Security

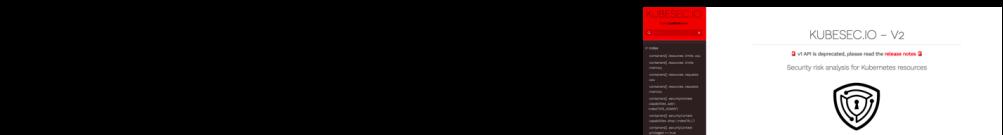


Kubernetes – Cluster Security – Secure Deployments

Scan for K8s Best Practices

kubesc – Validation of security best practices

Performs security risk analysis for Kubernetes resources and tells you what you should change in order to improve the security of those pods. It also gives you a score that you can use to create a minimum standard. The score incorporates a great number of Kubernetes best practices.



```
1  {
2    "object": "Deployment/k8sdemo.default",
3    "valid": true,
4    "message": "Passed with a score of 4 points",
5    "score": 4,
6    "scoring": {
7      "advise": [
8        {
9          "points": 1
10         },
11         {
12           "selector": ".spec .serviceAccountName",
13           "reason": "Service accounts restrict Kubernetes API access and should be configured with least privilege",
14           "points": 3
15         },
16         {
17           "selector": ".metadata .annotations .\\\"container.seccomp.security.alpha.kubernetes.io/pod\\\"",
18           "reason": "Seccomp profiles set minimum privilege and secure against unknown threats",
19           "points": 1
20         }
21       ],
22       "warning": [
23         {
24           "points": 1
25         }
26       ]
27     }
28   }
```

```
{  
  "selector": ".spec .serviceAccountName",  
  "reason": "Service accounts restrict Kubernetes API access and should be configured with least privilege",  
  "points": 3  
},
```

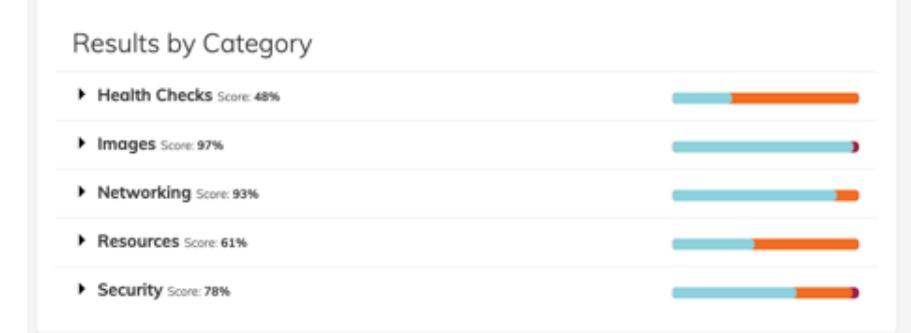
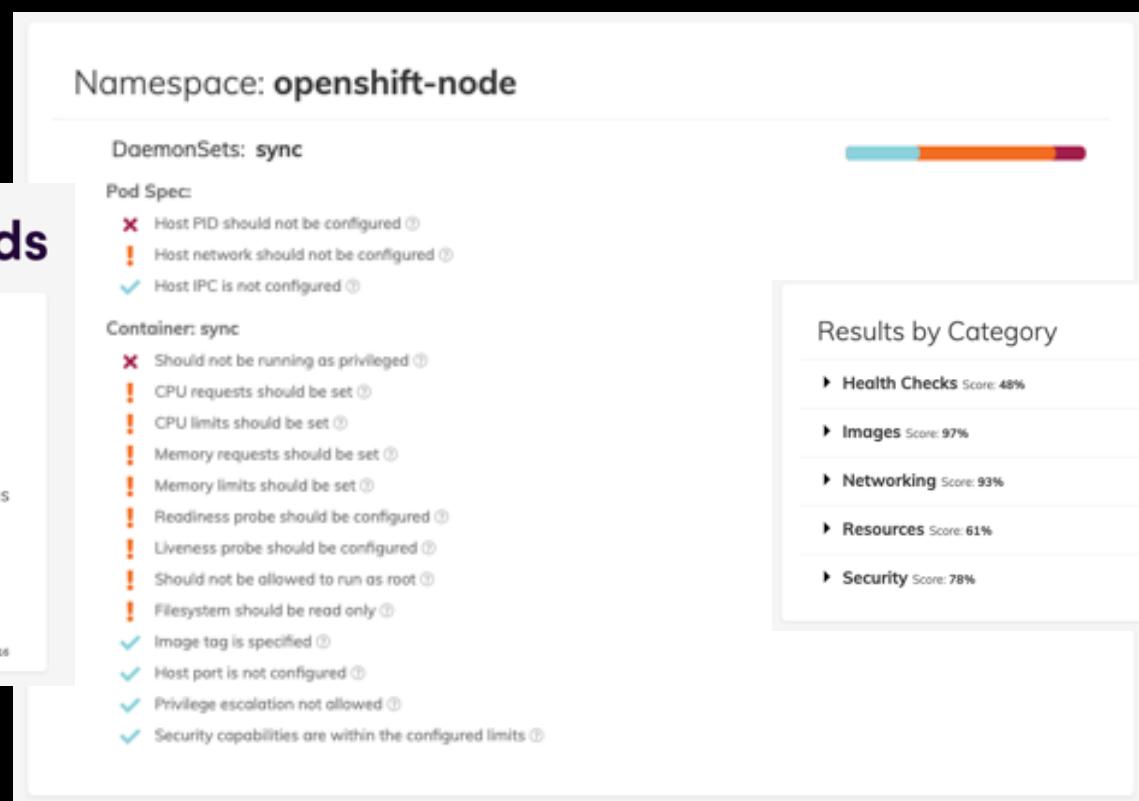
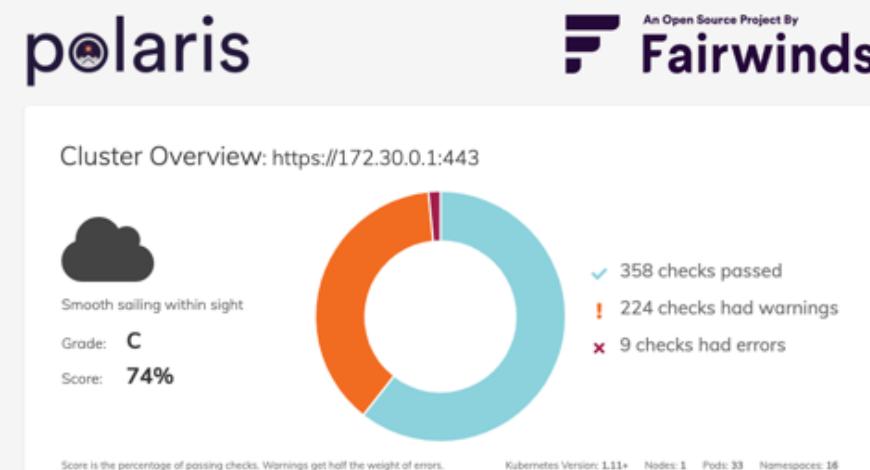
```
42   },
43   {
44     "points": 1
45     },
46     {
47       "selector": "containers[] .securityContext .runAsUser > 10000",
48       "reason": "Run as a high-UID user to avoid conflicts with the host's user table",
49       "points": 1
50     }
51   ]  
}
```

Kubernetes – Cluster Security – Secure Deployments

Scan for K8s Best Practices



Polaris – Validation of best practices



Kubernetes – Cluster Security – Secure Images

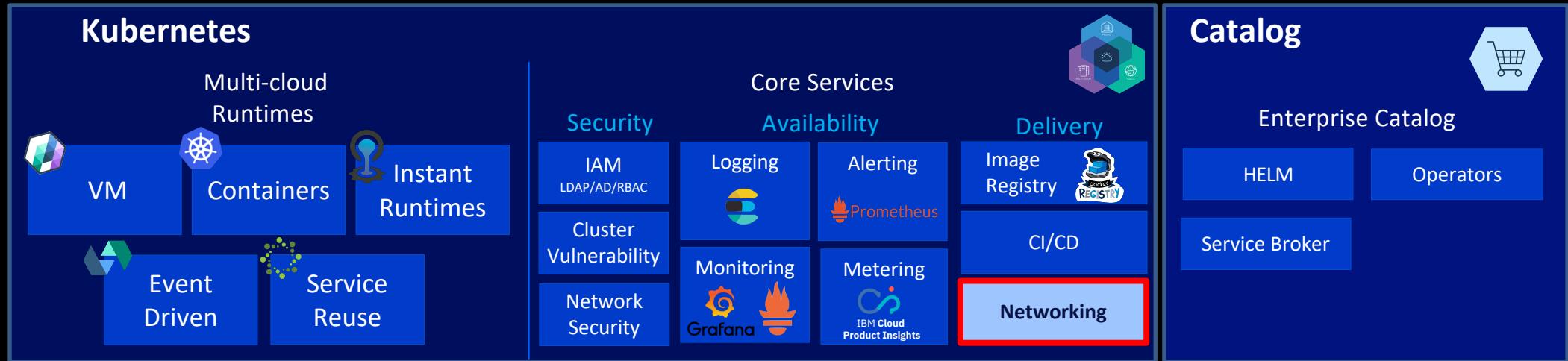
Scan images for vulnerabilities



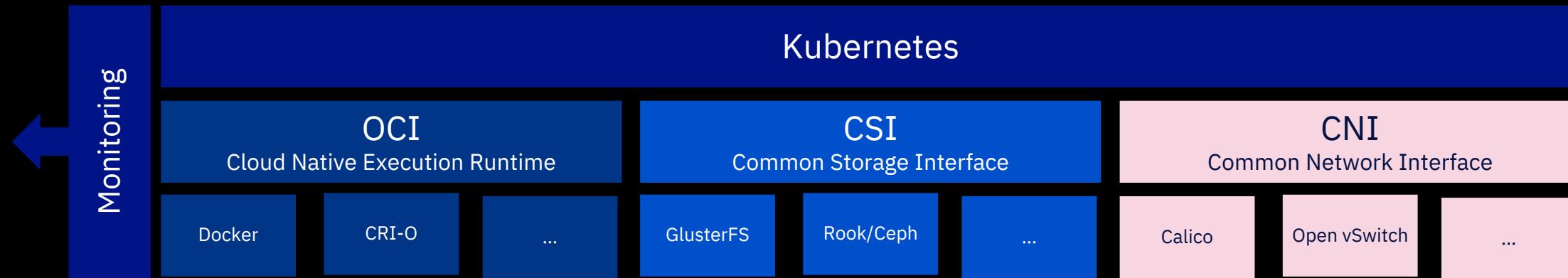
One of the best-known Image scanning tools is Clair. It is an open source project for the static analysis of vulnerabilities in application containers (currently including appc and docker).

clair timeout: 10ms docker timeout: 10ms no whitelist file Analysing 14 layers GET results from Clair API v1 Found 734 vulnerabilities Unknown: 223 Negligible: 345 Low: 184 Medium: 2																						
<table border="1"><thead><tr><th>SEVERITY</th><th>NAME</th><th>FEATURENAME</th><th>FEATUREVERSION</th><th>FIXEDBY</th><th>DESCRIPTION</th></tr></thead><tbody><tr><td>Medium</td><td>CVE-2009-3546</td><td>libwnf</td><td>0.2.6.4-18.6</td><td></td><td>The <code>_gdGetColors</code> function in <code>gd_gdvc</code> in PHP 5.2.31 and 5.3.x before 5.3.1, and the GD Graphics Library 2.x, does not properly verify a certain <code>colorstotal</code> structure member, which might allow remote attackers to conduct buffer overflow or buffer over-read attacks via a crafted GD file, a different vulnerability than CVE-2009-3293. NOTE: some of these details are obtained from third party information.</td></tr><tr><td>Medium</td><td>CVE-2007-3996</td><td>libwnf</td><td>0.2.6.4-18.6</td><td></td><td>Multiple integer overflows in <code>libgd</code> in PHP before 5.2.4 allow remote attackers to cause a denial of service (application crash) and possibly execute arbitrary code via a large (1) <code>srcW</code> or (2) <code>srcH</code> value to the (a) <code>gdImageCopyResized</code> function, or a large (3) <code>sy</code> (<code>height</code>) or (4) <code>sx</code> (<code>width</code>) value to the (b) <code>gdImageCreate</code> or the (c) <code>gdImageCreateTrueColor</code> function.</td></tr></tbody></table>					SEVERITY	NAME	FEATURENAME	FEATUREVERSION	FIXEDBY	DESCRIPTION	Medium	CVE-2009-3546	libwnf	0.2.6.4-18.6		The <code>_gdGetColors</code> function in <code>gd_gdvc</code> in PHP 5.2.31 and 5.3.x before 5.3.1, and the GD Graphics Library 2.x, does not properly verify a certain <code>colorstotal</code> structure member, which might allow remote attackers to conduct buffer overflow or buffer over-read attacks via a crafted GD file, a different vulnerability than CVE-2009-3293. NOTE: some of these details are obtained from third party information.	Medium	CVE-2007-3996	libwnf	0.2.6.4-18.6		Multiple integer overflows in <code>libgd</code> in PHP before 5.2.4 allow remote attackers to cause a denial of service (application crash) and possibly execute arbitrary code via a large (1) <code>srcW</code> or (2) <code>srcH</code> value to the (a) <code>gdImageCopyResized</code> function, or a large (3) <code>sy</code> (<code>height</code>) or (4) <code>sx</code> (<code>width</code>) value to the (b) <code>gdImageCreate</code> or the (c) <code>gdImageCreateTrueColor</code> function.
SEVERITY	NAME	FEATURENAME	FEATUREVERSION	FIXEDBY	DESCRIPTION																	
Medium	CVE-2009-3546	libwnf	0.2.6.4-18.6		The <code>_gdGetColors</code> function in <code>gd_gdvc</code> in PHP 5.2.31 and 5.3.x before 5.3.1, and the GD Graphics Library 2.x, does not properly verify a certain <code>colorstotal</code> structure member, which might allow remote attackers to conduct buffer overflow or buffer over-read attacks via a crafted GD file, a different vulnerability than CVE-2009-3293. NOTE: some of these details are obtained from third party information.																	
Medium	CVE-2007-3996	libwnf	0.2.6.4-18.6		Multiple integer overflows in <code>libgd</code> in PHP before 5.2.4 allow remote attackers to cause a denial of service (application crash) and possibly execute arbitrary code via a large (1) <code>srcW</code> or (2) <code>srcH</code> value to the (a) <code>gdImageCopyResized</code> function, or a large (3) <code>sy</code> (<code>height</code>) or (4) <code>sx</code> (<code>width</code>) value to the (b) <code>gdImageCreate</code> or the (c) <code>gdImageCreateTrueColor</code> function.																	
https://security-tracker.debian.org/tracker/CVE-2009-3546																						

Kubernetes – Core Services – Mesh Networking



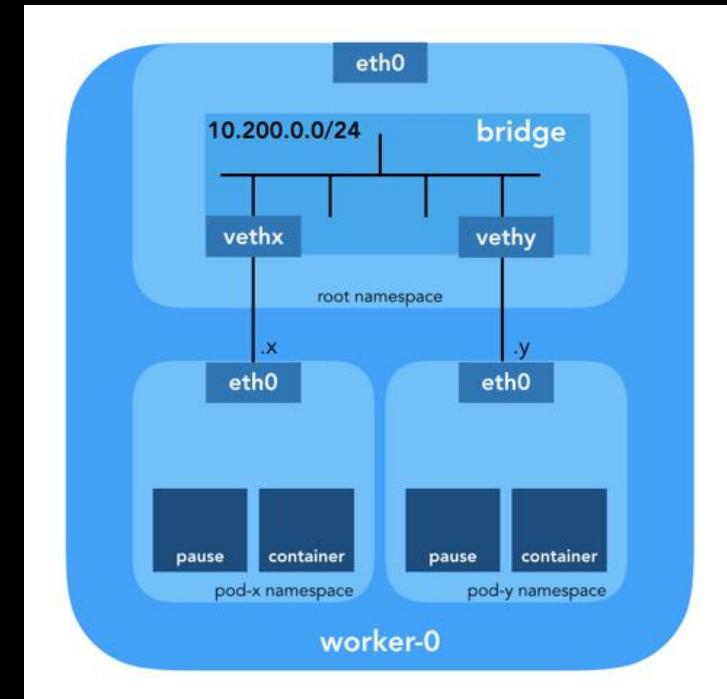
Kubernetes – Core Services – Network Security



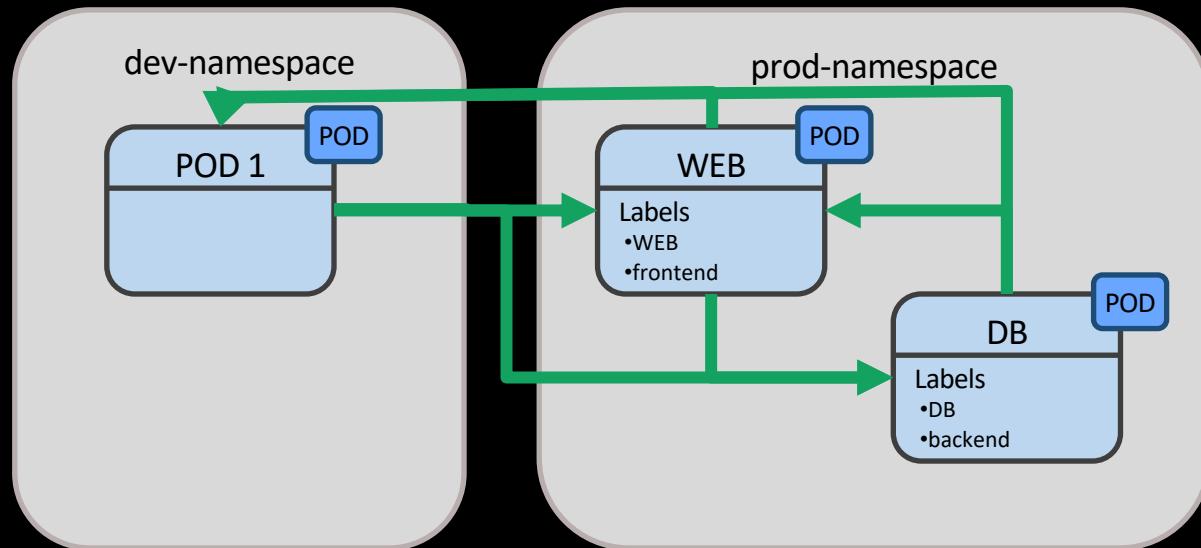
CNI – Common Network Interface

Provides a rich set of security enforcement capabilities running on top of a highly scalable and efficient virtual network

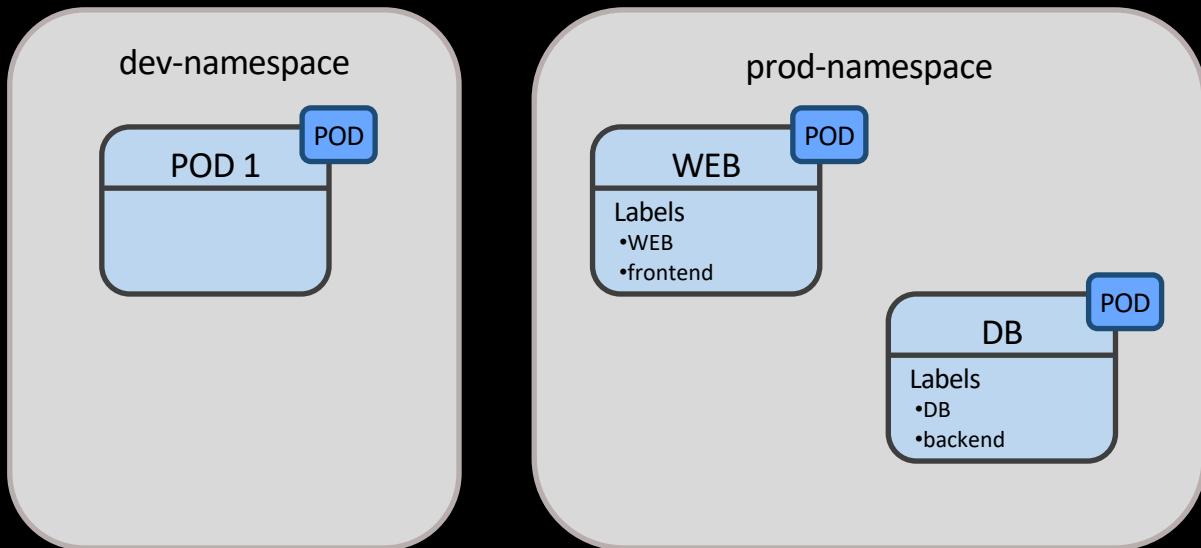
- **Calico**
Virtual networking and network security for containers, VMs, and bare metal services.
- **Open vSwitch**
Production quality, multilayer virtual switch
- ...



Kubernetes – Core Services – Network Security

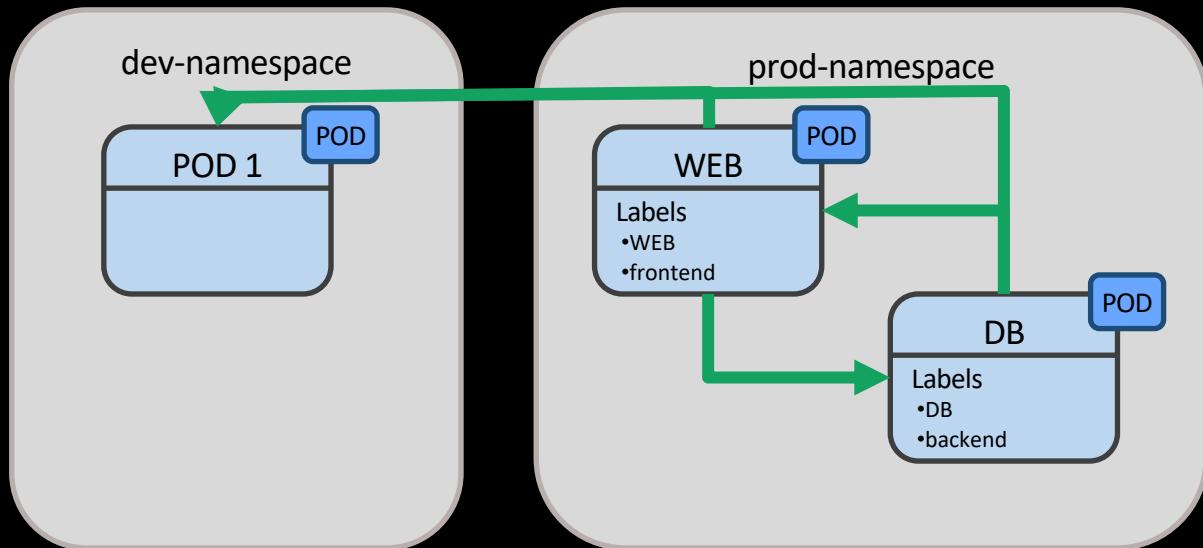


Kubernetes – Core Services – Network Security



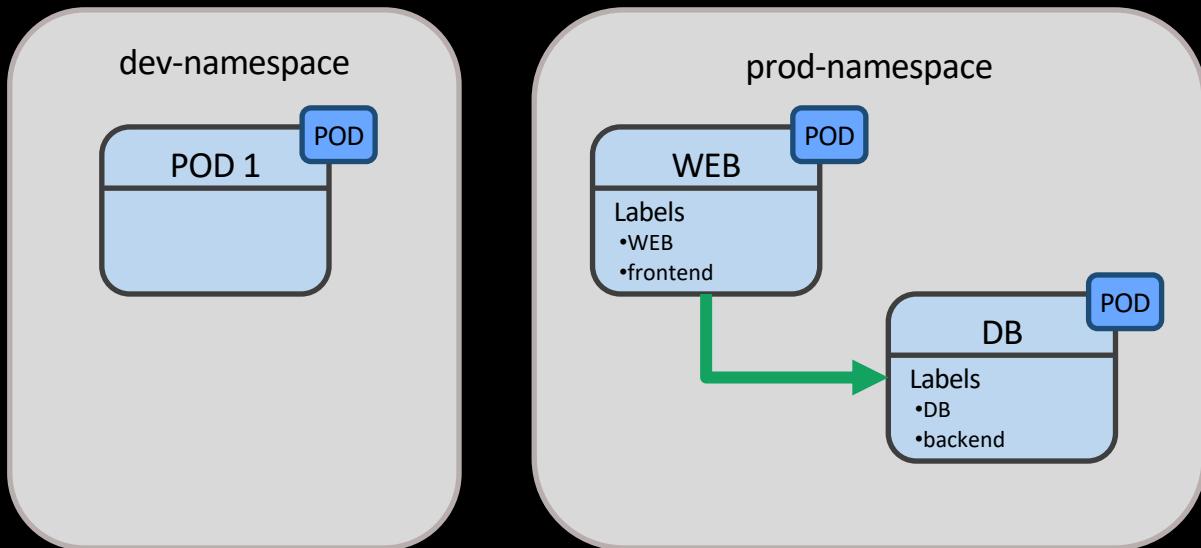
```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: default-deny-all
spec:
  podSelector:
    matchLabels: {}
```

Kubernetes – Core Services – Network Security



```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: default-deny
  namespace: dev-namespace
spec:
  podSelector:
    matchLabels: {}
```

Kubernetes – Core Services – Network Security



```
kind: NetworkPolicy
metadata:
  name: access-frontend-backend
  namespace: prod-namespace
spec:
  podSelector:
    matchLabels:
      run: DB
  ingress:
  - from:
    - podSelector:
        matchLabels:
          run: WEB
```

Kubernetes – Core Services – Network Security

Certificate Management for TLS

cert-manager is a native Kubernetes certificate management controller.

