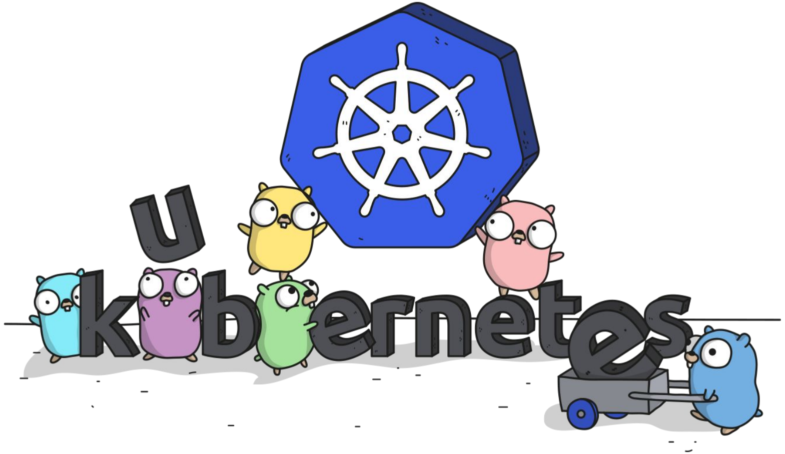


Intro to the cloud-native world with Kubernetes

Lucas Käldström - luxas labs

8th of June 2018 - Lyon

Image credit: @ashleymcnamara

$ whoami

CLOUD NATIVE COMPUTING FOUNDATION AMBASSADOR

Lucas Käldström, Upper Secondary School Student, 18 years old

CNCF Ambassador, **Certified Kubernetes Administrator** and **Kubernetes SIG Lead**

CERTIFIED kubernetes ADMINISTRATOR

Speaker at **KubeCon** in **Berlin**, **Austin and Copenhagen**

Kubernetes approver and subproject owner, active in the community for ~3 years

Driving **luxas labs** which currently performs contracting for Weaveworks

A guy that has never attended a computing class

CLOUD NATIVE COMPUTING FOUNDATION

= Open Source Cloud Computing For Applications

Cloud Native Landscape

v20180525

See the interactive landscape at l.cncf.io

Greyed logos are not open source

App Definition and Development

Database and Data Warehouse

Vitess — CNCF Incubating

Streaming

NATS — CNCF Incubating

cloudevents — CNCF Sandbox

Source Code Management

Application Definition and Image Build

CNCF Sandbox

Continuous Integration / Continuous Delivery (CI/CD)

Orchestration & Management

Scheduling & Orchestration

kubernetes — CNCF Graduated

Coordination & Service Discovery

CoreDNS — CNCF Incubating

Service Management

GRPC — CNCF Incubating

envoy — CNCF Incubating

LINKERD — CNCF Incubating

Open Policy Agent — CNCF Sandbox

Runtime

Cloud-Native Storage

ROOK — CNCF Sandbox

Container Runtime

containerd — CNCF Incubating

rkt — CNCF Incubating

Cloud-Native Network

CNI — CNCF Incubating

Provisioning

Host Management / Tooling

Infrastructure Automation

Container Registries

Secure Images

Notary — CNCF Incubating

TUF — CNCF Incubating

Key Management

spiffe — CNCF Sandbox

SPIRE — CNCF Sandbox

Cloud

Public

Private

l.cncf.io

This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.

