K8s Hands On

Salvo Nicotra

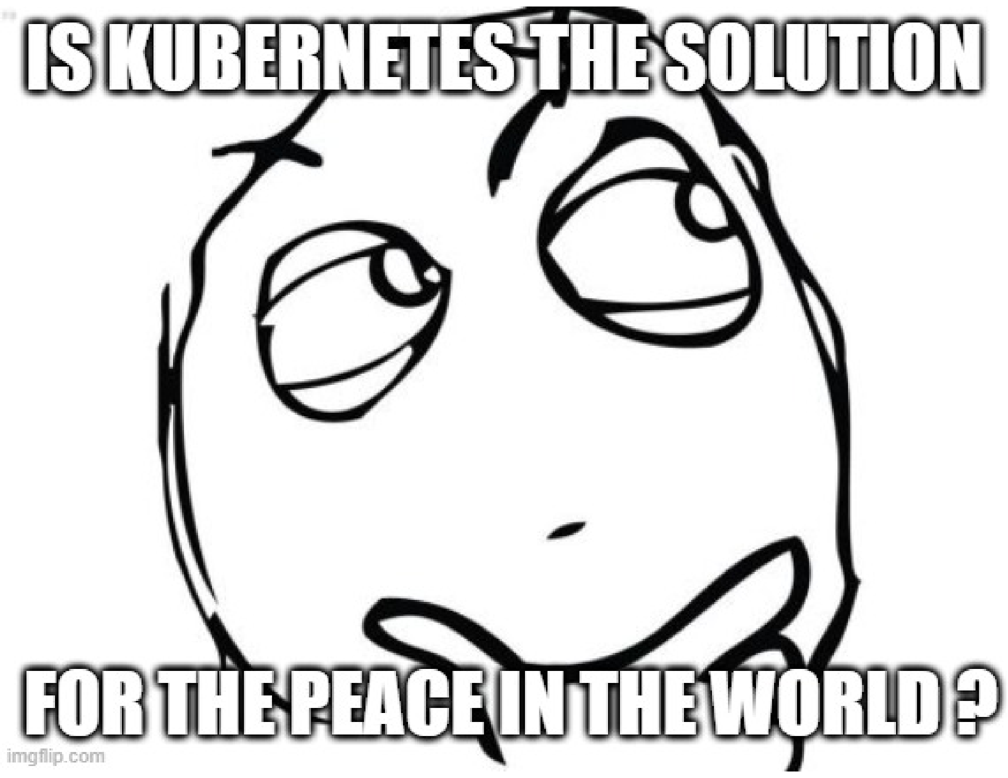
Agenda

- 5W (What, Why, When, Where, Who)
- 1H (HOW - Architecture & Concepts & Demo)

The 5W and (1H)

The 5 Ws (and 1 H) that should be asked of every project

What is our question today ?



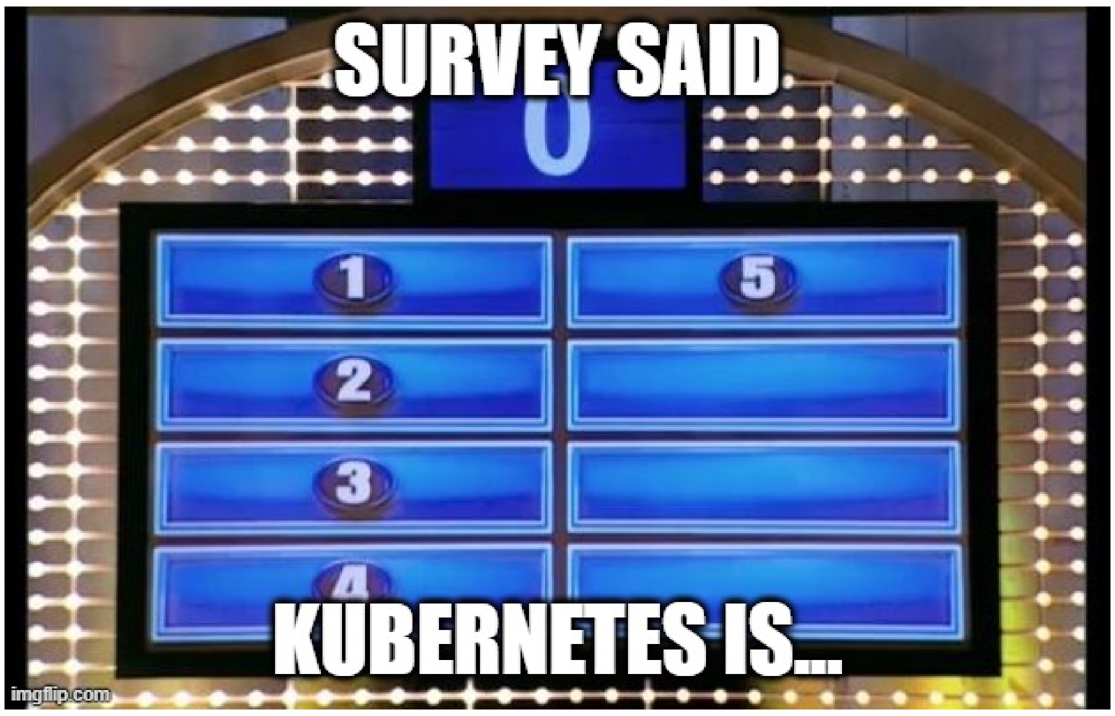
NicsMeme

More realistically…

My first steps with kubernetes

Kube Feud

Survey Said...

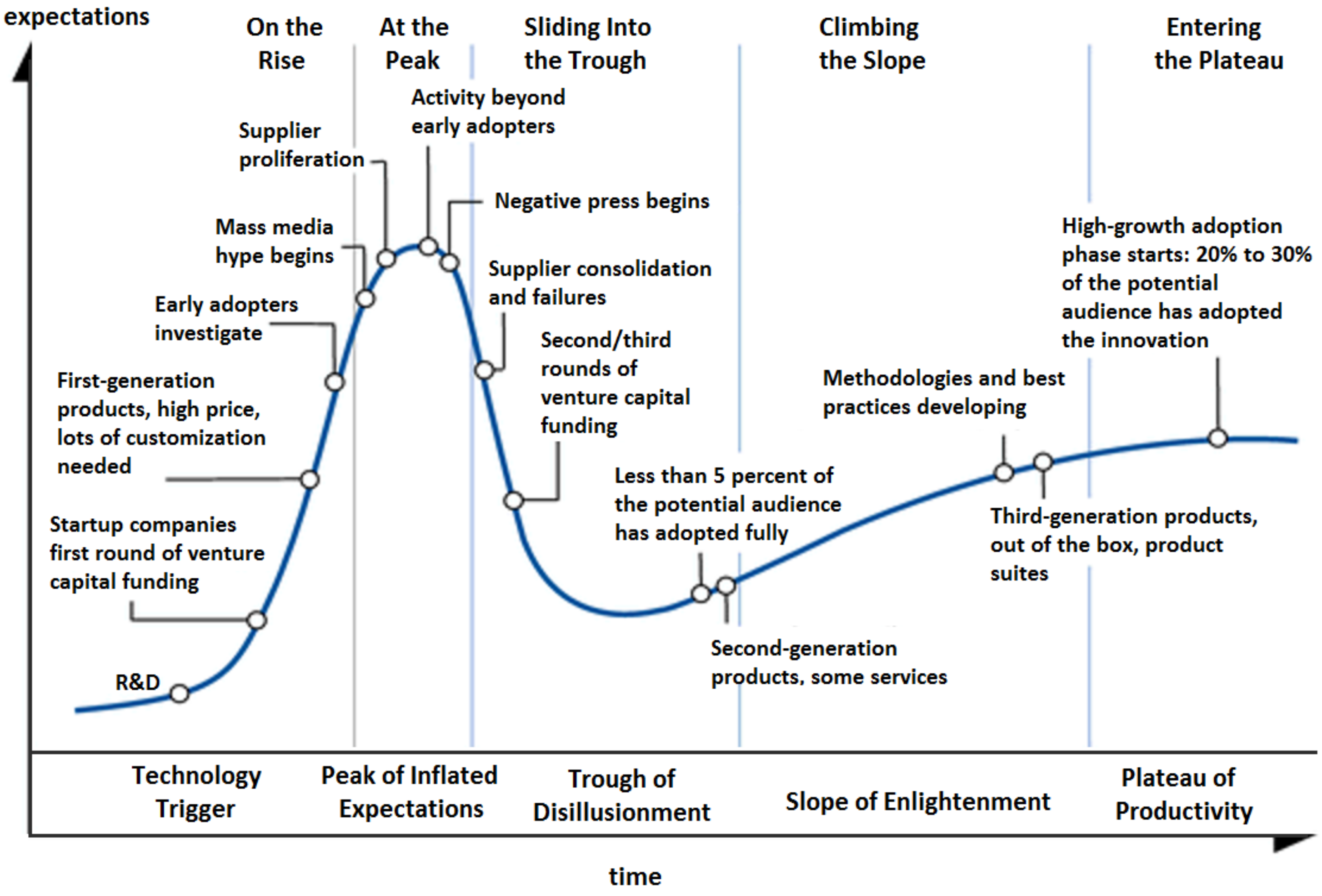


NicsMeme

Hype

The hype cycle is a branded graphical presentation developed and used by the American research, advisory and information technology firm Gartner to represent the maturity, adoption, and social application of specific technologies. The hype cycle claims to provide a graphical and conceptual presentation of the maturity of emerging technologies through five phases

Wikipedia


expectations
On the Rise
At the Peak
Sliding Into the Trough
Climbing the Slope
Entering the Plateau
Activity beyond early adopters
Supplier proliferation
Negative press begins
Mass media hype begins
Supplier consolidation and failures
High-growth adoption phase starts: 20% to 30% of the potential audience has adopted the innovation
Early adopters investigate
Second/third rounds of venture capital funding
Methodologies and best practices developing
First-generation products, high price, lots of customization needed
Less than 5 percent of the potential audience has adopted fully
Third-generation products, out of the box, product suites
Startup companies first round of venture capital funding
Second-generation products, some services
R&D
Technology Trigger
Peak of Inflated Expectations
Trough of Disillusionment
Slope of Enlightenment
Plateau of Productivity
time

Kuberbernets Hype Cycle

- 2017 - https://thenewstack.io/7-ways-kubenetes-avoids-openstack-like-hype-cycle/
- 2019 - https://amazicworld.com/kubernetes-and-the-hype-cycle/
- 2020 - https://www.weave.works/blog/navigating-the-kubernetes-hype-cycle

- Kubernetes is an Ecosystem, not a Monolith
- it depends, where you are on your personal Cloud Native Journey ?
- Companies at the beginning, developers are ahead

What ?

From What is Kubernetes

Kubernetes is a

portable

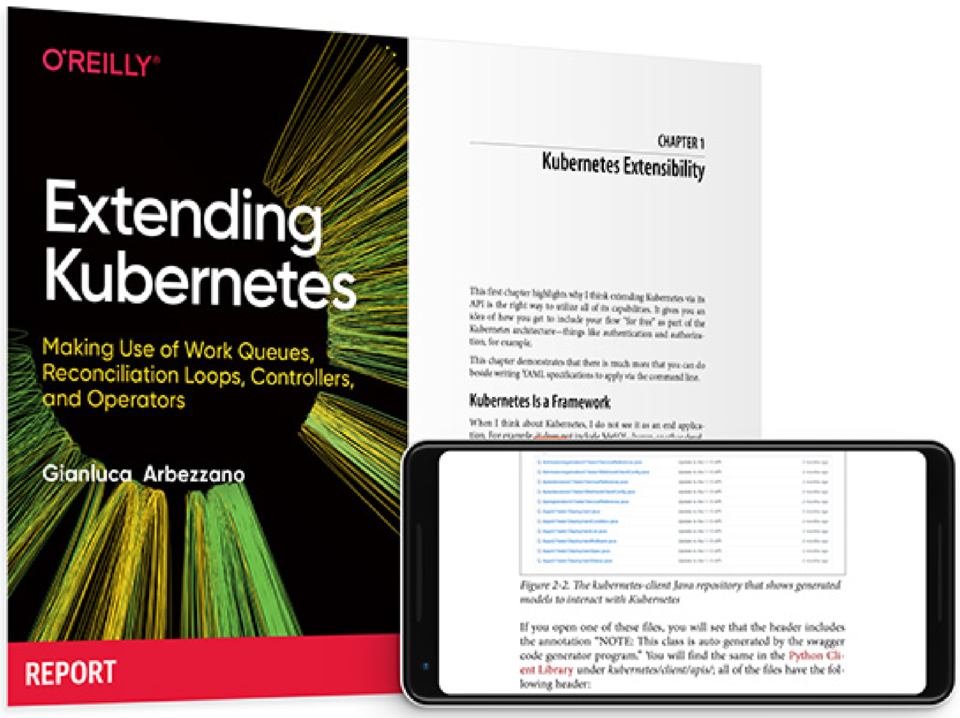
> Related to Kubernetes as software, to applications or destination ?

Kubernetes Portability: Must-Have or Shiny Object Syndrome?

extensible

> Kubernetes is highly configurable and extensible. As a result, there is rarely a need to fork or submit patches to the Kubernetes project code.

[Source](#)


O'REILLY
Extending Kubernetes
Making Use of Work Queues, Reconciliation Loops, Controllers, and Operators
Gianluca Arbezzano
REPORT

CHAPTER 1
Kubernetes Extensibility

This first chapter highlights why I think extending Kubernetes via its API is the right way to utilize all of its capabilities. It gives you an idea of how you get to include your flow "for free" as part of the Kubernetes architecture—things like authentication and authorization, for example.

This chapter demonstrates that there is much more that you can do beside writing YAML specifications to apply via the command line.

Kubernetes Is a Framework

When I think about Kubernetes, I do not see it as an end application. For example,

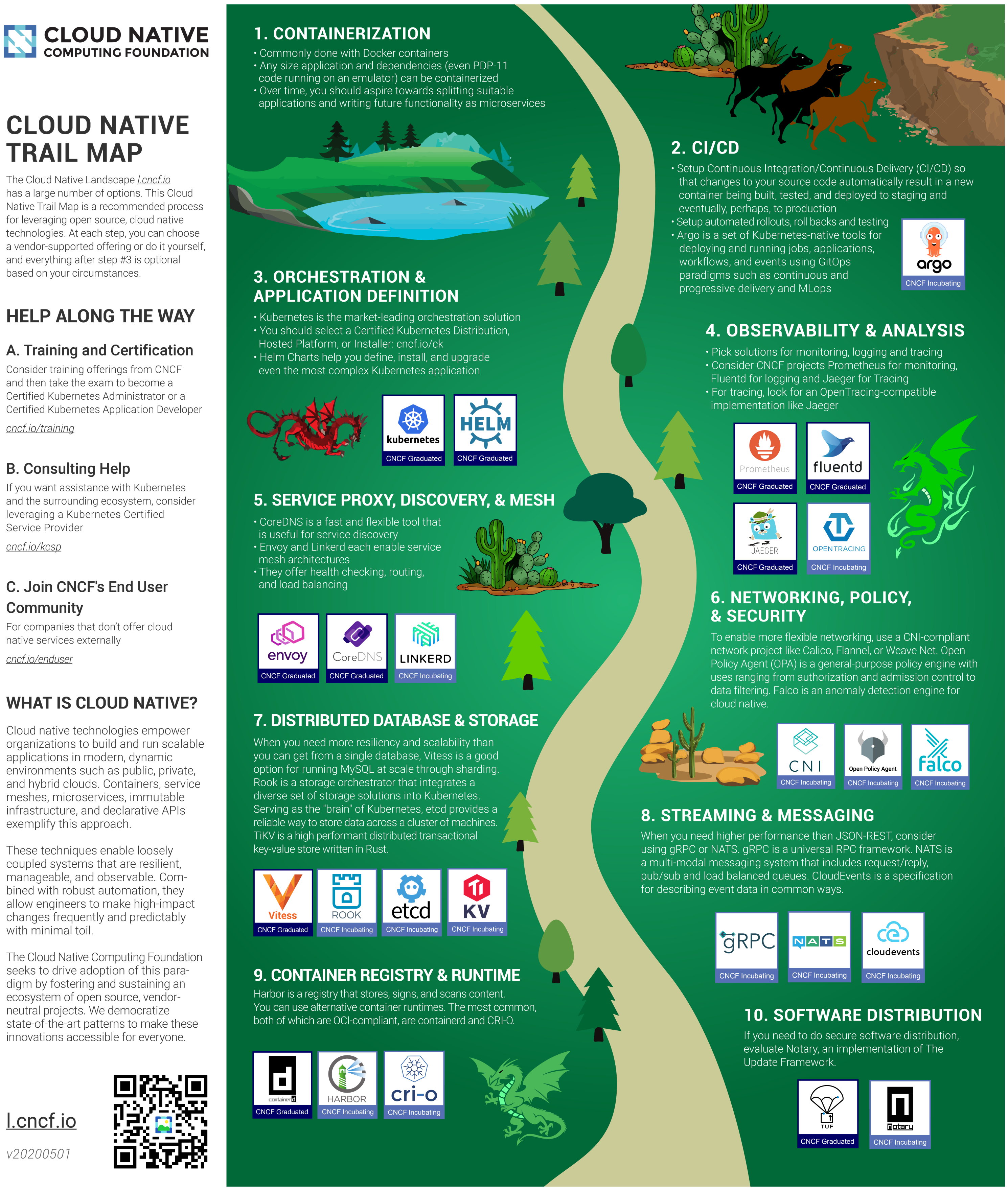
Figure 2-2. The kubernetes-client Java repository that shows generated models to interact with Kubernetes

If you open one of these files, you will see that the header includes the annotation "NOTE: This class is auto generated by the swagger code generator program." You will find the same in the Python Client Library under *kubernetes/client/apis/*; all of the files have the following header:

[Extending Kubernets](#)

open-source

>



CLOUD NATIVE
COMPUTING FOUNDATION

CLOUD NATIVE TRAIL MAP

The Cloud Native Landscape *l.cncf.io* has a large number of options. This Cloud Native Trail Map is a recommended process for leveraging open source, cloud native technologies. At each step, you can choose a vendor-supported offering or do it yourself, and everything after step #3 is optional based on your circumstances.

HELP ALONG THE WAY

A. Training and Certification

Consider training offerings from CNCF and then take the exam to become a Certified Kubernetes Administrator or a Certified Kubernetes Application Developer

cncf.io/training

B. Consulting Help

If you want assistance with Kubernetes and the surrounding ecosystem, consider leveraging a Kubernetes Certified Service Provider

cncf.io/kcsp

C. Join CNCF's End User Community

For companies that don't offer cloud native services externally

cncf.io/enduser

WHAT IS CLOUD NATIVE?

Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

The Cloud Native Computing Foundation seeks to drive adoption of this paradigm by fostering and sustaining an ecosystem of open source, vendor-neutral projects. We democratize state-of-the-art patterns to make these innovations accessible for everyone.

l.cncf.io

v20200501

1. CONTAINERIZATION

- Commonly done with Docker containers
- Any size application and dependencies (even PDP-11 code running on an emulator) can be containerized
- Over time, you should aspire towards splitting suitable applications and writing future functionality as microservices

2. CI/CD

- Setup Continuous Integration/Continuous Delivery (CI/CD) so that changes to your source code automatically result in a new container being built, tested, and deployed to staging and eventually, perhaps, to production
- Setup automated rollouts, roll backs and testing
- Argo is a set of Kubernetes-native tools for deploying and running jobs, applications, workflows, and events using GitOps paradigms such as continuous and progressive delivery and MLops

3. ORCHESTRATION & APPLICATION DEFINITION

- Kubernetes is the market-leading orchestration solution
- You should select a Certified Kubernetes Distribution, Hosted Platform, or Installer: cncf.io/ck
- Helm Charts help you define, install, and upgrade even the most complex Kubernetes application

4. OBSERVABILITY & ANALYSIS

- Pick solutions for monitoring, logging and tracing
- Consider CNCF projects Prometheus for monitoring, Fluentd for logging and Jaeger for Tracing
- For tracing, look for an OpenTracing-compatible implementation like Jaeger

5. SERVICE PROXY, DISCOVERY, & MESH

- CoreDNS is a fast and flexible tool that is useful for service discovery
- Envoy and Linkerd each enable service mesh architectures
- They offer health checking, routing, and load balancing

6. NETWORKING, POLICY, & SECURITY

To enable more flexible networking, use a CNI-compliant network project like Calico, Flannel, or Weave Net. Open Policy Agent (OPA) is a general-purpose policy engine with uses ranging from authorization and admission control to data filtering. Falco is an anomaly detection engine for cloud native.

7. DISTRIBUTED DATABASE & STORAGE

When you need more resiliency and scalability than you can get from a single database, Vitess is a good option for running MySQL at scale through sharding. Rook is a storage orchestrator that integrates a diverse set of storage solutions into Kubernetes. Serving as the "brain" of Kubernetes, etcd provides a reliable way to store data across a cluster of machines. TiKV is a high performant distributed transactional key-value store written in Rust.

8. STREAMING & MESSAGING

When you need higher performance than JSON-REST, consider using gRPC or NATS. gRPC is a universal RPC framework. NATS is a multi-modal messaging system that includes request/reply, pub/sub and load balanced queues. CloudEvents is a specification for describing event data in common ways.

9. CONTAINER REGISTRY & RUNTIME

Harbor is a registry that stores, signs, and scans content. You can use alternative container runtimes. The most common, both of which are OCI-compliant, are containerd and CRI-O.

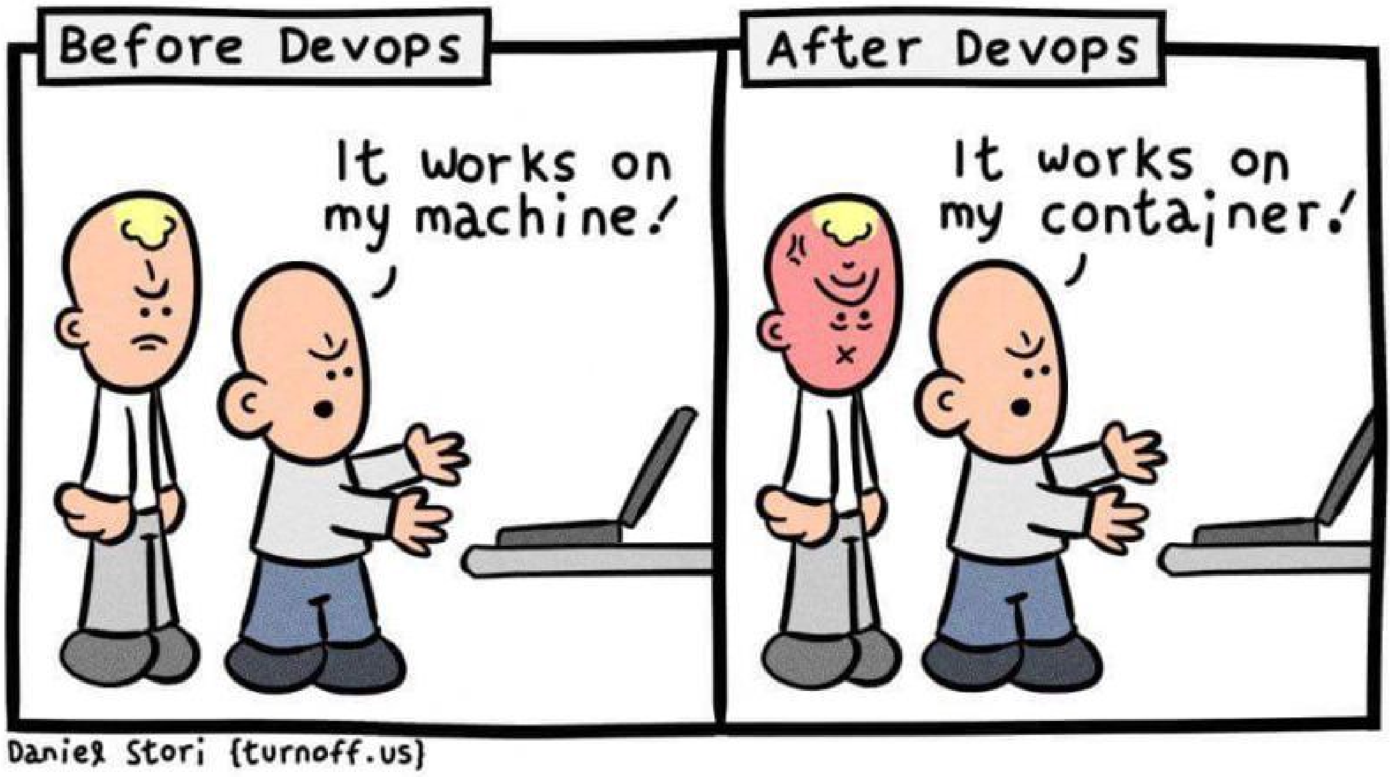
10. SOFTWARE DISTRIBUTION

If you need to do secure software distribution, evaluate Notary, an implementation of The Update Framework.