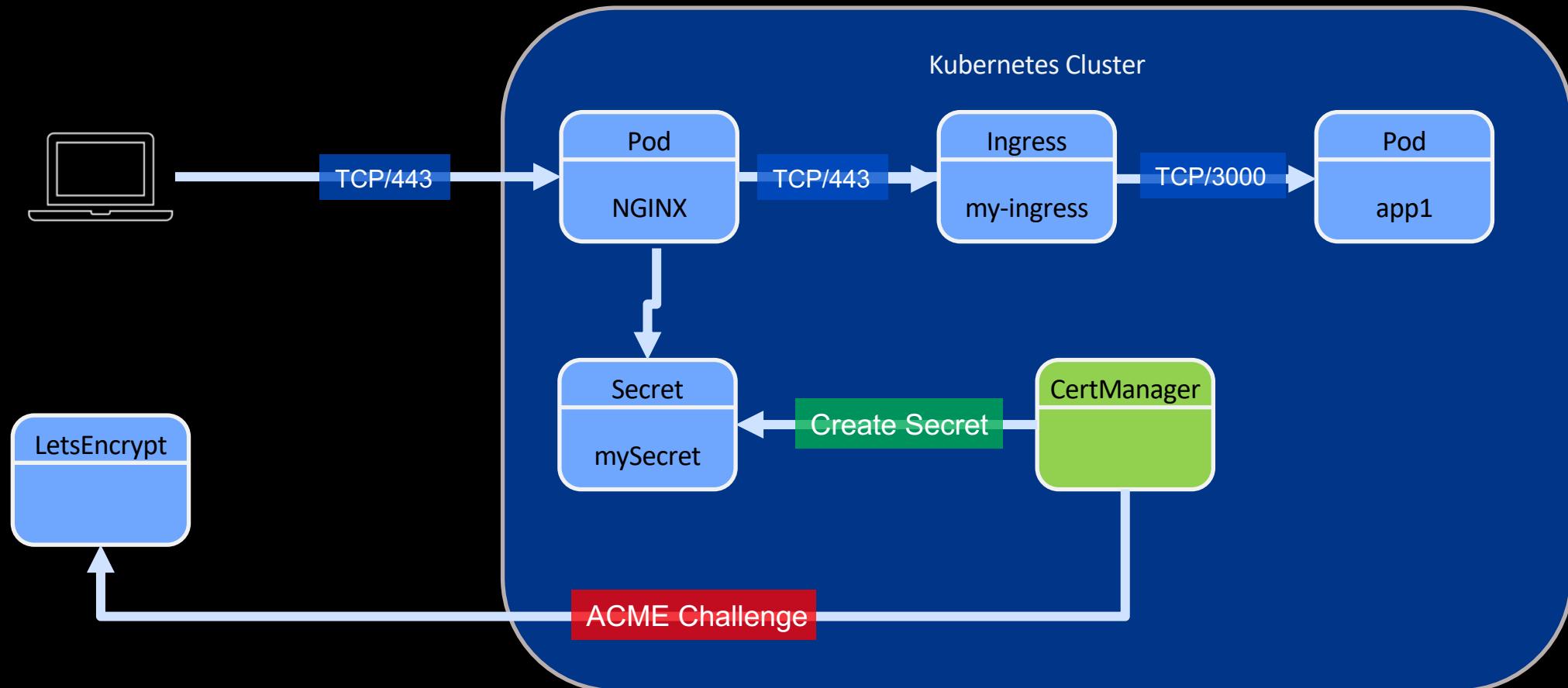
It can help with issuing certificates from a variety of sources, such as Let's Encrypt, HashiCorp Vault, Venafi, a simple signing keypair, or self signed.

It will ensure certificates are valid and up to date, and attempt to renew certificates at a configured time before expiry.



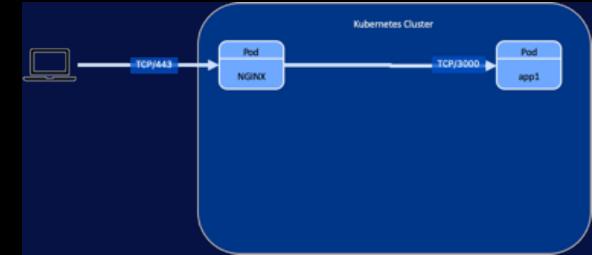
Tutorial: <https://docs.cert-manager.io/en/latest/tutorials/acme/quick-start/index.html#>

Kubernetes – Core Services – Network Security



Kubernetes – Core Services – Network Security

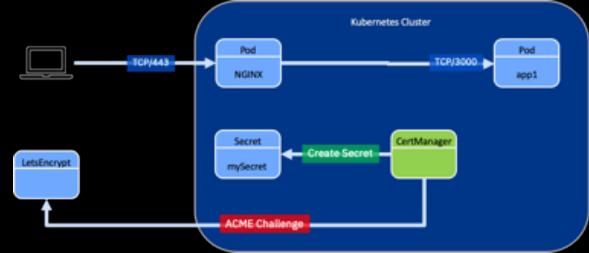
```
apiVersion: v1
kind: Service
metadata:
  name: app1
  labels:
    app: app1
    tier: frontend
spec:
  type: ClusterIP
  ports:
  - port: 80
    targetPort: 3000
  selector:
    app: app1
    tier: frontend
```



Kubernetes – Core Services – Network Security

```
apiVersion: v1
kind: Service
metadata:
  name: app1
  labels:
    app: app1
    tier: frontend
spec:
  type: ClusterIP
  ports:
  - port: 80
    targetPort: 3000
  selector:
    app: app1
    tier: frontend
```

```
apiVersion: certmanager.k8s.io/v1alpha1
kind: ClusterIssuer
metadata:
  name: letsencrypt-staging
spec:
  acme:
    server: https://acme-staging-v02.api.letsencrypt.org/directory
    email: <me@example.com>
    privateKeySecretRef:
      name: mySecret
    http01: {}
```

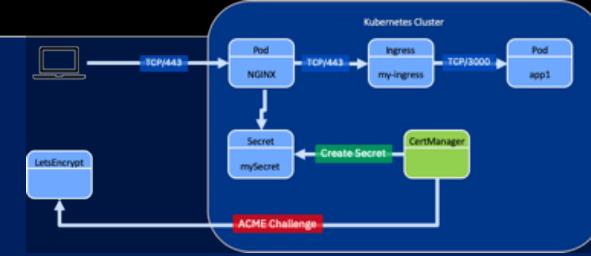


Kubernetes – Core Services – Network Security

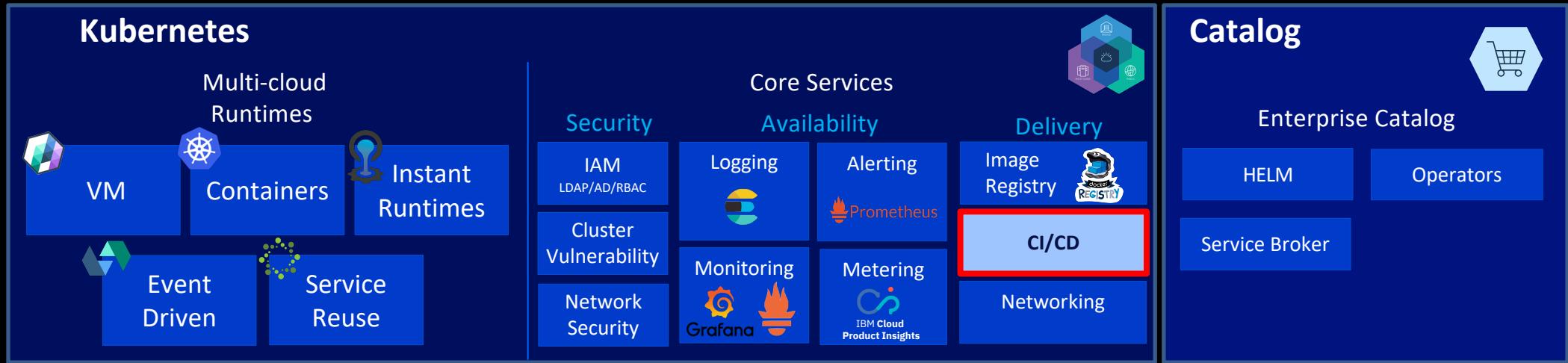
```
apiVersion: certmanager.k8s.io/v1alpha1
kind: ClusterIssuer
metadata:
  name: letsencrypt-staging
spec:
  acme:
    server: https://acme-staging-v02.api.letsencrypt.org/
    email: <me@example.com>
    privateKeySecretRef:
      name: mySecret
    http01: {}
```

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: my-ingress
  annotations:
    kubernetes.io/ingress.class: nginx
    certmanager.k8s.io/cluster-issuer: letsencrypt-staging
    kubernetes.io/tls-acme: "true"
spec:
  rules:
  - host: app.example.com
    http:
      paths:
      - path: /
        backend:
          serviceName: app1
          servicePort: 80
    tls:
    - secretName: mySecret
      hosts:
      - app.example.com
```