CLOUD NATIVE Landscape

CLOUD NATIVE COMPUTING FOUNDATION

Redpoint

Amplify PARTNERS

Platforms

Certified Kubernetes - Distribution

Certified Kubernetes - Hosted

Certified Kubernetes - Installer

Non-Certified Kubernetes

PaaS/Container Service

Observability & Analysis

Monitoring

Prometheus — CNCF Incubating

Logging

fluentd — CNCF Incubating

Tracing

JAEGER — CNCF Incubating

OPENTRACING — CNCF Incubating

Serverless

Special

Kubernetes Certified Service Provider

Kubernetes Training Partner

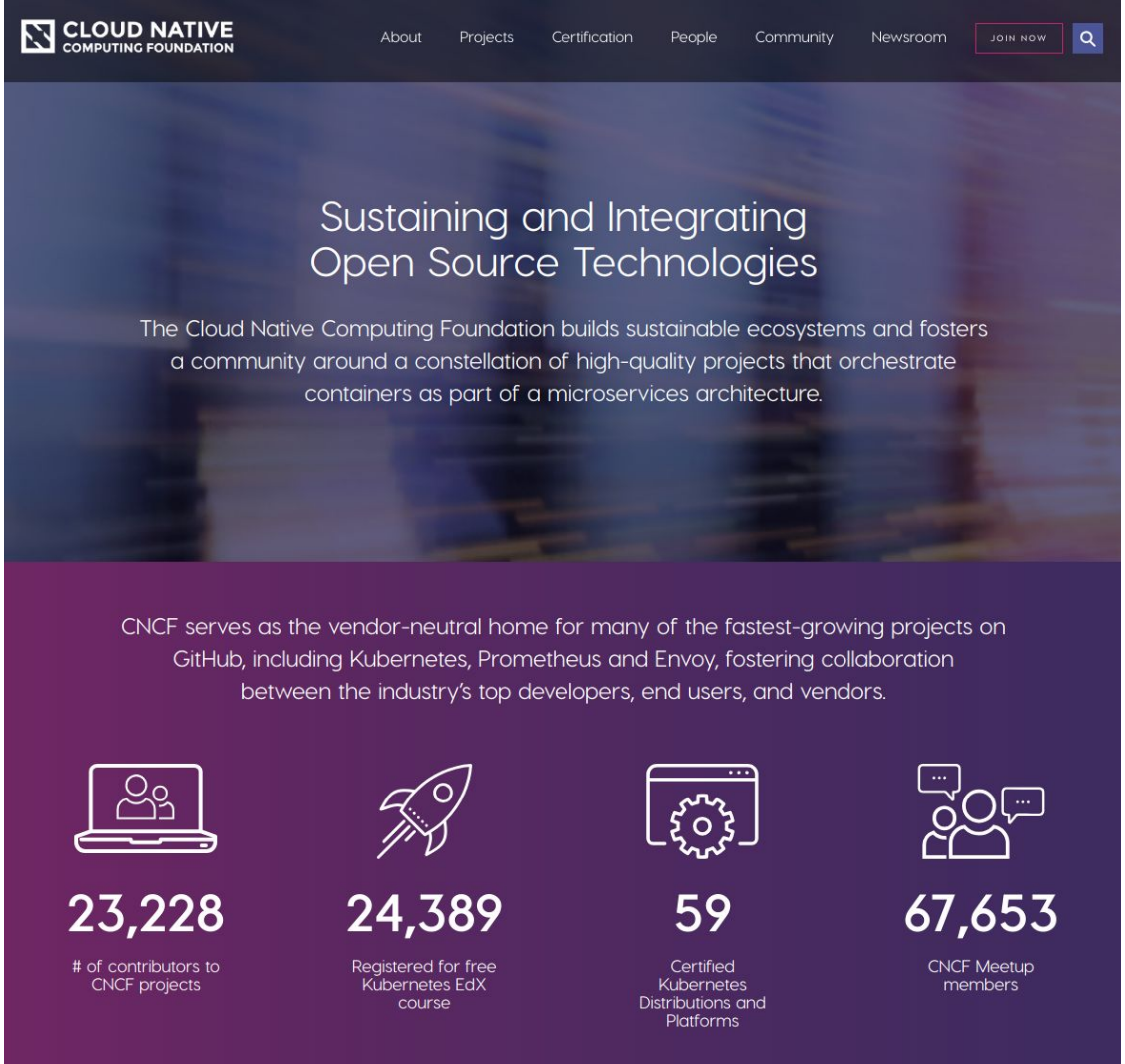
What is CNCF?

A non-profit foundation for getting Cloud Native:
a) open source projects
b) companies
c) enthusiasts
to **come together in a neutral place**.

CNCF was founded in December 2015 and is a part of **The Linux Foundation**.

CNCF curates and **promotes a toolkit of trusted projects** for modern applications.

Helps hosted **projects** to **succeed** in various ways, one of them is by **organizing events** where the community can meet in person.



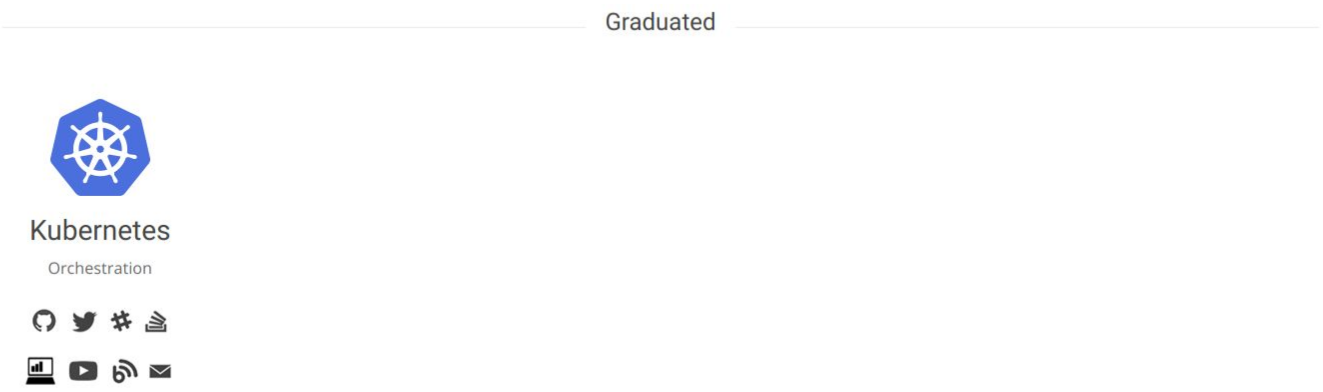
Sustaining and Integrating Open Source Technologies

The Cloud Native Computing Foundation builds sustainable ecosystems and fosters a community around a constellation of high-quality projects that orchestrate containers as part of a microservices architecture.