[Clound Native Computing Foundation]

platform

> #Kubernetes is not a product…it's a cloud native platform for building platforms (Bryan Liles of #VMware during his opening keynote @ #KubeCo 2019)

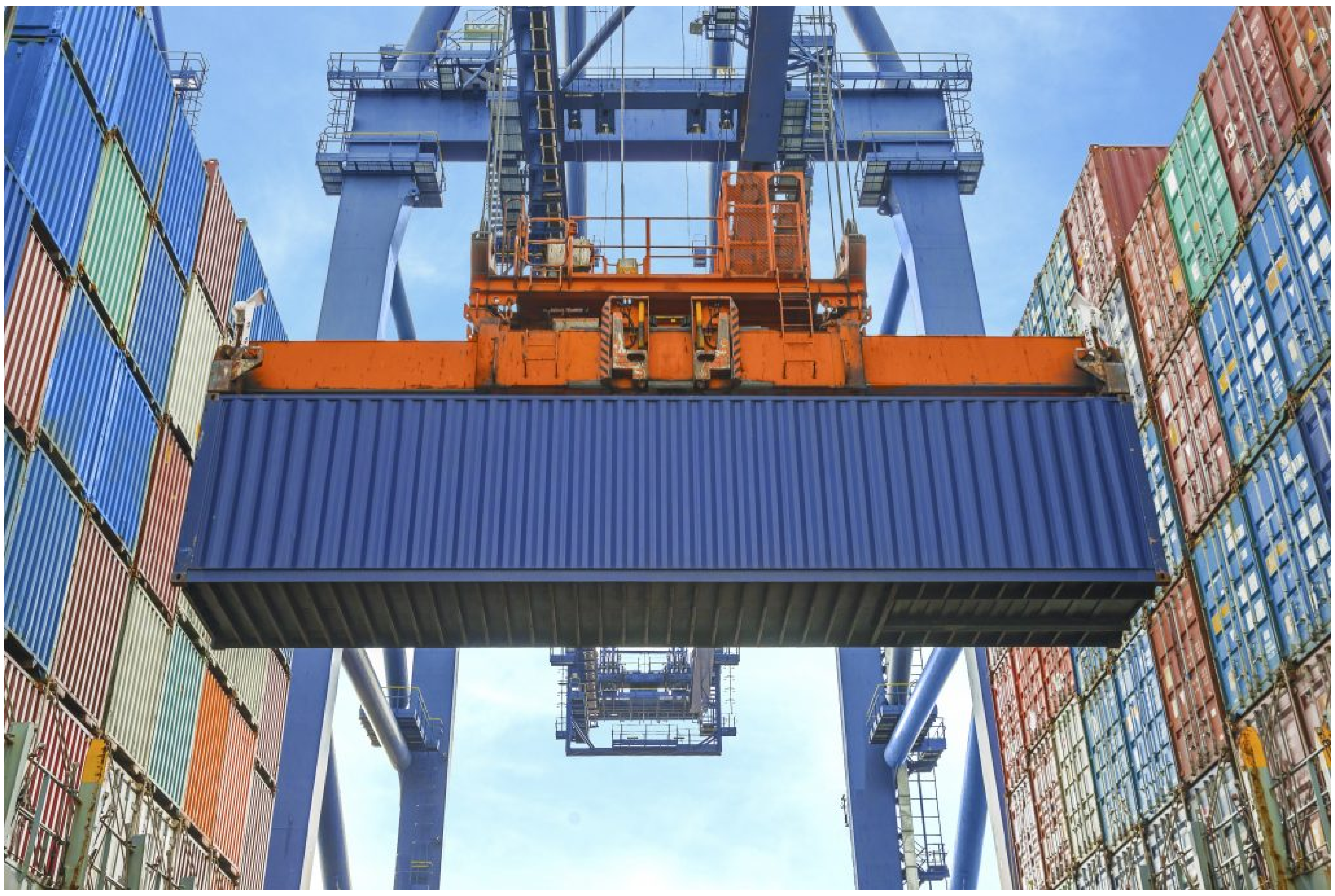


Source

Manage

for managing containerized workloads and services

> i.e less stress for devops/sysadmin, someone take care of application and handle problems



Help

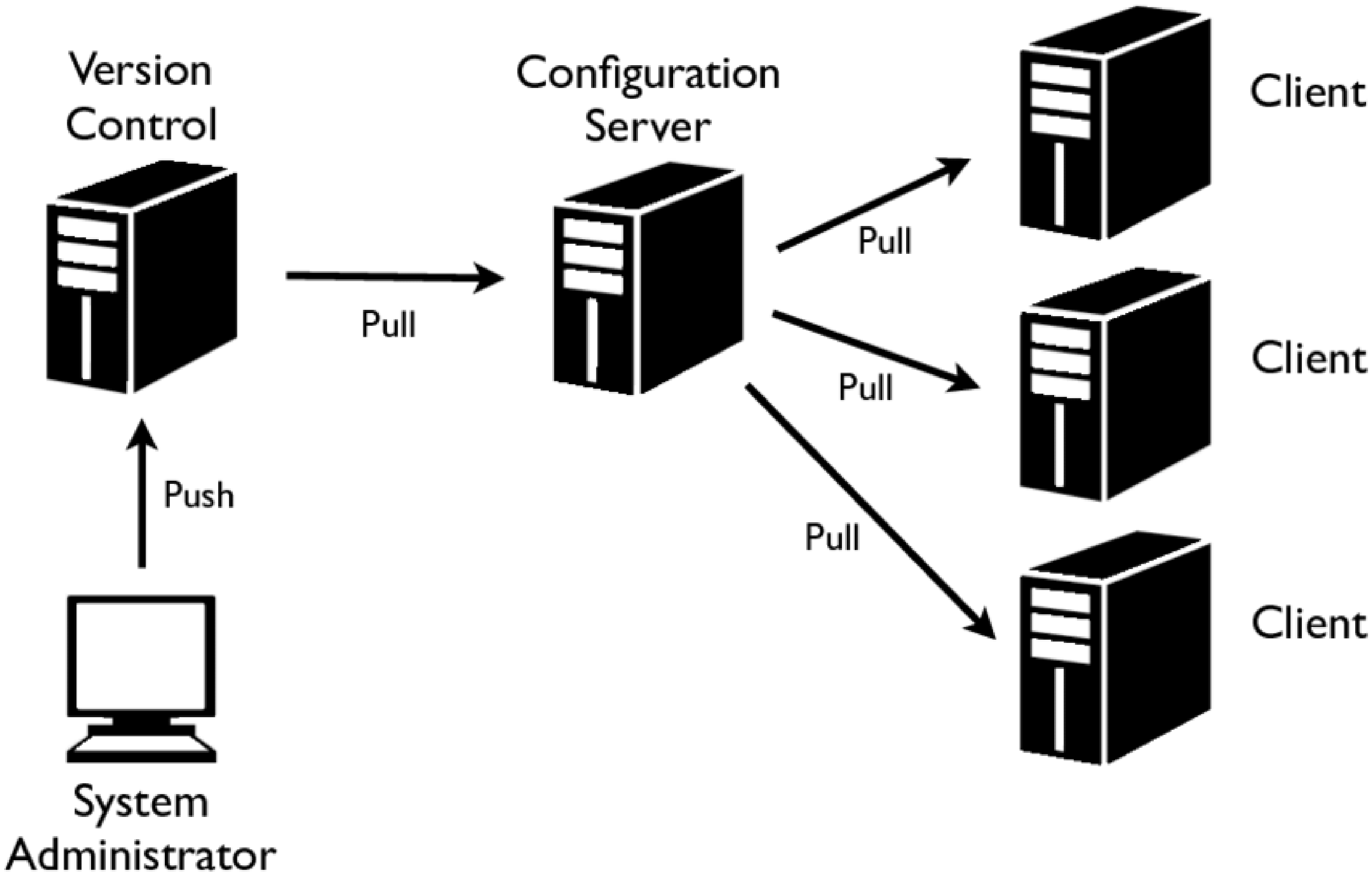
facilitates both

- declarative configuration

> “Infrastucture as a code” https://blog.nelhage.com/post/declarative-configuration-management/

- automation

> “Continuos Deployment”



A typical declarative configuration system. The system administrator authors a declarative specification which is stored in version control. The configuration server periodically retrieves the latest revision and computes the configuration for each of its clients. Clients periodically retrieve their configuration from the server in a voluntary manner.

Source

So What is Kubernetes

> Kubernetes is a portable, extensible, open-source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation. It has a large, rapidly growing ecosystem. Kubernetes services, support, and tools are widely available.

Let's see in practice

Inspired by https://medium.com/payscale-tech/imperative-vs-declarative-a-kubernetes-tutorial-4be66c5d8914

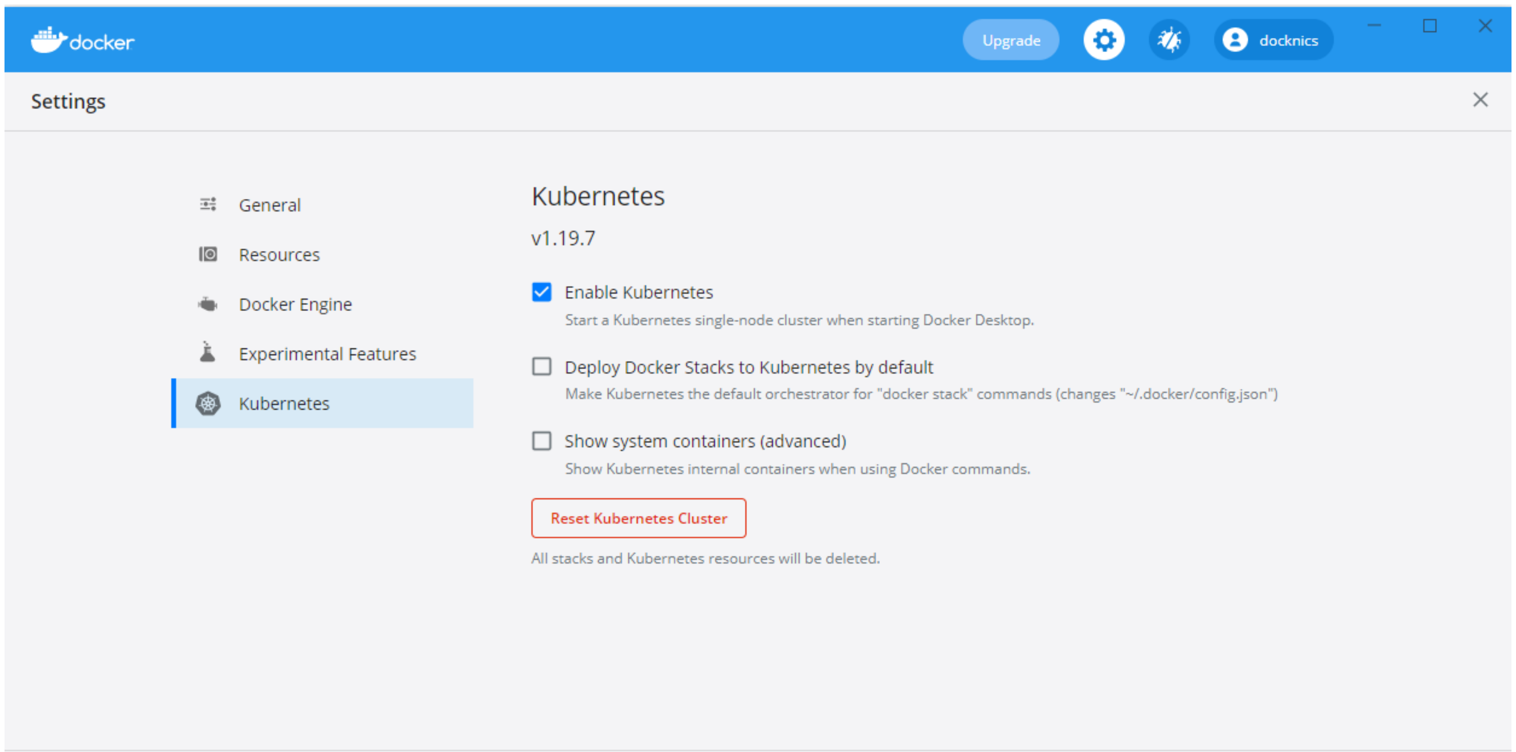
Don't Blame me

NicsMeme

Enable Kubernetes on Docker Desktop + WSL 2

https://kubernetes.io/blog/2020/05/21/wsl-docker-kubernetes-on-the-windows-desktop/

Enable Kubernetes



Kubernetes

v1.19.7

☑ Enable Kubernetes
Start a Kubernetes single-node cluster when starting Docker Desktop.

□ Deploy Docker Stacks to Kubernetes by default
Make Kubernetes the default orchestrator for "docker stack" commands (changes "~/.docker/config.json")

□ Show system containers (advanced)
Show Kubernetes internal containers when using Docker commands.

All stacks and Kubernetes resources will be deleted.

Build a image

A simple webserver in node.js

```
// app.js
const http = require('http');
const os = require('os');
const ip = '0.0.0.0';
const port = 3000;
const hostname = os.hostname();
const whoami = process.env['WHOAMI'] || 'Anonymous';
const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end(`Hi, I'm ${whoami}, from ${hostname}.\n`);
});
server.listen(port, ip, () => {
  console.log(`Server Running at http://${ip}:${port}/`);
});
```

Dockerfile

```
FROM node:8
COPY app.js .
ENTRYPOINT ["node", "app.js"]
```

Build with

```
cd code/payscale-example/
docker build -t k8s:payscaleapp app
```

Deploy

```
# kubectl apply -f app-deployment.yaml -f app-service.yaml
deployment.apps/pyscale-example created
service/pyscale-example created

# kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
pyscale-example-7c8499c88d-24jlw   1/1     Running   0          36s
pyscale-example-7c8499c88d-ntb59   1/1     Running   0          36s
pyscale-example-7c8499c88d-xsnkx   1/1     Running   0          36s
```

Proxy

```
# kubectl port-forward deployment/pyscale-example 3000
Forwarding from 127.0.0.1:3000 -> 3000
Forwarding from [::1]:3000 -> 3000
```

Test it

http://localhost:3000

Dashboard

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.0.0-rc6/aio/deploy/recommended.yaml
namespace/kubernetes-dashboard created
serviceaccount/kubernetes-dashboard created
service/kubernetes-dashboard created
secret/kubernetes-dashboard-certs created
secret/kubernetes-dashboard-csrf created
secret/kubernetes-dashboard-key-holder created
configmap/kubernetes-dashboard-settings created
role.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrole.rbac.authorization.k8s.io/kubernetes-dashboard created
rolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
deployment.apps/kubernetes-dashboard created
service/dashboard-metrics-scraper created
deployment.apps/dashboard-metrics-scraper created
nics@NICS:~$ kubectl get all -n kubernetes-dashboard
NAME                                             READY   STATUS              RESTARTS   AGE
pod/dashboard-metrics-scraper-74db988864-7lw55   1/1     Running             0          15s
pod/kubernetes-dashboard-847d8c7cdc-czqjk        0/1     ContainerCreating   0          15s

NAME                                TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)    AGE
service/dashboard-metrics-scraper   ClusterIP   10.101.36.54    <none>        8000/TCP   15s
service/kubernetes-dashboard        ClusterIP   10.107.135.78   <none>        443/TCP    16s

NAME                                        READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/dashboard-metrics-scraper   1/1     1            1           15s
deployment.apps/kubernetes-dashboard        0/1     1            0           15s

NAME                                                   DESIRED   CURRENT   READY   AGE
replicaset.apps/dashboard-metrics-scraper-74db988864   1         1         1       15s
replicaset.apps/kubernetes-dashboard-847d8c7cdc        1         1         0       15s

nics@NICS:~$ kubectl proxy
Starting to serve on 127.0.0.1:8001
```

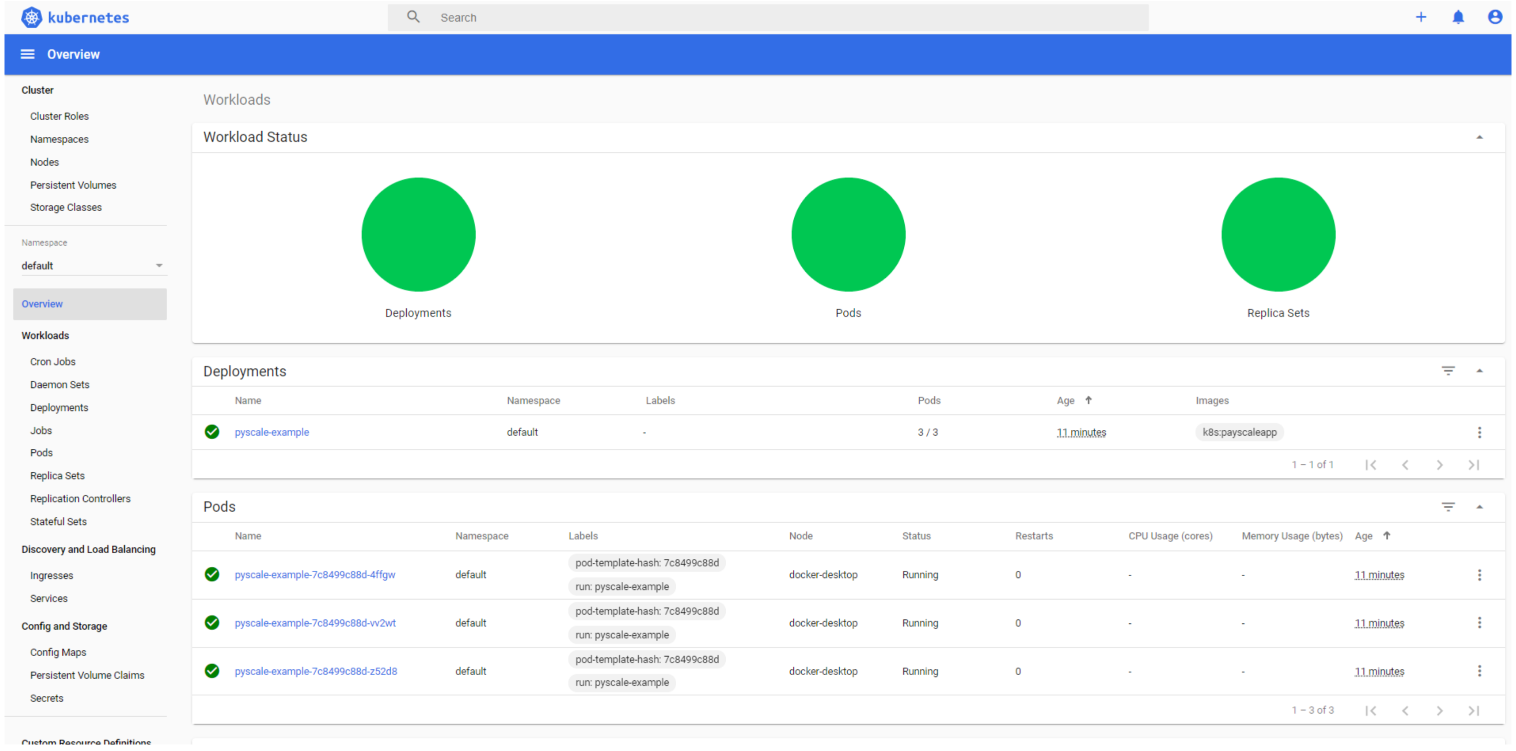
http://localhost:8001/api/v1/namespaces/kubernetes-dashboard/services/https:kubernetes-dashboard:/proxy/#/login

```
kubectl apply -f - <<EOF
apiVersion: v1
kind: ServiceAccount
metadata:
  name: admin-user
  namespace: kubernetes-dashboard
EOF
# Create a ClusterRoleBinding for the ServiceAccount
kubectl apply -f - <<EOF
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: admin-user
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: admin-user
  namespace: kubernetes-dashboard
EOF

nics@NICS:~$ kubectl -n kubernetes-dashboard describe secret $(kubectl -n kubernetes-dashboard get secret | grep
admin-user | awk '{print $1}')
Name:         admin-user-token-42dwr
Namespace:    kubernetes-dashboard
Labels:       <none>
Annotations:  kubernetes.io/service-account.name: admin-user
              kubernetes.io/service-account.uid: a7bbe800-c0fb-49fe-a5f0-ec8ec422b4f9