```
apiVersion: v1
kind: Service
metadata:
  name: app1
  labels:
    app: app1
    tier: frontend
spec:
  type: ClusterIP
  ports:
  - port: 80
    targetPort: 3000
  selector:
    app: app1
    tier: frontend
```

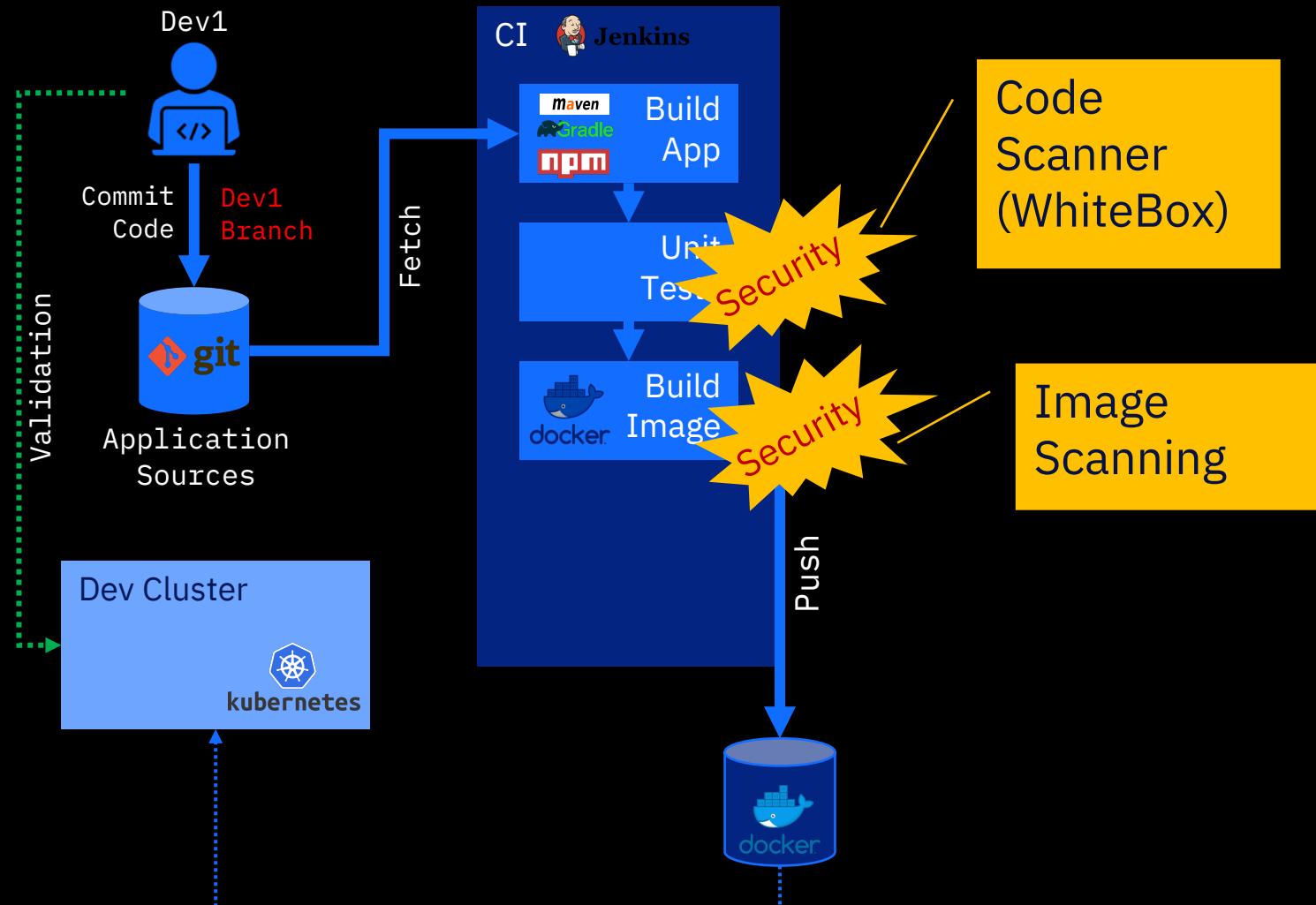


Kubernetes – Core Services – CI/CD



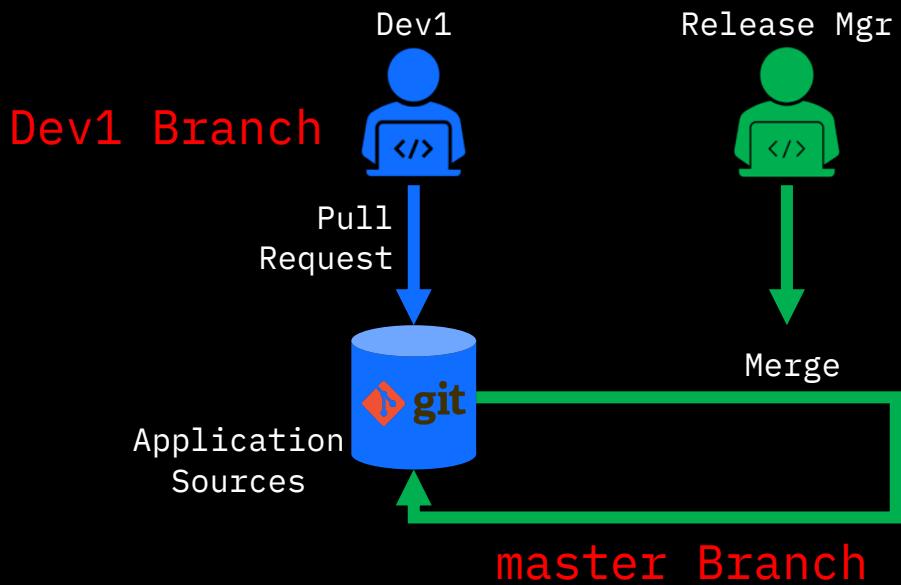
Kubernetes – CI/CD Pipeline

Development – Dev1 Branch



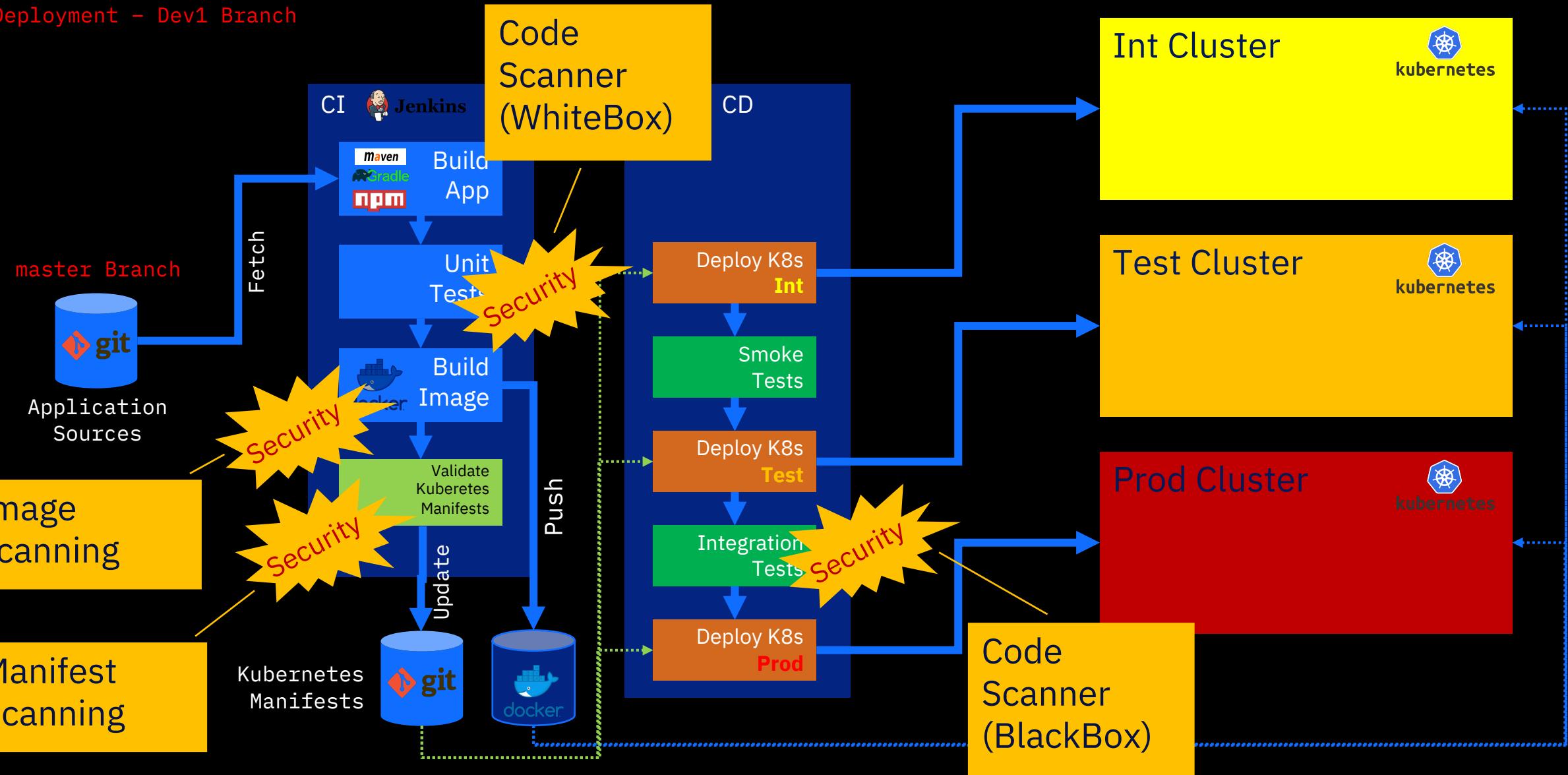
Kubernetes – CI/CD Pipeline

Merge – Dev1 Branch to master Branch

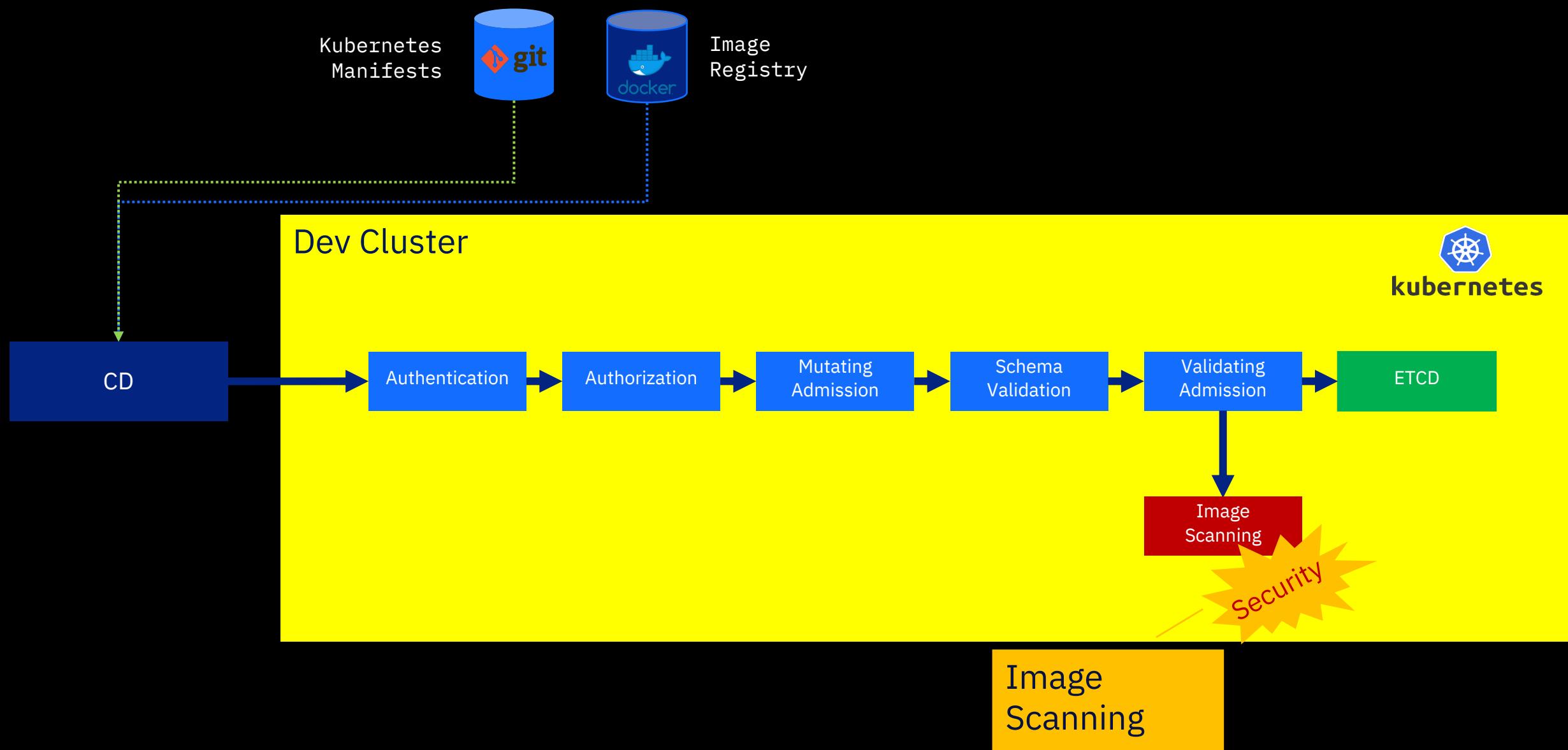


Kubernetes – CI/CD Pipeline

Deployment – Dev1 Branch



Kubernetes – CD Pipeline



DevOps – GitOps

Full Audit Trail

GitOps is using git, which is an excellent database to persist the changes made to the system.

Everything as a code

In the cloud, both CI and CD should be defined as code first. The desired state of environments should also be set in full as a declarative code.

Test Your Manifests Before You Commit

A lot of issues that escape into pre-production environments can be prevented by testing the changes before committing and pushing them.

Two Repos: One For App Source Code, Another For Manifests

Clean separation of application code vs. application config.

Separation of access

The developers who are developing the application, may not necessarily be the same people who can/should push to production environments, either intentionally or unintentionally.

Kubernetes Security – Some tools I use

RBAC

RBAC-lookup - <https://github.com/FairwindsOps/rbac-lookup>

rakkess - <https://github.com/corneliusweig/rakkess>

rbac-view – <https://github.com/jasonrichardsmith/rbac-view>

rback - <https://github.com/team-soteria/rback>

Scanning

Polaris - <https://github.com/FairwindsOps/polaris>

Clair - <https://github.com/coreos/clair>

Management

K9s - <https://github.com/derailed/k9s>

Kubernetes Security – Some Reading Tips

<https://kubernetespodcast.com/episode/065-attacking-and-defending-kubernetes/>

https://en.wikipedia.org/wiki/Red_team

<https://blog.ropnop.com/attacking-default-installs-of-helm-on-kubernetes/>

<https://kubernetes.io/docs/tasks/administer-cluster/encrypt-data/>

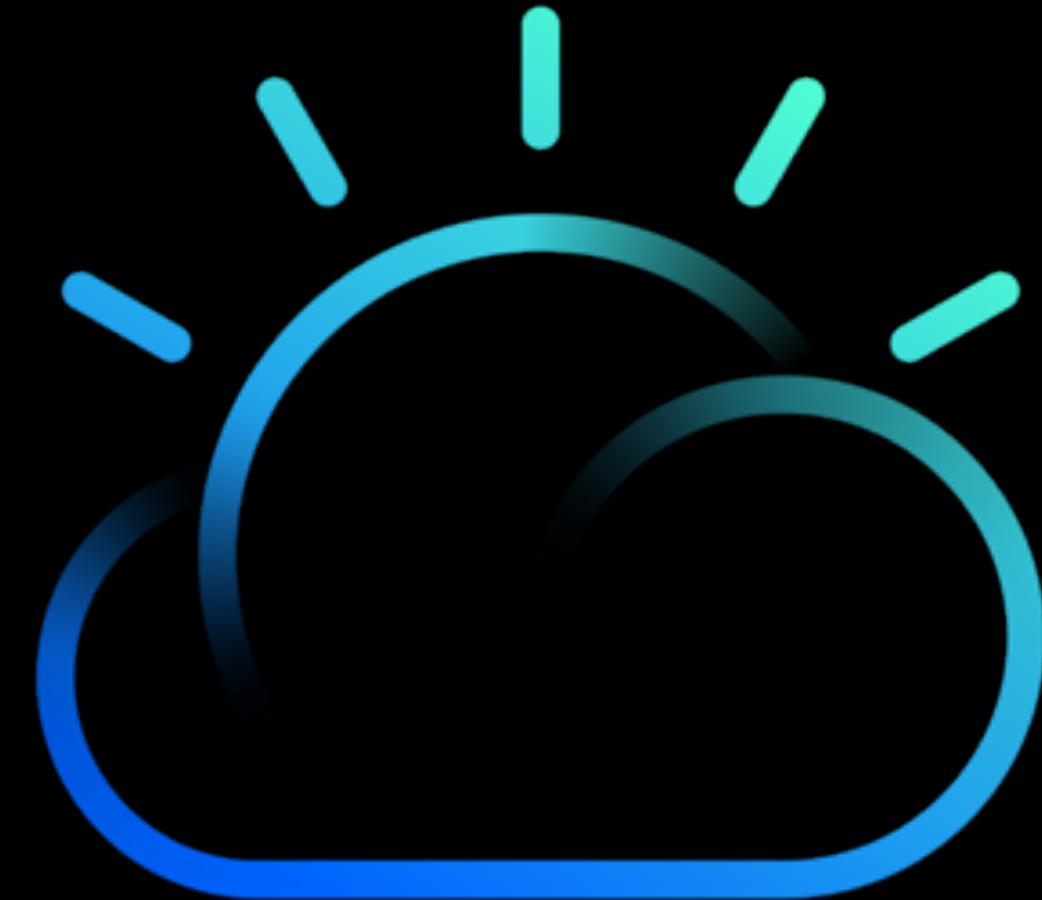
<https://kubernetes.io/blog/2019/03/21/a-guide-to-kubernetes-admission-controllers/>

<https://kubernetes.io/docs/tutorials/clusters/apparmor/>

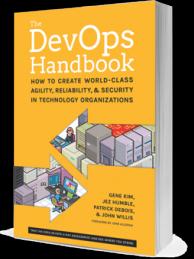
<https://kubernetes.io/docs/concepts/policy/pod-security-policy/>

<https://kubernetes.io/docs/concepts/storage/volumes/#hostpath>

QUESTIONS?

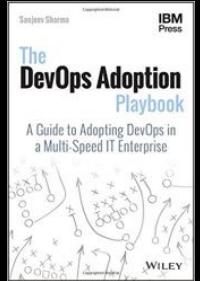


DevOps - – Some Reading Tips



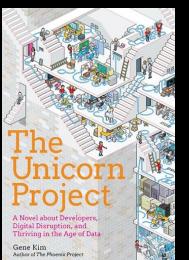
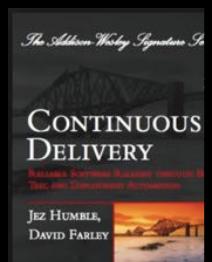
How To Create World-Class
Agility, Reliability, &
Security in Technology
Organizations

Achieve business goals,
maximize IT productivity, and
deliver innovation with
enterprise-scale DevOps.



A view into the cultural
challenges of adopting DevOps
and best practices

Automate deployments using
production-like environments and
accelerate delivery cycles



A Novel about Developers, Digital
Disruption, and Thriving in the Age
of Data

Kubernetes – Some Reading Tips

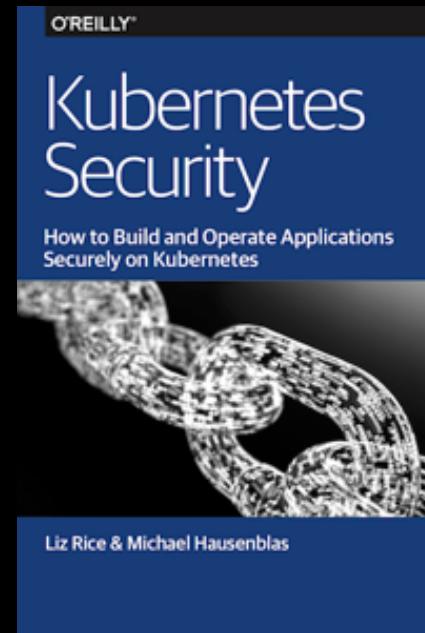


We wrote the books on
cloud adoption...

ibm.com/cloud/garage/adoption

...and Kubernetes in
the enterprise

<https://ibm.co/2LQketN>

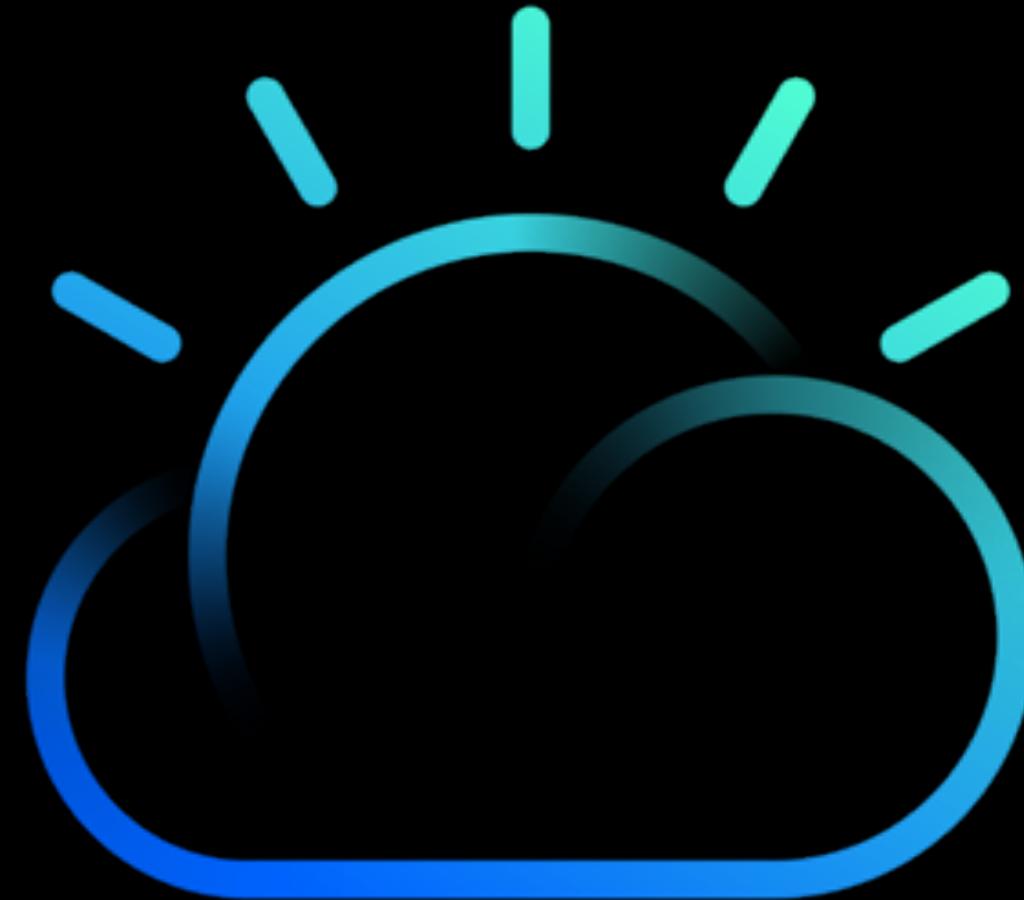


<https://kubernetes-security.info/>



°° The Journey to Cloud
Kubernetes Security -
Hands-On

10





JTC16 – Kubernetes Labs

~~Lab 1: Network Policies~~

~~Lab 2: Role Based Access Control (RBAC)~~

~~Lab 3: Service Accounts~~

~~Lab 4: Security Tooling~~

Lab 5: Image scanning



READY
SET
GO!!!!

<http://158.177.137.195:{port#}>

Duration: 30 mins

QUESTIONS?





Documentation

https://github.com/niklaushirt/k8s_training_public

Sources

<https://github.com/niklaushirt/training>

Always-on training environment

<http://158.177.137.195:31999/>

Day 02



IBM Cloud

Agenda – DevSecOps & Kubernetes Basic Concepts

Module 0: Prepare the Labs

Module 1: Kubernetes Mesh Networking

Module 2: Hands-on Istio

Lunch

Module 2: Kubernetes Best Practices

END



°° The Journey to Cloud
Prepare the Labs



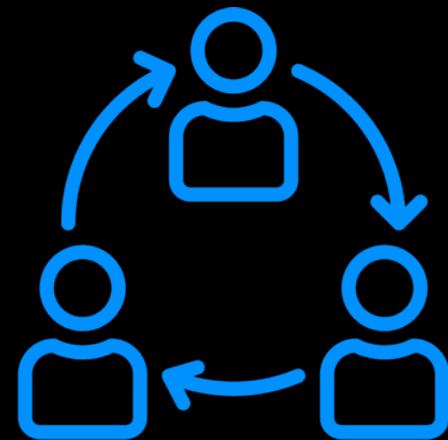
IBM Cloud

Session Objectives

Attendees will be grouped in **teams** wherever it makes sense to facilitate collaborative work.



Following some lectures will be **hands-on** work that each team collaborates to complete.



Teams

black 31701

olive 31711

peru 31715

white 31702

brown 31712

chocolate 31716

red 31703

lightblue 31713

orchid 31717

blue 31704

orange 31708

gold 31718

yellow 31705

purple 31709

pink 31719

lime 31706

maroon 31710

violet 31720

cyan 31707

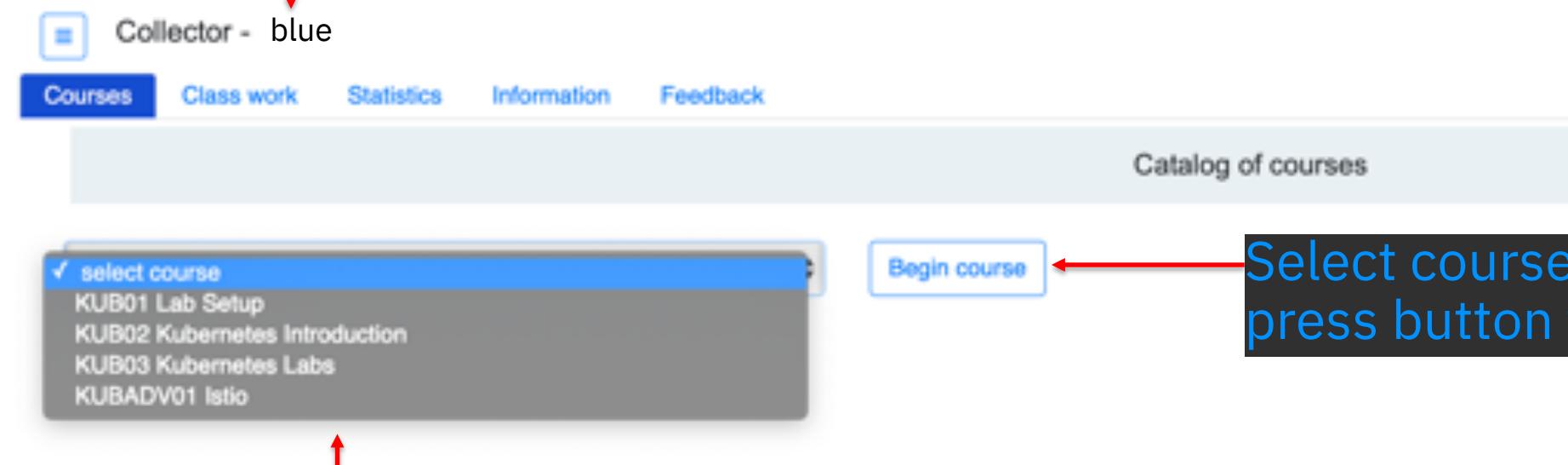
firebrick 31714

Collector - Accessing team web site

`http://158.177.137.195:{port#}`

Team name / color will be shown

blue **31704**



The screenshot shows a web browser displaying the 'Collector - blue' website. At the top, there is a navigation bar with tabs: Courses (selected), Class work, Statistics, Information, and Feedback. Below the navigation bar, a section titled 'Catalog of courses' contains a list of courses under the heading 'select course'. The courses listed are: KUB01 Lab Setup, KUB02 Kubernetes Introduction, KUB03 Kubernetes Labs, and KUBADV01 Istio. To the right of this list is a button labeled 'Begin course'. A large callout box with a red border and white text is positioned over the 'Begin course' button, containing the instruction 'Select course and press button to begin'. A red arrow points from the text 'Current course catalog' at the bottom left to the 'select course' list. Another red arrow points from the text 'Team name / color will be shown' at the top left to the 'blue' text in the top right corner of the screenshot.

Current course catalog

Team name / color will be shown

blue 31704

Collector - blue

Courses Class work Statistics Information Feedback

Catalog of courses

select course

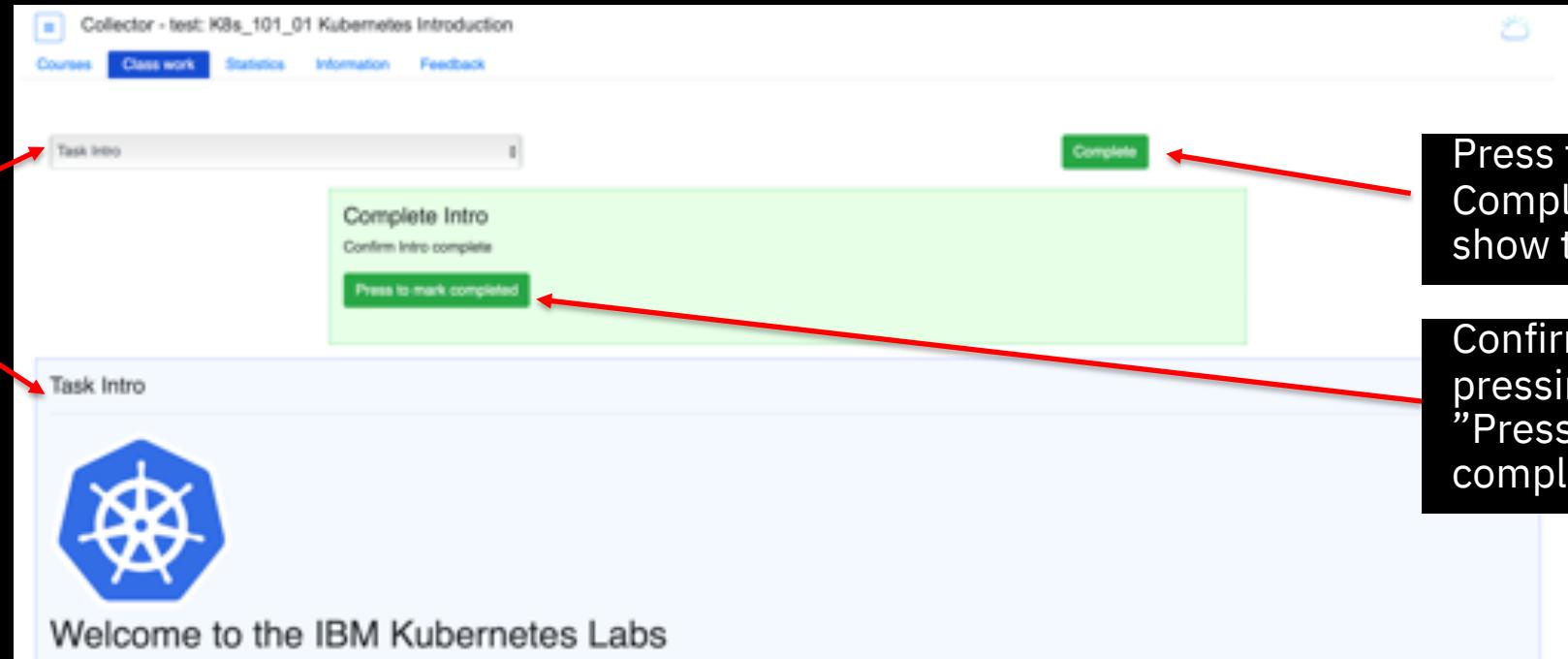
- KUB01 Lab Setup
- KUB02 Kubernetes Introduction
- KUB03 Kubernetes Labs
- KUBADV01 Istio

Begin course

Select course and press button to begin

Collector – Class work

Select class work and the blue portion of the screen is shown



Press the green Complete button to show the green portion.

Confirm completion by pressing the green "Press to mark completed" button.



The Complete Button might not show instantly depending on the course settings

Collector – Track course completed work

Course title

Collector - test: KUB03 Kubernetes Labs

Courses Class work Statistics Information Feedback

Completed Work - It may take a second or two to show the updated progress

Team	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
test	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Collector - test: KUB03 Kubernetes Labs

Courses Class work Statistics Information Feedback

Completed Work - It may take a second or two to show the updated progress

Team	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
test	✓	✓	✓	✓	✓	✓	○	○	○	○	○	○	○	○	○

Green checkmark - item is completed
Red circle - item is waiting to be completed

The number of items tracked will change based on the current course selected.

Collector – Instructor Dashboard

Remaining Time for the Lab

Collector - instructor: K8s_101_01 Kubernetes Introduction

Remaining time: 0h 29m 50s

Courses Class work Statistics Information Feedback Insight

Completed Work - It may take a second or two to show the updated progress

Team	01	02	03	04	05	06
instructor	✓	○	○	○	○	0
lightblue	✓	✓	✓	✓	✓	1
olive	✓	✓	○	○	○	2
peru	○	○	○	○	○	3
chocolate	○	○	○	○	○	4
pink	○	○	○	○	○	5
violet	○	○	○	○	○	6

Collector - instructor: K8s_101_01 Kubernetes Introduction





JTC90 Lab Setup

EVERYBODY

Task 2: Setup Kubectl

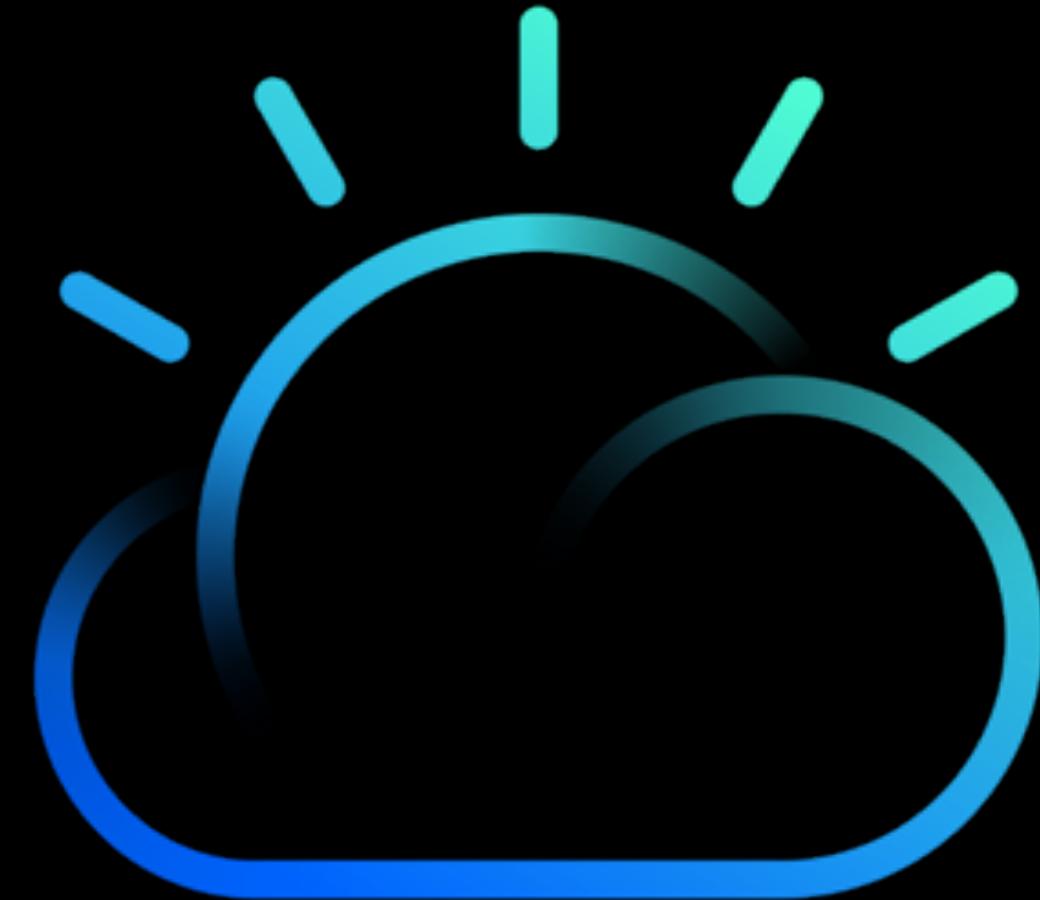
OK

Task 3: Setup git

Task 4: Final Check

?

QUESTIONS?



The Journey to Cloud **Kubernetes Mesh** Networking

01



IBM Cloud

The trade off

Improved delivery velocity

in exchange for

increased operational complexity



Common DevOps Challenge 1

How do I **roll out** a newer version of my microservice
without down time?

How do I **ensure traffic** continue to go to the current version
before the newer version is tested and ready?

Common DevOps Challenge 2

How do **canary testing**?

How do I proceed to a **full rollout** after satisfactory testing of the new version?

Common DevOps Challenge 3

How do I do **A/B testing**?

- Release a new version to a subset of users in a precise way

I want to leverage crowdsourced testing. How do I **test** the new version **with a subset of users**?

I have **launched B in the dark**, but how can I keep B to myself or a small testing group?

Other common DevOps Challenges

4. Things don't always go correctly in production... How do I **inject fault** to my microservices to prepare myself?
5. My services can only **handle certain rate**, how can I limit rate for some of my services?

Other common DevOps Challenges

6. I need to **view and monitor** what is going on with each of my services when crisis arises.
7. How can I **secure my services**.

Service Mesh

**Dedicated infrastructure layer
to make
service-to-service communication
fast, safe and reliable**

Istio



A service mesh designed to connect, manage and secure micro services

The image shows two news snippets side-by-side. The left snippet is from Forbes, dated May 25, 2017, with 3,859 views. It title is "Google, IBM And Lyft Want To Simplify Microservices Management With Istio". The right snippet is from ZDNet, dated May 24, 2017, with the title "Google, IBM, and Lyft launch open source project Istio". Both snippets mention the project's purpose of providing a vendor-neutral way to connect, secure, manage, and monitor microservices.

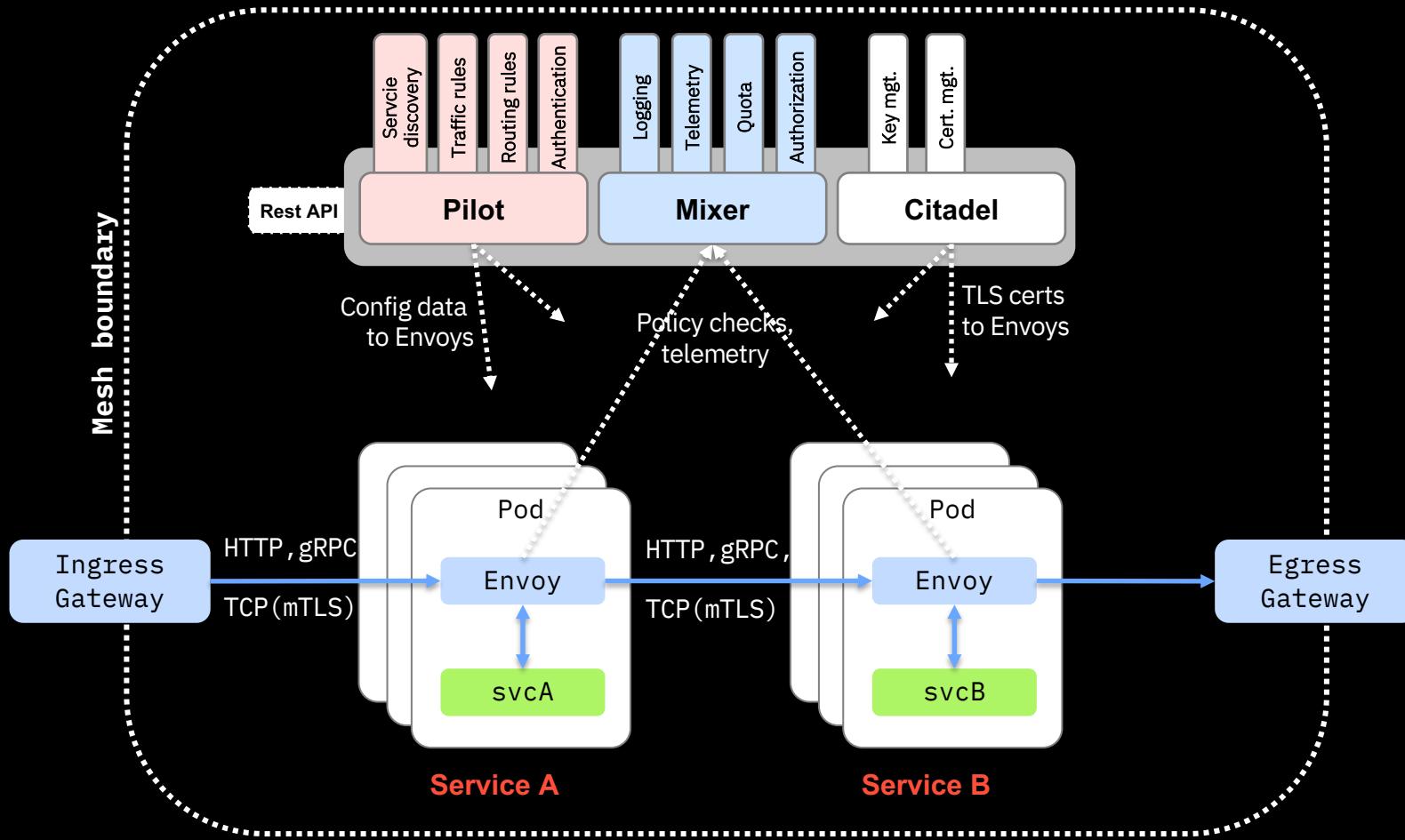
The image shows a blog post from Google Cloud, dated May 24, 2017, by Varun Talwar. The title is "Istio: a modern approach to developing and managing microservices". The post discusses the alpha release of Istio and its benefits for microservices management. It includes social sharing icons for Facebook, Twitter, LinkedIn, and Email.

Launched in May 2017 by Google, Lyft and IBM

Open Source

Zero Code Changes

ISTIO - Architecture

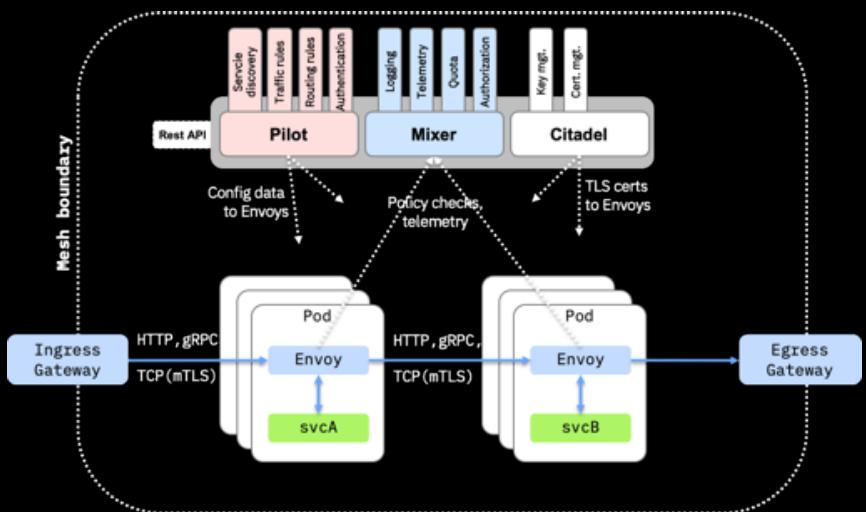


●—● Istio control plane traffic. Request routing rules, resilience configuration (circuit breakers, timeouts, retries), policies (ACLs, rate limits, auth), and metrics/reports from proxies.

●—● User/application traffic. HTTP/1.1, HTTP/2, gRPC, TCP with or without TLS

Operates at **Layer 7**

- policies can be applied based on virtual host, URL, or other HTTP headers.
- Flexibility in processing.
- Allows it to be distributed



Custom resource definitions

Ingress Configuration