CNCF serves as the vendor-neutral home for many of the fastest-growing projects on GitHub, including Kubernetes, Prometheus and Envoy, fostering collaboration between the industry's top developers, end users, and vendors.

23,228	24,389	59	67,653
# of contributors to CNCF projects	Registered for free Kubernetes EdX course	Certified Kubernetes Distributions and Platforms	CNCF Meetup members

What projects does CNCF host?

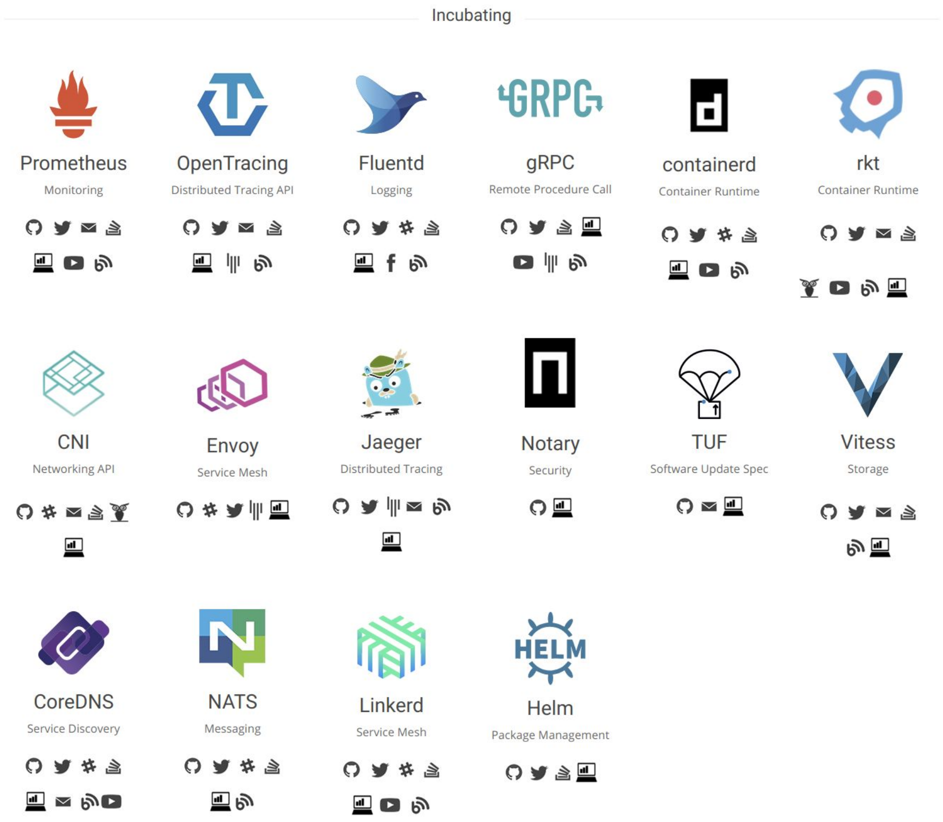


Graduated

Kubernetes

Orchestration

What projects does CNCF host?