Type:  kubernetes.io/service-account-token
```

Et voila



Deployments

	Name	Namespace	Labels	Pods	Age ↑	Images
✓	pyscale-example	default	-	3 / 3	11 minutes	k8s:payscaleapp

1 – 1 of 1

Pods

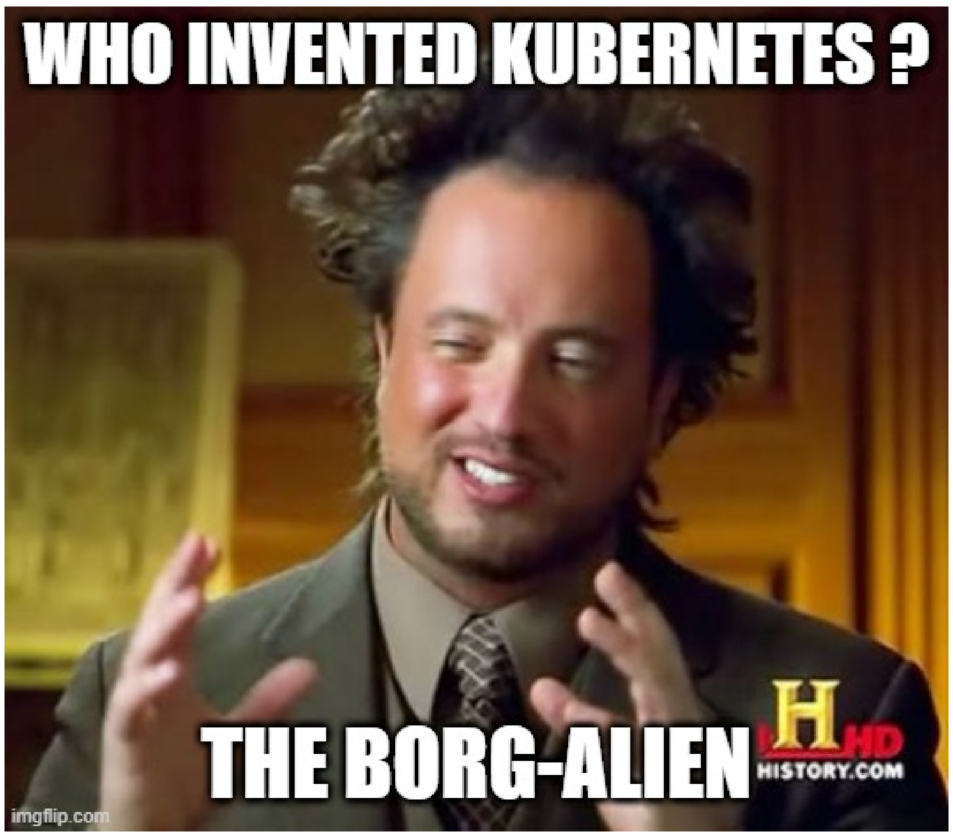
	Name	Namespace	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Age ↑
✓	pyscale-example-7c8499c88d-4ffgw	default	pod-template-hash: 7c8499c88d run: pyscale-example	docker-desktop	Running	0	-	-	11 minutes
✓	pyscale-example-7c8499c88d-vv2wt	default	pod-template-hash: 7c8499c88d run: pyscale-example	docker-desktop	Running	0	-	-	11 minutes
✓	pyscale-example-7c8499c88d-z52d8	default	pod-template-hash: 7c8499c88d run: pyscale-example	docker-desktop	Running	0	-	-	11 minutes

1 – 3 of 3

Destroy