```
kind: Gateway
metadata:
  name: helloworld-gateway
spec:
  selector:
    istio: ingressgateway
  servers:
    - hosts:
        - myapp.demo.com
    port:
      name: http
      number: 80
      protocol: HTTP
```



<https://myapp.demo.com/demo>

URL Routing

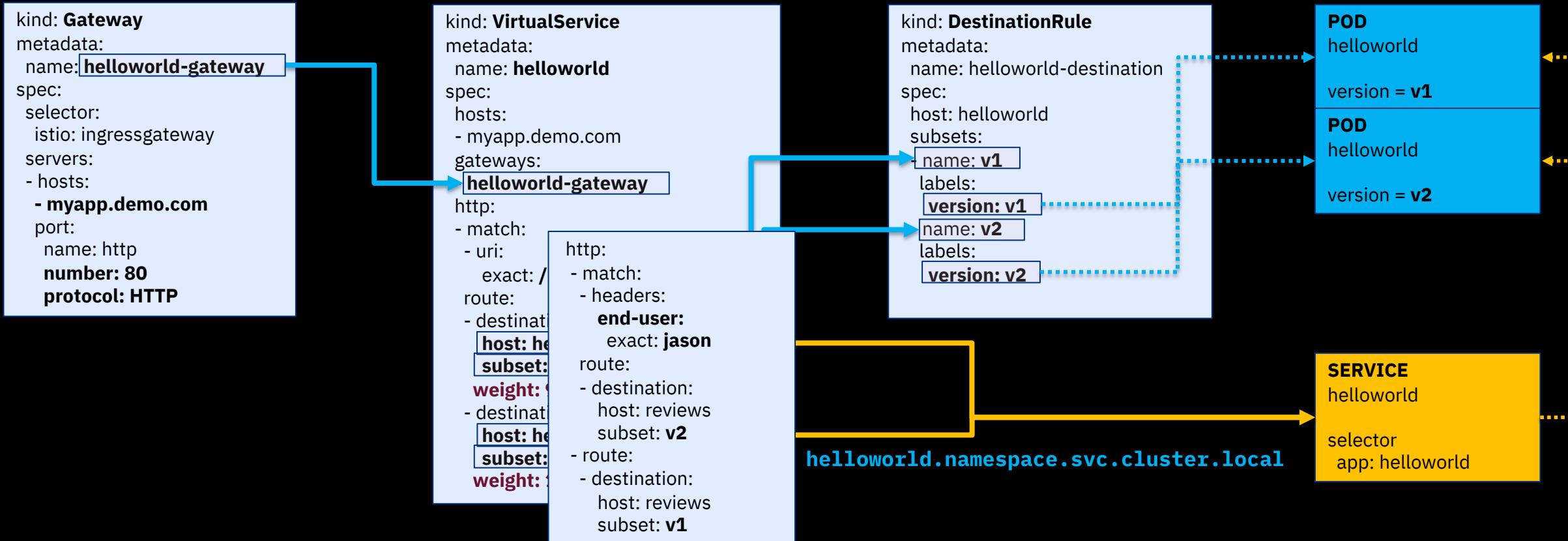
```
kind: VirtualService
metadata:
  name: helloworld
spec:
  hosts:
    - myapp.demo.com
  gateways:
    - helloworld-gateway
  http:
    - match:
        - uri:
            exact: /demo
      route:
        - destination:
            host: helloworld
```



DestinationRule

```
kind: DestinationRule
metadata:
  name: helloworld-destination
spec:
  host: helloworld
  subsets:
    - name: v1
      labels:
        version: v1
    - name: v2
      labels:
        version: v2
```

Custom resource definitions



Sidecar injection

Manual Injection

```
kubectl apply -f <(istioctl kube-inject -f myapp.yaml)
```

Automatic Injection

For automatic sidecar injection, Istio relies on Mutating Admission Webhooks.

Label the namespace where you are deploying the app with **istio-injection=enabled**

```
root@please1:~# kubectl get namespaces --show-labels
NAME        STATUS    AGE      LABELS
default     Active    35d     istio-injection=enabled
dev-namespace Active    35d     <none>
godemo      Terminating   16d    istio-injection=enabled
istio-system Active    35d     icp=system
kube-public  Active    35d     <none>
kube-system  Active    35d     icp=system
platform     Active    35d     <none>
prod-namespace Active    35d     <none>
services     Active    35d     <none>
test-namespace Active    35d     <none>
```

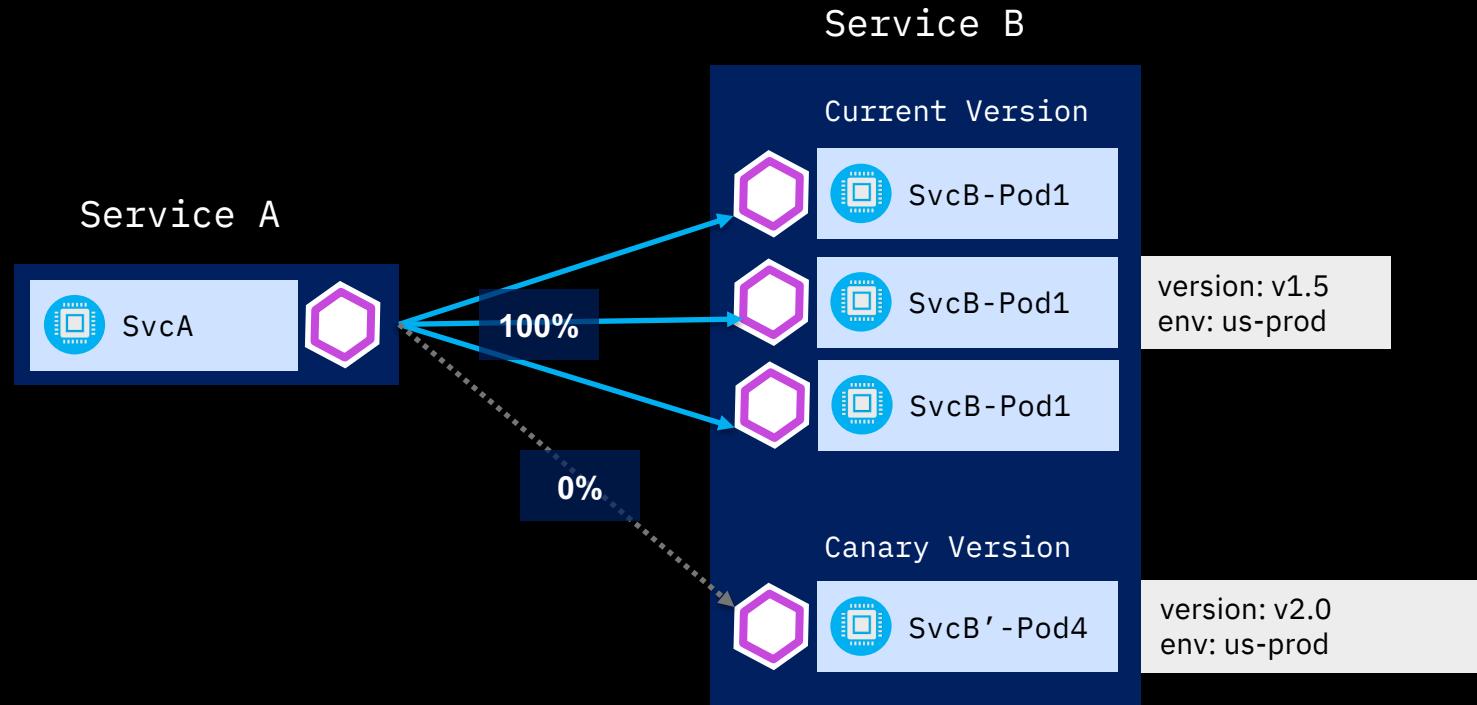
```
kind: Deployment
metadata:
  name: no-injection
spec:
  template:
    metadata:
      annotations:
        sidecar.istio.io/inject: "false"
    spec:
      containers:
        - name: no-injection
          image: nginx
```

Addressing DevOps Challenges



#	CHALLENGE	ISTIO SOLUTION
CHALLENGE 1	ROLL OUT NEW VERSION WITHOUT DOWNTIME OR CHANGING CODE	TRAFFIC CONTROL
CHALLENGE 2	HOW TO DO CANARY TESTING	TRAFFIC SPLITTING
CHALLENGE 3	HOW TO DO A/B TESTING	TRAFFIC STEERING
CHALLENGE 4	THINGS DON'T ALWAYS GO CORRECTLY IN PRODUCTION...	TRAFFIC MIRRORING RESILIENCY RESILIENCY TESTING
CHALLENGE 5	HOW CAN I LIMIT RATE FOR SOME OF MY SERVICES?	RATE LIMITING
CHALLENGE 6	I NEED TO VIEW AND MONITOR WHAT IS GOING ON WHEN CRISIS ARISES	TELEMETRY
CHALLENGE 7	HOW CAN I SECURE MY SERVICES?	AUTHENTICATION AUTHORIZATION CALICO

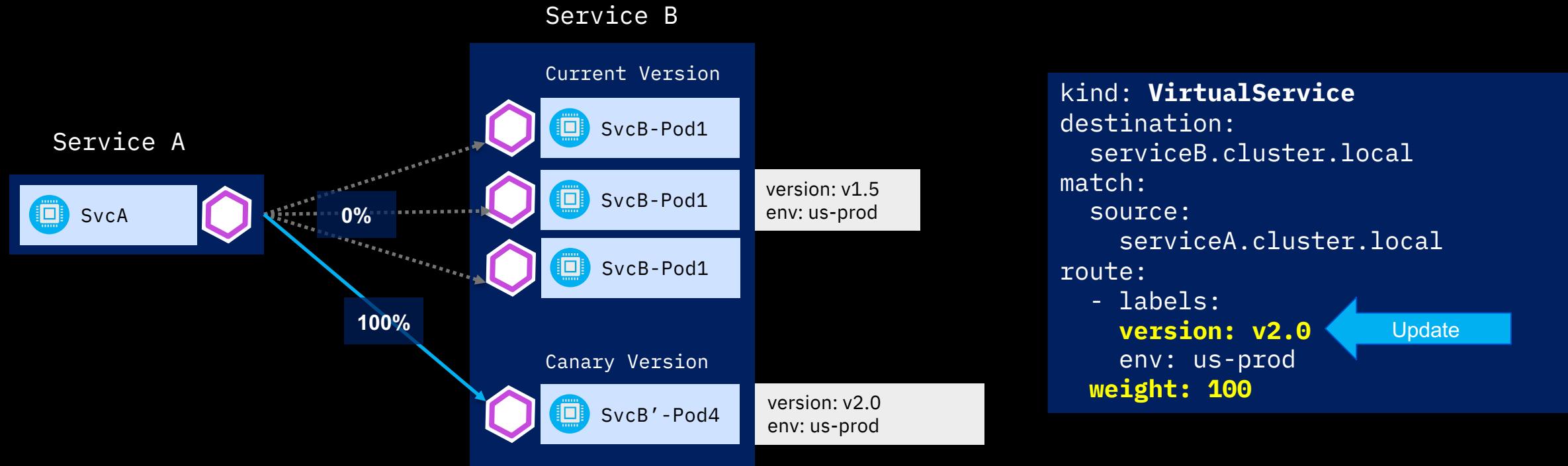
Traffic Control



```
kind: VirtualService
destination:
  serviceB.cluster.local
match:
  source:
    serviceA.cluster.local
route:
  - labels:
      version: v1.5
    env: us-prod
    weight: 100
```

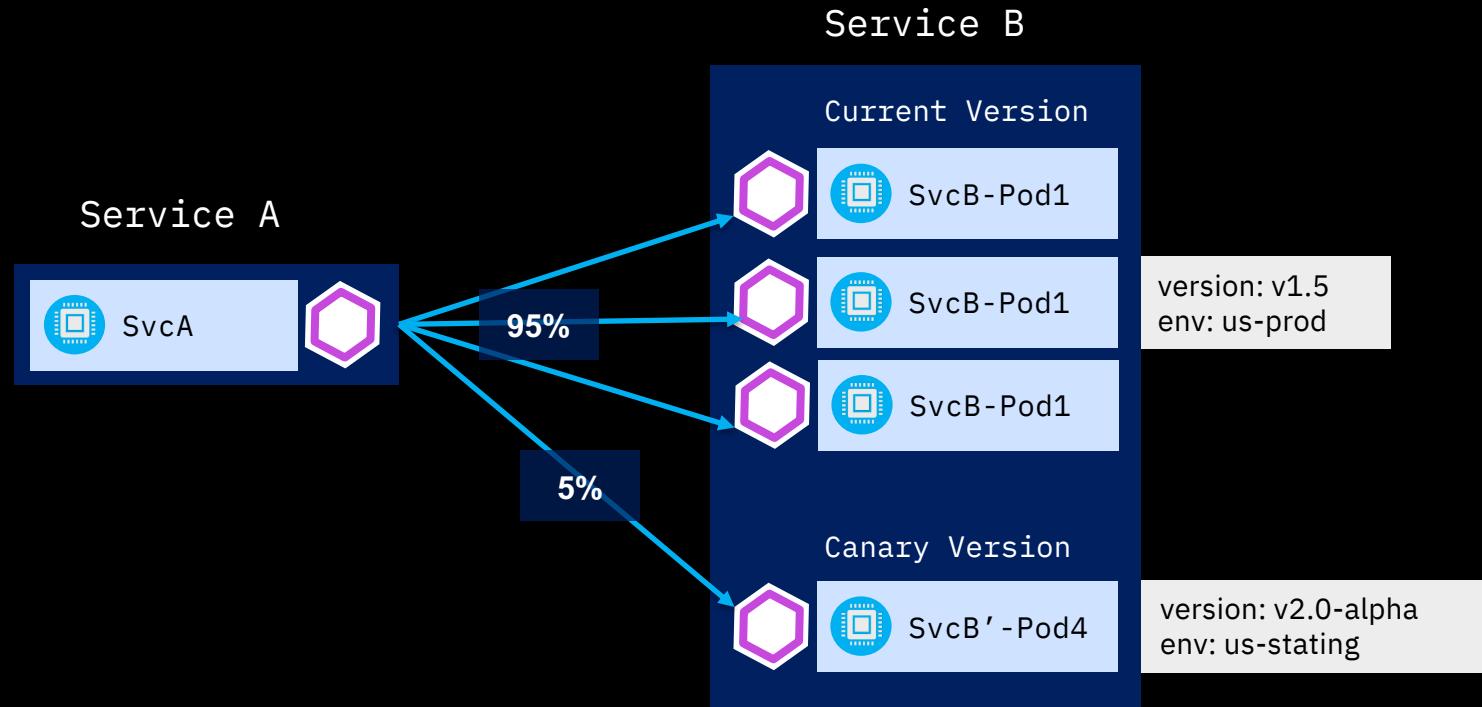
CHALLENGE 1
ROLL OUT NEW VERSION WITHOUT DOWNTIME OR CHANGING CODE

Traffic Control



CHALLENGE 1
ROLL OUT NEW VERSION WITHOUT DOWNTIME OR CHANGING CODE

Traffic Splitting

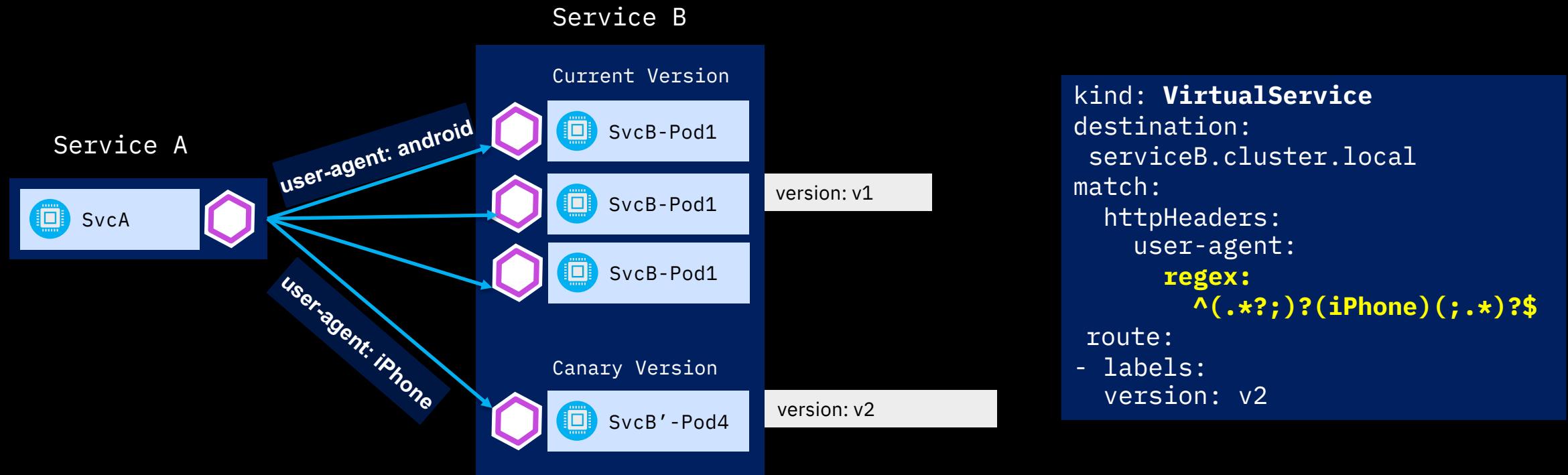