Incubating

Prometheus
Monitoring

OpenTracing
Distributed Tracing API

Fluentd
Logging

gRPC
Remote Procedure Call

containerd
Container Runtime

rkt
Container Runtime

CNI
Networking API

Envoy
Service Mesh

Jaeger
Distributed Tracing

Notary
Security

TUF
Software Update Spec

Vitess
Storage

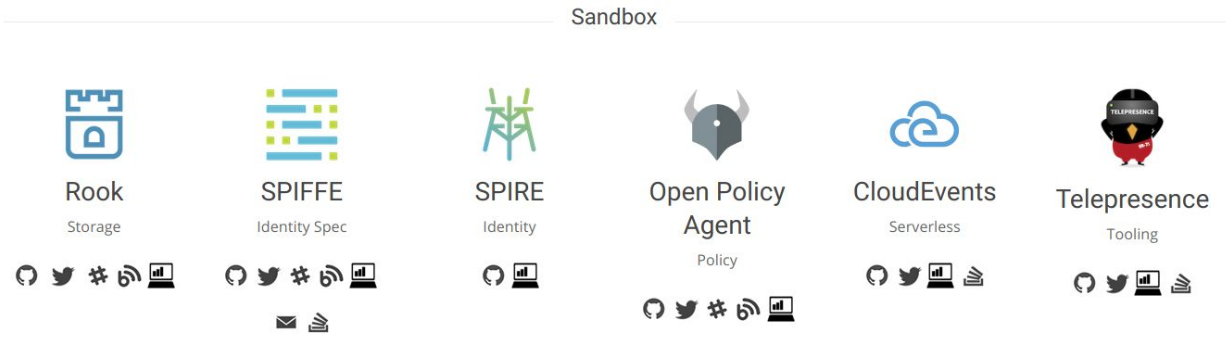
CoreDNS
Service Discovery

NATS
Messaging

Linkerd
Service Mesh

Helm
Package Management

What projects does CNCF host?