```
kubectl delete deployment pyscale-example
kubectl delete deployments -n kubernetes-dashboard
```

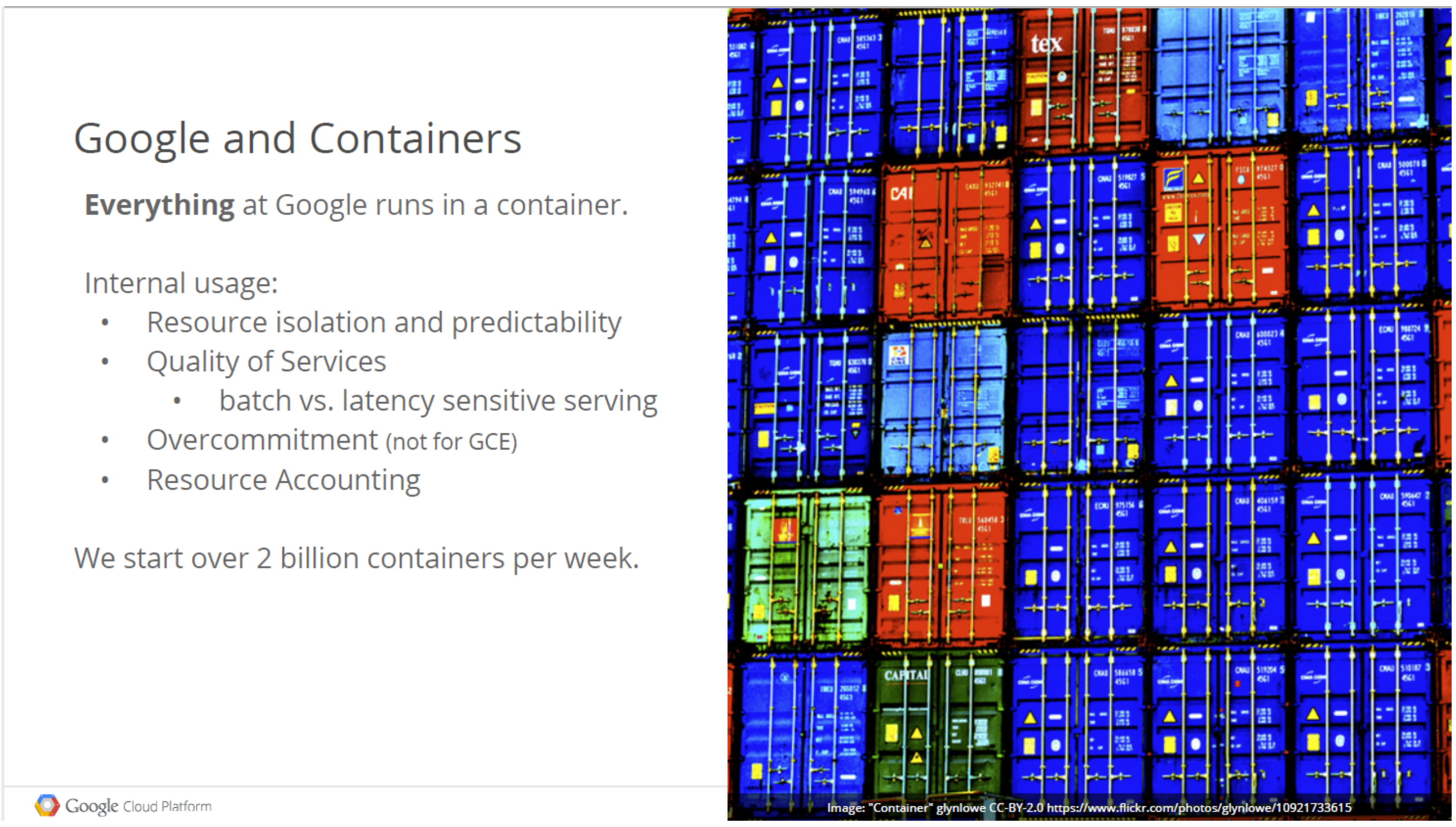
When ?



NicsMeme

Kubernetes combines over 15 years of Google’s experience running production workloads at scale with best-of-breed ideas and practices from the community

The system used in Google was called Borg



Google and Containers

Everything at Google runs in a container.

Internal usage:

- Resource isolation and predictability
- Quality of Services
    - batch vs. latency sensitive serving
- Overcommitment (not for GCE)
- Resource Accounting

We start over 2 billion containers per week.

Google Cloud Platform

Image: "Container" glynlowe CC-BY-2.0 https://www.flickr.com/photos/glynlowe/10921733615

Source

Google open-sourced the Kubernetes project in 2014.

> And it’s in GitHub https://github.com/kubernetes

Source

Origin of the name

The name Kubernetes originates from Greek (κυβερνήτης) , meaning helmsman or pilot.

NicsMeme

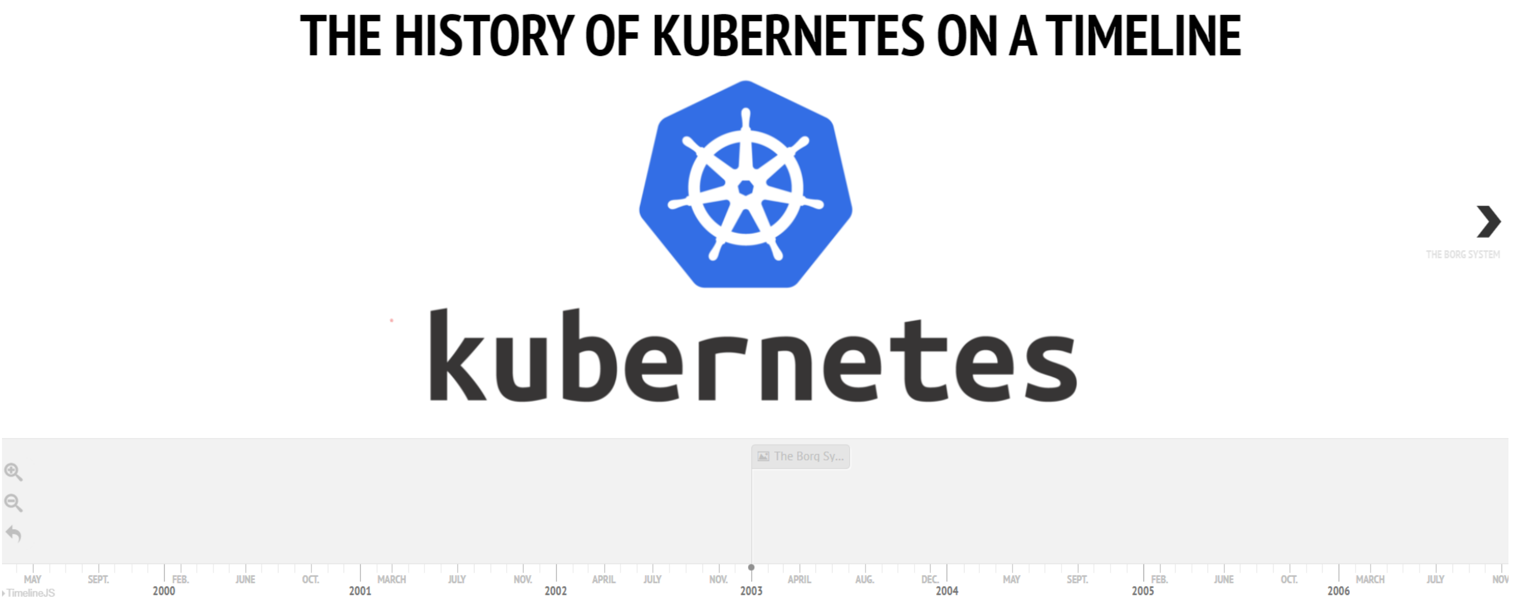
K8s as an abbreviation results from counting the eight letters between the “K” and the “s”.

> K.......s

Kubernetes evolution

Timeline from https://blog.risingstack.com/the-history-of-kubernetes/

Using https://timeline.knightlab.com/



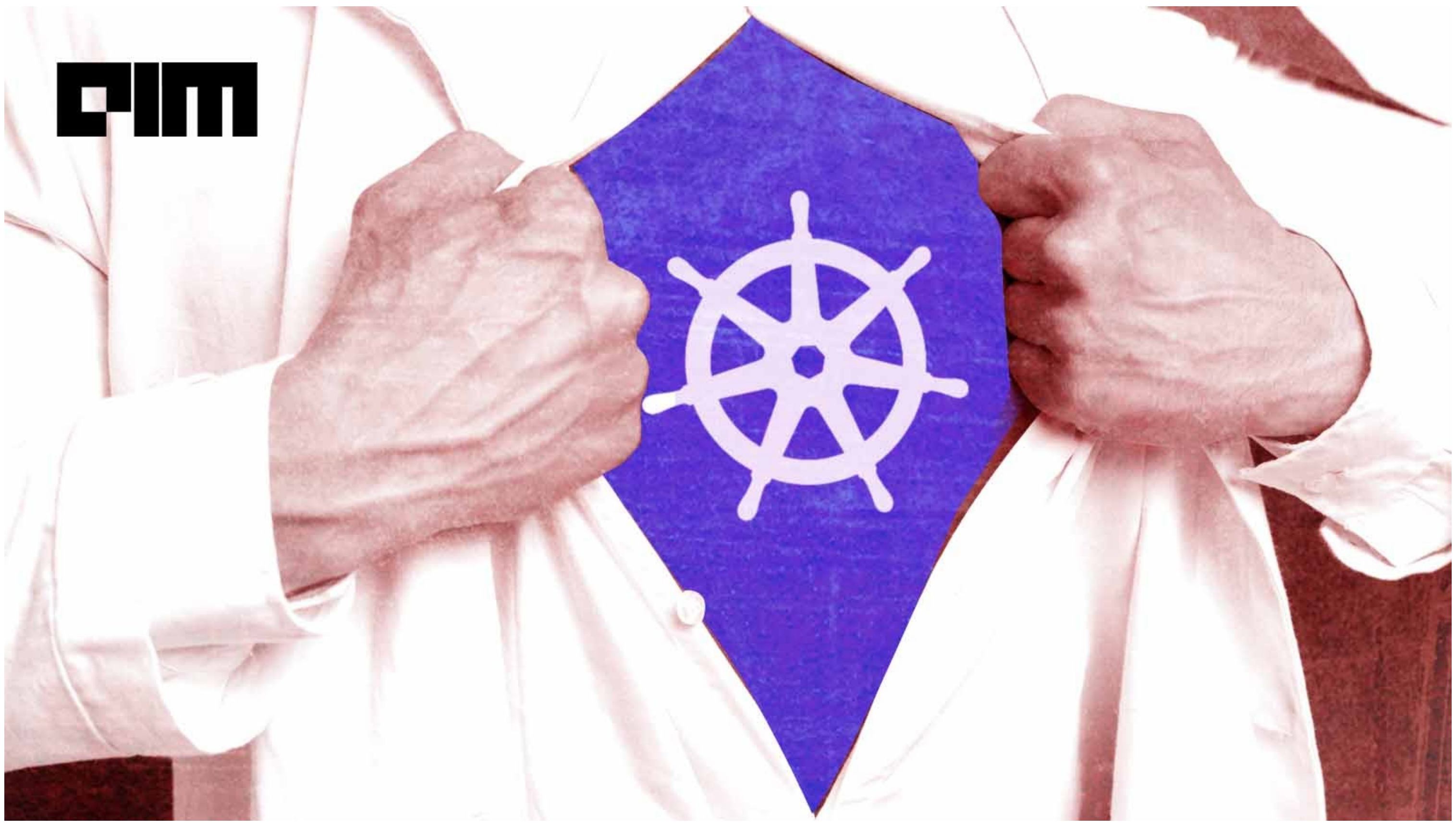
THE HISTORY OF KUBERNETES ON A TIMELINE

Why ?

Why is so popular ?

16 Dec 2019

https://www.forbes.com/sites/janakirammsv/2019/12/16/how-kubernetes-has-changed-the-face-of-hybrid-cloud/?sh=36f92c61228d



16 Mar 2021 https://analyticsindiamag.com/why-is-kubernetes-so-popular/

6 May 2021

https://containerjournal.com/features/findings-from-the-2021-kubernetes-adoption-report/

Adoption

A RightScale report titled, State of the Cloud, said container adoption increased from 49 percent in 2018 to 57 percent in 2019.

Containers run complex and critical enterprise applications, and the rise in their numbers have necessitated the need for a managing system.

Worldwide IT shifts and the agile creed are prompting more and more Kubernetes usage across the board. New research shows that 68% of IT professionals increased their Kubernetes use due to the pandemic.

Community

Kubernetes is one of the largest open source communities, with 75,200 (edit 77k on 23May) stars on GitHub and contributions from thousands of organizations: One of the reasons why it is rated higher than the competitors such as Docker Swarm and Apache Mesos.

The Certified Kubernetes Conformance Program ensures that every vendor's version of Kubernetes supports the required APIs, as do open source community versions.

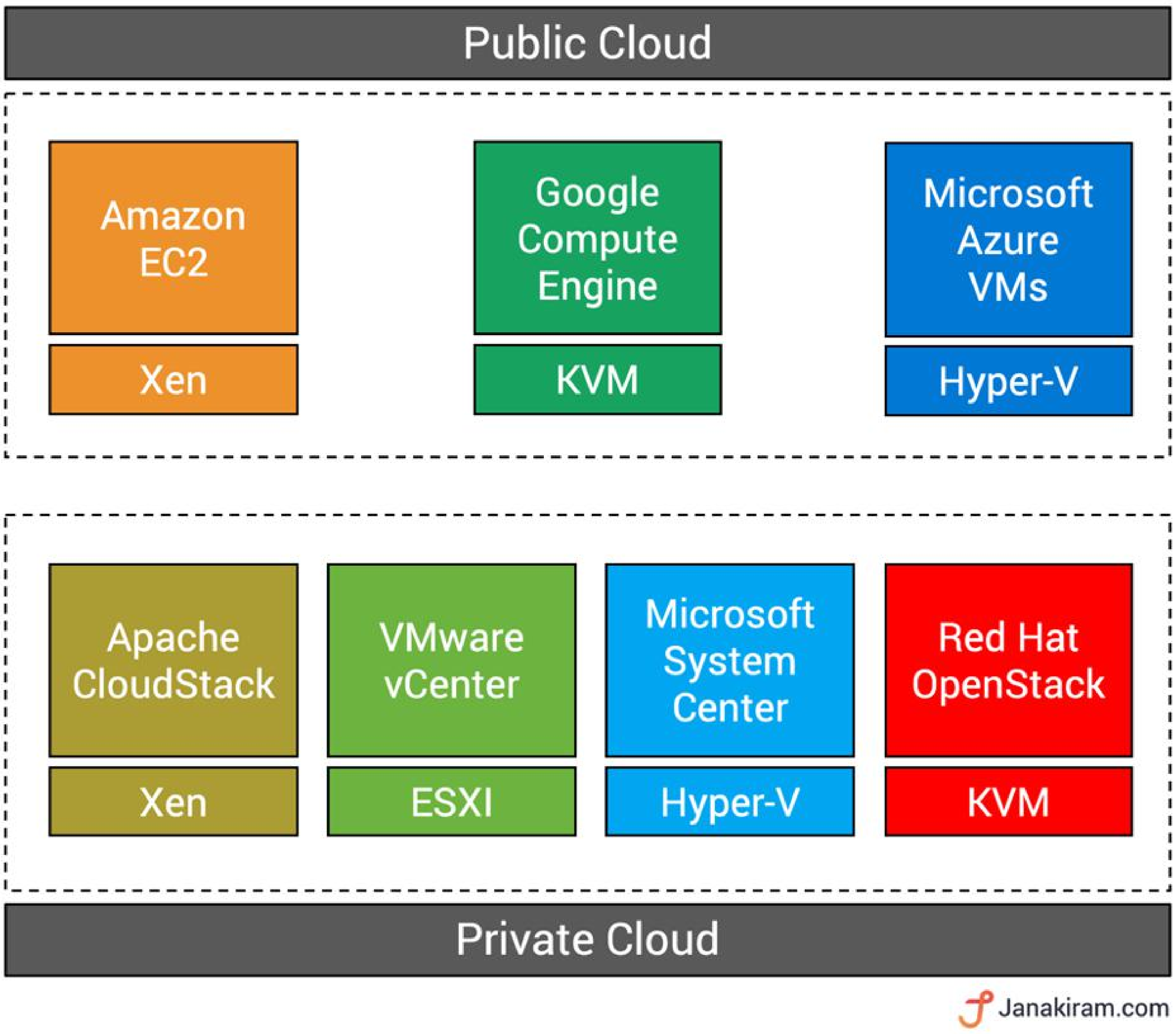
The dream of hybrid cloud

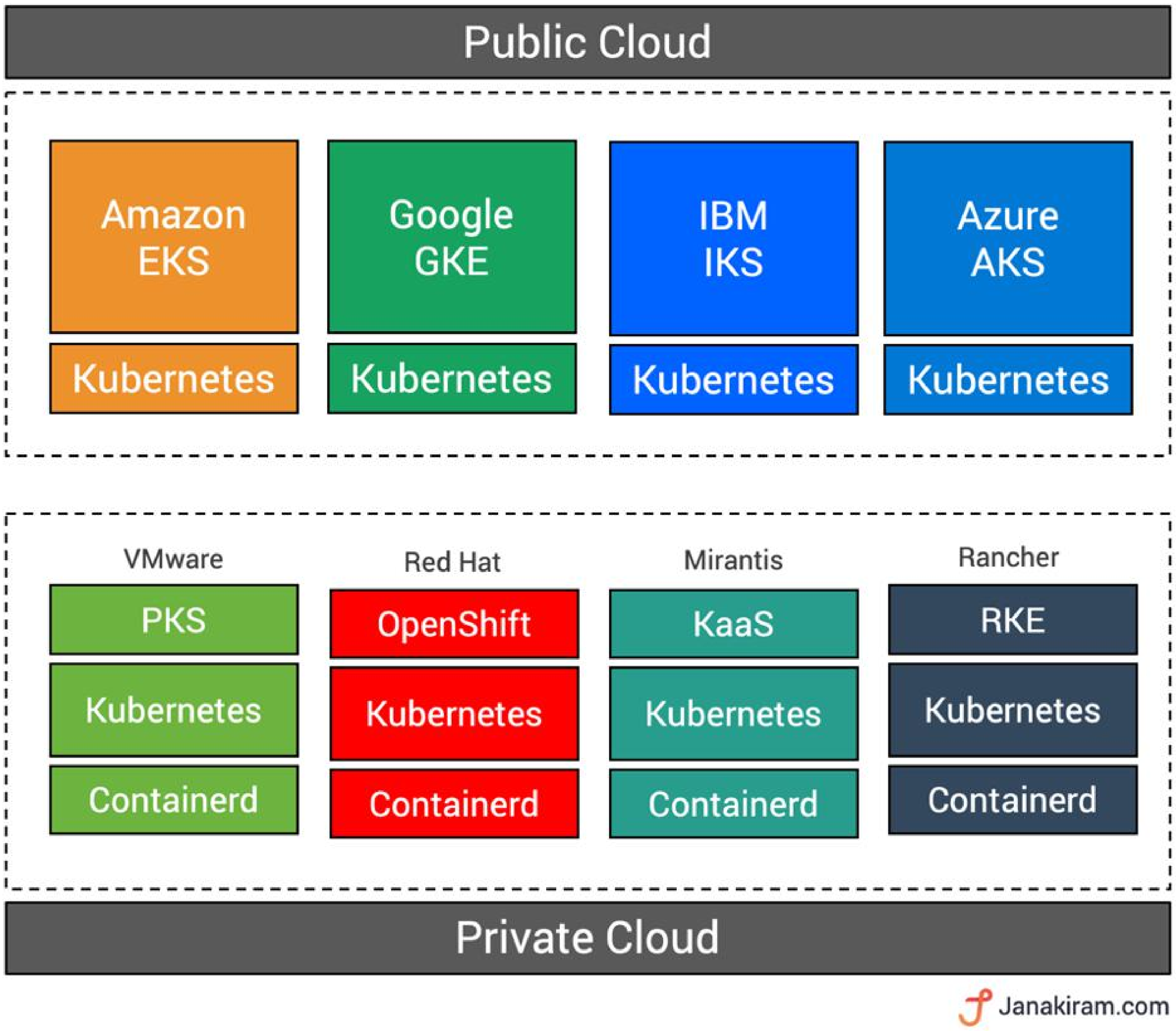
- It provides consistency in both on-premise and public cloud.
- Kubernetes allows users to deploy applications based on their business needs.
- It also offers the ability to automatically scale the applications, leading to better utilisation of the underlying infrastructure.
- Kubernetes automates the deployment of containerized workloads across the hybrid architectures, allowing organisations to deploy and run their containers on servers at different locations.
- Developers can add additional clusters to their existing infrastructure if needed. This reduces the application downtime and improves overall performance.

Hybrid cloud is an IT architecture that incorporates some degree of workload portability, orchestration, and management across 2 or more environments.

Think about it like this: Instead of building a local 2-lane road (fixed middleware instances) to connect 2 interstate highways (a public cloud and a private cloud), you could instead focus on creating an all-purpose vehicle that can drive, fly, and float. Either strategy still gets you from one place to another, but there's a lot less permitting, construction, permanancy, and ecological impact if you focus on a universally capable vehicle.

https://www.redhat.com/en/topics/cloud-computing/what-is-hybrid-cloud



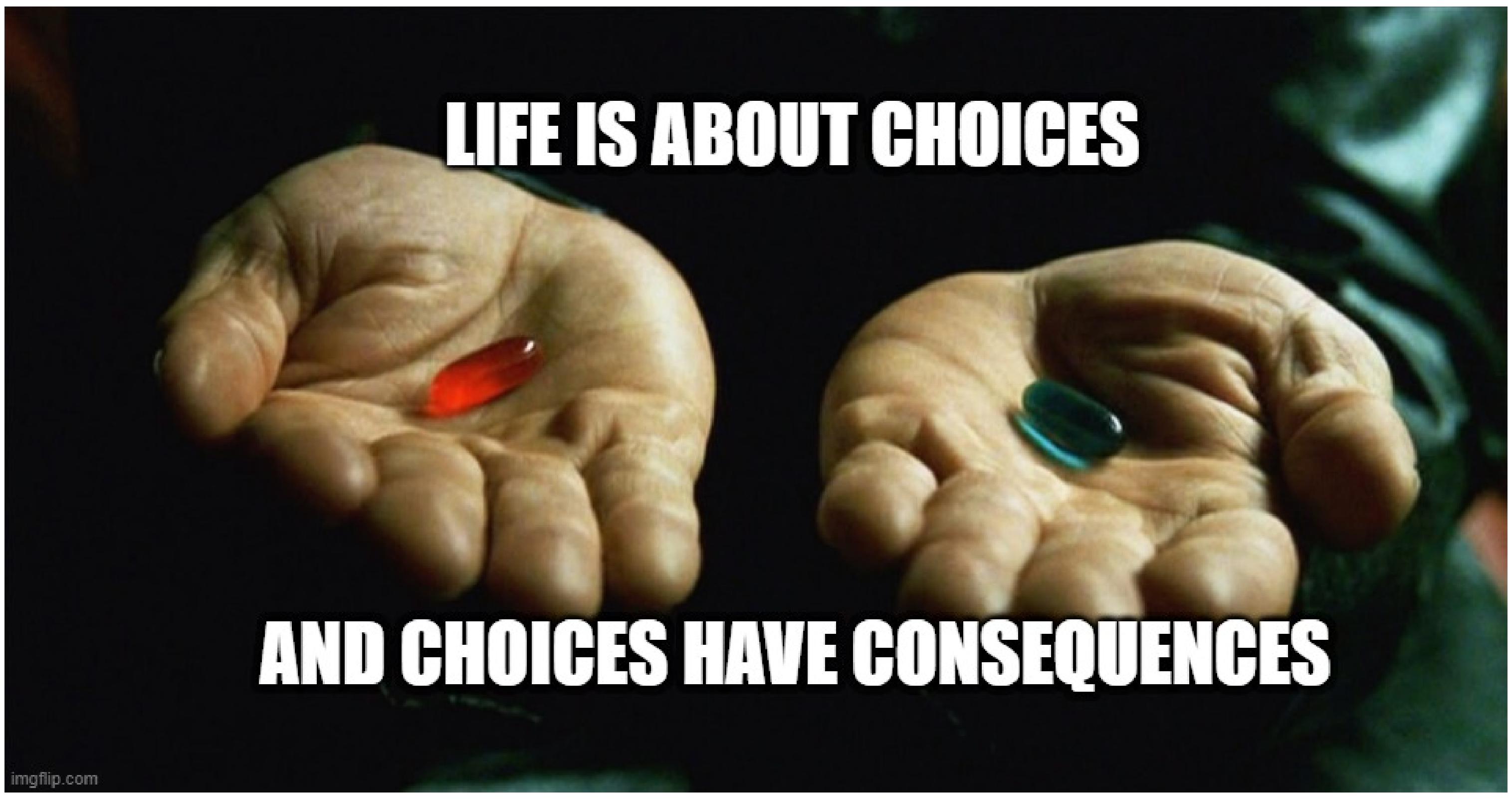


Cost

> I think the value for the CIO level is the following– Today on average 70 percent of your total cost and people are tied up in maintaining what you have, 30 percent is on new. That's the rough rule of thumb. Technologies, like Kubernetes, have taken to where we wanted to go, can flip that to 30%-70%, meaning you need to spend only 30 percent maintaining what you have, and you could, then, go spend 70% on doing innovation, which is going to make your end-client happier, and your business happier, said IBM CEO Arvind Krishna in an interview.

Raising profits by using Kubernetes is likely more of an indirect than direct result, as more than a quarter of respondents said they expect to reduce IT costs by 30% or more annually as a result of Kubernetes.

Why should I learn/use ?



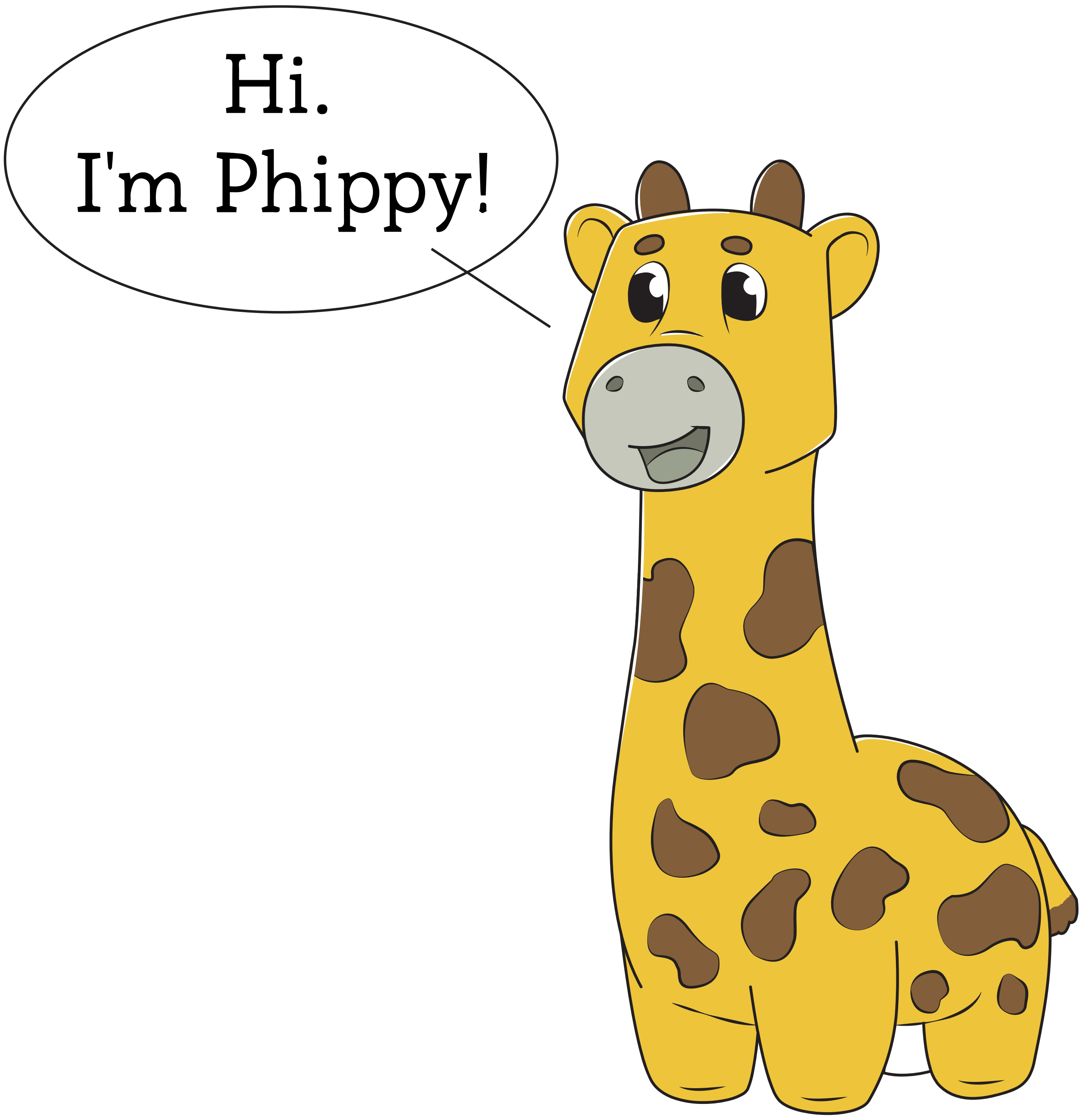
NicsMeme

The terms "red pill" and "blue pill" refer to a choice between the willingness to learn a potentially unsettling or life-changing truth, by taking the red pill, or remaining in contented ignorance with the blue pill. The terms refer to a scene in the 1999 film The Matrix.