```
kind: VirtualService
destination:
  serviceB.cluster.local
match:
  source:
    serviceA.cluster.local
route:
  - labels:
      version: v1.5
      env: us-prod
  weight: 95
  - labels:
      version: v2.0-alpha
      env: us-staging
  weight: 5
```

CHALLENGE 2 HOW TO DO CANARY TESTING

Routing not based on the request content.
Staged rollouts with %-based traffic splits.

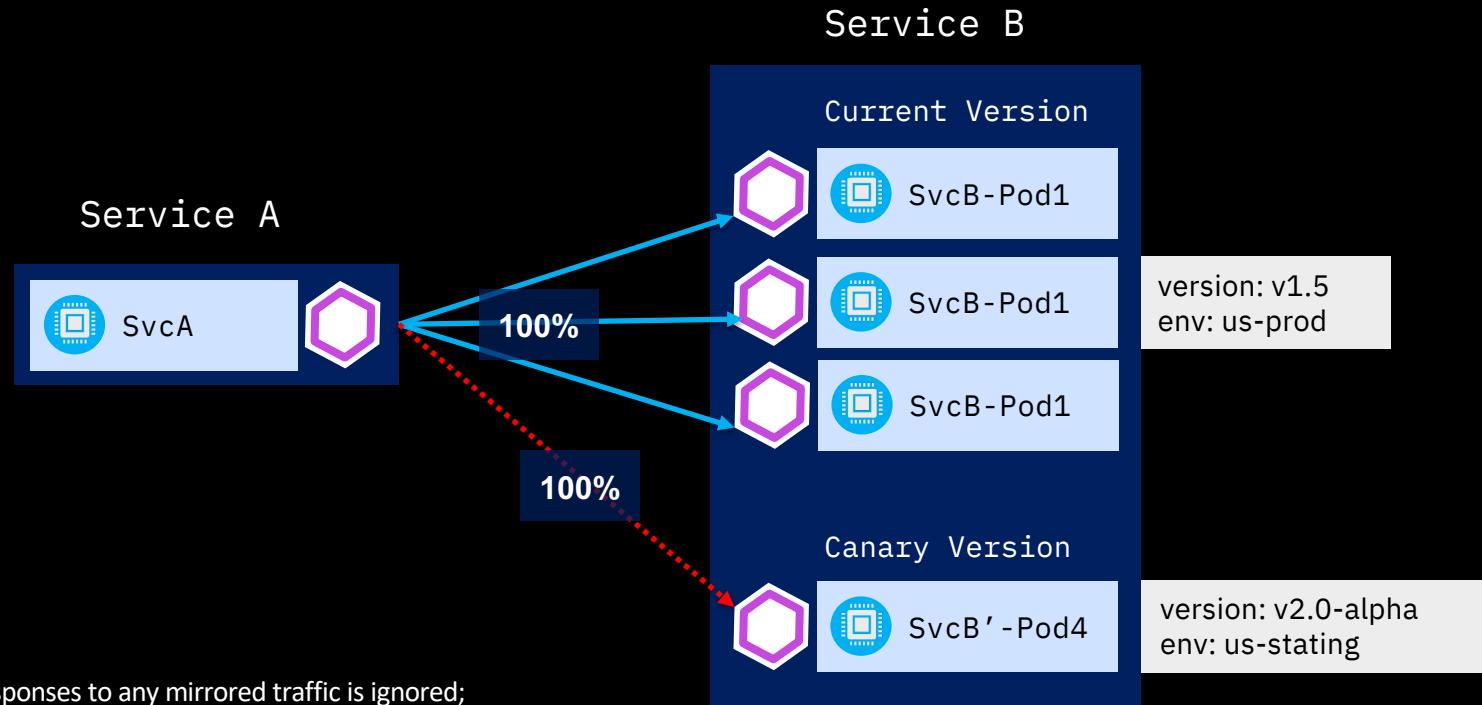
Traffic Steering



Routing based on the request content

CHALLENGE 3
HOW TO DO A/B TESTING

Traffic Mirroring



```
kind: VirtualService
destination:
  serviceB.cluster.local
match:
  source:
    serviceA.cluster.local
route:
  - labels:
      version: v1.5
      env: us-prod
    weight: 100
  - labels:
      version: v2.0-alpha
      env: us-staging
    weight: 0
    mirror:
      name: httpbin
      labels:
        version: v2.0-alpha
        env: us-staging
```

CHALLENGE 4
THINGS DON'T ALWAYS GO CORRECTLY IN PRODUCTION...

Resiliency

Istio adds fault tolerance to your application without any changes to code

```
// Circuit breakers
destination: serviceB.example.cluster.local
policy:
- labels:
  version: v1
circuitBreaker:
  simpleCb:
    maxConnections: 100
    httpMaxRequests: 1000
    httpMaxRequestsPerConnection: 10
    httpConsecutiveErrors: 7
    sleepWindow: 15m
    httpDetectionInterval: 5m
```

Resilience features

- ❖ Timeouts
- ❖ Retries with timeout budget
- ❖ Circuit breakers
- ❖ Health checks
- ❖ AZ-aware load balancing w/ automatic failover
- ❖ Control connection pool size and request load

CHALLENGE 4
THINGS DON'T ALWAYS GO CORRECTLY IN PRODUCTION...

Rate limiting

Istio protects your application from rogue actors by imposing ratelimits

Quotas:

```
- name: requestcount.quota.istio-system
  maxAmount: 5000
  validDuration: 1s
  overrides:
    - dimensions:
        destination: ratings
        source: reviews
        sourceVersion: v3
        maxAmount: 1
        validDuration: 1s
    - dimensions:
        destination: ratings
        maxAmount: 100
        validDuration: 1s
```

Rate limit

- ❖ Configurable limits with overrides
- ❖ Multiple rate limiting backends
- ❖ Conditional rate limiting

CHALLENGE 5
HOW CAN I LIMIT RATE FOR SOME OF MY SERVICES?

Telemetry

Monitoring & tracing should not be an afterthought in the infrastructure

Goals

- Metrics without instrumenting apps
- Consistent metrics across fleet
- Trace flow of requests across services
- Portable across metric backend providers



CHALLENGE 6
I NEED TO VIEW WHAT IS GOING ON WHEN CRISIS ARISES

Kiali

Kiali (greek κιάλι)
monocular or spyglass

Visualise the service mesh topology, features like circuit breakers or request rates

Features

Graph

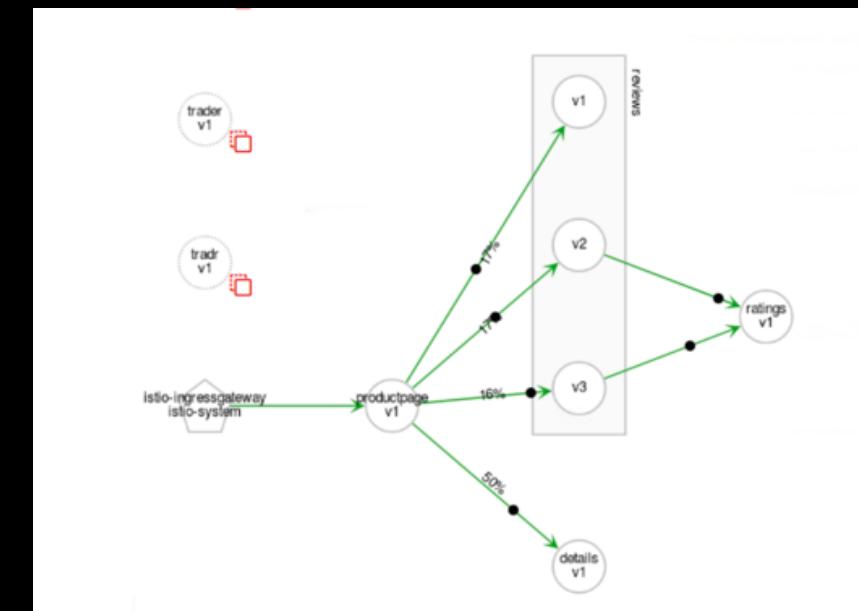
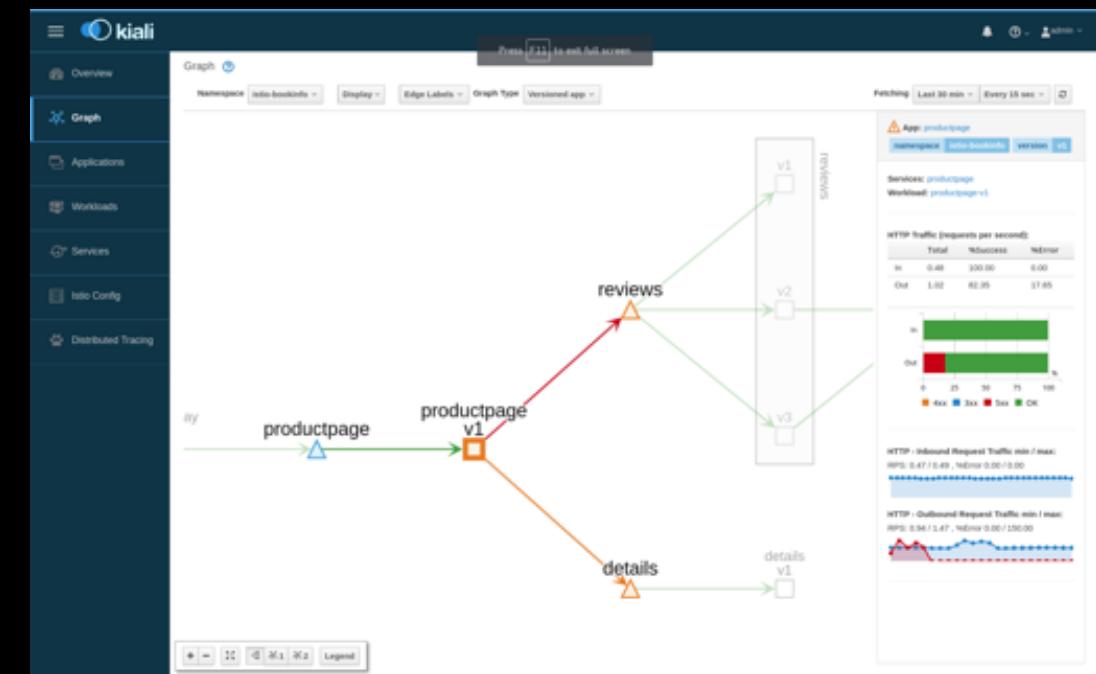
- Health
- Types
- Side Panel
- Traffic Animation

Applications, Workloads and Services

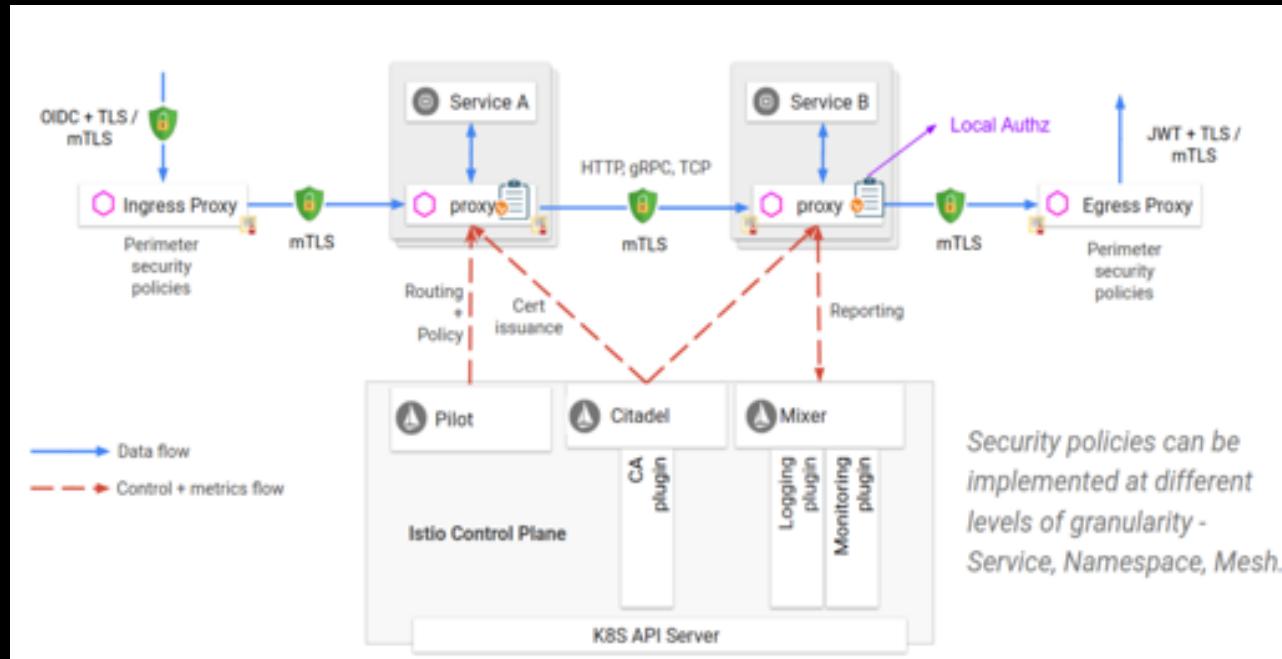
- Detailed Metrics
- Traffic Routing
- Istio compliance
- Istio Configuration

CHALLENGE 6

I NEED TO VIEW WHAT IS GOING ON WHEN CRISIS ARISES



Security



Authentication

Transport authentication, also known as service-to-service authentication
Origin authentication, also known as end-user authentication

Authorization

Based on RBAC
Namespace-level, service-level and method-level access control for services

CHALLENGE 7
HOW CAN I SECURE MY SERVICES?

Service Mesh - Bad Idea ?

A Service Mesh is not always the right solution...

- ▶ **Service Meshes are Opinionated**

They are a *platform* solution. “Work their way”

- ▶ **Service Meshes are Complex**

Adds considerable complexity with sidecars and control plane

- ▶ **Service Meshes can be Slow**

Routing traffic through a series of proxies can get painfully slow (about 700 nodes → reflector)

- ▶ **Service Meshes are for Developers**

Focused primarily on Developer view.

Getting started

- ▶ Go to <https://istio.io/>

Download ISTIO Release

With Kubectl

```
$ kubectl apply -f install/kubernetes/helm/istio/templates/crds.yaml
```

```
$ kubectl apply -f install/kubernetes/istio-demo.yaml
```

With HELM Templating

```
$ kubectl create namespace istio-system
```

```
$ helm template --name istio  
  --namespace istio-system  
  --set grafana.enabled=true  
  --set servicegraph.enabled=true  
  --set kiali.enabled=true > istio.yaml
```

```
$ kubectl apply -f istio.yaml
```

Getting started

- ▶ IBMs ISTIO 101 Hands On
- ▶ Go to <https://github.com/IBM/istio101>

Exercise 3 - Deploy the Guestbook app with Istio Proxy

The Guestbook app is a sample app for users to leave comments. It consists of a web front end, Redis master for storage, and replicated set of Redis slaves. We will also integrate the app with Watson Tone Analyzer that detects the sentiment in user's comments and replies with emoticons. Here are the steps to deploy the app on your Kubernetes cluster:

Download the Guestbook app

1. Open your preferred terminal and download the Guestbook app from GitHub.

```
git clone https://github.com/IBM/guestbook.git
```

2. Navigate into the app directory.

```
cd guestbook/v2
```

Create a Redis database

The Redis database is a service that you can use to persist the data of your app. The Redis database comes with a master and slave modules.

1. Create the Redis controllers and services for both the master and the slave.

```
kubectl create -f redis-master-deployment.yaml  
kubectl create -f redis-master-service.yaml  
kubectl create -f redis-slave-deployment.yaml  
kubectl create -f redis-slave-service.yaml
```

Useful links

- Web istio.io
- Twitter: [@Istiomesh](https://twitter.com/Istiomesh)
- Istio 101: <https://github.com/IBM/istio101>
- Traffic management using Istio: <https://ibm.co/2F7xSnf>
- Resiliency and fault-tolerance using Istio:
<https://bit.ly/2qStF2B>
- Reliable application roll out and operations using Istio:
<https://bit.ly/2K9IRQX>

QUESTIONS?





°° The Journey to Cloud
ISTIO - Hands-On

02



IBM Cloud

Remember your Team Color

black 31701

olive 31711

peru 31715

white 31702

brown 31712

chocolate 31716

red 31703

lightblue 31713

orchid 31717

blue 31704

orange 31708

gold 31718

yellow 31705

purple 31709

pink 31719

lime 31706

maroon 31710

violet 31720

cyan 31707

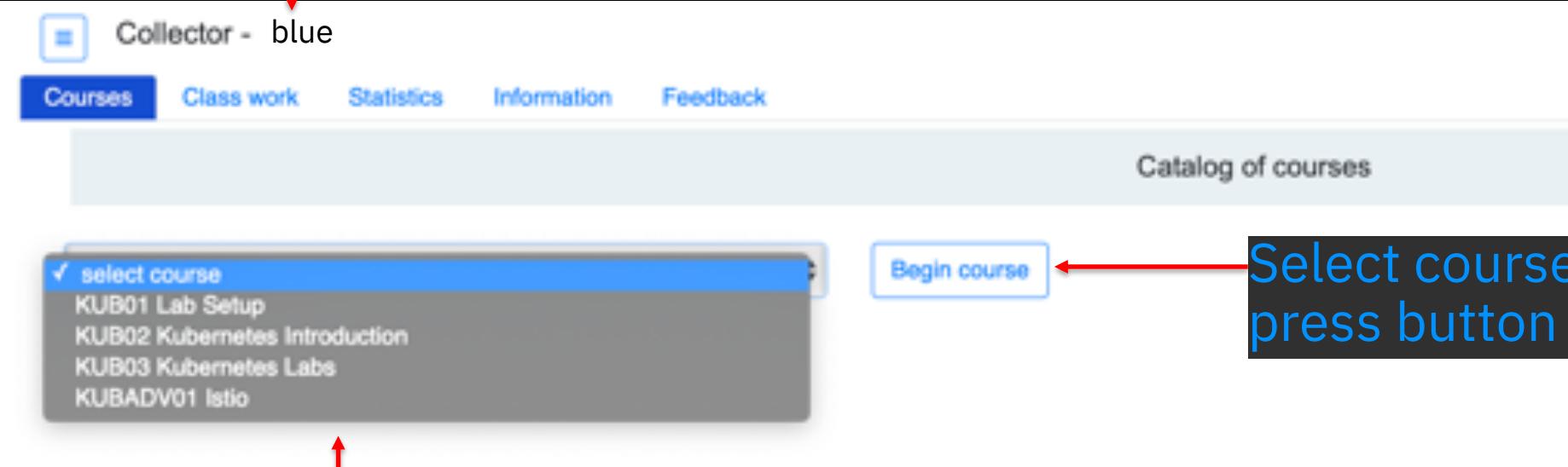
firebrick 31714

Collector - Accessing team web site

`http://158.177.137.195:{port#}`

Team name / color will be shown

blue **31704**



The screenshot shows a web browser window with the title "Collector - blue". The navigation bar includes links for "Courses", "Class work", "Statistics", "Information", and "Feedback". Below the navigation bar, a section titled "Catalog of courses" displays a list of courses under the heading "select course". The listed courses are: KUB01 Lab Setup, KUB02 Kubernetes Introduction, KUB03 Kubernetes Labs, and KUBADV01 Istio. To the right of the course list is a button labeled "Begin course". A red arrow points from the text "Select course and press button to begin" to the "Begin course" button.

Current course catalog



JTC11 Istio

Lab 0 : Introduction

Lab 1 - Make sure minikube is running

Lab 2 - Installing Istio

Lab 3 - Deploy the Bookinfo App

Lab 4 - Monitoring with Kiali

Lab 5 - Traffic flow management

Lab 6 - Access policy enforcement

Lab 7 - Telemetry data aggregation



READY
SET
GO!!!!

<http://158.177.137.195:{port#}>

Duration: 60 mins

QUESTIONS?



The Journey to Cloud **Kubernetes Best Practices**

03



IBM Cloud

Developer Productivity

Developers want Kubernetes

And once they get it

They hate it

Serverless paradigm

Source-to-image

Simply provide code to the platform and the platform manages all of the hosting aspects (e.g., building, hosting, scaling, etc.) for them.

Auto-scaling/scale-to-zero

Scales the application based on the load it is experiencing. Including scaling the application down to zero instances when it is not in use.

Short-lived functions

Splitting up the microservices into even smaller “functions” allows for a more fine-grained hosting model, meaning better resource utilization.

Event-driven

Optimized scaling by responding to events rather than simply always running and waiting for something to happen. This allows for a much more loosely-coupled architecture.

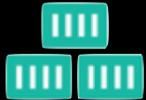


**A platform for developers
to
build and run
serverless applications
atop Kubernetes**

Open source project being developed by some of the key cloud innovators,
including IBM, RedHat, Google, Pivotal, and SAP

Knative - building blocks

Knative solves these concerns through its three main components:



Build: Integrate building of container images into the configuration of how to encapsulate the configuration of their application. **DEPRECATED in 0.8**  application can be built all at once.

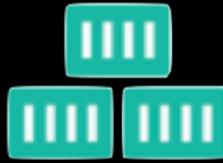


Serving: **Event-driven** hosting scheme to ensure that the **applications are scaled** based on actual need, including **scaling down to zero** when appropriate. Also automatically **manages the rolling-out** of newer versions of the code and allow for advanced traffic routing (such as A/B testing), relying on Istio.



Eventing: **Core eventing primitives** to allow for the specification of interest in events from event sources (both internal and external to the cluster), as well as simple orchestration.

Tekton Pipelines



Tekton Pipelines are **Cloud Native**

- Run on Kubernetes
- Have Kubernetes clusters as a first class type
- Use containers as their building blocks

Tekton Pipelines are **Decoupled**

- One Pipeline can be used to deploy to any k8s cluster
- The Tasks which make up a Pipeline can easily be run in isolation
- Resources such as git repos can easily be swapped between runs

Tekton Pipelines are **Typed**

- The concept of typed resources means that for a resource such as an Image, implementations can easily be swapped out (e.g. building with kaniko v.s. buildkit)

Tekton Pipelines

```
apiVersion: tekton.dev/v1alpha1
kind: Pipeline
metadata:
  name: demo-pipeline
spec:
  tasks:
    - name: build-skaffold-web
      taskRef:
        name: build-docker-image-from-git-source
      resources:
        inputs:
          - name: docker-source
            resource: myproject-git
    - name: deploy-web
      taskRef:
        name: deploy-using-kubectl
...
...
```