Sandbox

Rook
Storage

SPIFFE
Identity Spec

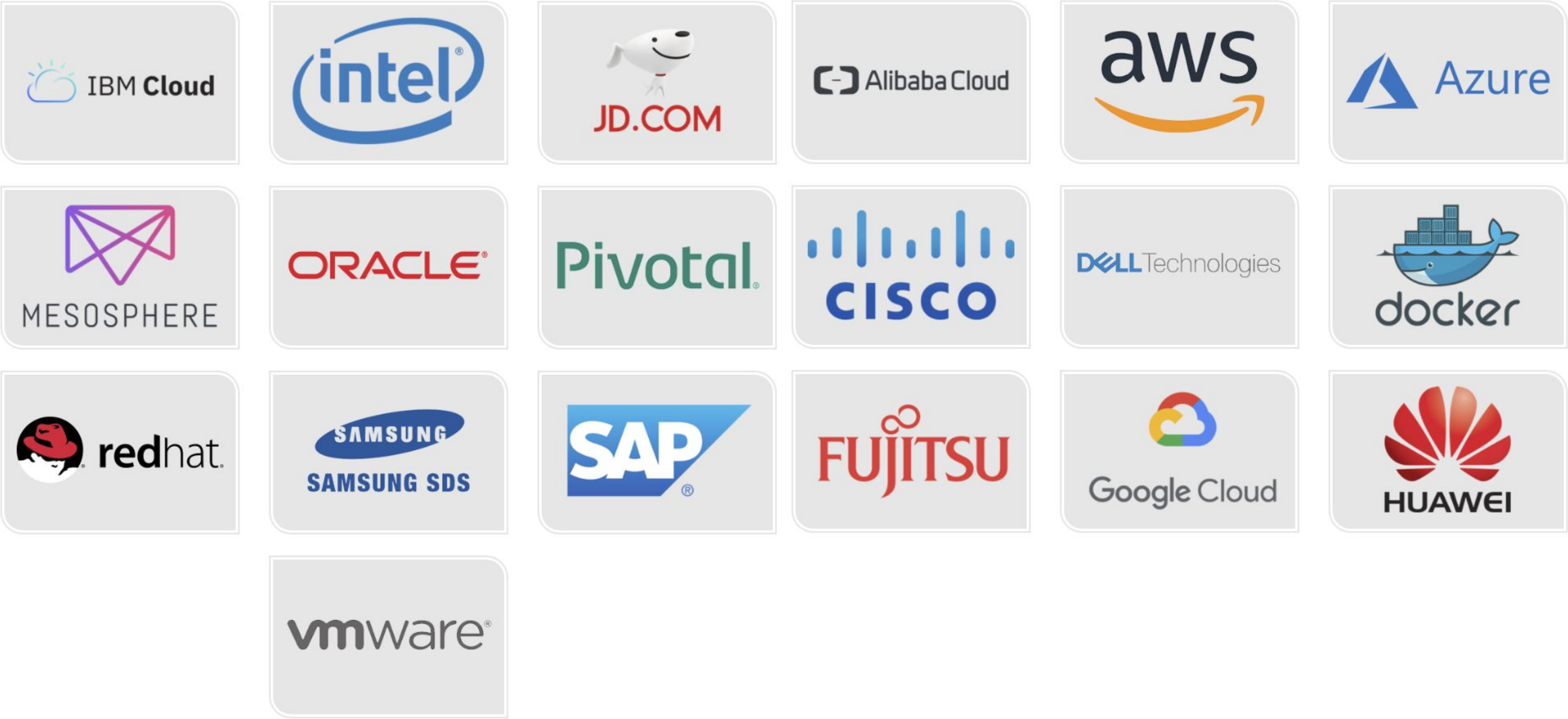
SPIRE
Identity

Open Policy Agent
Policy

CloudEvents
Serverless

Telepresence
Tooling

CNCF Platinum members

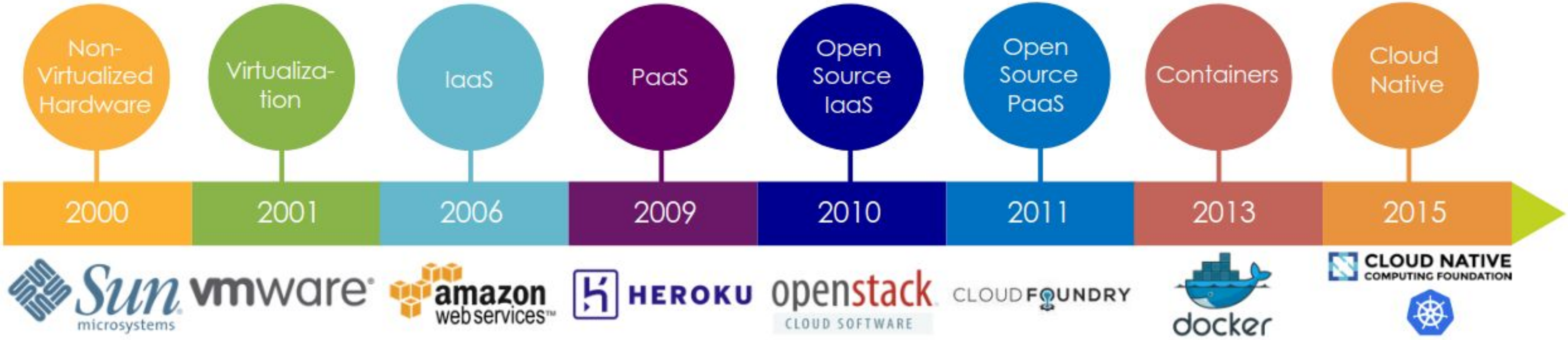


From Virtualization to Cloud Native

CLOUD NATIVE
COMPUTING FOUNDATION

kubernetes

- Cloud native computing uses an open source software stack to:
  - segment applications into *microservices*,
  - packaging each part into its own *container*
  - and dynamically *orchestrating* those containers to optimize resource utilization



Source

What is the “Cloud Native” mindset?

Cloud Native computing uses an open source software stack that is:

1. Containerized
2. Dynamically orchestrated
3. Microservices oriented

There are three main keywords:

1. Speed
2. Freedom
3. Trust

Alexis Richardson, CEO of Weaveworks, gave a good keynote on this topic at KubeCon Berlin 2017

EVERYONE’S EXCITED ABOUT

KUBERNETES

Most importantly: What does “Kubernetes” mean?

Kubernetes
= Greek for “pilot” or
“helmsman of a ship”

What is Kubernetes?

= A Production-Grade Container Orchestration System

Google-grown, based on **Borg** and **Omega**, systems that run inside of Google right now and are proven to work at Google for over 10 years.
Google spawns *2 billion containers per week* with these systems.

Created by three Google employees initially during the **summer of 2014**;
grew exponentially and became the first project to get donated to the CNCF.

Hit the first production-grade version **v1.0.1 in July 2015**.

Has continually released a new minor version every three months since
v1.2.0 in March 2016. Lately **v1.10.0** was released in **March 2018**.

So what does Kubernetes *actually* do?

One thing: ***Abstract away the underlying hardware***. Abstract away the concept **Node.**

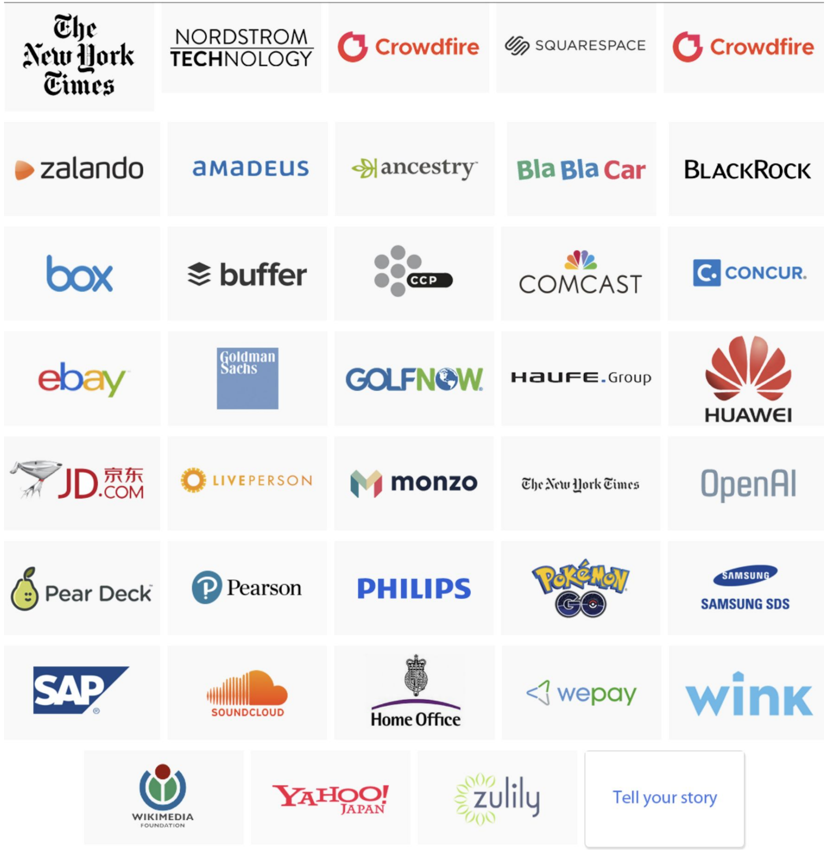
Principle: Manage your applications like **Cattle** (generic, bulk operations) instead of like **Pets** (every operation is customized with care and love for the individual)