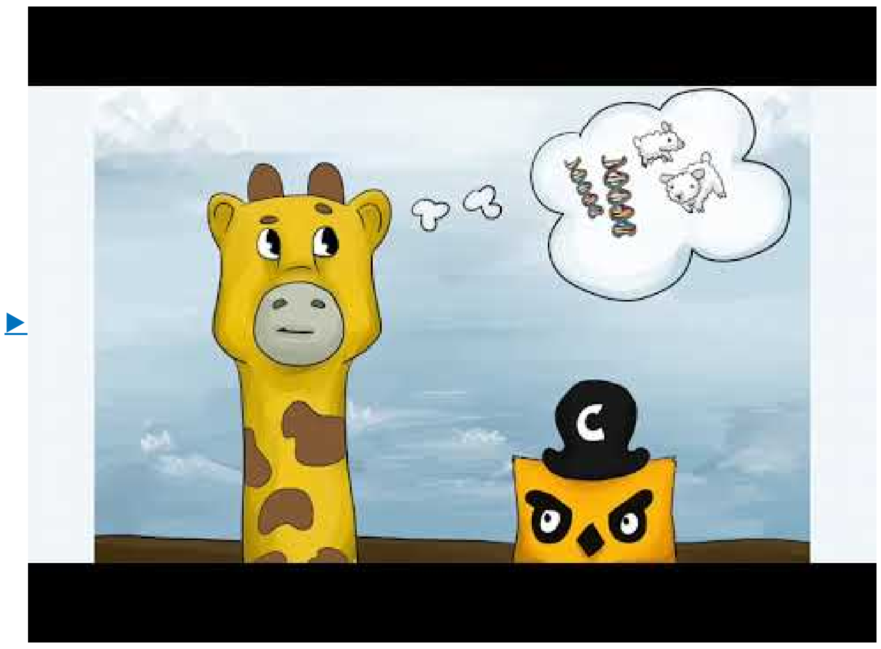
Who ?

Phippy is a simple PHP app, trying to find a home in a cloud native world.

https://www.cncf.io/phippy/



The Illustrated Children’s Guide to Kubernetes

▶ 

Let’s do a music game

Beethoven - Symphony No. 5 (Proms 2012)

▶

So what ?

- Kubernetes is the director
- Containers are the musicians that play their scores

but what is really important is that

- the user just need to listen the symphony (application)

GREAT TECHNOLOGY IS INVISIBLE

June 1995: “And it’s the same with Toy Story. The audience isn’t gonna care about the Pixar animation system, they’re not gonna care about the Pixar production system, they’re not gonna care about anything–except what they will be able to judge for themselves, and that’s the end result, which they can appreciate without having to understand what went into it, what went into creating it. And that, I love. “

Source

Where ?

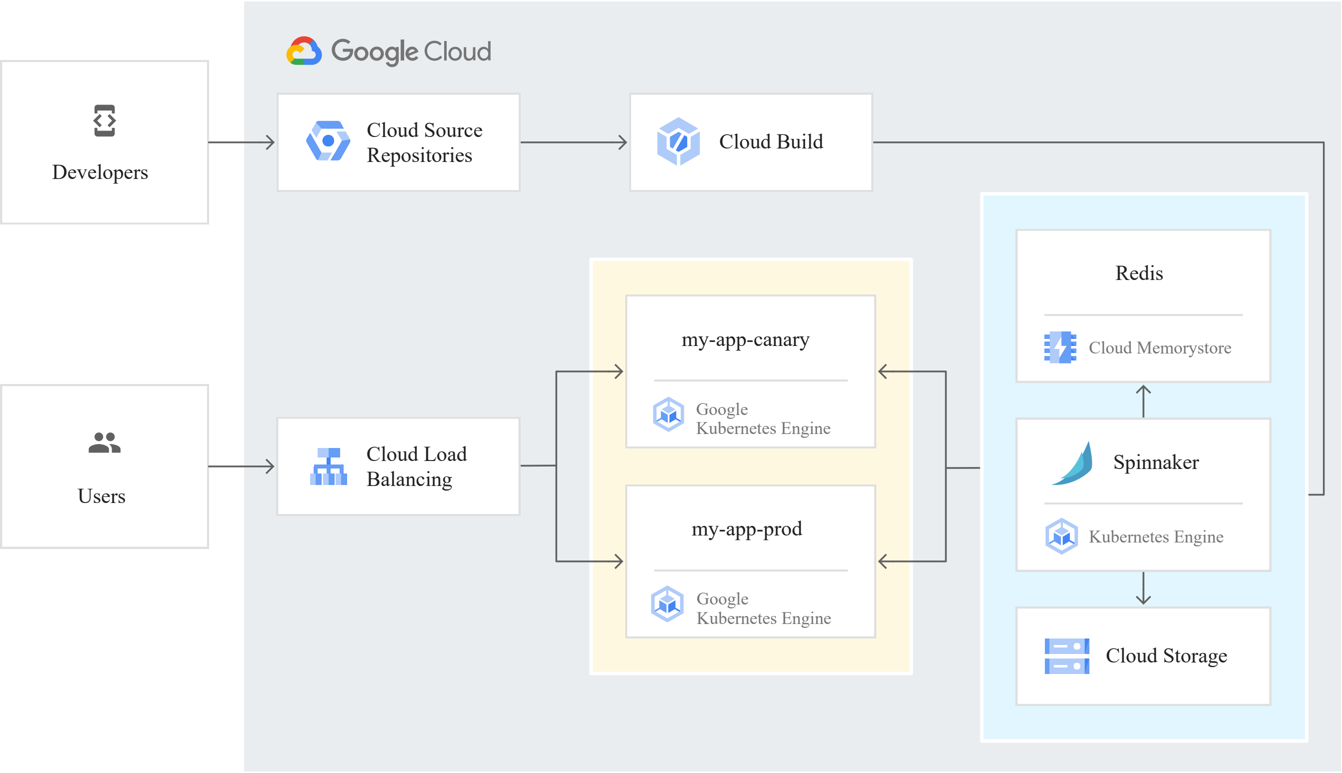
NicsMeme

In the public cloud

Google Kubernetes Engine

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

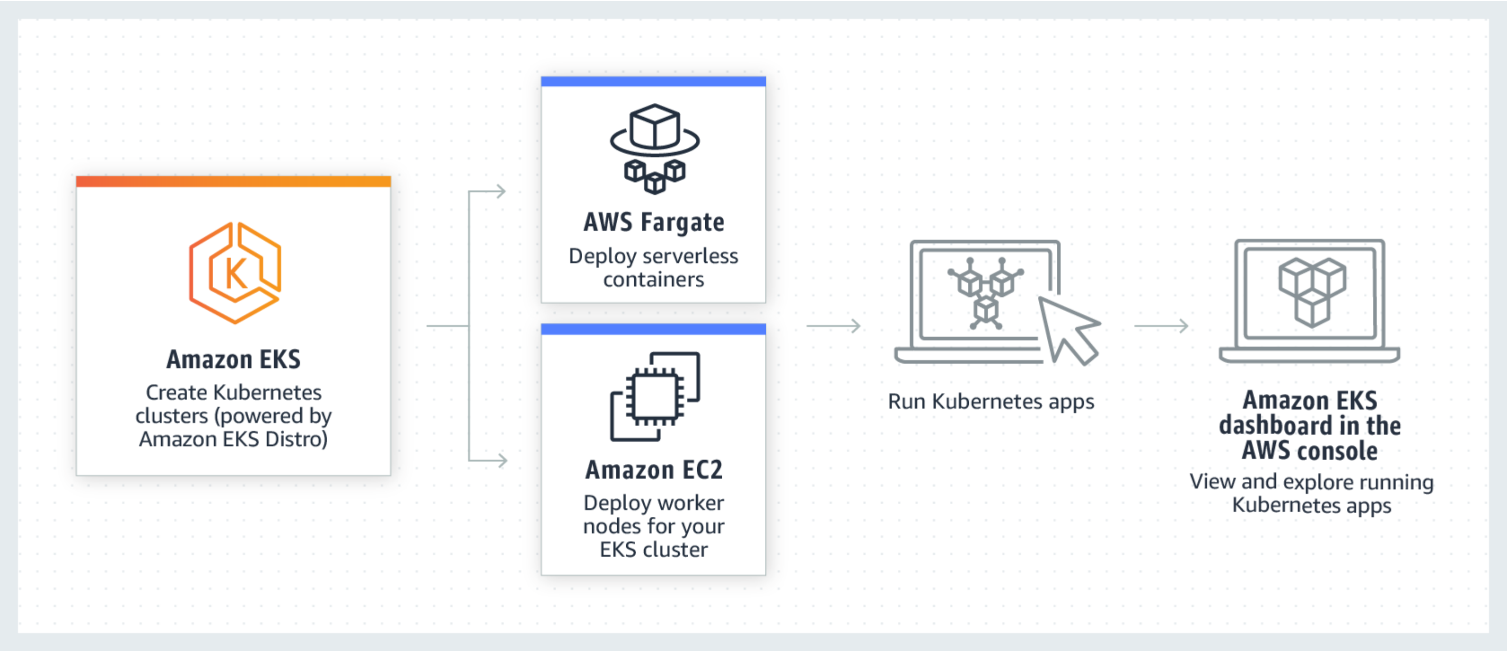
> Secured and fully managed Kubernetes service with revolutionary autopilot mode of operation.



AWS Amazon Elastic Kubernetes Service

https://aws.amazon.com/it/eks/

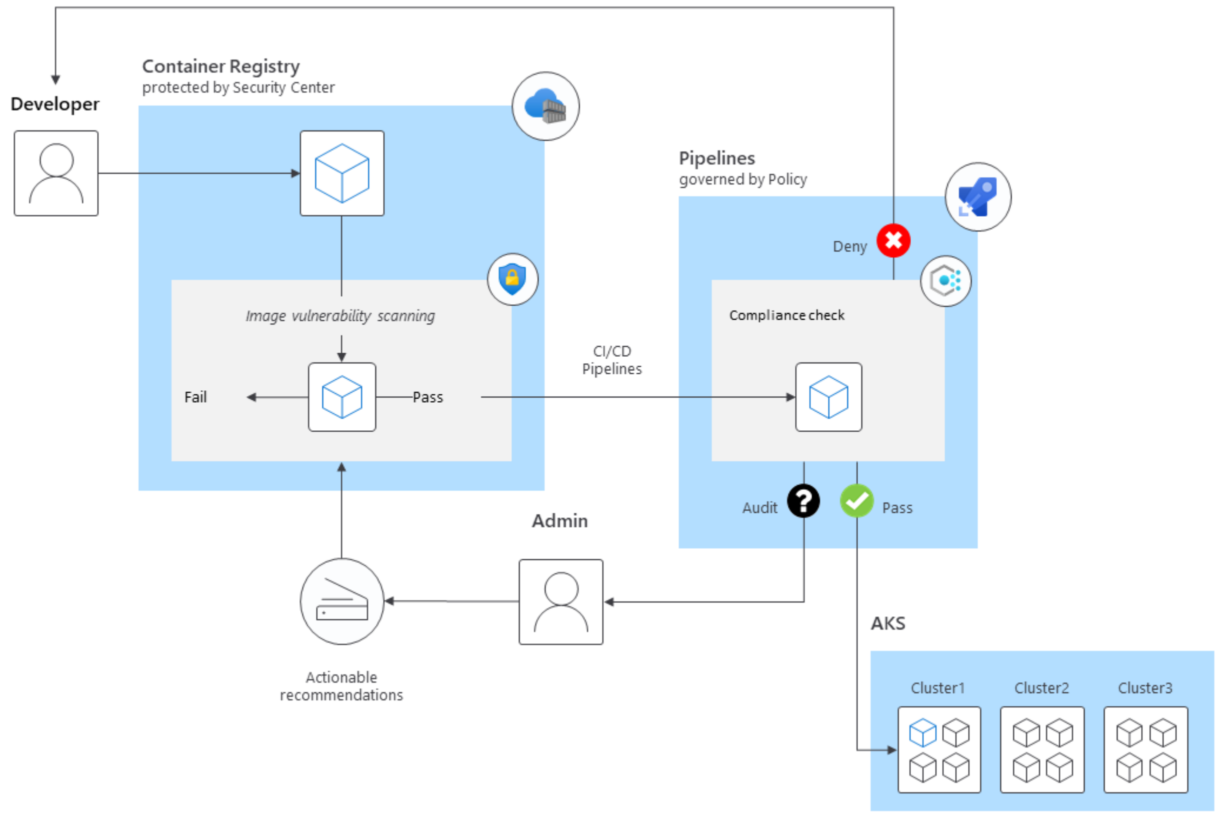
> The most trusted way to run Kubernetes

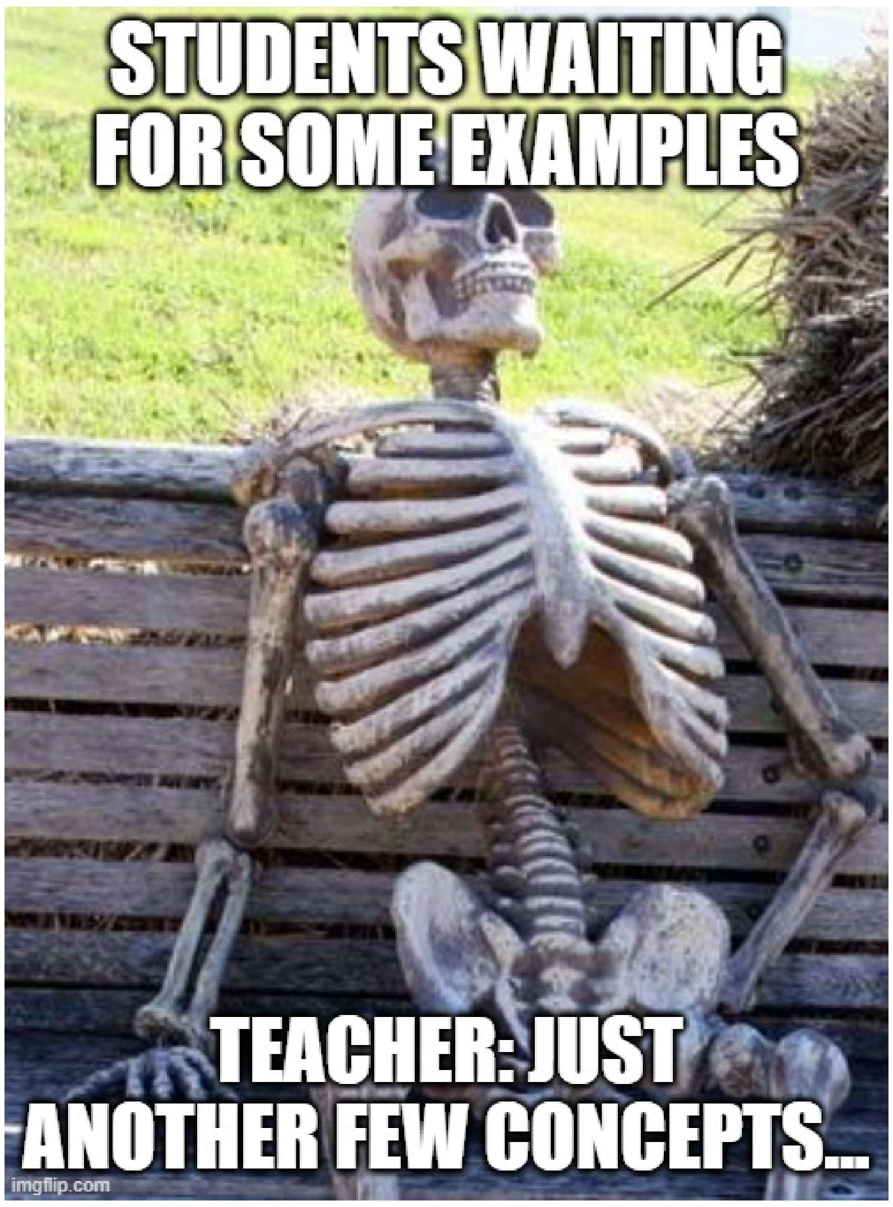

Amazon EKS
Create Kubernetes clusters (powered by Amazon EKS Distro)
AWS Fargate
Deploy serverless containers
Amazon EC2
Deploy worker nodes for your EKS cluster
Run Kubernetes apps
Amazon EKS dashboard in the AWS console
View and explore running Kubernetes apps

Azure

https://azure.microsoft.com/en-us/services/kubernetes-service/

> Highly available, secure, and fully managed Kubernetes service

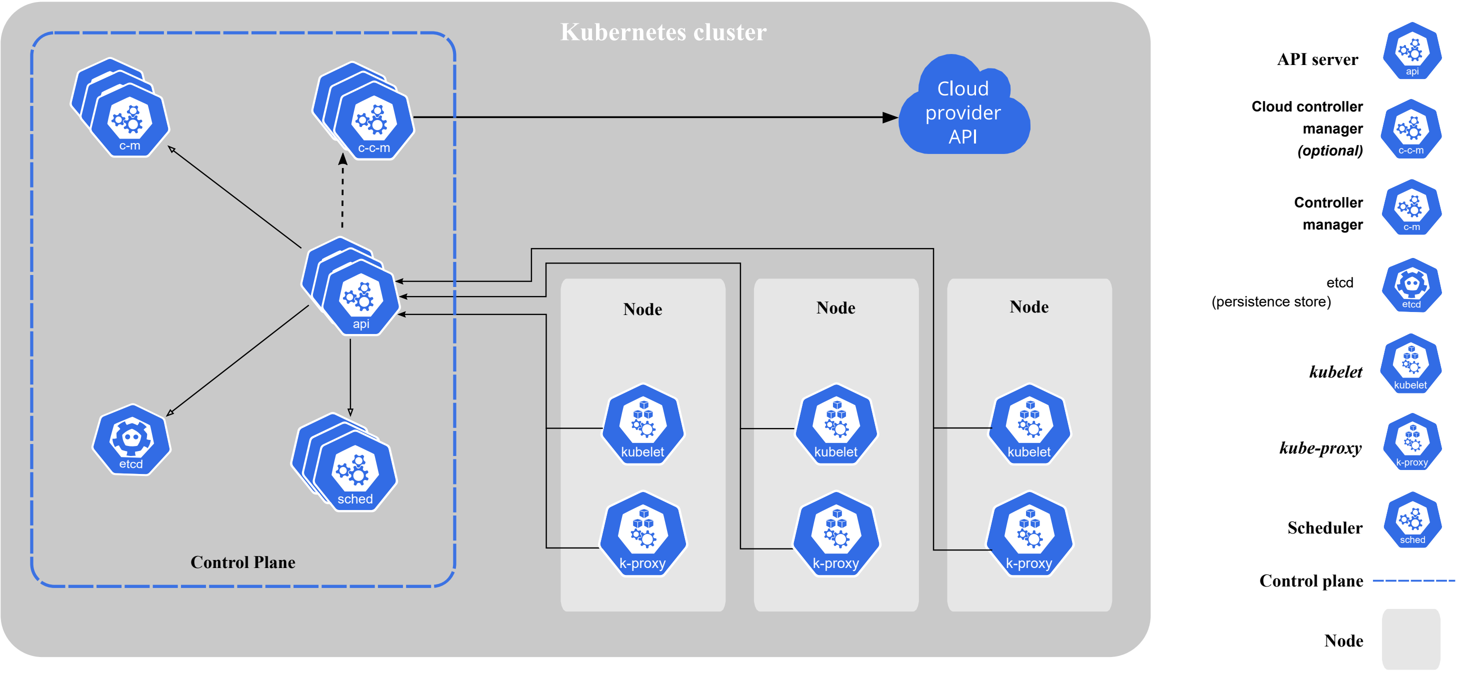



STUDENTS WAITING FOR SOME EXAMPLES
TEACHER: JUST ANOTHER FEW CONCEPTS...
imgflip.com

NicsMeme

How

Components



https://kubernetes.io/docs/concepts/overview/components/

Control Panel

The control plane's components make global decisions about the cluster (for example, scheduling), as well as detecting and responding to cluster events (for example, starting up a new pod when a deployment's replicas field is unsatisfied).

The Kubernetes control plane consists of various components, each its own process, that can run both on a single master node or on multiple masters supporting high-availability clusters.[

etcd

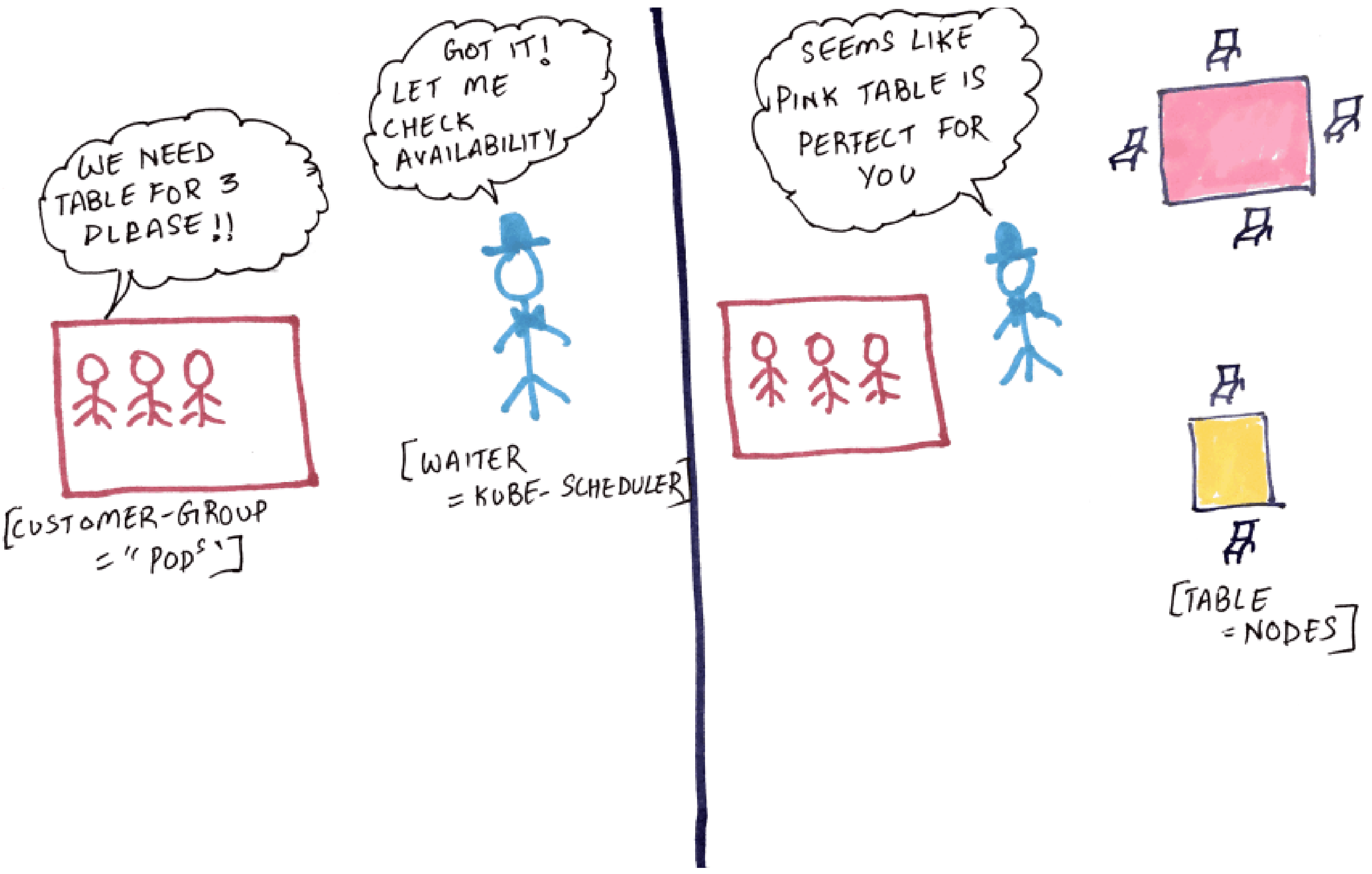
etcd
A distributed, reliable key-value store for the most critical data of a distributed system

https://etcd.io/ is a CNCF project

Consistent and highly-available key value store used as Kubernetes' backing store for all cluster data.

If your Kubernetes cluster uses etcd as its backing store, make sure you have a back up plan for those data.

scheduler


WE NEED TABLE FOR 3 PLEASE !!
GOT IT! LET ME CHECK AVAILABILITY
[CUSTOMER-GROUP = "PODS"]
[WAITER = KUBE- SCHEDULER]
SEEMS LIKE PINK TABLE IS PERFECT FOR YOU
[TABLE = NODES]

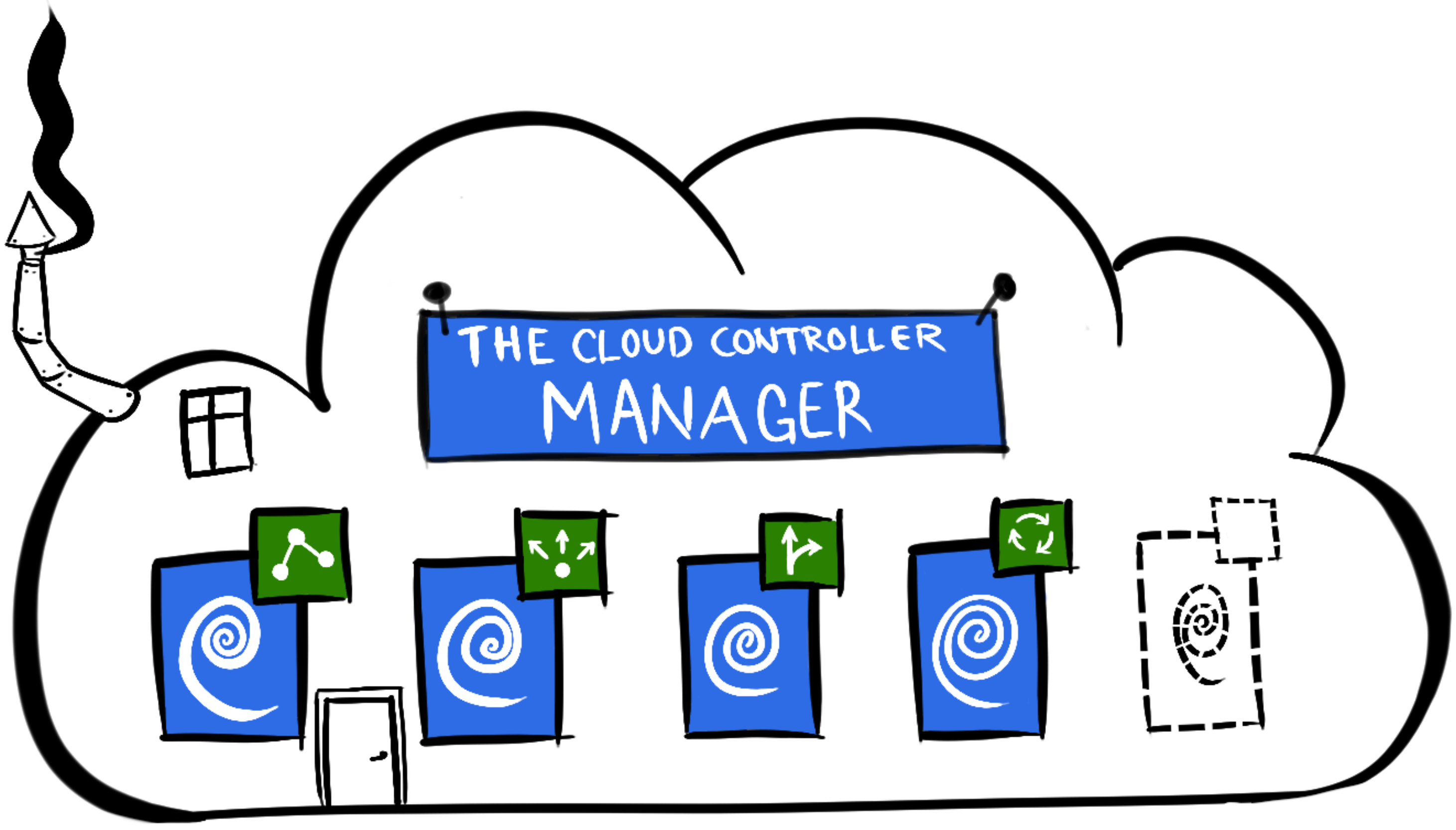
https://dev.to/ranand12/kubernetes-scheduler-visually-explained-in-plain-english-with-a-story-5h0g

Control plane component that watches for newly created Pods with no assigned node, and selects a node for them to run on.

Factors taken into account for scheduling decisions include: individual and collective resource requirements, hardware/software/policy constraints, affinity and anti-affinity specifications, data locality, inter-workload interference, and deadlines

kube-scheduler is the default scheduler for Kubernetes and runs as part of the control plane. kube-scheduler is designed so that, if you want and need to, you can write your own scheduling component and use that instead.

control manager



https://medium.com/@m.json/the-kubernetes-cloud-controller-manager-d440af0d2be5

Control Plane component that runs controller processes.

Logically, each controller is a separate process, but to reduce complexity, they are all compiled into a single binary and run in a single process.

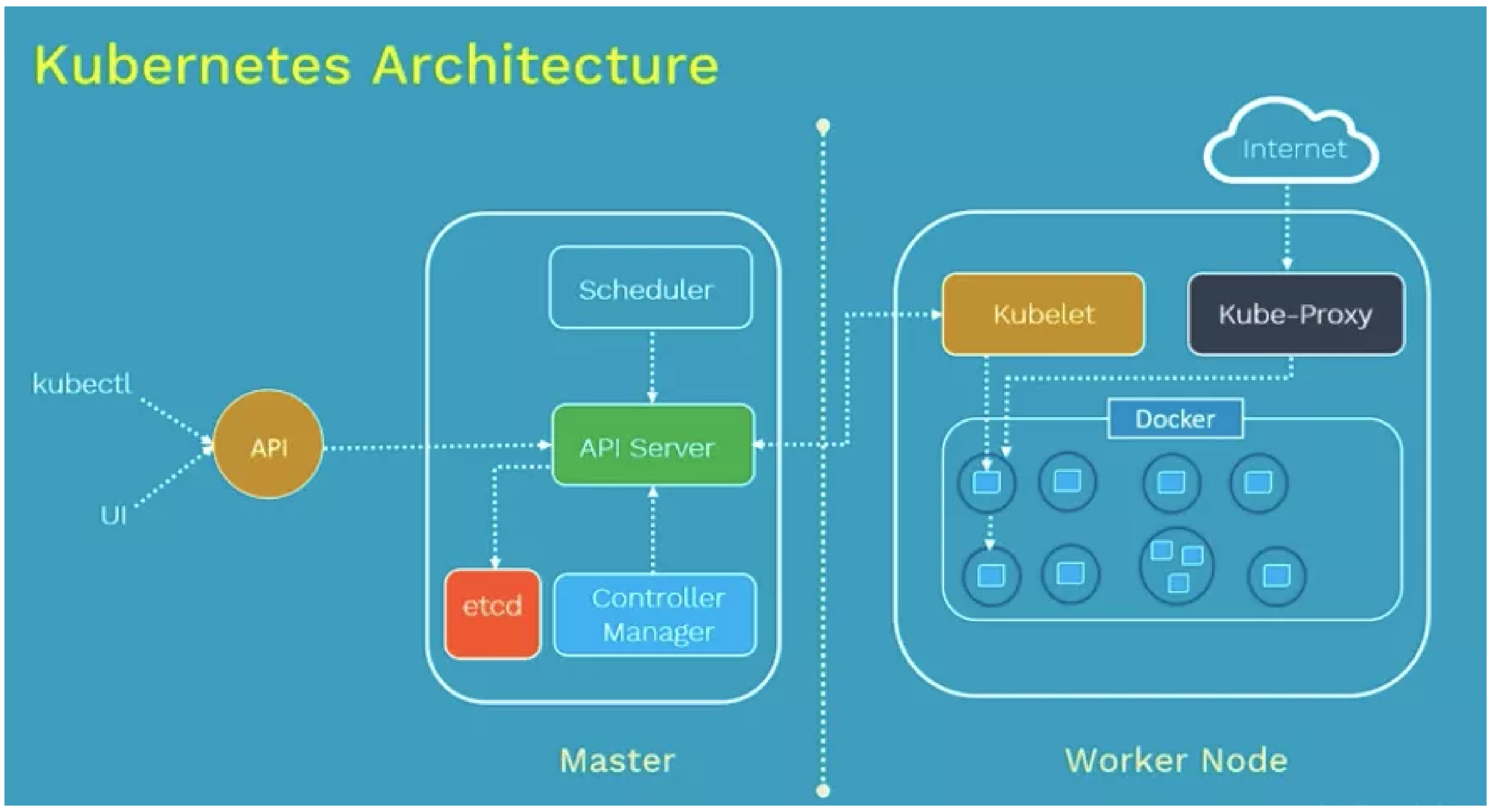
Some types of these controllers are:

- Node controller: Responsible for noticing and responding when nodes go down.
- Job controller: Watches for Job objects that represent one-off tasks, then creates Pods to run those tasks to completion.
- Endpoints controller: Populates the Endpoints object (that is, joins Services & Pods).
- Service Account & Token controllers: Create default accounts and API access tokens for new namespaces

api server

The API server is a component of the Kubernetes control plane that exposes the Kubernetes API. The API server is the front end for the Kubernetes control plane.

The main implementation of a Kubernetes API server is kube-apiserver. kube-apiserver is designed to scale horizontally—that is, it scales by deploying more instances. You can run several instances of kube-apiserver and balance traffic between those instances



https://managedserver.it/kubernetes-tanto-potente-quanto-difficile/

Yes they are running