```
apiVersion: tekton.dev/v1alpha1
kind: Task
metadata:
  name: echo-hello-world
spec:
  inputs:
  resources:
    - name: docker-source
      type: git
  steps:
    - name: echo
      image: ubuntu
      command:
        - echo
      args:
        - "hello world"
```

```
apiVersion: tekton.dev/v1alpha1
kind: PipelineResource
metadata:
  name: myproject-git
spec:
  type: git
  params:
    - name: revision
      value: master
    - name: url
      value: https://github.com/nick/myproject
```

Knative – Serving

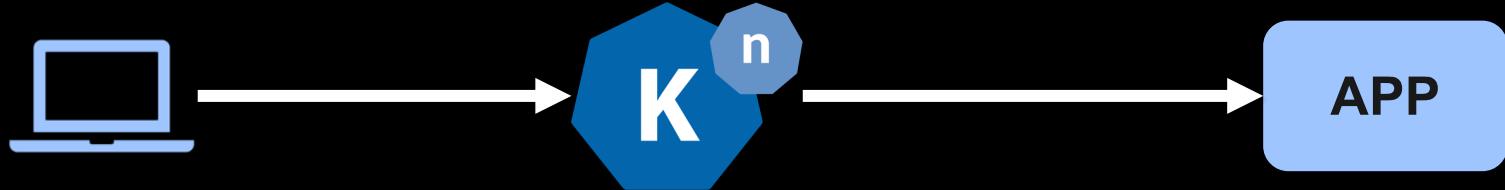


The Knative Serving project provides middleware primitives that enable:

- Automatically deploy containers and configure routing
- Automatically scale up and down, including scale-to-zero
- Point-in-time snapshots of deployments allows multiple versions of applications at once
- Easy rollbacks, blue-green deployments, partial load testing, etc.

Knative – Serving

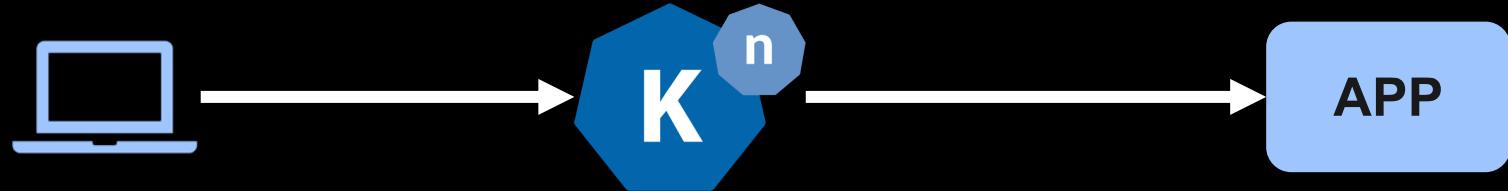
Deployment



```
apiVersion: serving.knative.dev/v1alpha1
kind: Service
metadata:
  name: knative-helloworld
  namespace: default
spec:
  runLatest:
    configuration:
      revisionTemplate:
        spec:
          container:
            image: docker.io/gswk/knative-helloworld:latest
```

Knative – Serving

Deployment



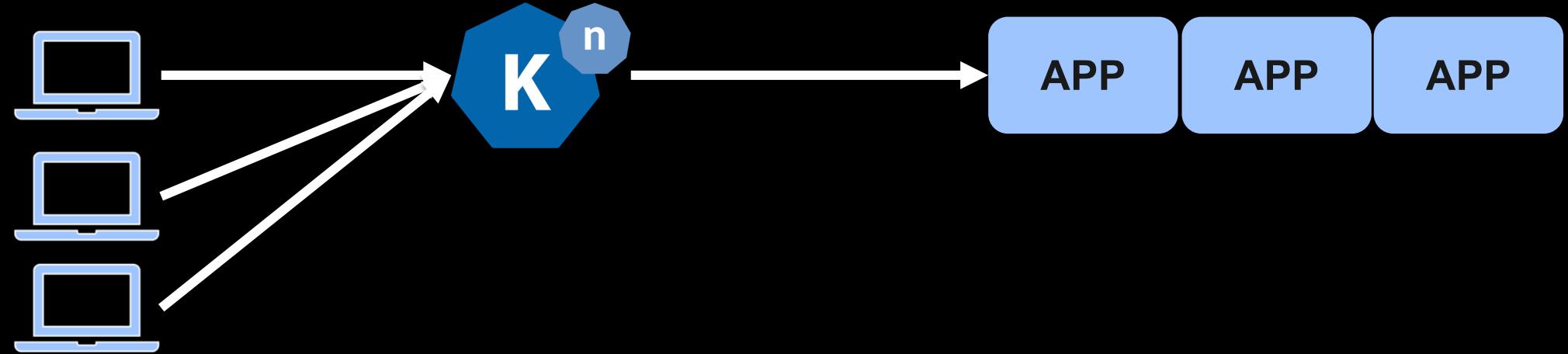
App reachable at:

{app-name}.{namespace}.{custom-domain}

knative-helloworld.default.example.com

Knative – Serving

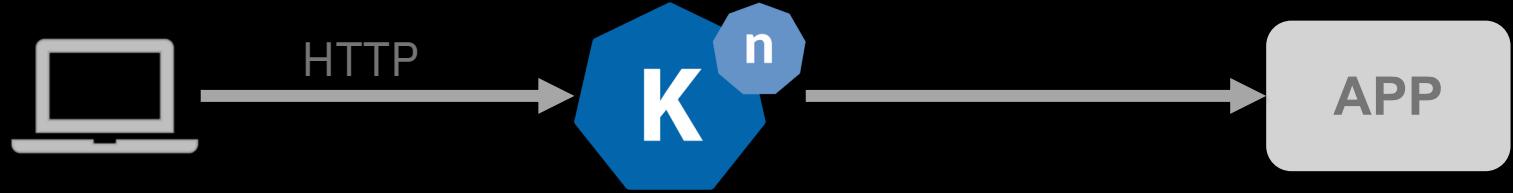
Scale up





Knative – Serving

Scale down



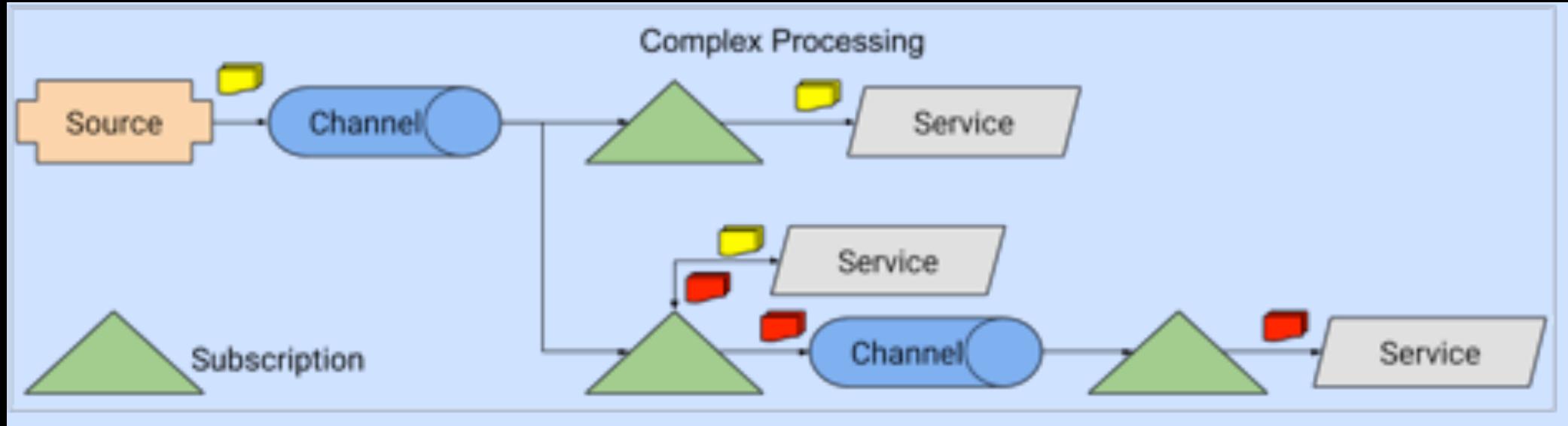
Knative – Eventing



Designed to address a common need for cloud native development, providing composable primitives to enable late-binding event sources and event consumers.

- Services are loosely coupled and can be developed and deployed independently on, and across a variety of platforms (for example Kubernetes, VMs, SaaS or FaaS).
- Event producers and event sources are independent.
- Other services can be connected to the Eventing system.
- Ensure cross-service interoperability.

Knative – Eventing



Some Event Sources

Kubernetes	Brings Kubernetes cluster events into Knative.
GitHub	Registers for events of the specified types on the specified GitHub organization/repository
Cron Job	Uses an in-memory timer to produce events on the specified Cron schedule.
....	



ArgoCD

A tool for developers
for
declarative, GitOps continuous delivery
atop Kubernetes

GitOps

At its core, GitOps refers to
a set of practices and tooling
that put
Git at the center of the DevOps toolchain
and as the
source of truth
for what should be deployed on the cluster.

With GitOps, developers and operators use familiar Git workflows to define, review, approve, and audit changes to their infrastructure and applications, whereas automated tools take care of synchronizing the live state of their cluster with the desired state described in Git.

ArgoCD

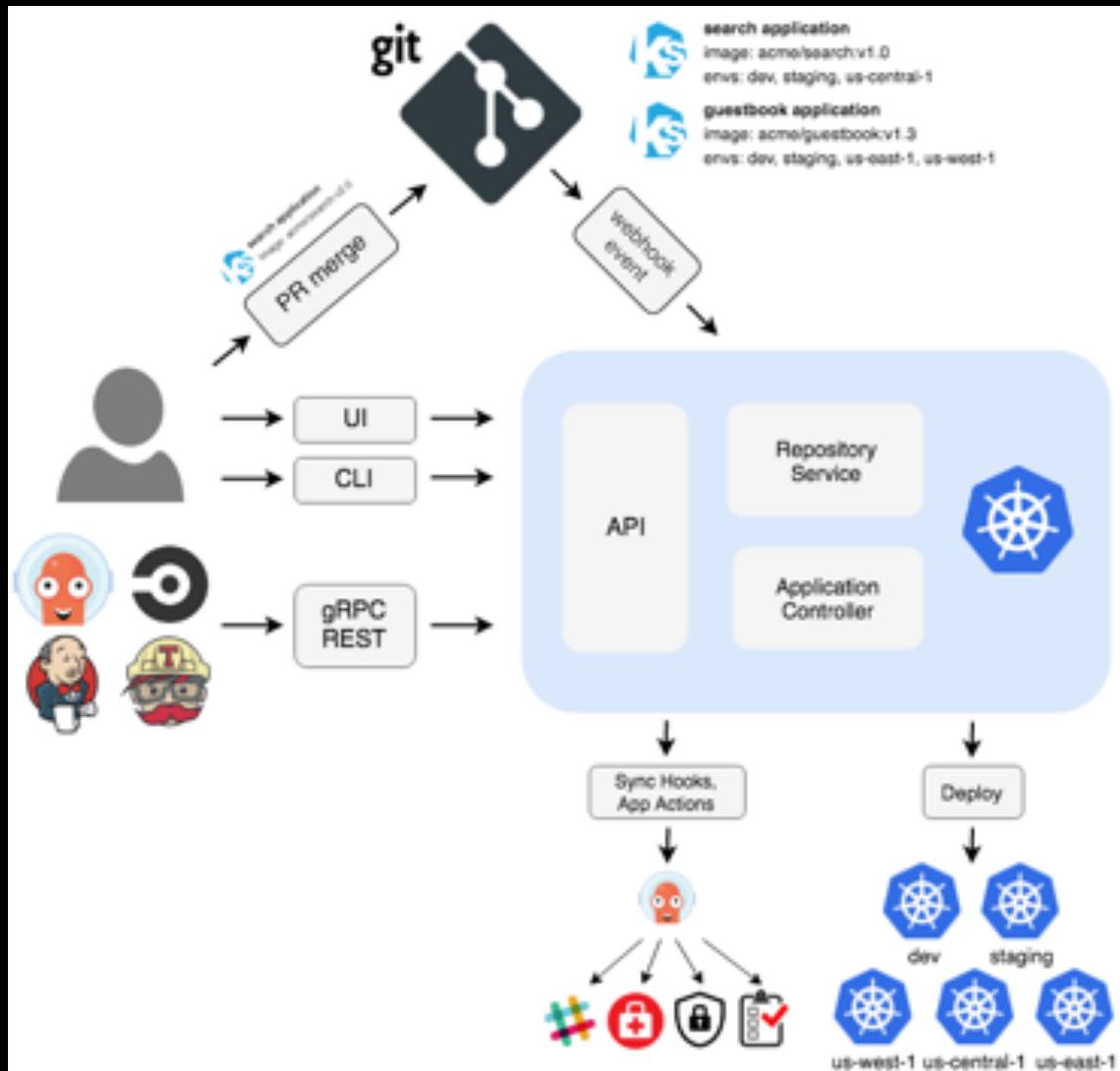
Argo CD is a declarative, GitOps continuous delivery tool for Kubernetes.

Application definitions, configurations, and environments should be declarative and version controlled. Application deployment and lifecycle management should be automated, auditable, and easy to understand.

- kustomize applications
- helm charts
- ksonnet applications
- jsonnet files
- Plain directory of YAML/json manifests

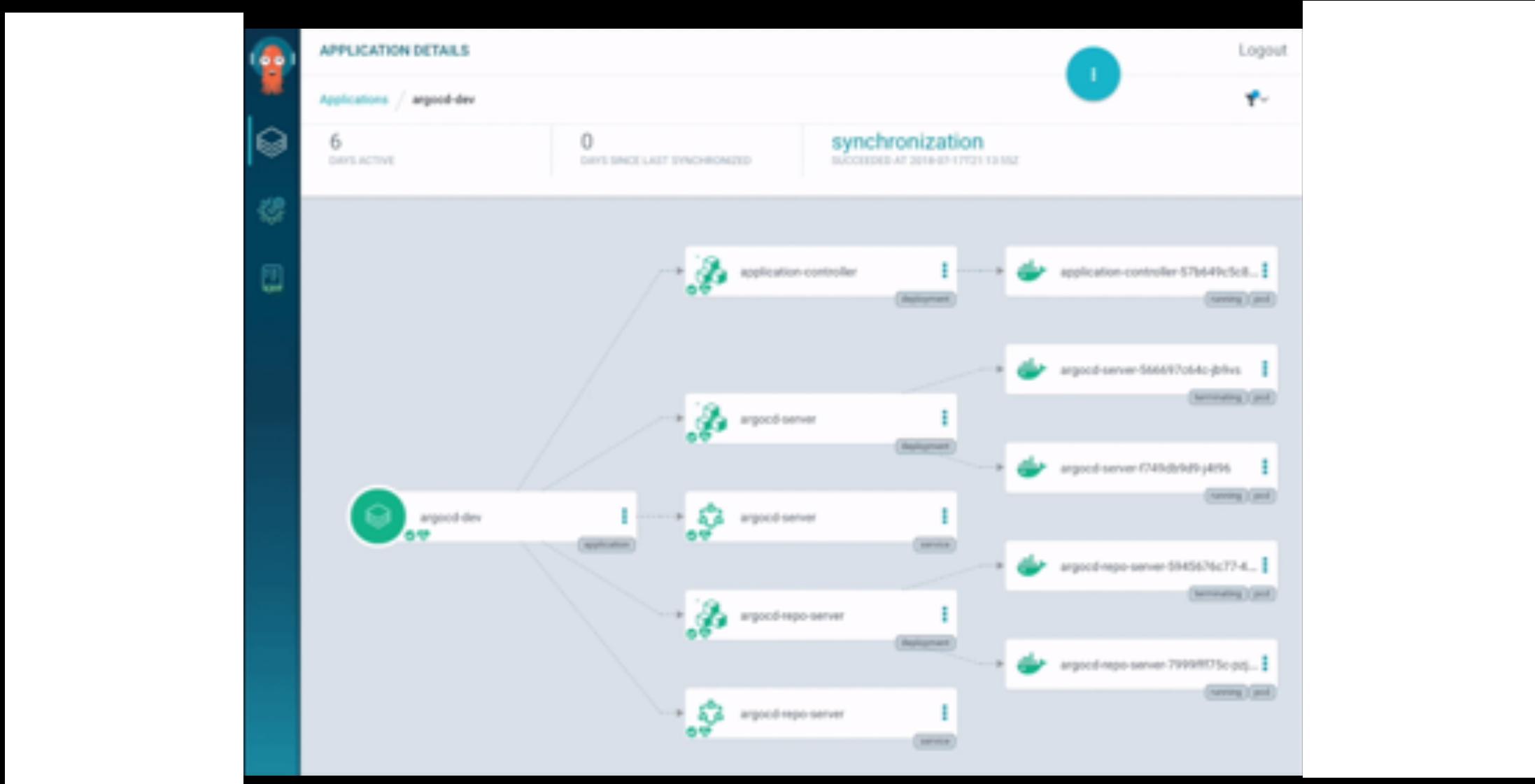
ArgoCD

Argo CD is a declarative, GitOps continuous delivery tool for Kubernetes.



ArgoCD

Argo CD is a declarative, GitOps continuous delivery tool for Kubernetes.



Security

Kubernetes – Tip 1

Use **Namespaces** Liberally

“Namespaces are cheap.”

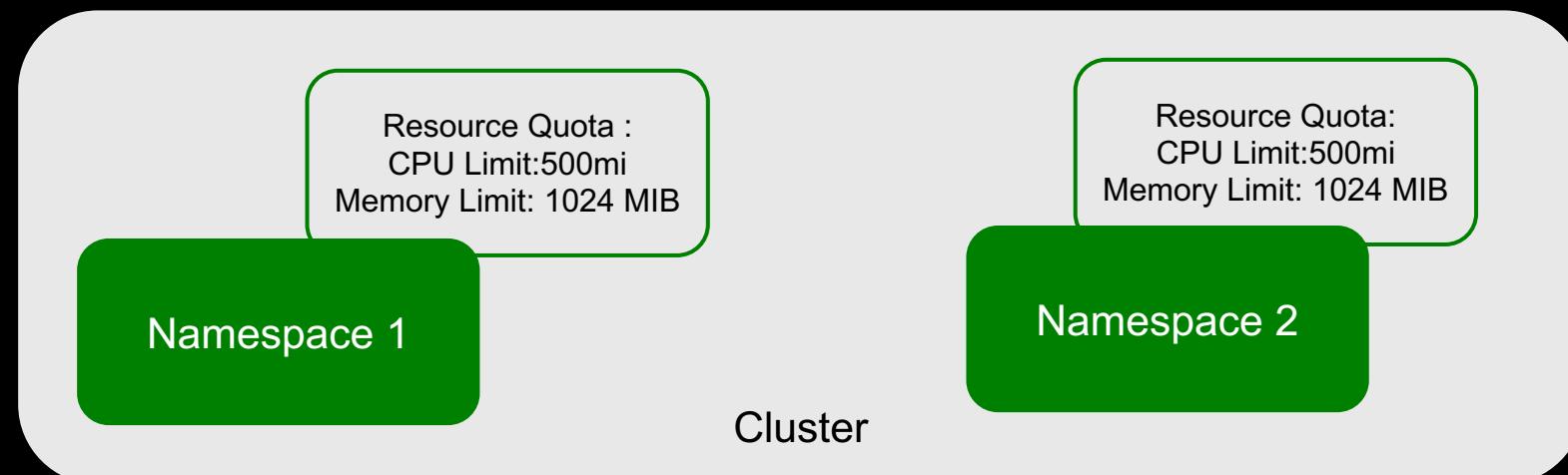
Use them to separate things like infrastructure tooling and applications.

This allows you to restrict access easily using **RBAC** and to limit the scope of applications. It will also make your **network policy** creation easier when you decide to do it.

Kubernetes – Tip 2

Set resource quota

Once quota in a namespace for compute resources set, the users are forced to set requests or limits for those values. Avoids starving of Cluster.



Kubernetes – Tip 3

Scan your containers for vulnerabilities

Known vulnerabilities account for a large portion of breaches. Use a tool to scan your containers for them and then mitigate them.

[Anchore](#), [Clair](#), and [Quay](#) are among my favorite tools, but there are others out there.



Kubernetes – Tip 4

Occam's razor (also **Ockham's razor** or Ocham's **razor**: Latin: novacula Occami; or law of parsimony: Latin: lex parsimoniae) is the problem-solving principle that states:

“Entities should not be multiplied without necessity.”

Or

“other things being equal, simpler explanations are generally better than more complex ones””

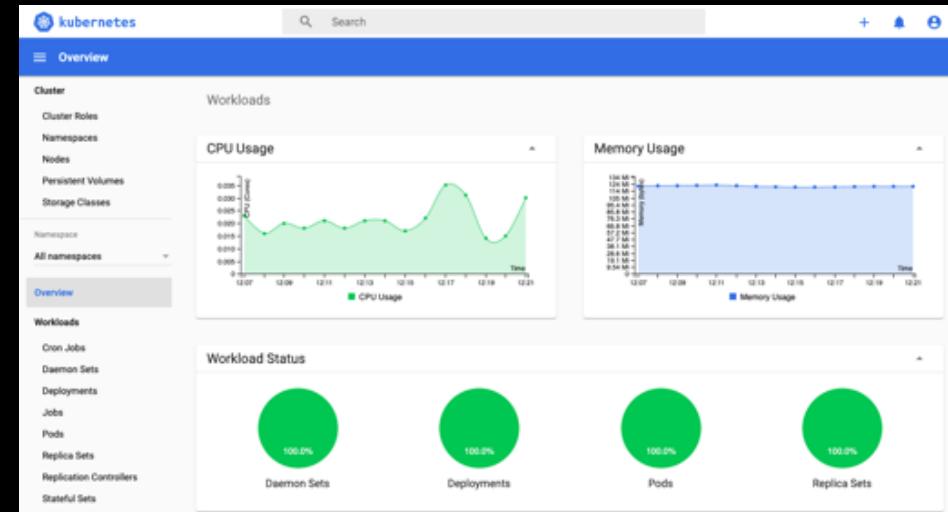
Kubernetes – Tip 5

Secure Kubernetes Dashboard

The Kubernetes Dashboard has often been used for attacks.
Easy to deploy with high privileges.

Best practice is to:

- Allow only authenticated access
- Use RBAC
- Ensure Dashboard service account has limited access
- Don't expose to the internet (use kubectl proxy for local access)



Kubernetes – Tip 6

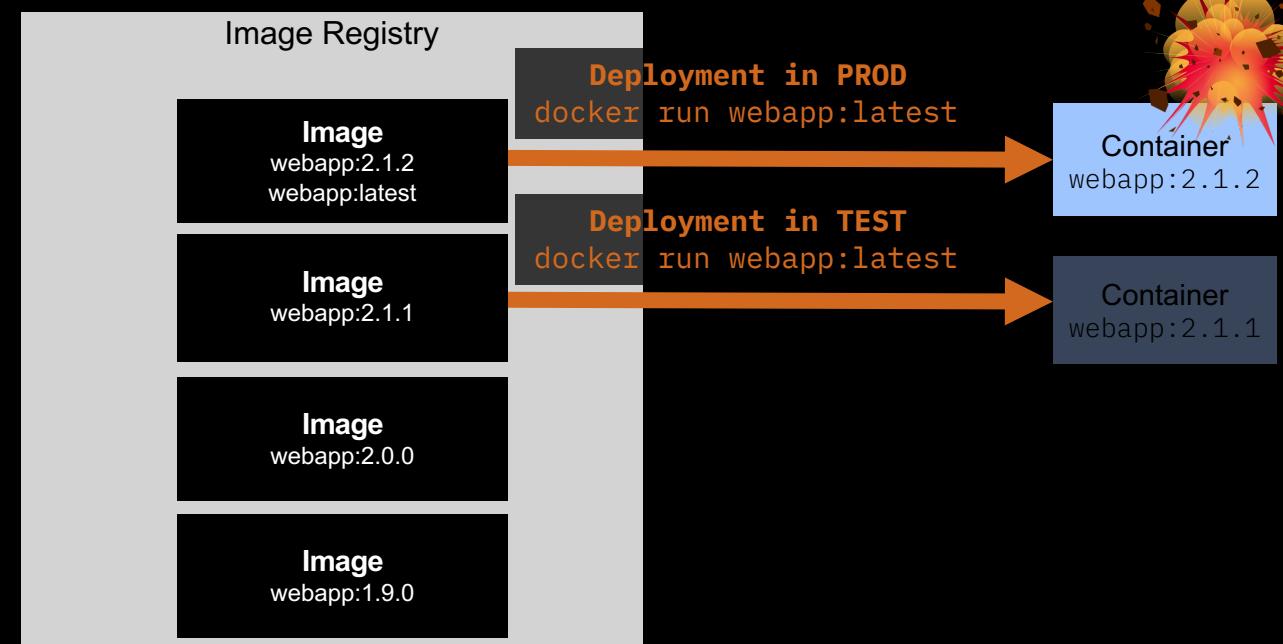
Avoid **Rolling Tags** at Any Cost

Avoid commands like `docker pull myregistry/myimage:latest`.

This “latest” is an example of a rolling tag (i.e. a tag that will point to different images over time).

If you want your deployments to be secure and maintainable, make sure that you use immutable images (for example: “`myregistry/myimage:1.1.2`”).

With this approach, every time you deploy or scale, you know what image you are using and you will have the guarantee that the deployed image has been tested and validated.



Kubernetes – Tip 7

Don't run **multiple processes per container**

Docker general best practices suggest a single process per container simply for usability and size reasons but it also **keeps your attack surface small** and limits the number of potential vulnerabilities.

Kubernetes – Tip 8

Don't run containers as the root user

By default, containers run as the root user inside the container. This is easy to avoid using the pod specification to set a high-numbered UID.