Kubernetes is the ***Linux for distributed systems***.

In the same manner Linux (an OS) abstracts away the hardware differences (with different CPU types, etc.), Kubernetes abstracts away the fact that you have 5 000 nodes in the node pool and provides consistent UX and operation methods for apps

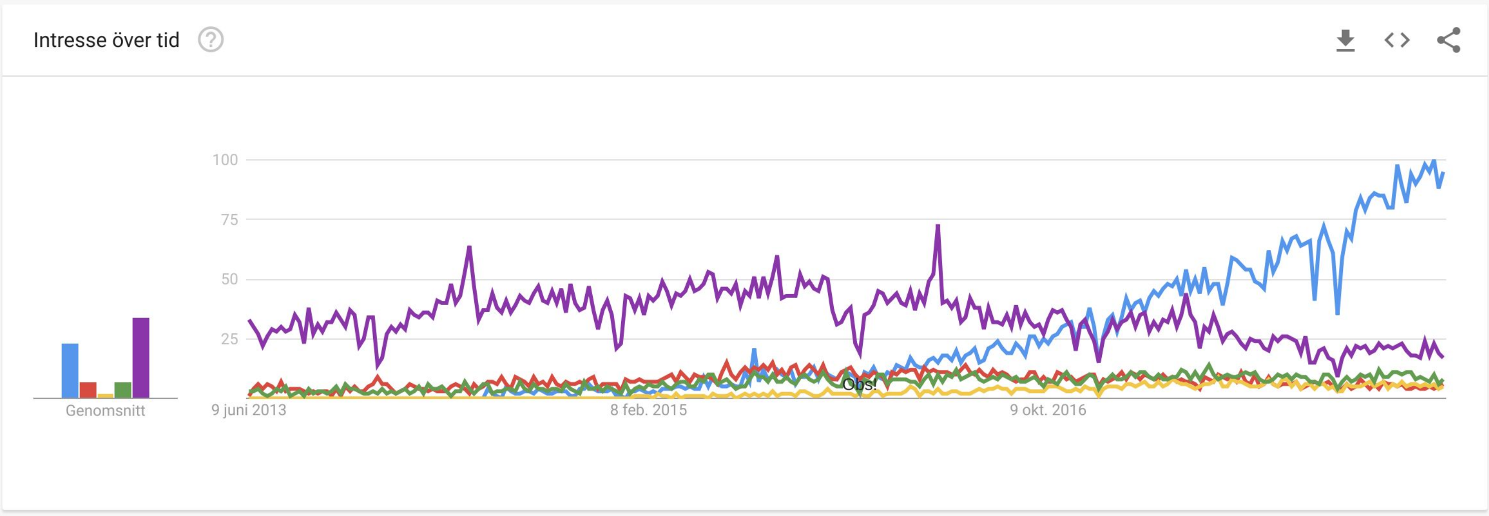
You (the admin) declares the **desired state**, Kubernetes' main task is to ***make the desired state the actual state*****.**

A couple of Kubernetes case stories



Kubernetes' popularity measured briefly

Google Search interest over time in the 8.6.2013-8.6.2018 timespan



KUBERNETES

MESOS

DOCKER SWARM

CLOUD FOUNDRY

OPENSTACK

Kubernetes is one of the fastest moving open source projects in history (5th of June, 2017)
Measuring the popularity of Kubernetes using BigQuery (27th of February, 2017)