```
%%bash
kubectl get pods -A
```

```
NAMESPACE              NAME                                         READY   STATUS
RESTARTS   AGE
default                pyscale-example-7c8499c88d-8n7md             1/1     Running   1
48m
default                pyscale-example-7c8499c88d-jqkn2             1/1     Running   0
48m
default                pyscale-example-7c8499c88d-vx8dx             1/1     Running   0
48m
kube-system            coredns-f9fd979d6-4tb7l                      1/1     Running   5
14d
kube-system            coredns-f9fd979d6-9pstt                      1/1     Running   5
14d
kube-system            etcd-docker-desktop                          1/1     Running   5
14d
kube-system            kube-apiserver-docker-desktop                1/1     Running   6
14d
kube-system            kube-controller-manager-docker-desktop       1/1     Running   5
14d
kube-system            kube-proxy-wztvz                             1/1     Running   5
14d
kube-system            kube-scheduler-docker-desktop                1/1     Running   10
14d
kube-system            storage-provisioner                          1/1     Running   8
14d
kube-system            vpnkit-controller                            1/1     Running   5
14d
kubernetes-dashboard   dashboard-metrics-scraper-74db988864-rrjfq   1/1     Running   0
42m
kubernetes-dashboard   kubernetes-dashboard-847d8c7cdc-vws57        1/1     Running   0
42m
```

Node

Kubernetes runs your workload by placing containers into Pods to run on Nodes.

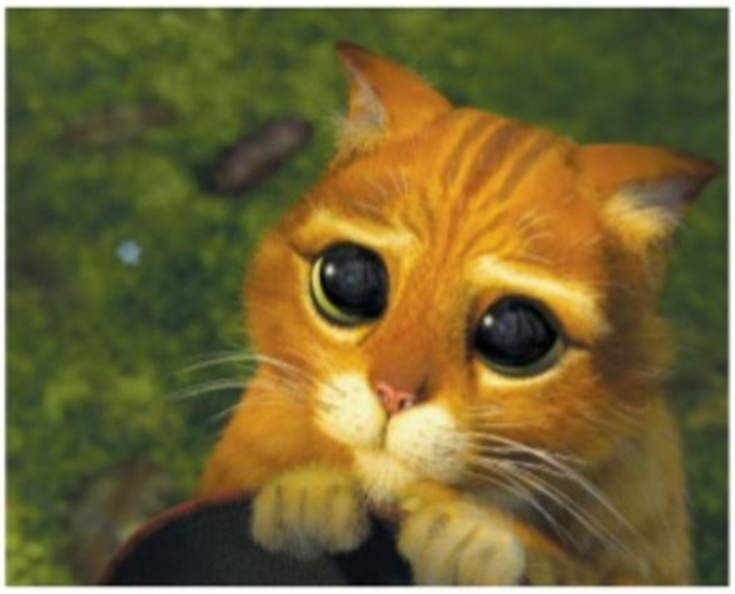
A node may be a virtual or physical machine, depending on the cluster.

Each node is managed by the control plane and contains the services necessary to run Pods

Nodes are generic Linux machines, they need to be installed / managed...

> If you have adopted the cattle-not-pets view of container management — destroying a container and launching a new version when an update or fix is to be deployed — then it makes sense to ensure the same approach is adopted for the infrastructure that supports the containers.

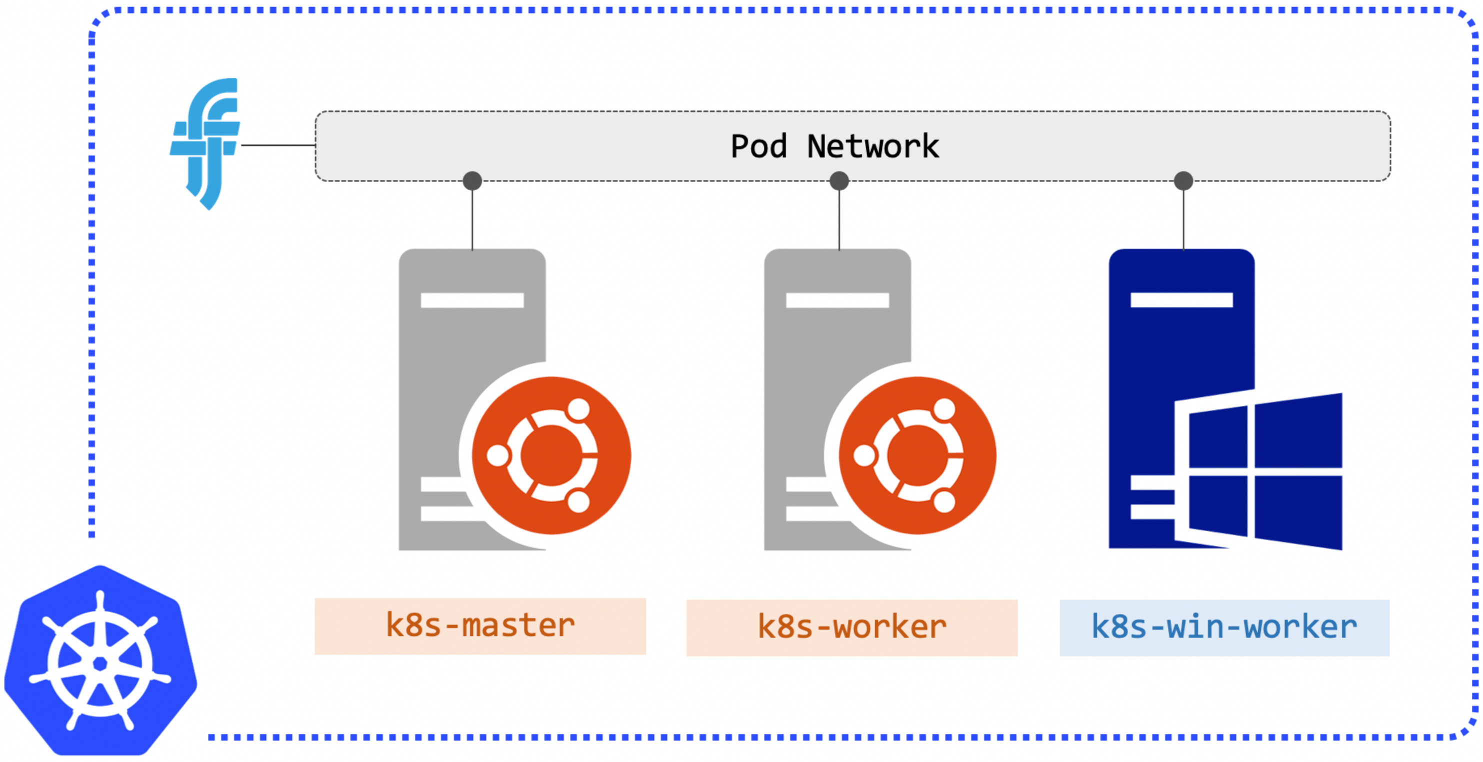
https://thenewstack.io/a-guide-to-linux-operating-systems-for-kubernetes/



https://devops.stackexchange.com/questions/653/what-is-the-definition-of-cattle-not-pets

However windows nodes are coming...

https://kubernetes.io/docs/setup/production-environment/windows/intro-windows-in-kubernetes/



https://blog.sixeyed.com/getting-started-with-kubernetes-on-windows/

- Nodes can joins the cluster using kubelet or it can manually added
- The name of a Node object must be a valid DNS subdomain name.
- The name identifies a Node. Two Nodes cannot have the same name at the same time. Kubernetes also assumes that a resource with the same name is the same object.
- The components on a node include the kubelet, a container runtime, and the kube-proxy.

https://kubernetes.io/docs/concepts/architecture/nodes/

kubelet

An agent that runs on each node in the cluster.

It makes sure that containers are running in a Pod.

The kubelet takes a set of PodSpecs that are provided through various mechanisms and ensures that the containers described in those PodSpecs are running and healthy.

The kubelet doesn't manage containers which were not created by Kubernetes

NicsMeme

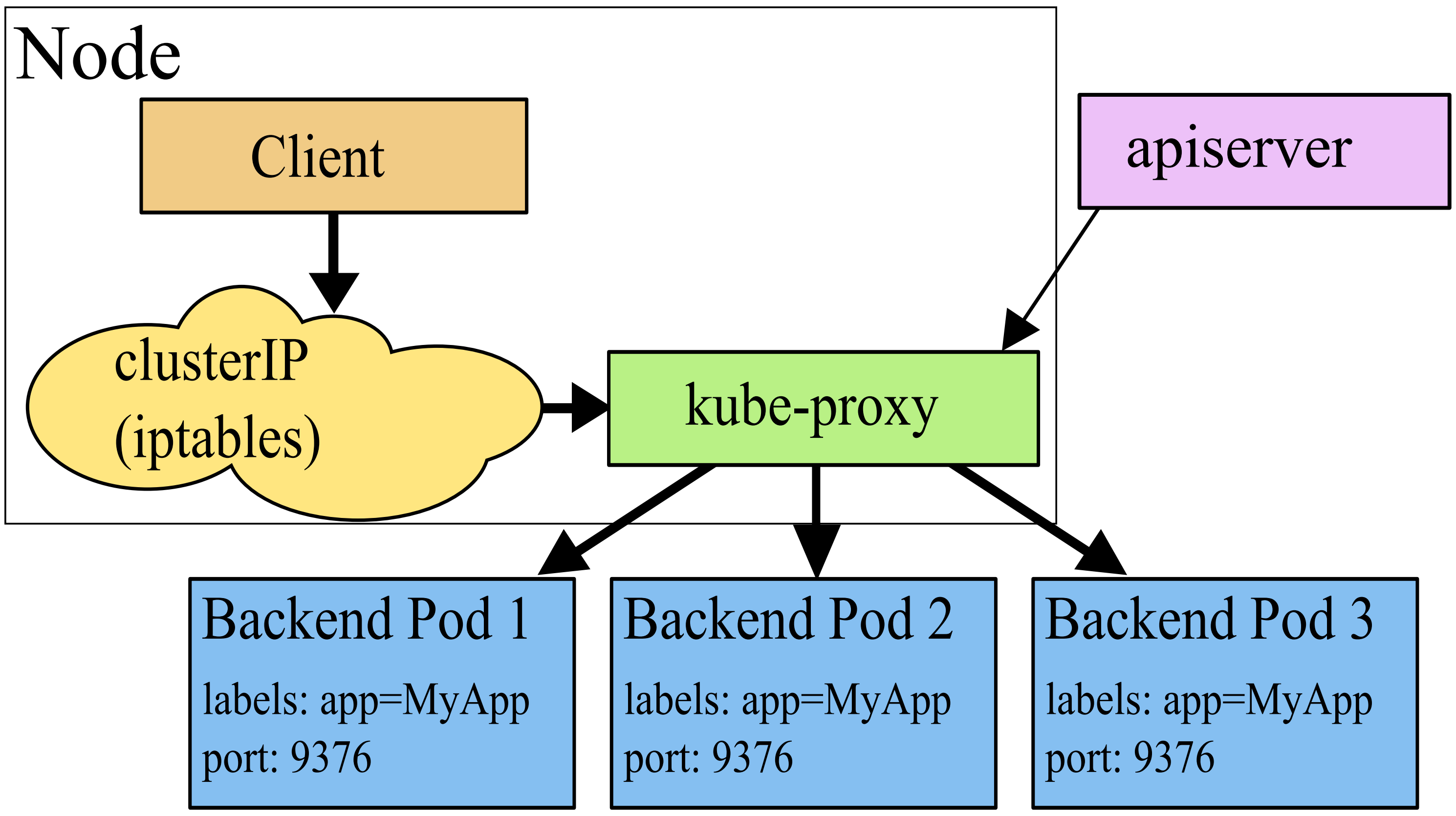
kube-proxy

kube-proxy is a network proxy that runs on each node in your cluster, implementing part of the Kubernetes Service concept.

kube-proxy maintains network rules on nodes. These network rules allow network communication to your Pods from network sessions inside or outside of your cluster.

kube-proxy uses the operating system packet filtering layer if there is one and it's available. Otherwise, kube-proxy forwards the traffic itself.

https://kubernetes.io/docs/reference/command-line-tools-reference/kube-proxy/



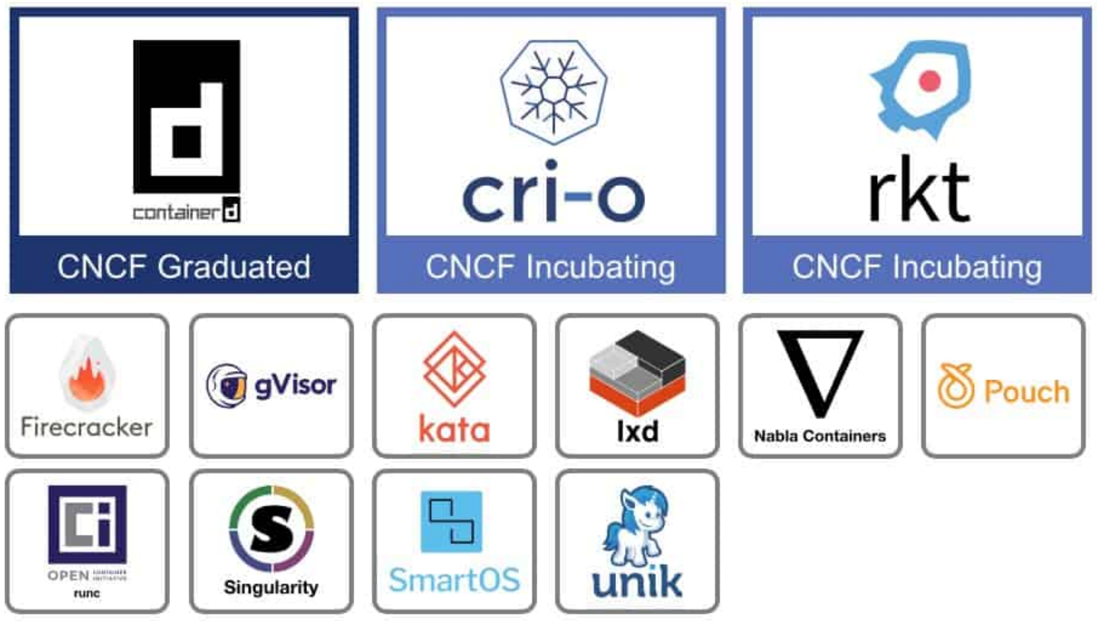
https://kubernetes.io/docs/concepts/services-networking/service/

container runtime

The container runtime is the software that is responsible for running containers.

Kubernetes supports several container runtimes: Docker, containerd, CRI-O, and any implementation of the Kubernetes CRI (Container Runtime Interface).

https://kubernetes.io/docs/setup/production-environment/container-runtimes/



https://www.cncf.io/blog/2019/07/15/demystifying-containers-part-ii-container-runtimes/

Do we have one ?

```
%%bash
kubectl get nodes
```

```
NAME             STATUS   ROLES    AGE   VERSION
docker-desktop   Ready    master   14d   v1.19.3
```

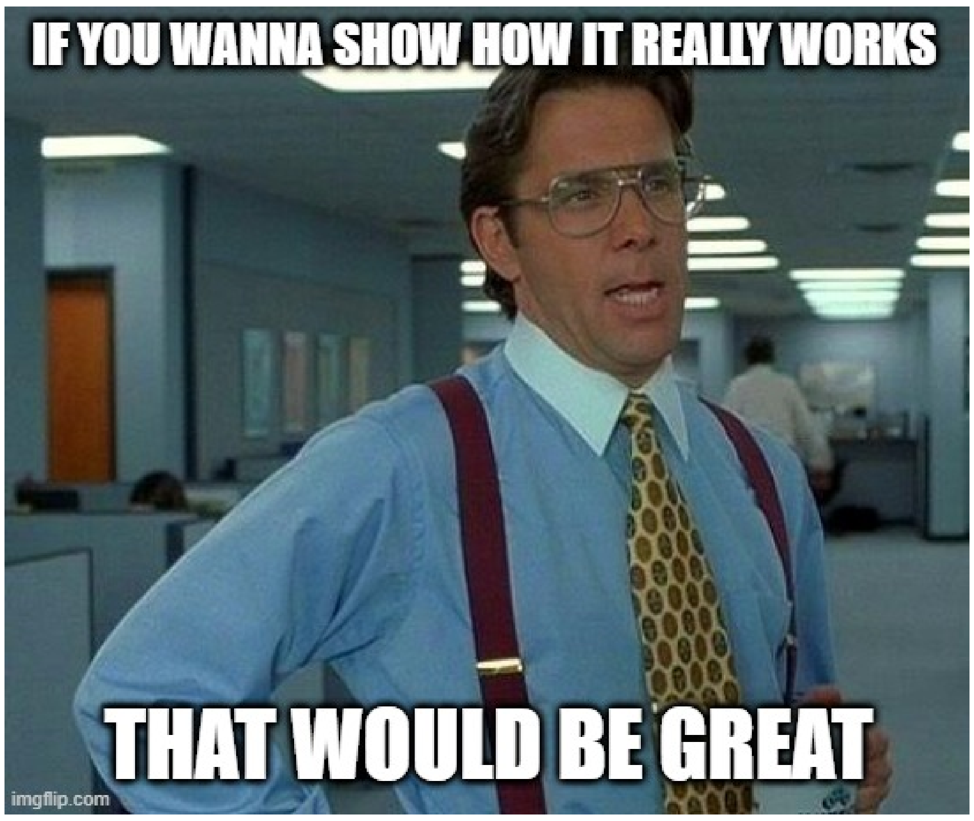
```
%%bash
kubectl describe node
```

```
Name:               docker-desktop
Roles:              master
Labels:             beta.kubernetes.io/arch=amd64
                    beta.kubernetes.io/os=linux
                    kubernetes.io/arch=amd64
                    kubernetes.io/hostname=docker-desktop
                    kubernetes.io/os=linux
                    node-role.kubernetes.io/master=
Annotations:        kubeadm.alpha.kubernetes.io/cri-socket: /var/run/dockershim.sock
                    node.alpha.kubernetes.io/ttl: 0
                    volumes.kubernetes.io/controller-managed-attach-detach: true
CreationTimestamp:  Fri, 14 May 2021 23:00:39 +0200
Taints:             <none>
Unschedulable:      false
Lease:
  HolderIdentity:  docker-desktop
  AcquireTime:     <unset>
  RenewTime:       Sat, 29 May 2021 12:04:19 +0200
Conditions:
  Type             Status  LastHeartbeatTime                 LastTransitionTime
Reason                       Message
  ----             ------  -----------------                 ------------------
------                       -------
  MemoryPressure   False   Sat, 29 May 2021 12:04:04 +0200   Tue, 25 May 2021 22:20:43 +0200
KubeletHasSufficientMemory   kubelet has sufficient memory available
  DiskPressure     False   Sat, 29 May 2021 12:04:04 +0200   Tue, 25 May 2021 22:20:43 +0200
KubeletHasNoDiskPressure     kubelet has no disk pressure
  PIDPressure      False   Sat, 29 May 2021 12:04:04 +0200   Tue, 25 May 2021 22:20:43 +0200
KubeletHasSufficientPID      kubelet has sufficient PID available
  Ready            True    Sat, 29 May 2021 12:04:04 +0200   Tue, 25 May 2021 22:20:43 +0200
KubeletReady                 kubelet is posting ready status
Addresses:
  InternalIP:  192.168.65.4
  Hostname:    docker-desktop
Capacity:
  cpu:                12
  ephemeral-storage:  263174212Ki
  hugepages-1Gi:      0
  hugepages-2Mi:      0
  memory:             8033972Ki
  pods:               110
Allocatable:
  cpu:                12
  ephemeral-storage:  242541353378
  hugepages-1Gi:      0
  hugepages-2Mi:      0
  memory:             7931572Ki
  pods:               110
System Info:
  Machine ID:                 aad4606b-d2a1-43fd-acab-75c4c33a1a01
  System UUID:                aad4606b-d2a1-43fd-acab-75c4c33a1a01
  Boot ID:                    e4a875a1-e805-4004-826d-c539527dd091
  Kernel Version:             5.10.16.3-microsoft-standard-WSL2
  OS Image:                   Docker Desktop
  Operating System:           linux
  Architecture:               amd64
  Container Runtime Version:  docker://20.10.5
  Kubelet Version:            v1.19.3
  Kube-Proxy Version:         v1.19.3
Non-terminated Pods:          (14 in total)
  Namespace                  Name                                          CPU Requests  CPU
Limits  Memory Requests  Memory Limits  AGE
  ---------                  ----                                          ------------  ---
-------  ---------------  -------------  ---
  default                    pyscale-example-7c8499c88d-8n7md              0 (0%)        0
(0%)      0 (0%)           0 (0%)         58m
  default                    pyscale-example-7c8499c88d-jqkn2              0 (0%)        0
(0%)      0 (0%)           0 (0%)         58m
  default                    pyscale-example-7c8499c88d-vx8dx              0 (0%)        0
(0%)      0 (0%)           0 (0%)         58m
  kube-system                coredns-f9fd979d6-4tb7l                       100m (0%)     0
(0%)      70Mi (0%)        170Mi (2%)     14d
  kube-system                coredns-f9fd979d6-9pstt                       100m (0%)     0
(0%)      70Mi (0%)        170Mi (2%)     14d
  kube-system                etcd-docker-desktop                           0 (0%)        0
(0%)      0 (0%)           0 (0%)         14d
  kube-system                kube-apiserver-docker-desktop                 250m (2%)     0
(0%)      0 (0%)           0 (0%)         14d
  kube-system                kube-controller-manager-docker-desktop        200m (1%)     0
(0%)      0 (0%)           0 (0%)         14d
  kube-system                kube-proxy-wztvz                              0 (0%)        0
(0%)      0 (0%)           0 (0%)         14d
  kube-system                kube-scheduler-docker-desktop                 100m (0%)     0
(0%)      0 (0%)           0 (0%)         14d
  kube-system                storage-provisioner                           0 (0%)        0
(0%)      0 (0%)           0 (0%)         14d
  kube-system                vpnkit-controller                             0 (0%)        0
(0%)      0 (0%)           0 (0%)         14d
  kubernetes-dashboard       dashboard-metrics-scraper-74db988864-rrjfq    0 (0%)        0
(0%)      0 (0%)           0 (0%)         51m
  kubernetes-dashboard       kubernetes-dashboard-847d8c7cdc-vws57         0 (0%)        0
(0%)      0 (0%)           0 (0%)         51m
Allocated resources:
  (Total limits may be over 100 percent, i.e., overcommitted.)
  Resource           Requests    Limits
  --------           --------    ------
  cpu                750m (6%)   0 (0%)
  memory             140Mi (1%)  340Mi (4%)
  ephemeral-storage  0 (0%)      0 (0%)
  hugepages-1Gi      0 (0%)      0 (0%)
  hugepages-2Mi      0 (0%)      0 (0%)
Events:              <none>
```

Addons

- Dns
- Web UI
- Container Resource Monitoring
- Cluster-level Logging

And many others https://kubernetes.io/docs/concepts/cluster-administration/addons/



NicsMeme

The Kubernetes API

https://kubernetes.io/docs/concepts/overview/kubernetes-api/

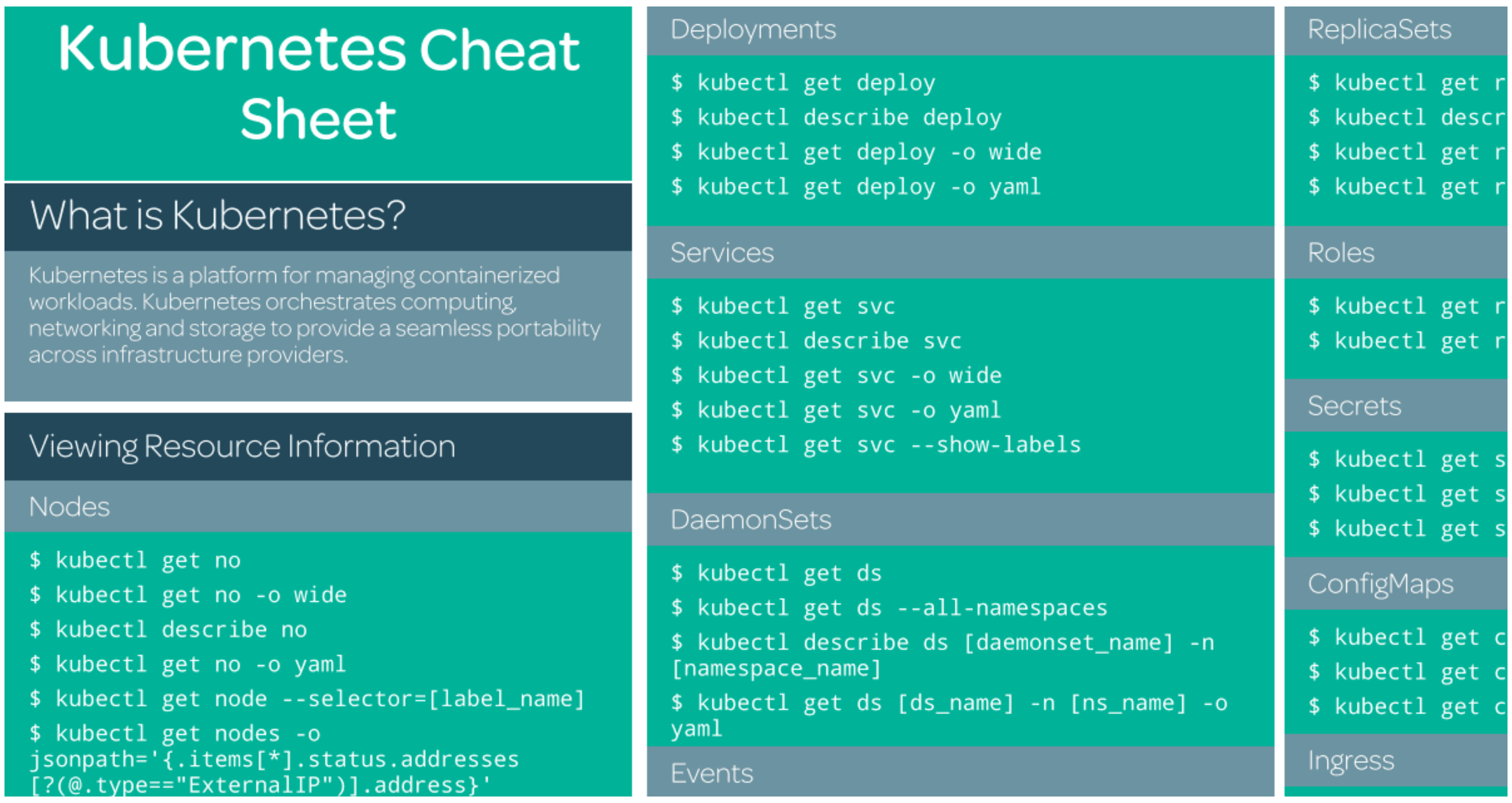
The core of Kubernetes' control plane is the API server.

The API server exposes an HTTP API that lets end users, different parts of your cluster, and external components communicate with one another.

The Kubernetes API lets you query and manipulate the state of API objects in Kubernetes (for example: Pods, Namespaces, ConfigMaps, and Events).

Most operations can be performed through the kubectl command-line interface or other command-line tools, such as kubeadm, which in turn use the API.

However, you can also access the API directly using REST calls.



Kubernetes Cheat Sheet

What is Kubernetes?

Kubernetes is a platform for managing containerized workloads. Kubernetes orchestrates computing, networking and storage to provide a seamless portability across infrastructure providers.

Viewing Resource Information

Nodes

```
$ kubectl get no
$ kubectl get no -o wide
$ kubectl describe no
$ kubectl get no -o yaml
$ kubectl get node --selector=[label_name]
$ kubectl get nodes -o
jsonpath='{.items[*].status.addresses
[?(@.type=="ExternalIP")].address}'
```

Deployments