```
spec:  
  securityContext:  
    runAsUser: 10324
```

Kubernetes – Tip 9

Don't run containers as privileged

Running a container or pod as privileged gives it the ability to make modifications to the host. This is a large security issue that is easy to mitigate by just not doing it.

Kubernetes – Tip 10

Don't use the default list of capabilities

Docker runs containers with a significant set of Linux capabilities by default, many of which your app might not require.

The following config will drop all Linux capabilities by default, allowing you to add only the specific capabilities your app actually needs:

```
spec:  
  containers:  
    - name: foo  
      securityContext:  
        capabilities:  
          drop:  
            - ALL
```

Kubernetes – Tip 11

Disable **Anonymous access**

By default, requests to the K8s API and kubelet's HTTPS endpoint that are not rejected by other configured authentication methods are treated as anonymous requests, and given a username of `system:anonymous` and a group of `system:unauthenticated`.

Anonymous authentication provides full access to the *kubelet* API, the only requirement being network access to the service.

To disable anonymous access pass `--anonymous-auth=false` to the API server and start the kubelet with the `--anonymous-auth=false` flag

If RBAC is enabled this risk is mitigated.

Kubernetes – Tip 12

Secure **kubelet**

Set `--anonymous-auth=false`

Set `--authorization-mode` to something other than `AlwaysAllow`

Set `--read-only-port=0`

Kubernetes – Tip 13

Secure **etcd**

Set --cert-file and --key-file to enable HTTPS connections

Set --client-cert-auth=true

Kubernetes – Tip 14

Secure API

The Kubernetes API server can serve requests on two ports:

- localhost port (8080) and
- secure port

The localhost port is intended for testing purposes and for other master components to talk to the API. Requests to the Localhost port **bypass authentication and authorization modules**.

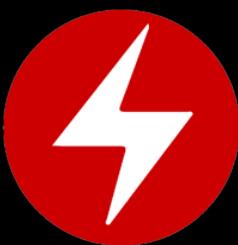
Best practice is to set `--insecure-port flag=0` and remove the `--insecure-bind-address` flag from the API server manifest.

And remove the `--secure-port` flag from the API server spec to ensure that all requests to the secure port are authenticated and authorized.

Note: Has been deprecated in Kubernetes 1.10 and should be removed in the future

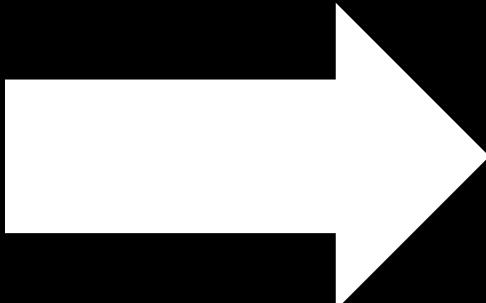
Operators

Kubernetes Operators



From this...

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: k8sdemo
  namespace: default
spec:
  replicas: 1
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxUnavailable: "30%"
      maxSurge: "90%"
  template:
    metadata:
      labels:
        app: k8sdemo
        version: current
    spec:
      containers:
        - name: k8sdemo
          image: niklaushirt/k8sdemo:1.0.0
          imagePullPolicy: Always
          livenessProbe:
            httpGet:
              path: sudo chmod -R 777 /home/training/.kube
              port: kubelet get nodes
              initialDelaySeconds: 10
              periodSeconds: 5
            readinessProbe:
              httpGet:
                path: sudo chmod -R 777 /home/training/.minikube
                port: kubelet get nodes
                initialDelaySeconds: 10
                periodSeconds: 5
              ports:
                - containerPort: 3000
              resources:
```



... to this

```
apiVersion: demo.ibm.com/v1beta1
kind: MyDemoBackend
metadata:
  name: example-mydemobackend
spec:
  # Add fields here
  size: 3
  label: backend
  image: niklaushirt/k8sdemo-backend:1.0.0
  message: "Hello from the Operator Lab..."
```

Kubernetes Operators

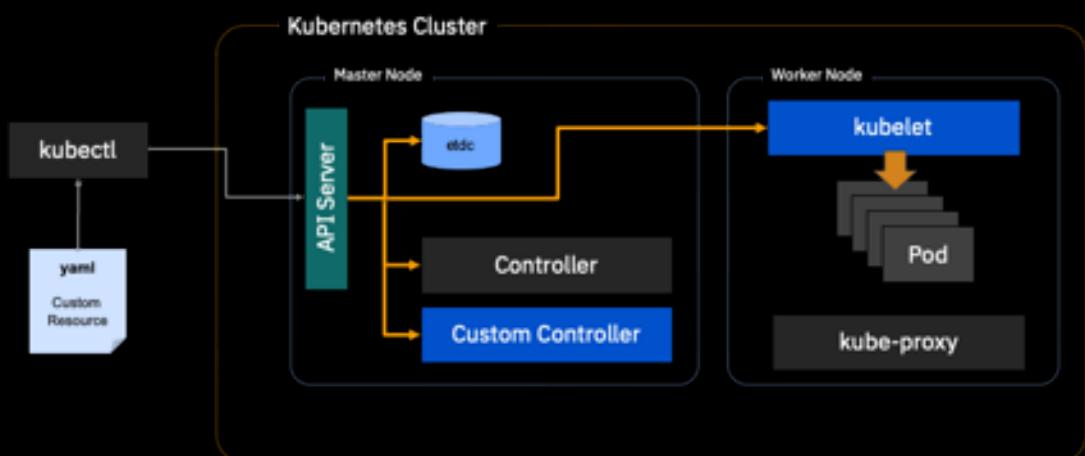


Operators are a **design pattern** made public in a 2016 CoreOS [blog](#) post.

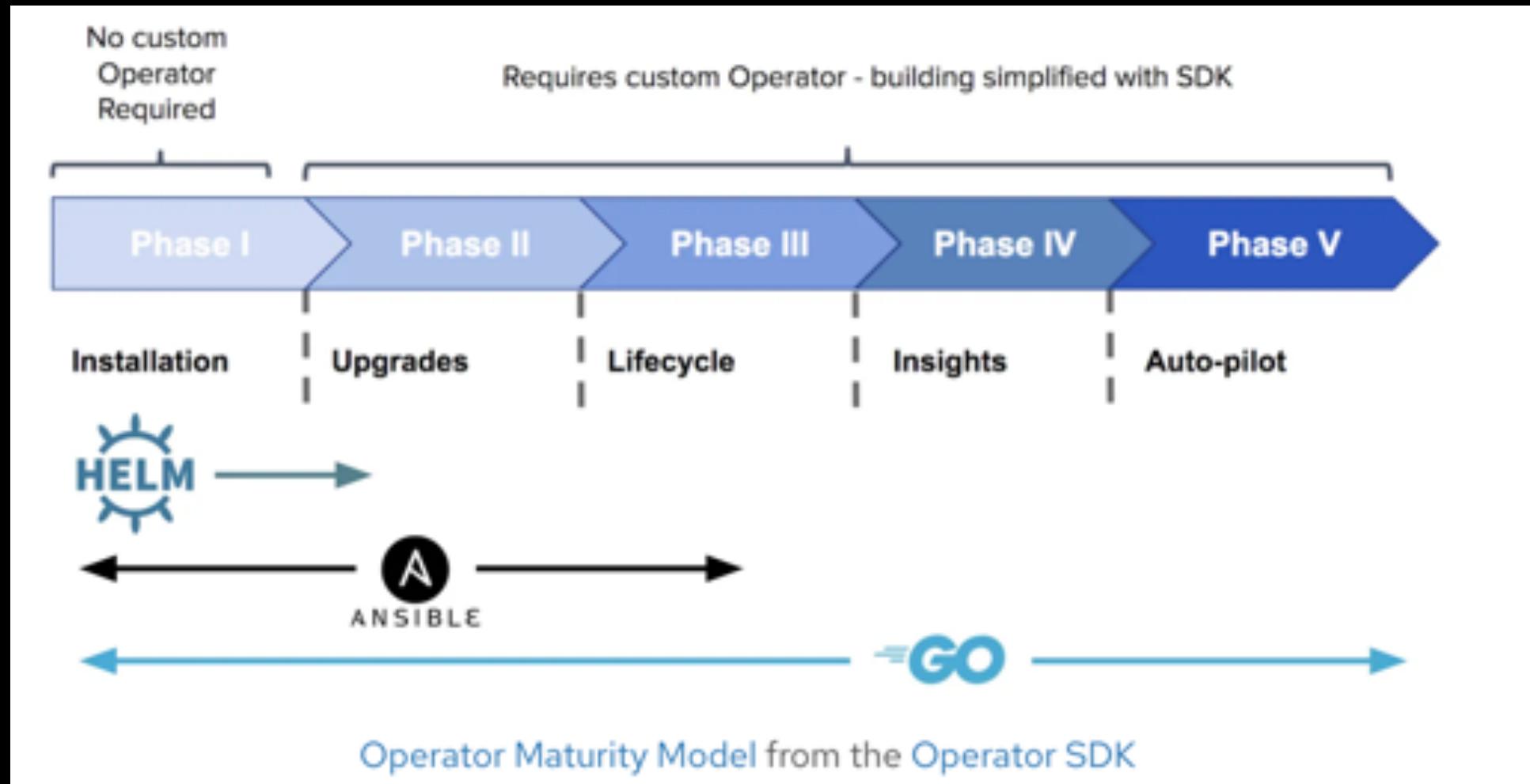
The goal of an Operator is to put **operational knowledge into software**.

Previously this knowledge only resided in the minds of administrators, various combinations of shell scripts or automation software like Ansible. It was outside of your Kubernetes cluster and hard to integrate.

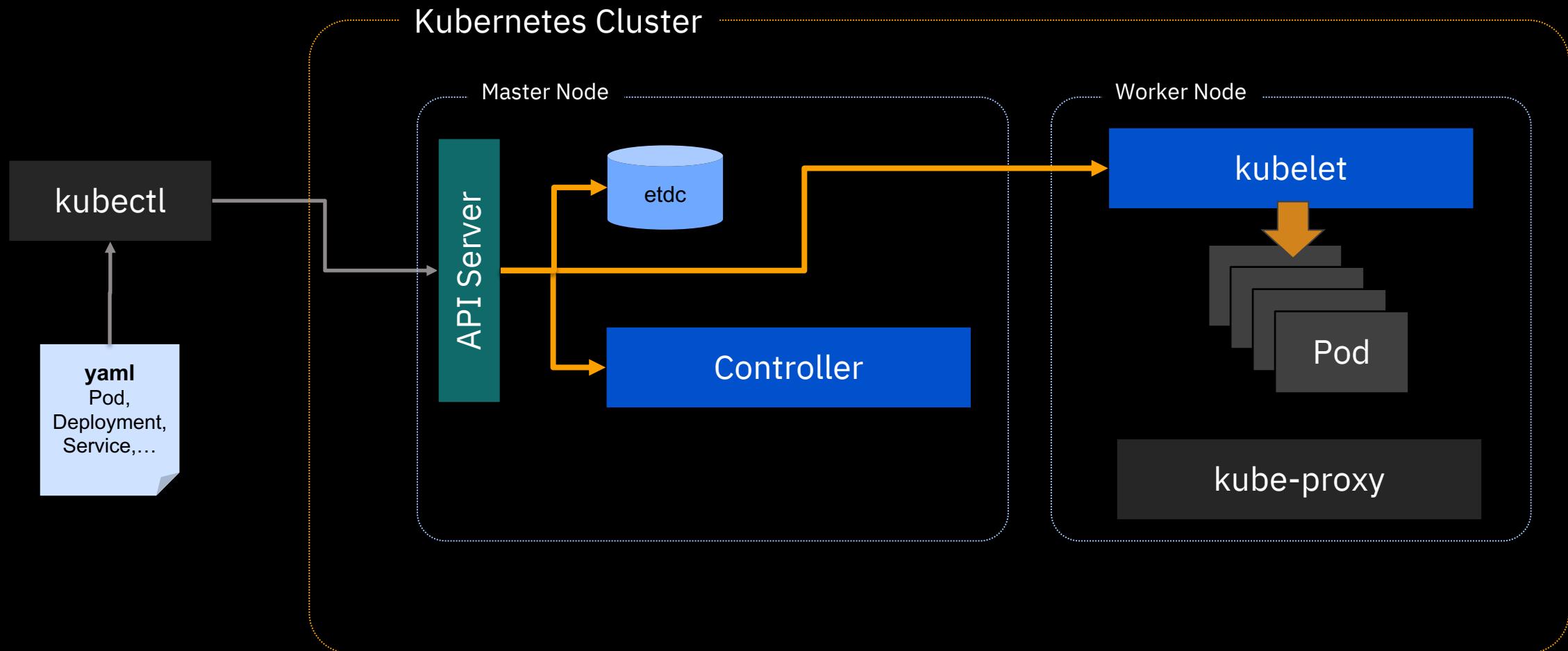
Operators implement and automate common **Day-1** (installation, configuration, etc) and **Day-2** (re-configuration, update, backup, failover, restore, etc.) **activities** in a piece of software running inside your Kubernetes cluster, by integrating natively with Kubernetes concepts and APIs.



Kubernetes Operators



Kubernetes Operators

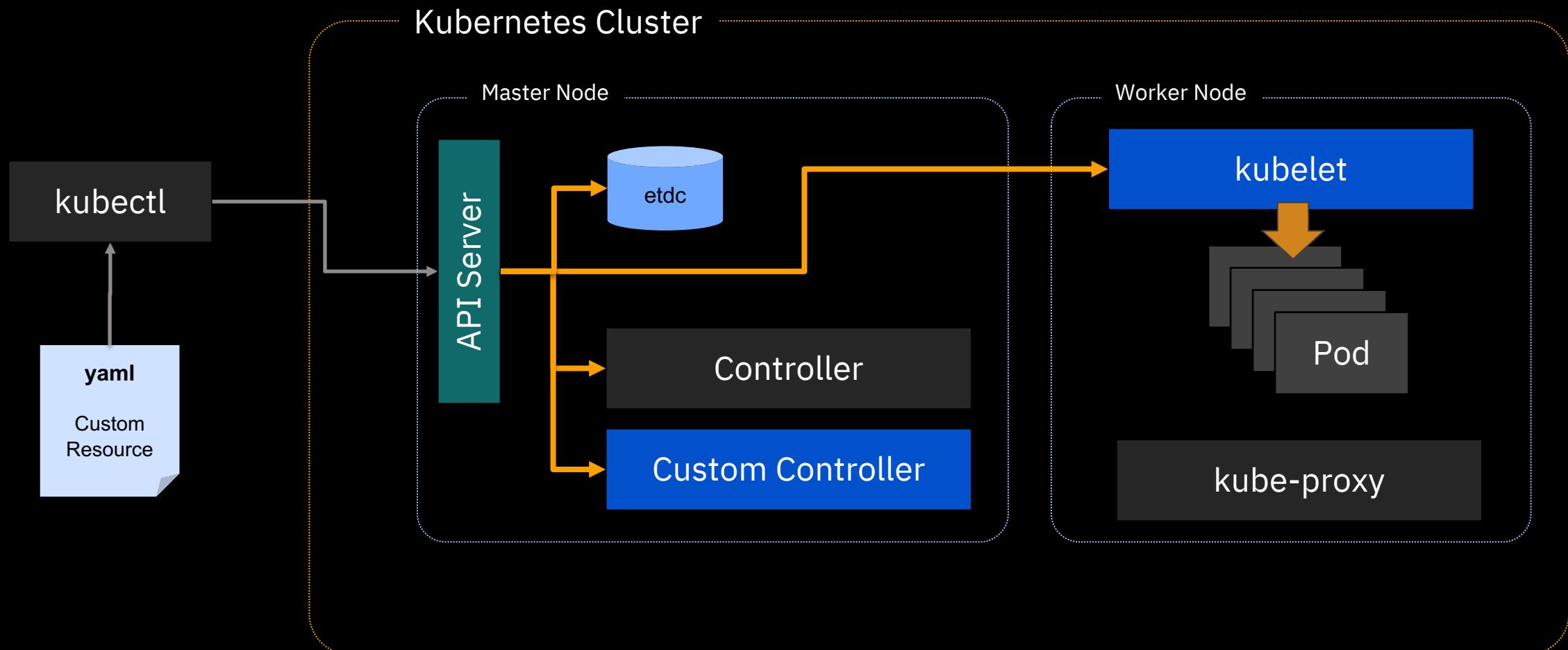


Kubernetes Operators



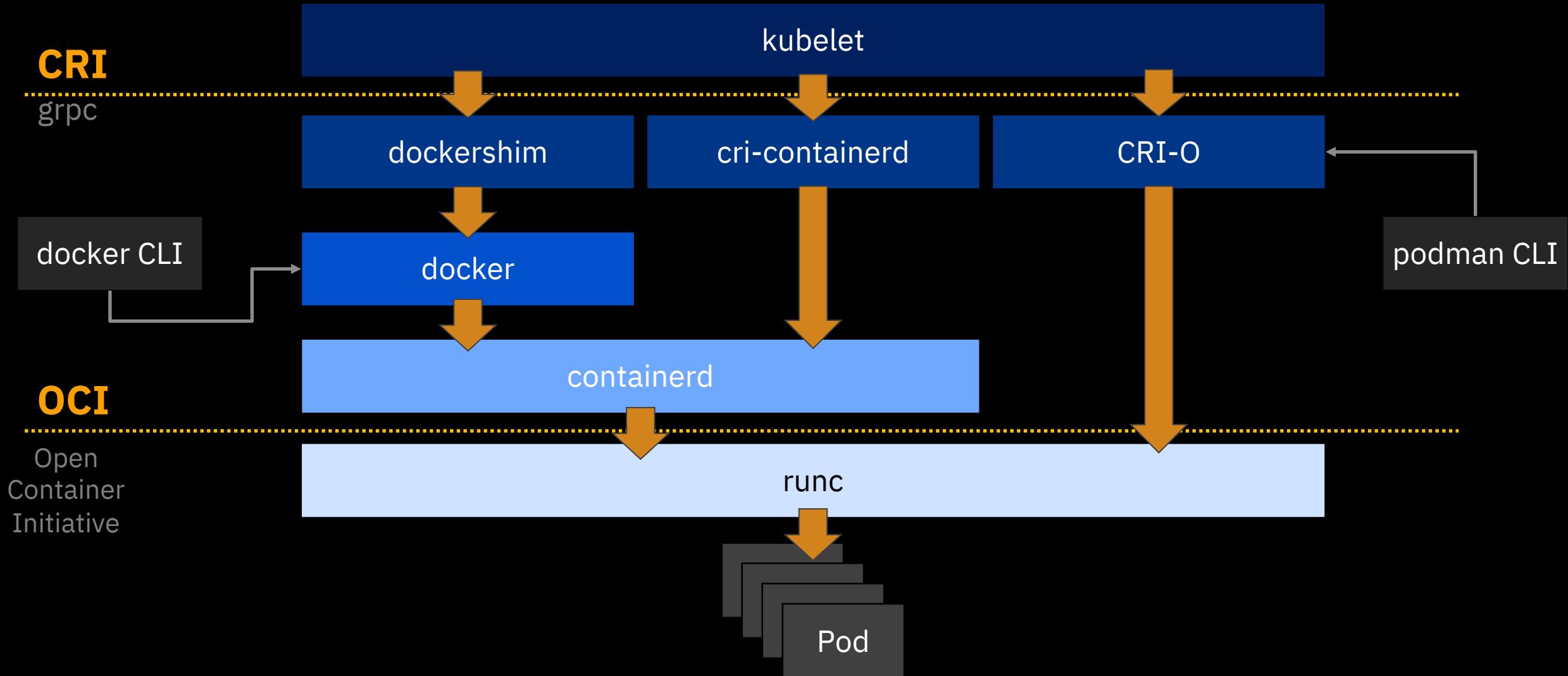
- A **Custom Resource Definition** (CRD) spec that defines the format of the Custom Resource
- A **Custom Controller** to watch our application
 - Custom code within the new controller that dictates how to reconcile our CR against the spec
- An **Operator** to manage the Custom Controller
- A **Custom Resource** (CR) spec that defines the application we want to watch
- A **deployment** for the Operator and Custom Resource

Kubernetes Operators



CRI - Runtimes

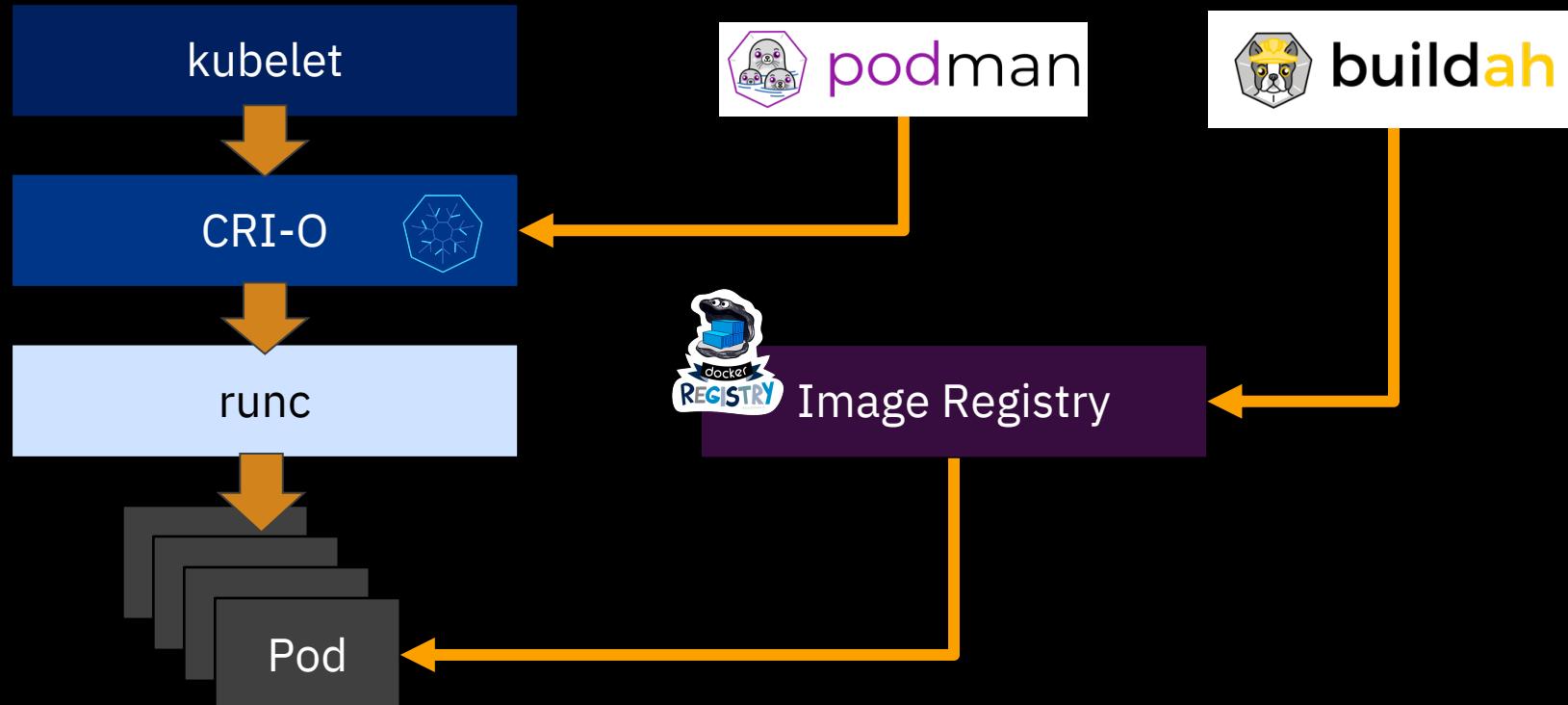
Kubernetes – Common Runtime Interface



Kubernetes – CRI-O



cri-O



Kubernetes – Why CRI-O?