```
$ kubectl get deploy
$ kubectl describe deploy
$ kubectl get deploy -o wide
$ kubectl get deploy -o yaml
```

Services

```
$ kubectl get svc
$ kubectl describe svc
$ kubectl get svc -o wide
$ kubectl get svc -o yaml
$ kubectl get svc --show-labels
```

DaemonSets

```
$ kubectl get ds
$ kubectl get ds --all-namespaces
$ kubectl describe ds [daemonset_name] -n
[namespace_name]
$ kubectl get ds [ds_name] -n [ns_name] -o
yaml
```

Events

ReplicaSets

```
$ kubectl get r
$ kubectl descr
$ kubectl get r
$ kubectl get r
```

Roles

```
$ kubectl get r
$ kubectl get r
```

Secrets

```
$ kubectl get s
$ kubectl get s
$ kubectl get s
```

ConfigMaps

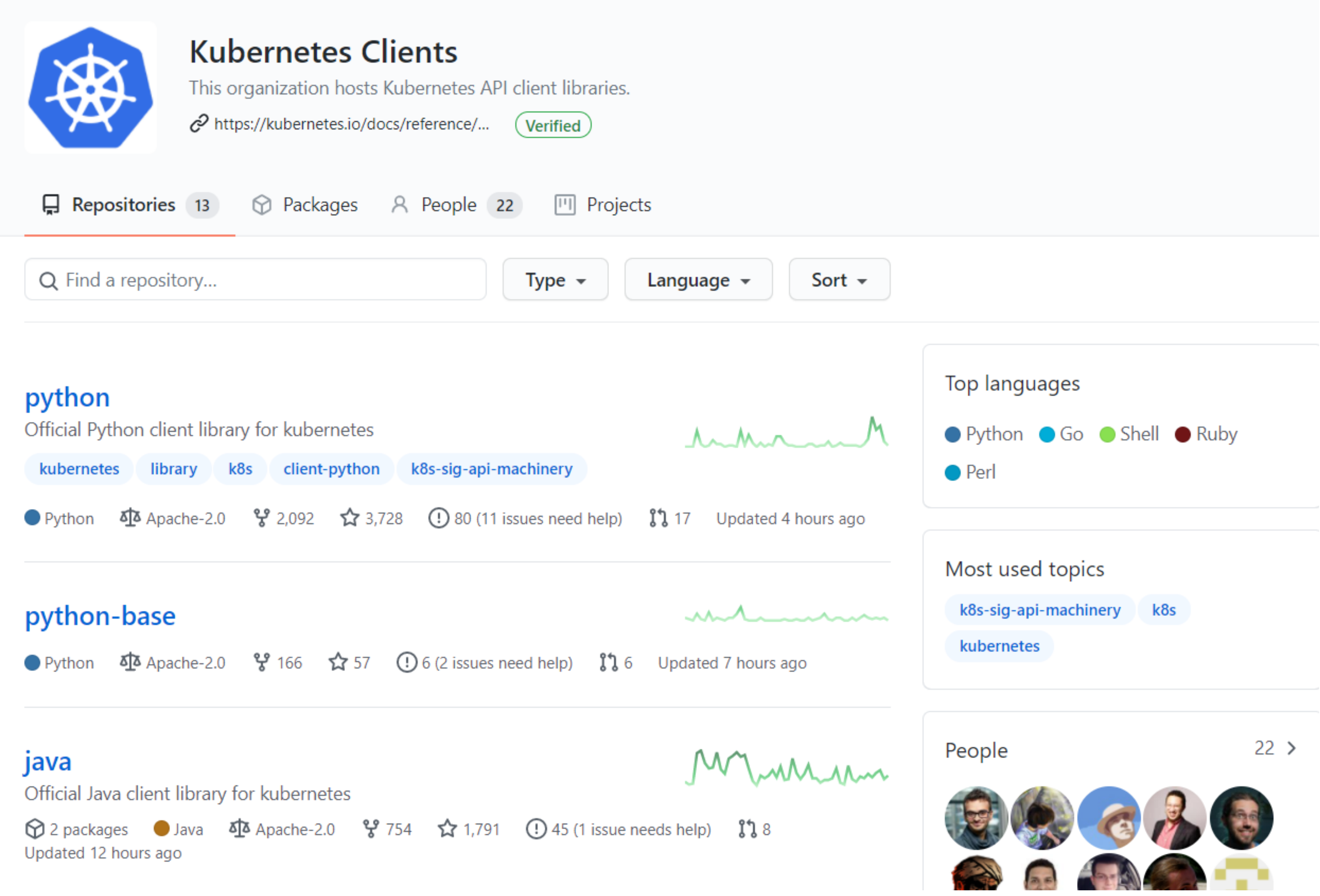
```
$ kubectl get c
$ kubectl get c
$ kubectl get c
```

Ingress

https://acloudguru.com/blog/engineering/kubernetes-cheat-sheet

https://kubernetes.io/docs/reference/kubectl/cheatsheet/

Client Libraries are also available https://kubernetes.io/docs/reference/using-api/client-libraries/



Kubernetes Clients

This organization hosts Kubernetes API client libraries.

https://kubernetes.io/docs/reference/... Verified

python

Official Python client library for kubernetes

kubernetes library k8s client-python k8s-sig-api-machinery

Python Apache-2.0 2,092 3,728 80 (11 issues need help) 17 Updated 4 hours ago

python-base

Python Apache-2.0 166 57 6 (2 issues need help) 6 Updated 7 hours ago

java

Official Java client library for kubernetes

2 packages Java Apache-2.0 754 1,791 45 (1 issue needs help) 8

Updated 12 hours ago

Top languages

Python Go Shell Ruby

Perl

Most used topics

k8s-sig-api-machinery k8s

kubernetes

People 22

OpenApi

kubectl proxy –port=8080

http://localhost:8080/openapi/v2

http://localhost:8080/api

Objects

What they are ?

Objects are persistent entities that represent the status of the cluster.

Kubernetes object is a “record of intent”, that’s is to say creating/updating the object is a ***desired state***, the final goal you want to achieve.

There two parts for the object

- spec: typically defined in yaml, containing the desired status
- status: is the actual status

https://kubernetes.io/docs/concepts/overview/working-with-objects/kubernetes-objects/

Example

```
cd code/ngnix-deployment
kubectl apply -f app-deployment.yaml
kubectl get pods
kubectl port-forward deployment/nginx-deployment 8080:80
http://localhost:8080/
kubectl describe pod nginx-deployment-66b6c48dd5-6ln4b
kubectl exec -it  nginx-deployment-66b6c48dd5-6ln4b -- /bin/bash
kubectl get -f app-deployment.yaml
kubectl scale deployment/nginx-deployment --replicas=4
kubectl delete deployment nginx-deployment
```

How to manage

Management technique	Operates on	Recommended environment	Supported writers	Learning curve
Imperative commands	Live objects	Development projects	1+	Lowest
Imperative object configuration	Individual files	Production projects	1	Moderate
Declarative object configuration	Directories of files	Production projects	1+	Highest

https://kubernetes.io/docs/concepts/overview/working-with-objects/object-management/

Name

Each object in your cluster has a Name that is unique for that type of resource. Every Kubernetes object also has a UID that is unique across your whole cluster.

For example, you can only have one Pod named myapp-1234 within the same namespace, but you can have one Pod and one Deployment that are each named myapp-1234.

For non-unique user-provided attributes, Kubernetes provides labels and annotations.

https://kubernetes.io/docs/concepts/overview/working-with-objects/names/

Namespaces

When to Use Multiple Namespaces Namespaces are intended for use in environments with many users spread across multiple teams, or projects. For clusters with a few to tens of users, you should not need to create or think about namespaces at all. Start using namespaces when you need the features they provide.

Namespaces provide a scope for names. Names of resources need to be unique within a namespace, but not across namespaces. Namespaces cannot be nested inside one another and each Kubernetes resource can only be in one namespace.

Namespaces are a way to divide cluster resources between multiple users (via resource quota).

It is not necessary to use multiple namespaces to separate slightly different resources, such as different versions of the same software: use labels to distinguish resources within the same namespace.

https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/

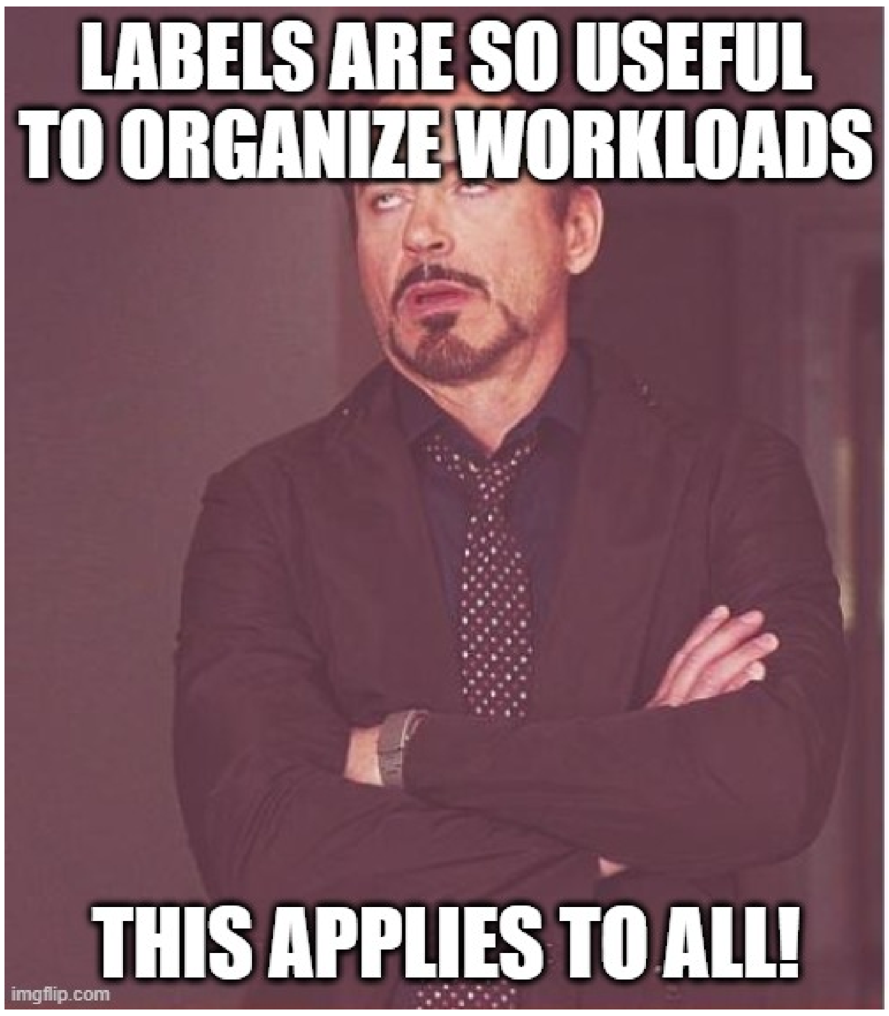
```
%%bash
kubectl get namespace
```

```
NAME                   STATUS   AGE
default                Active   14d
kube-node-lease        Active   14d
kube-public            Active   14d
kube-system            Active   14d
kubernetes-dashboard   Active   14d
```

Labels and selector

Labels are key/value pairs that are attached to objects, such as pods. Labels are intended to be used to specify identifying attributes of objects that are meaningful and relevant to users, but do not directly imply semantics to the core system. Labels can be used to organize and to select subsets of objects. Labels can be attached to objects at creation time and subsequently added and modified at any time. Each object can have a set of key/value labels defined. Each Key must be unique for a given object.

https://kubernetes.io/docs/concepts/overview/working-with-objects/labels/



NicsMeme

Workloads

Pods

https://www.wired.it/play/cinema/2016/03/09/traduzione-sbagliata-film-libri-hunger-games/

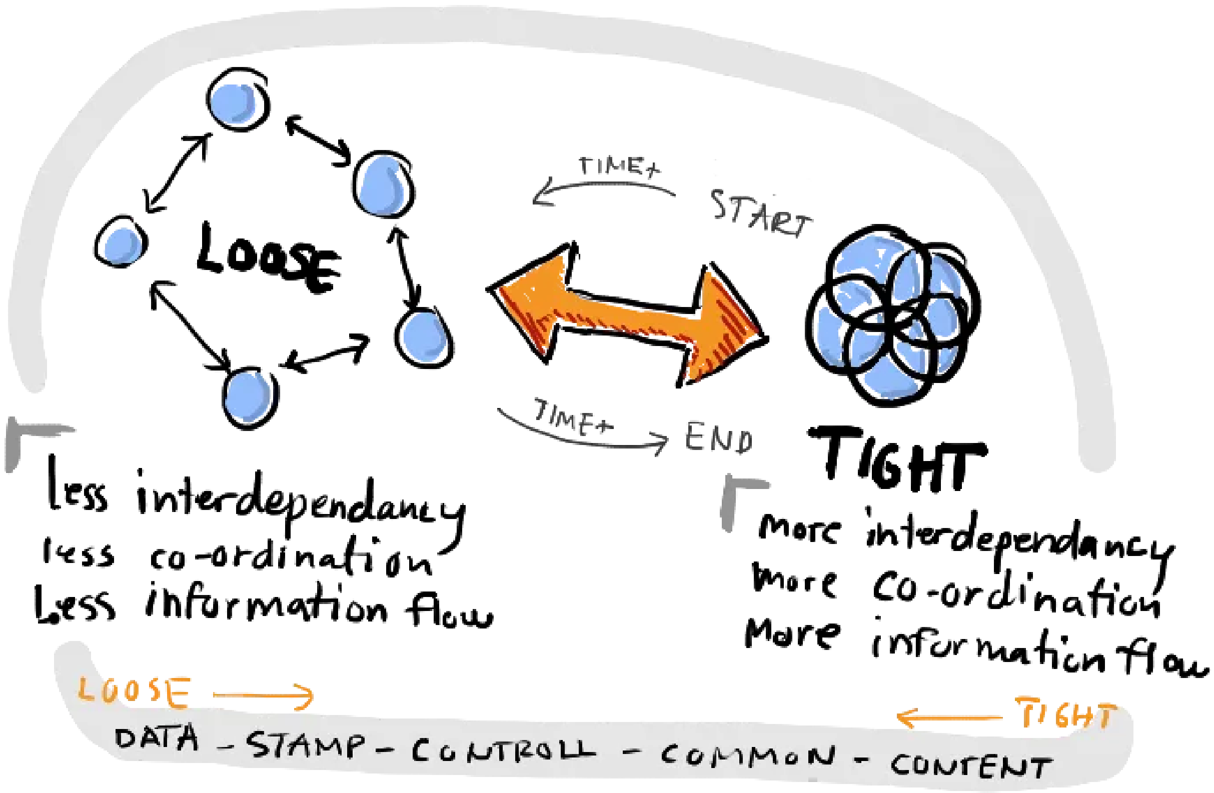
Pods are the smallest deployable **units of computing** that you can create and manage in Kubernetes.

A Pod (as in a pod of whales or pea pod) is a group of one or more containers, with shared storage and network resources, and a specification for how to run the containers.

https://commons.wikimedia.org/wiki/File:Green_pea_pod_8872.jpg

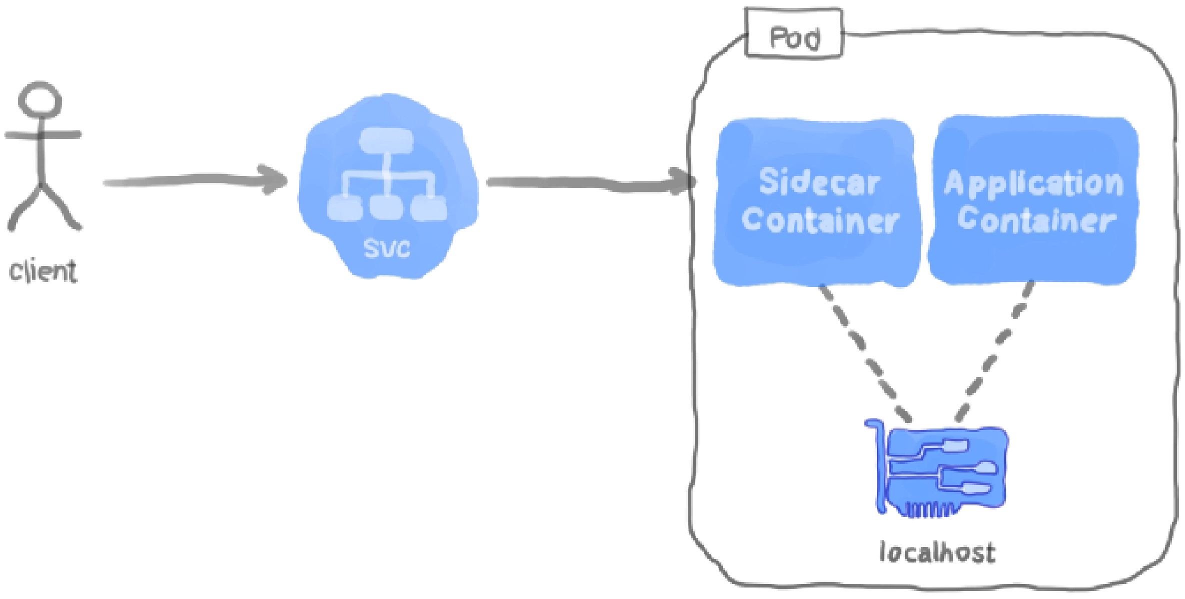
A Pod’s contents are always co-located and co-scheduled, and run in a shared context.

A Pod models an application-specific “logical host”: it contains one or more application containers which are relatively tightly coupled.



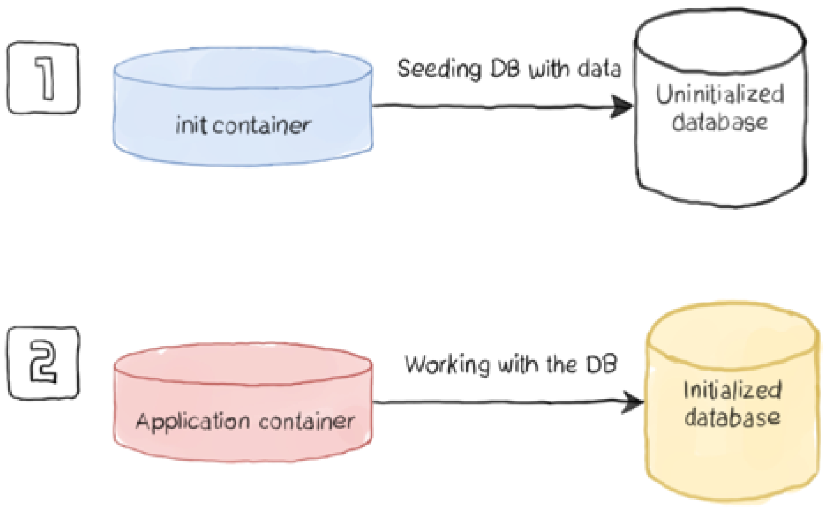
https://dailyfintech.com/2017/02/20/applying-loose-coupling-software-principles-to-enterprise-digital-transformation/

In non-cloud contexts, applications executed on the same physical or virtual machine are analogous to cloud applications executed on the same logical host.



https://levelup.gitconnected.com/implemeting-a-reverse-proxy-server-in-kubernetes-using-the-sidecar-eebba956801a

As well as application containers, a Pod can contain init containers that run during Pod startup. You can also inject ephemeral containers for debugging if your cluster offers this



https://www.magalix.com/blog/kubernetes-patterns-the-init-container-pattern

The shared context of a Pod is a set of Linux namespaces, cgroups, and potentially other facets of isolation - the same things that isolate a Docker container. Within a Pod's context, the individual applications may have further sub-isolations applied.

In terms of Docker concepts, a Pod is similar to a group of Docker containers with shared namespaces and shared filesystem volumes

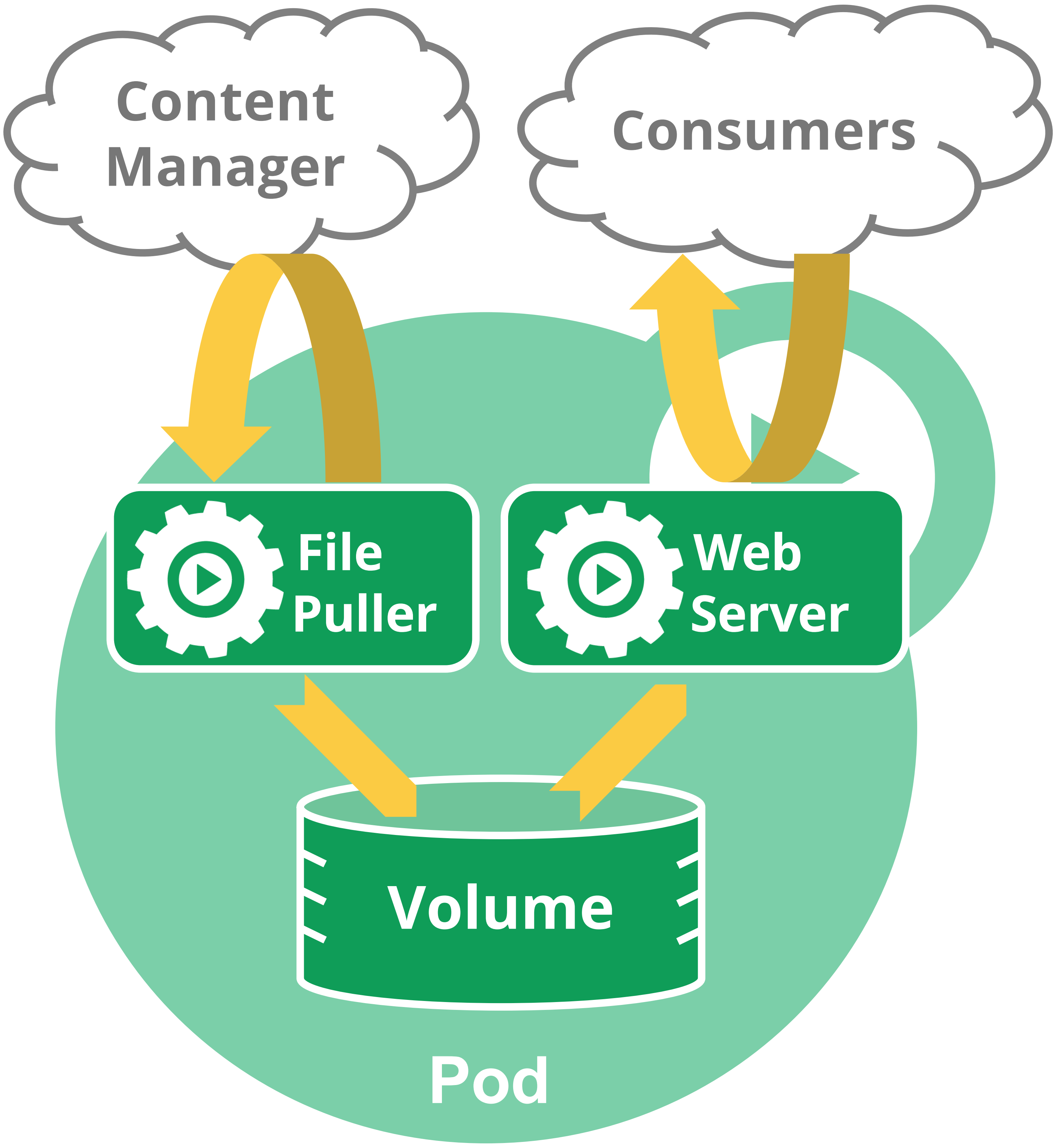
Usually you don't need to create Pods directly, even singleton Pods. Instead, create them using workload resources such as Deployment or Job. If your Pods need to track state, consider the StatefulSet resource.

Pods in a Kubernetes cluster are used in two main ways:

- Pods that run a single container. The "one-container-per-Pod" model is the most common Kubernetes use case; in this case, you can think of a Pod as a wrapper around a single container; Kubernetes manages Pods rather than managing the containers directly.
- Pods that run multiple containers that need to work together. A Pod can encapsulate an application composed of multiple co-located containers that are tightly coupled and need to share resources. These co-located containers form a single cohesive unit of service—for example, one container serving data stored in a shared volume to the public, while a separate sidecar container refreshes or updates those files. The Pod wraps these containers, storage resources, and an ephemeral network identity together as a single unit.

> Note: Grouping multiple co-located and co-managed containers in a single Pod is a relatively advanced use case. You should use this pattern only in specific instances in which your containers are tightly coupled.

Use case of multiple container in same pod