- **A Truly Open Project:** Operated as part of the broader Kubernetes community. Contributors from companies including Red Hat, SUSE, Intel, Google, Alibaba, IBM and more.
- **Lightweight:** CRI-O is made of lots of small components. In comparison, the Docker Engine is a heavyweight daemon which is communicated to using the docker CLI tool in a client/server fashion.
- **More Securable:** As CRI-O containers are children of the process that spawned it (not the daemon) they're fully compatible with tooling like cgroups & security constraints to provide an extra layer of protection to your containers. Every container run using the Docker CLI is a 'child' of that large Docker Daemon. This complicates or outright prevents the use of this tooling.
- **Aligned with Kubernetes:** As an official Kubernetes project, CRI-O releases in lock step with Kubernetes, with similar version numbers. ie. CRI-O 1.11.x works with Kubernetes 1.11.x.

API Deprecations

Kubernetes 1.16 APIs

As the Kubernetes API evolves, APIs are periodically reorganized or upgraded. When APIs evolve, the old API is deprecated and eventually removed..



Deprecated APIs in 1.16

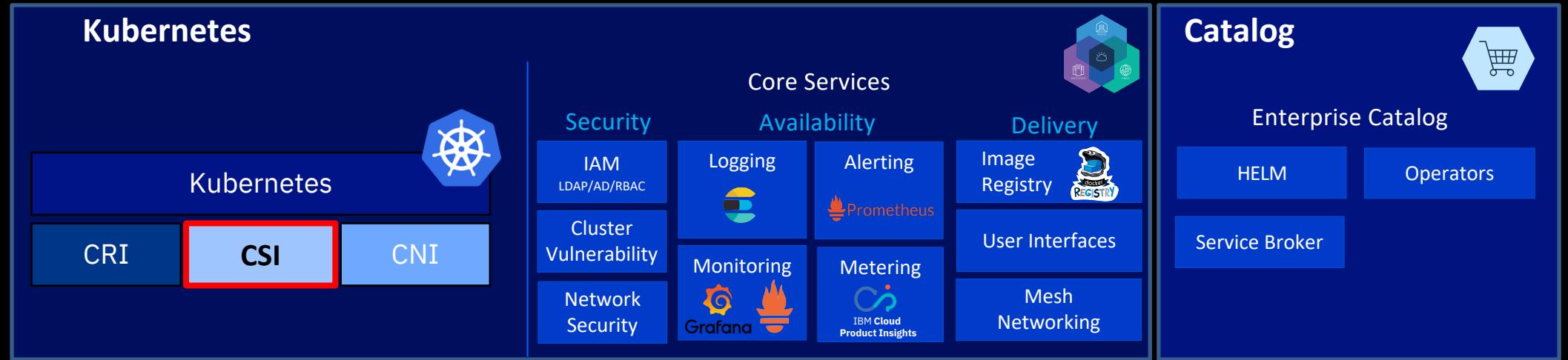
- **NetworkPolicy**
Migrate from extensions/v1beta1 to networking.k8s.io/v1 API
- **PodSecurityPolicy**
Migrate from extensions/v1beta1 to policy/v1beta1 API
- **DaemonSet, Deployment, StatefulSet, and ReplicaSet**
Migrate from extensions/v1beta1 or apps/v1beta2 to apps/v1 API

Remediation

- Rewrite or
- `kubectl convert -f ./my-deployment.yaml --output-version apps/v1`

Storage

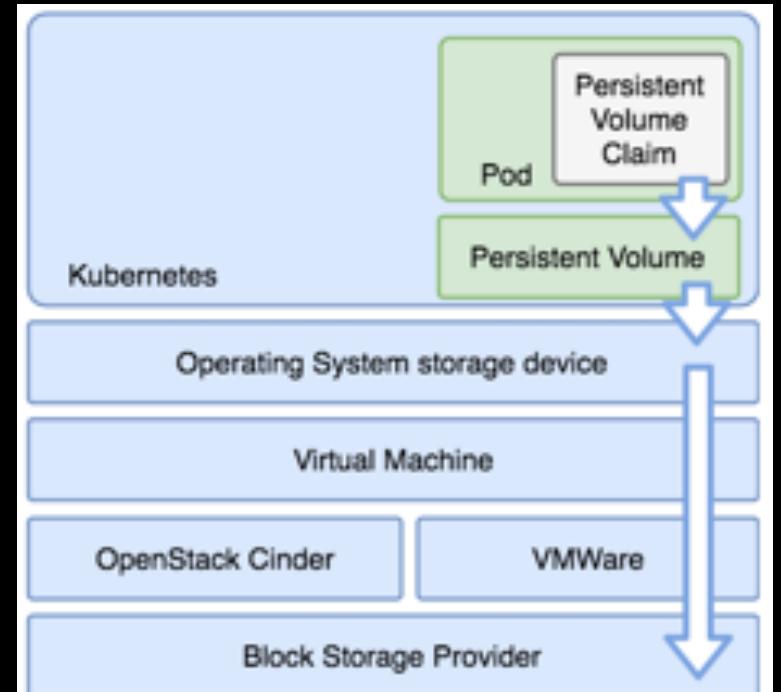
Kubernetes – Storage



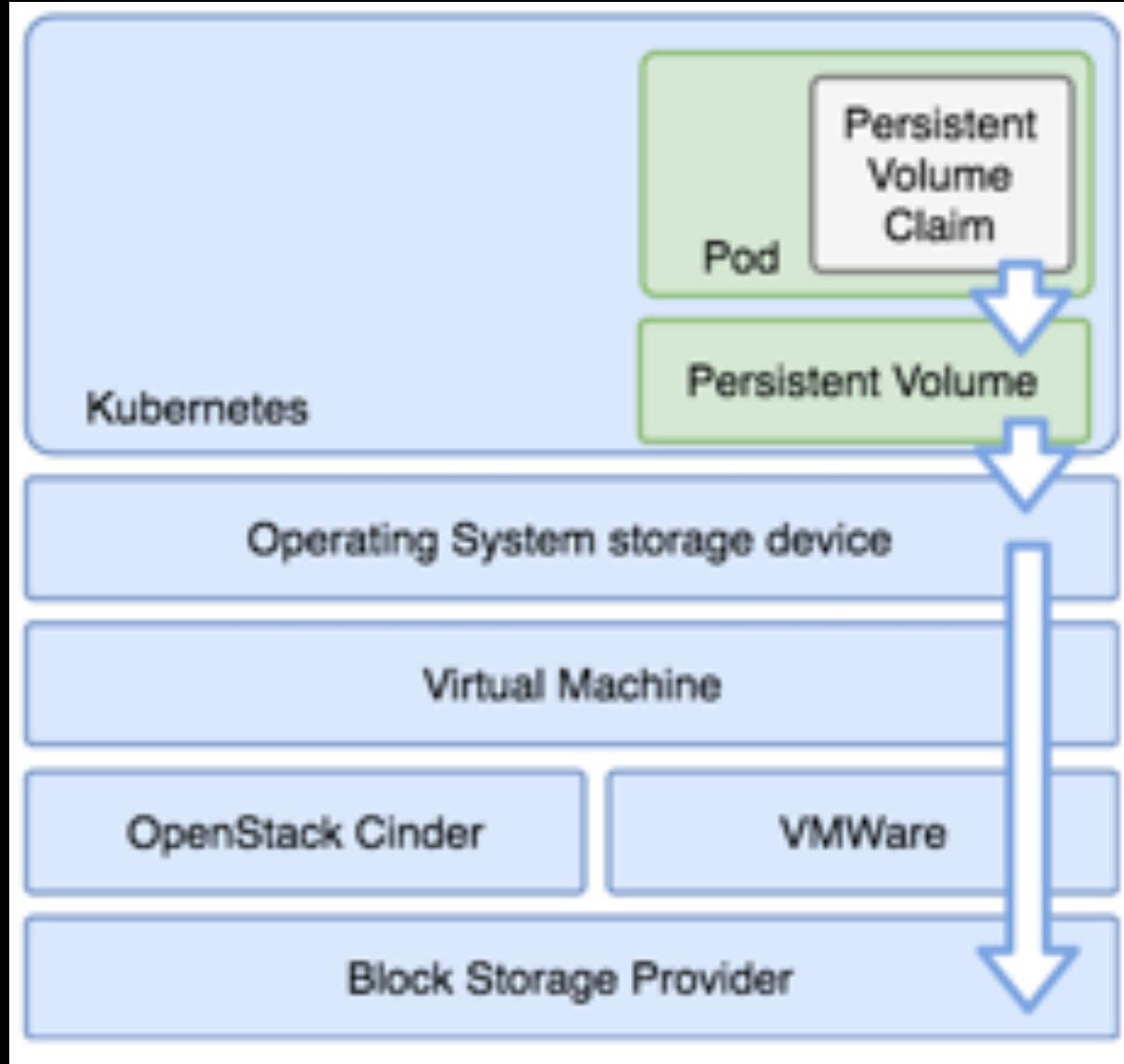
Kubernetes Persistent Storage Support:
HostPath, NFS, GlusterFS, vSphereVolume,

Access Modes:

- **ReadWriteOnce** – the volume can be mounted as read-write by a single node
- **ReadOnlyMany** – the volume can be mounted read-only by many nodes
- **ReadWriteMany** – the volume can be mounted as read-write by many nodes



Kubernetes – Storage



Kubernetes – Common Storage Interface

- GlusterFS
- Rook/Ceph
- OpenEBS
- NFS



GlusterFS

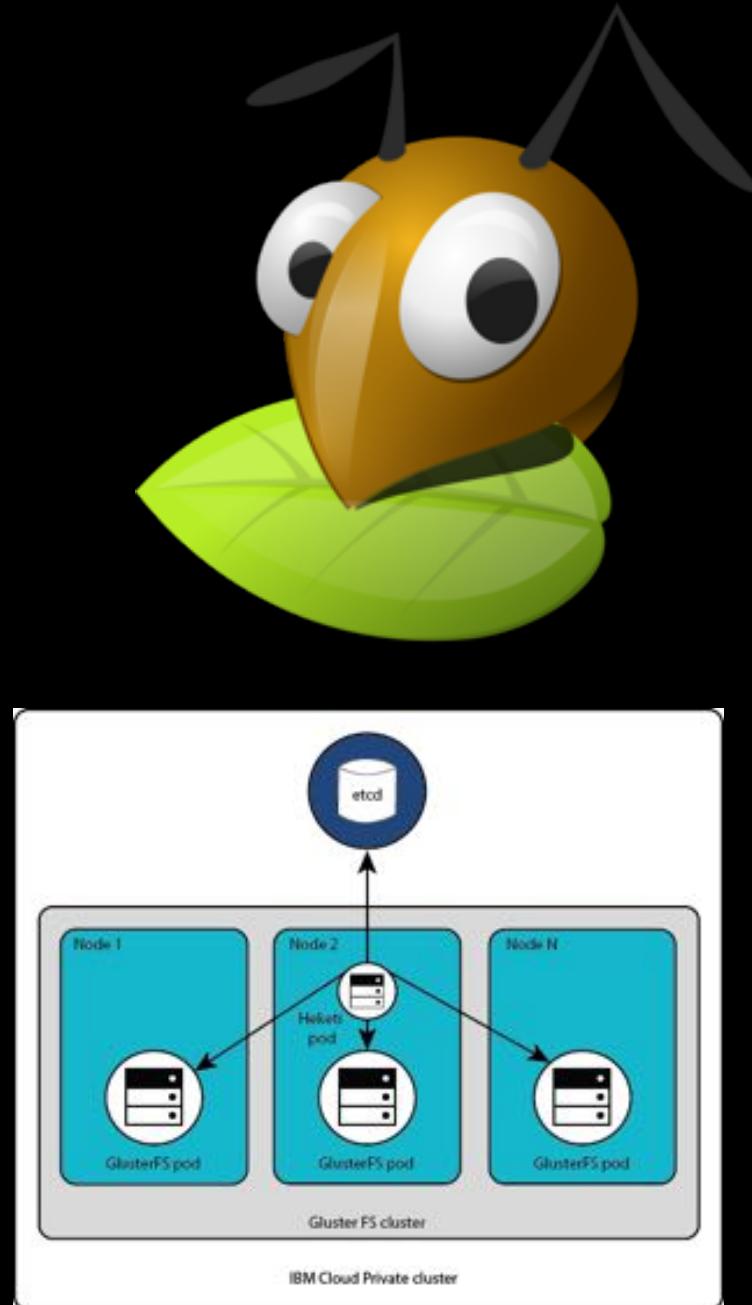
Gluster is a scalable, distributed network filesystem. Using common off-the-shelf hardware, you can create large, distributed storage solutions for media streaming, data analysis, and other data- and bandwidth-intensive tasks. Gluster is free.

Physical install

- Format and mount the bricks
- Installing GlusterFS
- Configure the trusted pool
- Set up a GlusterFS volume
- Install Heketi & Topology (for dynamic provisioning)
- Create K8s Storage Class

Or container based

- https://hub.docker.com/r/glusterfs/gluster_centos
- Must run as privileged → OpenShift?
<https://docs.gluster.org/en/latest/Quick-Start-Guide/Quickstart/>



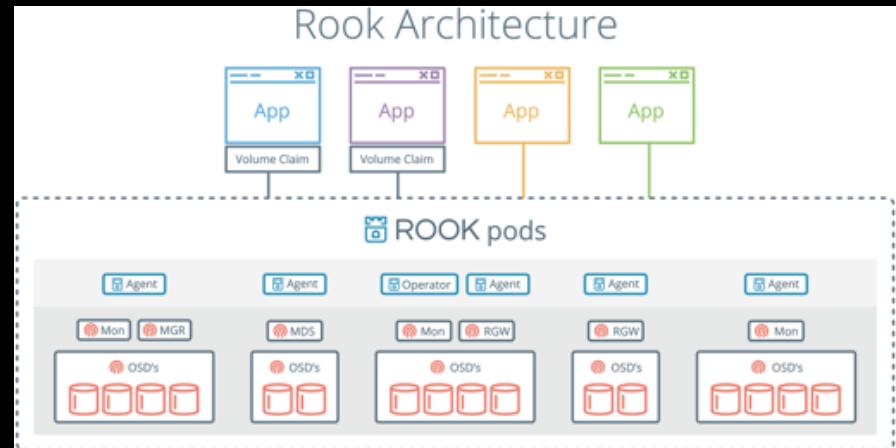
Rook/Ceph

Rook turns distributed storage systems into self-managing, self-scaling, self-healing storage services.

Ceph is a highly scalable distributed storage solution for **block storage**, **object storage**, and **shared file systems** with years of production deployments.

Install on K8s

- git clone <https://github.com/rook/rook.git>
- kubectl create -f common.yaml
- kubectl create -f operator(-openshift).yaml
- kubectl create -f cluster-test.yaml
- kubectl create -f csi/rbd/storageclass-test.yaml



<https://rook.io/docs/rook/v0.9/ceph-quickstart.html>

cassandra	crds: Set annotations pods, deployments and so on
ceph	We were defaulting to 'ext4' at first and then moved to
cockroachdb	crds: Set annotations pods, deployments and so on
edgefs	edgefs image version update in examples and docs
minio	minio: add necessary update verb in minio RBAC
nfs	NFS: Fix operator.yaml line endings
noobaa	Rook-NooBaa Design Doc
yugabytedb	yugabytedb: Documentation, user guides & examples

OpenEBS

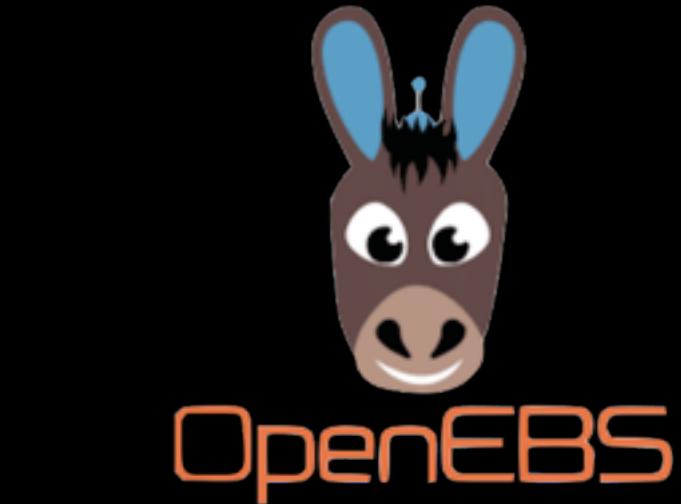
OpenEBS is truly Kubernetes native.

It adopts Container Attached Storage (CAS) approach, where each workload is provided with a dedicated storage controller. It implements granular storage policies and isolation that enable users to optimize storage for each specific workload.

Install on K8s

- sudo apt-get install open-iscsi
- kubectl apply -f <https://openebs.github.io/charts/openebs-operator.yaml>
- kubectl create -f csi/rbd/storageclass-test.yaml
- kubectl apply -f openebs-storageclasses.yaml

<https://docs.openebs.io/docs/next/installation.html>



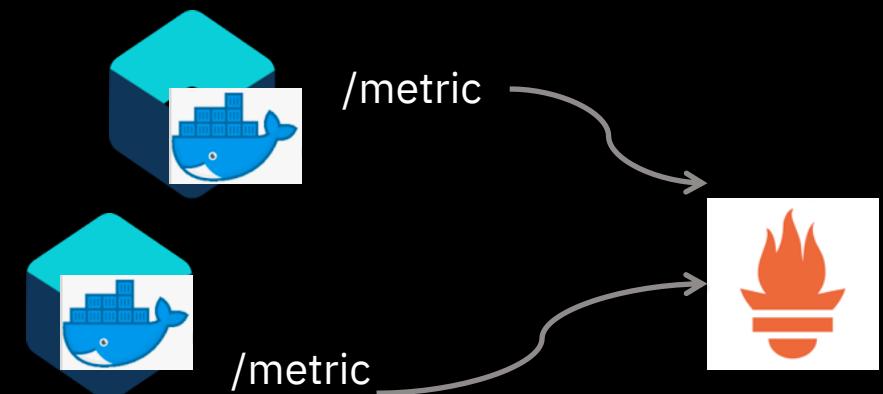
Observability

Observable: Application health

Know your application health

- Kubernetes probes
 - Is the app ready to accept traffic?: Readiness
 - Is the app responsive? : Liveliness
- Is this enough?
 - What about transactions, traffic, memory usage ?

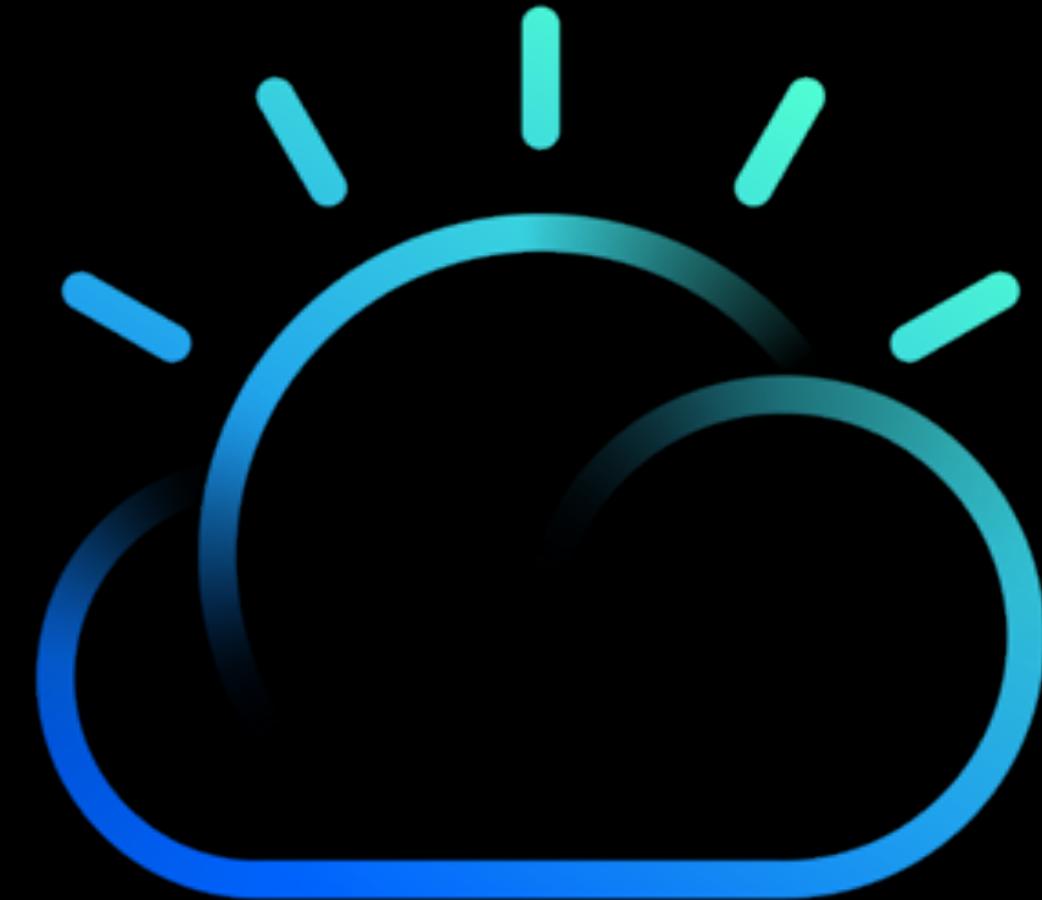
```
readinessProbe:  
# an http probe  
httpGet:  
path: /readiness  
port: 8080  
initialDelaySeconds: 20  
periodSeconds: 5  
  
livenessProbe:  
# an http probe  
httpGet:  
path: /healthcheck  
port: 8080  
initialDelaySeconds: 15  
timeoutSeconds: 1
```



QUESTIONS?



QUESTIONS?



IBM Cloud

The Journey to Cloud Q&A

04



IBM Cloud

QUESTIONS?



IBM