```yaml
apiVersion: v1
kind: Pod
metadata:
  name: two-containers
spec:

  restartPolicy: Never

  volumes:
  - name: shared-data
    emptyDir: {}

  containers:

  - name: nginx-container
    image: nginx
    volumeMounts:
    - name: shared-data
      mountPath: /usr/share/nginx/html

  - name: debian-container
    image: debian
    volumeMounts:
    - name: shared-data
      mountPath: /pod-data
    command: ["/bin/sh"]
    args: ["-c", "echo Hello from the debian container > /pod-data/index.html"]
```

https://kubernetes.io/docs/tasks/access-application-cluster/communicate-containers-same-pod-shared-volume/

Deployment

A Deployment provides declarative updates for Pods and ReplicaSets.

You describe a desired state in a Deployment, and the Deployment Controller changes the actual state to the desired state at a controlled rate. You can define Deployments to create new ReplicaSets, or to remove existing Deployments and adopt all their resources with new Deployments.

NicsMeme

Deploy K8S

Katakoda

Learn new technologies using real environments right in your browser

https://www.katacoda.com/

Minikube

minikube is local Kubernetes, focusing on making it easy to learn and develop for Kubernetes.

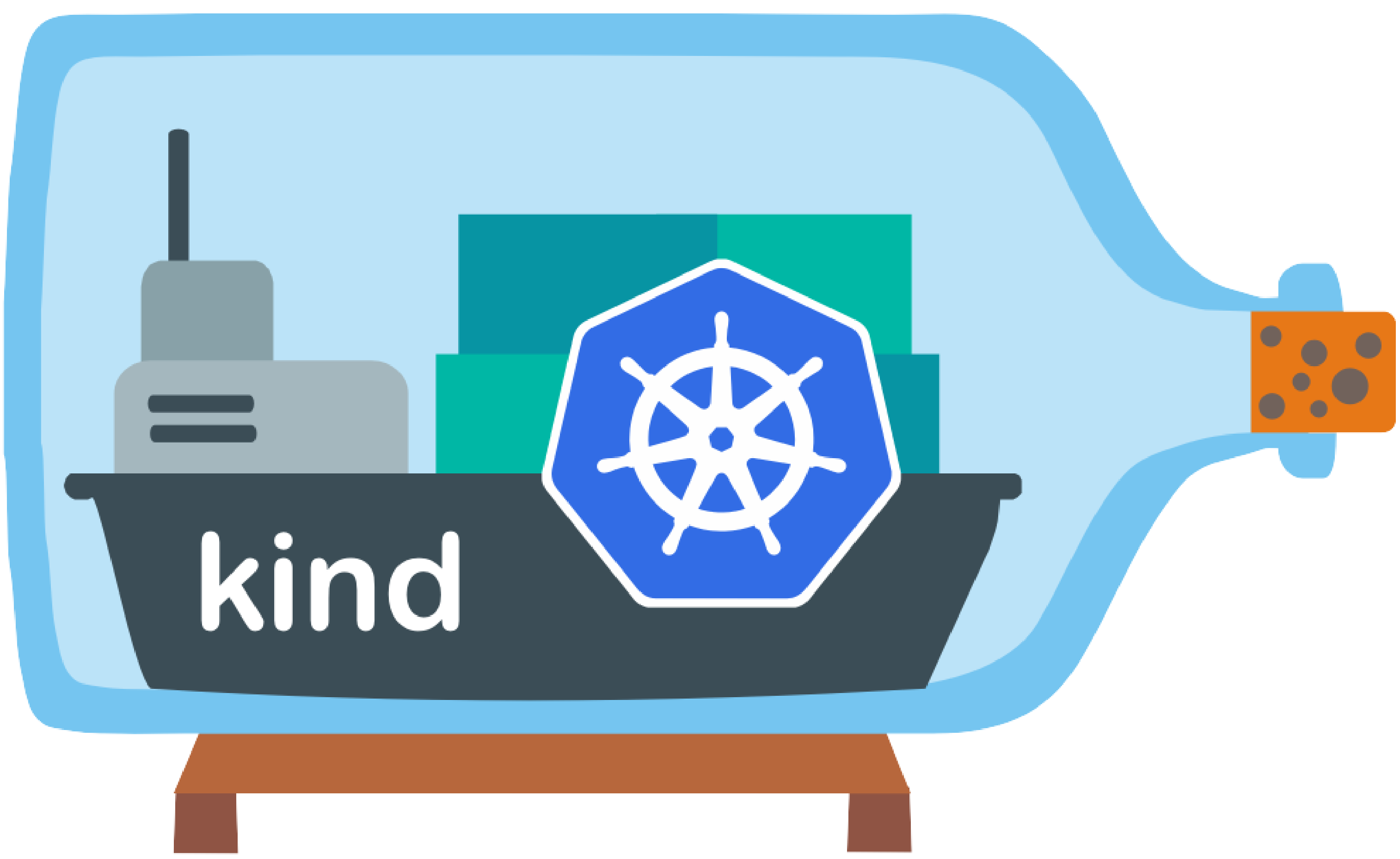
All you need is Docker (or similarly compatible) container or a Virtual Machine environment, and Kubernetes is a single command away: minikube start

https://minikube.sigs.k8s.io/docs/start/ https://kubernetes.io/docs/tutorials/hello-minikube/

kind

ind is a tool for running local Kubernetes clusters using Docker container “nodes”. kind was primarily designed for testing Kubernetes itself, but may be used for local development or CI.

https://kind.sigs.k8s.io/

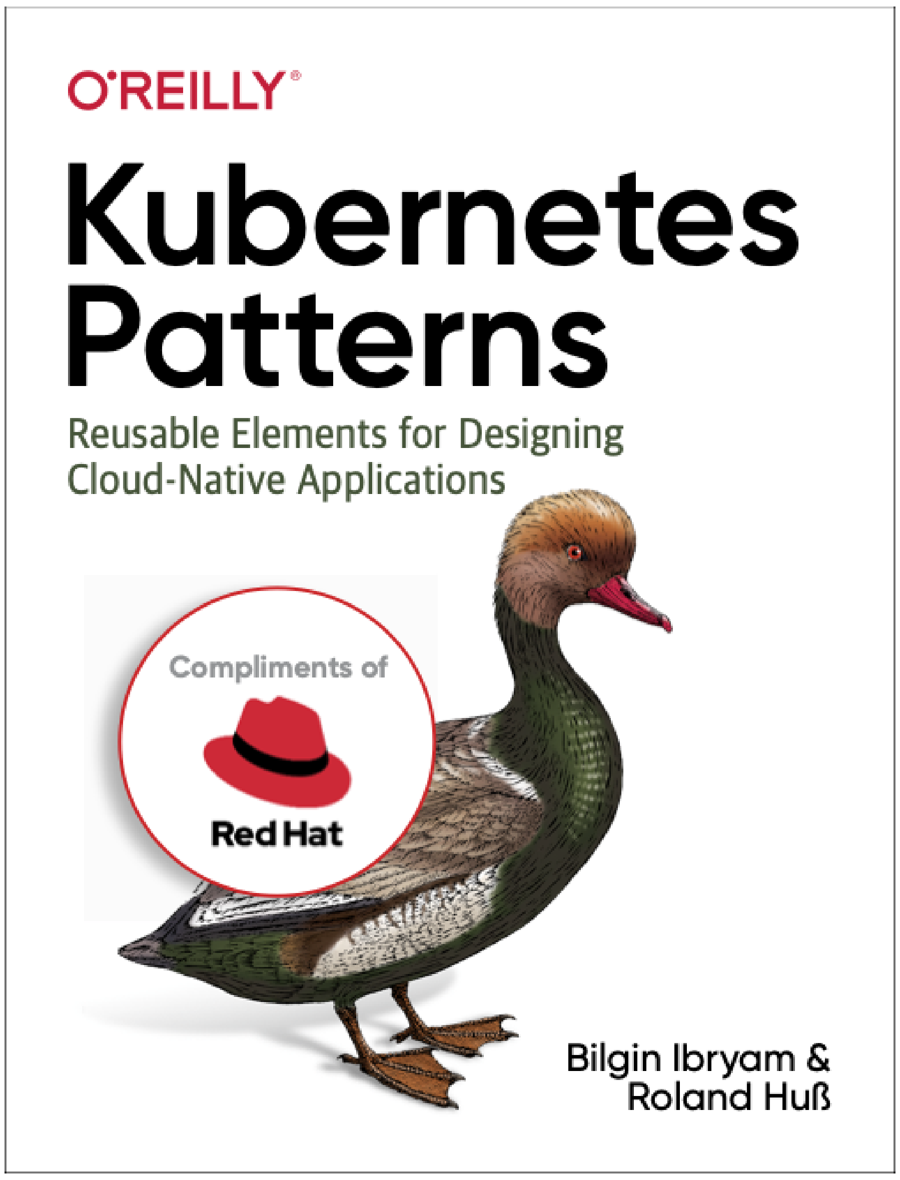


Kubeadm is a tool built to provide kubeadm init and kubeadm join as best-practice “fast paths” for creating Kubernetes clusters.

kubeadm performs the actions necessary to get a minimum viable cluster up and running. By design, it cares only about bootstrapping, not about provisioning machines. Likewise, installing various nice-to-have addons, like the Kubernetes Dashboard, monitoring solutions, and cloud-specific addons, is not in scope.

Patterns

https://developers.redhat.com/books/kubernetes-patterns


O'REILLY
Kubernetes Patterns
Reusable Elements for Designing Cloud-Native Applications
Compliments of
Red Hat
Bilgin Ibryam & Roland Huß

Helm

> Helm is the best way to find, share, and use software built for Kubernetes.

https://helm.sh/

Helm on WSL 2

https://codelabs.solace.dev/codelabs/helm-environment-setup/#0

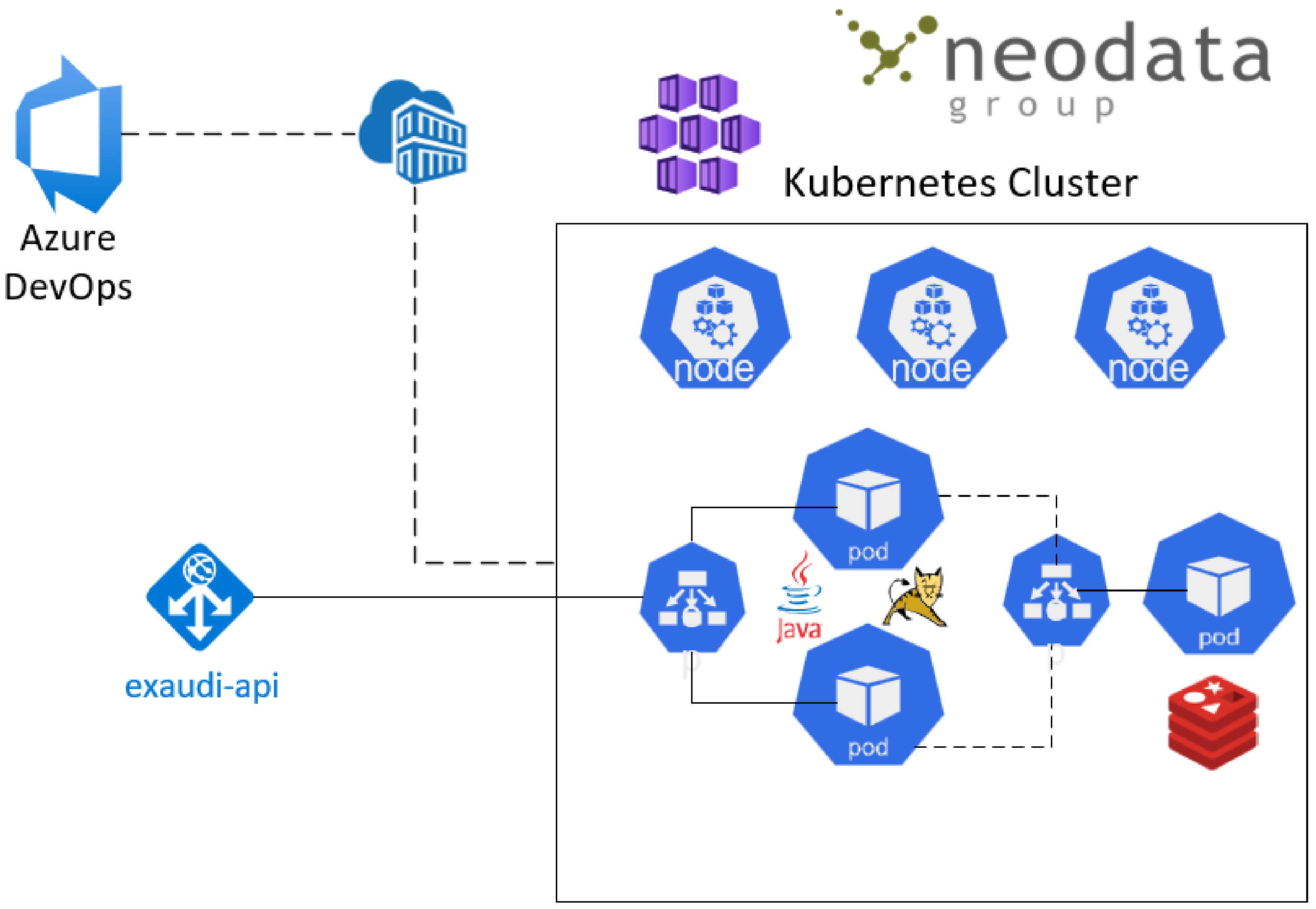
In Neodata

API

Rest API to manage UI Request

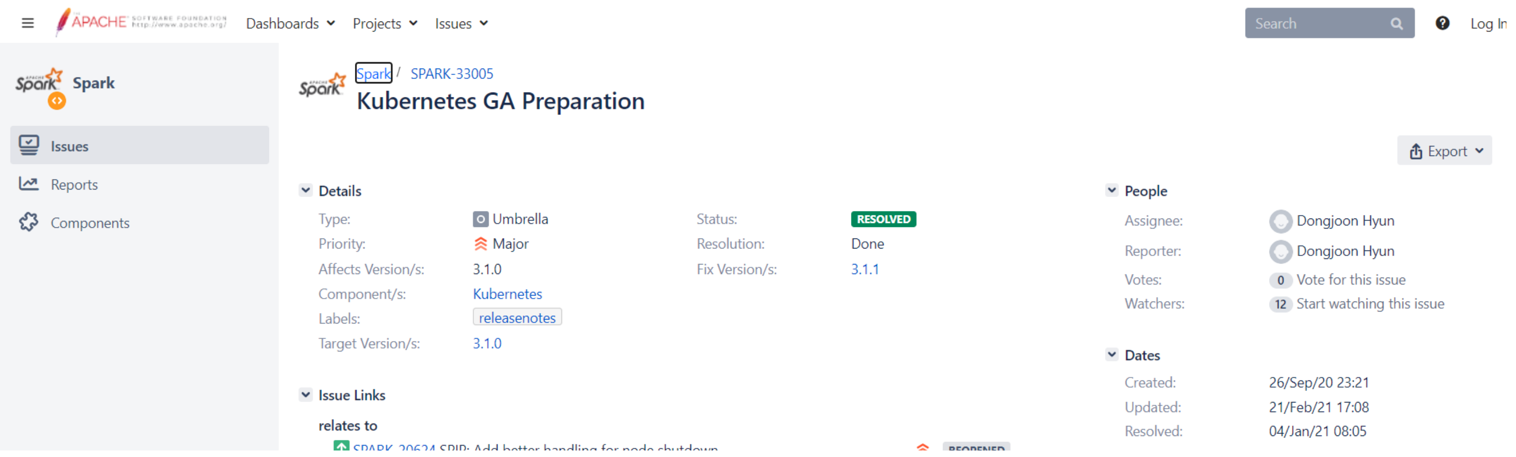
Framework: Tomcat+Java+Jersey+Spring / Redis

Live since 2019



Spark

Test in progress on Spark + K8S deployment



Spark / SPARK-33005

Kubernetes GA Preparation

Details

Type: Umbrella

Priority: Major

Affects Version/s: 3.1.0

Component/s: Kubernetes

Labels: releasenotes

Target Version/s: 3.1.0

Status: RESOLVED

Resolution: Done

Fix Version/s: 3.1.1

People

Assignee: Dongjoon Hyun

Reporter: Dongjoon Hyun

Votes: 0 Vote for this issue

Watchers: 12 Start watching this issue

Dates

Created: 26/Sep/20 23:21

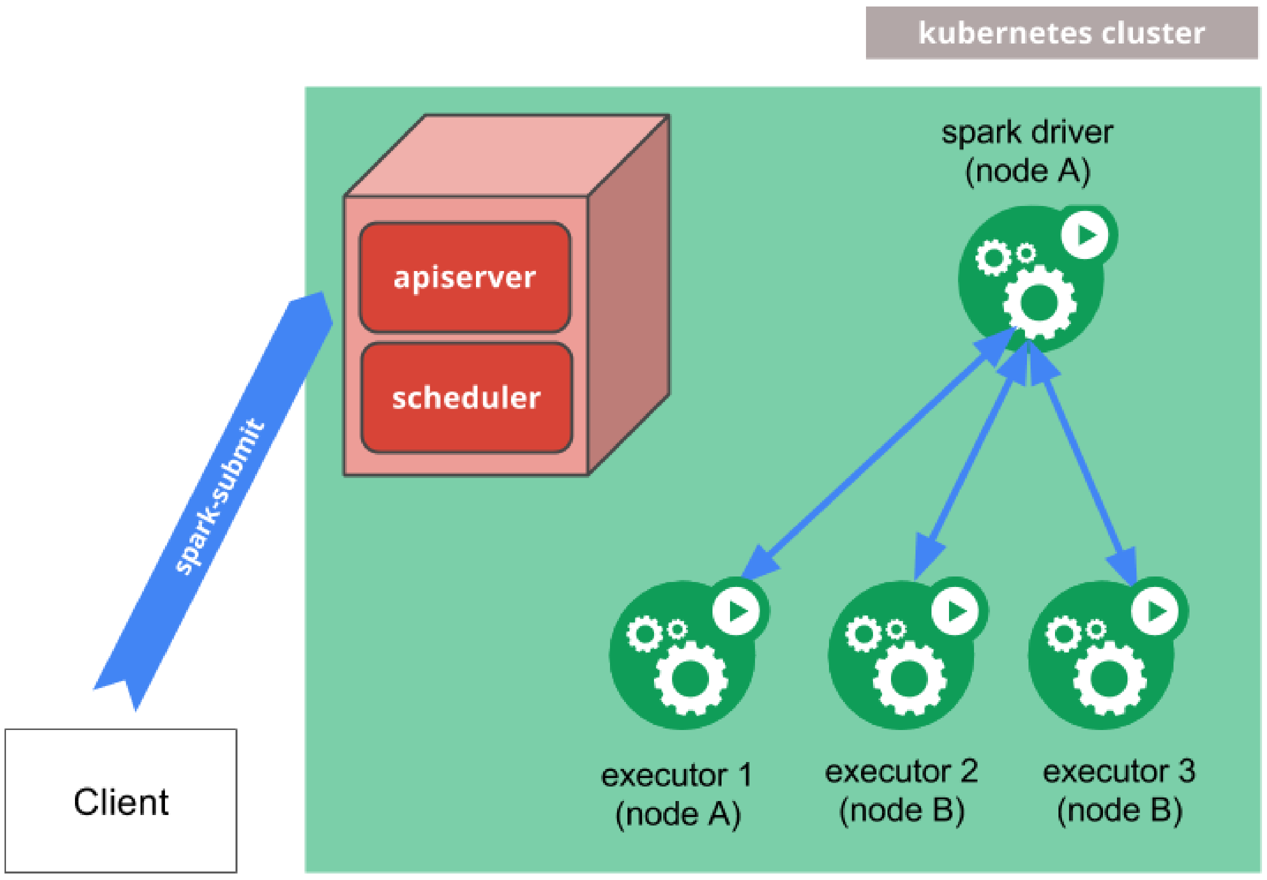
Updated: 21/Feb/21 17:08

Resolved: 04/Jan/21 08:05

Issue Links

relates to

SPARK-20624 SPIP: Add better handling for node shutdown REOPENED



Slides here

https://github.com/salvo-nicotra/notebooks/blob/master/K8s.ipynb

Biblio

General

- https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/
- https://thenewstack.io/kubernetes-portability-must-have-or-shiny-object-syndrome/
- https://blogs.gartner.com/marco-meinardi/2020/09/04/adopting-kubernetes-application-portability-not-good-idea/
- https://dev.to/stanley/my-first-steps-with-kubernetes-a3f
- https://stackoverflow.blog/2020/05/29/why-kubernetes-getting-so-popular/
- https://github.com/kubernetes/kubernetes
- https://cloud.google.com/blog/products/containers-kubernetes/from-google-to-the-world-the-kubernetes-origin-story
- https://medium.com/payscale-tech/imperative-vs-declarative-a-kubernetes-tutorial-4be66c5d8914
- https://www.aquasec.com/cloud-native-academy/kubernetes-101/kubernetes-complete-guide/
- https://betterprogramming.pub/why-kubernetes-bbb7d66fccf5
- https://www.techwell.com/techwell-insights/2019/07/why-use-kubernetes-your-container-management
- https://faun.pub/benefits-of-kubernetes-for-microservices-architecture-a04704f0d3a0
- https://www.barrons.com/articles/kubernetes-is-the-next-big-thing-in-cloud-computing-and-its-free-51575576969
- https://towardsdatascience.com/5-reason-for-using-kubernetes-b7ade82eda90
- https://blog.true-kubernetes.com/is-kubernetes-worth-learning-part-1-the-big-picture/
- https://www.youtube.com/watch?v=eXZ7lPBSM-k&t=2737s

WSL2

- https://andrewlock.net/running-kubernetes-and-the-dashboard-with-docker-desktop/
- https://mohitgoyal.co/2021/03/19/setup-local-kubernetes-cluster-with-docker-wsl2-and-kind/
- https://kubernetes.io/blog/2020/05/21/wsl-docker-kubernetes-on-the-windows-desktop/
- https://codefresh.io/kubernetes-tutorial/local-kubernetes-windows-minikube-vs-docker-desktop/
- https://www.codenotary.com/blog/combine-docker-kubernetes-and-windows-wsl/