The Kubernetes project’s incredible velocity

39 000+
users on Slack

14 000+
Kubernetes jobs

43 000+
commits
the latest year

40 000+
opened
Pull Requests
the latest year

50 000+
Kubernetes
professionals

24 000+
edX course enrolls

20 000+
opened issues
the latest year

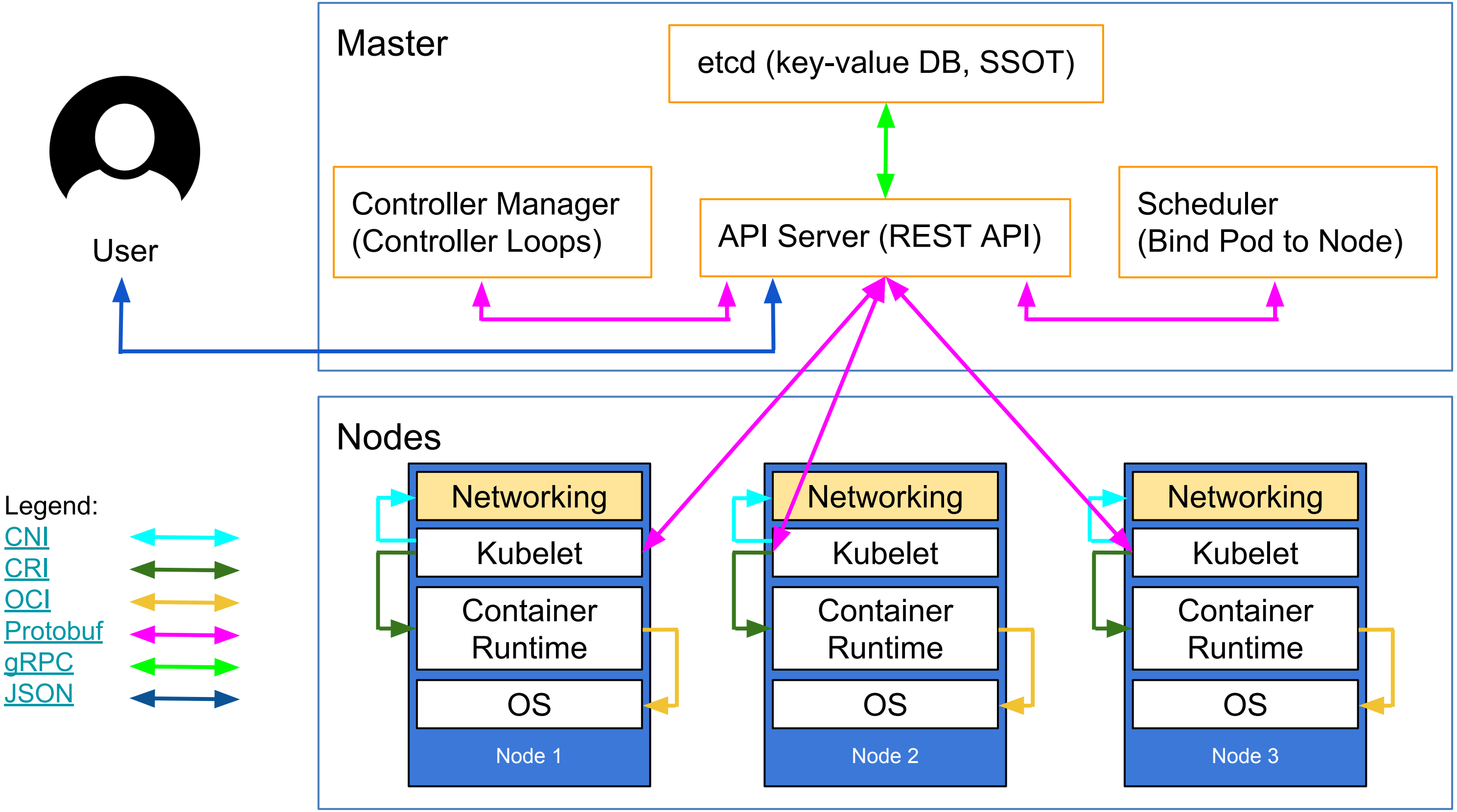
5 000+
unique authors

~20 PRs
merges/day
in the core repo

Last updated: 8.6.2018

Source 1 Source 2 Source 3 Source 4

Kubernetes’ high-level component architecture



User

Master

etcd (key-value DB, SSOT)

Controller Manager (Controller Loops)

API Server (REST API)

Scheduler (Bind Pod to Node)

Nodes

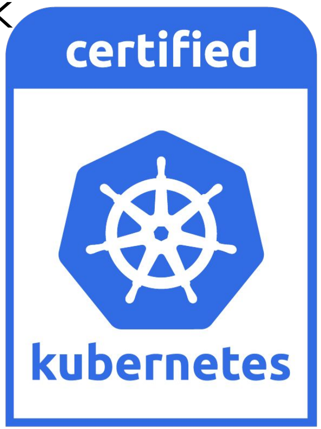
Node 1	Node 2	Node 3
Networking	Networking	Networking
Kubelet	Kubelet	Kubelet
Container Runtime	Container Runtime	Container Runtime
OS	OS	OS

Legend:

- CNI ⟷
- CRI ⟷
- OCI ⟷
- Protobuf ⟷
- gRPC ⟷
- JSON ⟷

Certified Kubernetes Conformance

- CNCF launched the software conformance program for Kubernetes
  - Implementations run conformance tests and upload results, which can be rerun by end users
  - New mark and more flexible use of Kubernetes trademark for conformant implementations
  - 40 Certified Kubernetes Partners
  - Now working on profiles (e.g., bare-metal, cloud, beta)
  - Amazon has committed to certifying EKS, which is the last of the distributions and platforms of any size
  - Great press pickup



Training and Certification

- Over 16,000 people have registered for the free Introduction to Kubernetes course
- 2,451 people have registered for the $299 Kubernetes Fundamentals course
- 487 people have already enrolled to take the CKA exam
- 23 companies are Kubernetes Certified Service Providers

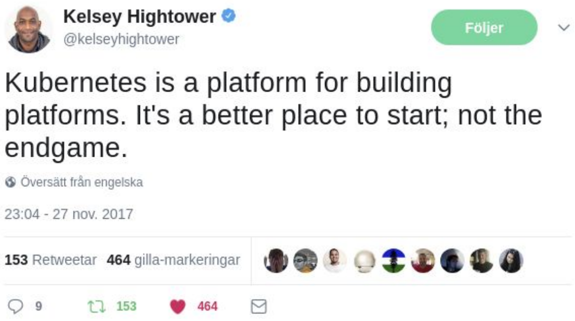
CERTIFIED
kubernetes
ADMINISTRATOR

Fresh docs on how to extend Kubernetes

Brand new docs on how to extend Kubernetes

Kubernetes has *many* extension mechanisms:

- API Aggregation (beta)
- **kubectl** plugins (alpha)
- CustomResourceDefinitions, Example intro (beta)
- Container Network Interface plugins (stable)
- Scheduler webhook & multiple (beta)
- Device plugins (alpha)
- Initializers & Admission webhook (beta)
- External Cloud Provider Integrations (alpha)
- API Server authn / authz webhooks (stable)
- Container Runtime Interface plugins (alpha)
- Container Storage Interface plugins (alpha)



Kelsey Hightower
@kelseyhightower

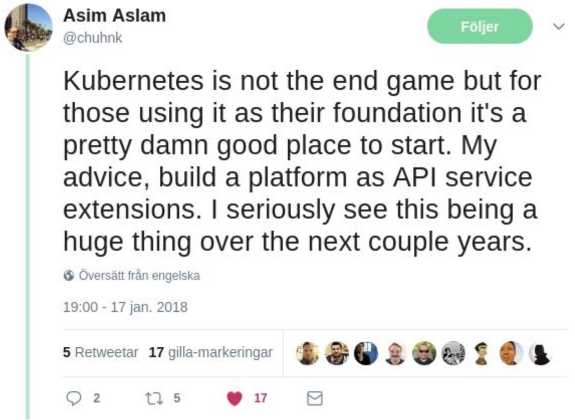
Kubernetes is a platform for building platforms. It's a better place to start; not the endgame.

Översätt från engelska

23:04 - 27 nov. 2017

153 Retweetar **464** gilla-markeringar

9 153 464



Asim Aslam
@chuhnk

Kubernetes is not the end game but for those using it as their foundation it's a pretty damn good place to start. My advice, build a platform as API service extensions. I seriously see this being a huge thing over the next couple years.

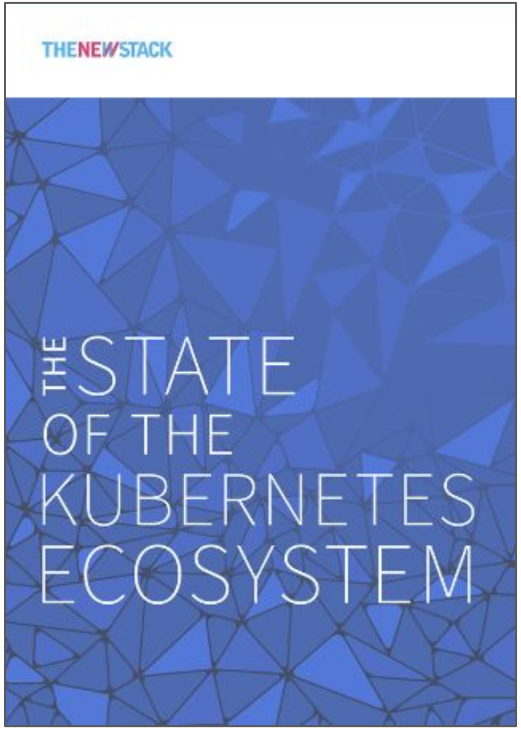
Översätt från engelska

19:00 - 17 jan. 2018

5 Retweetar **17** gilla-markeringar

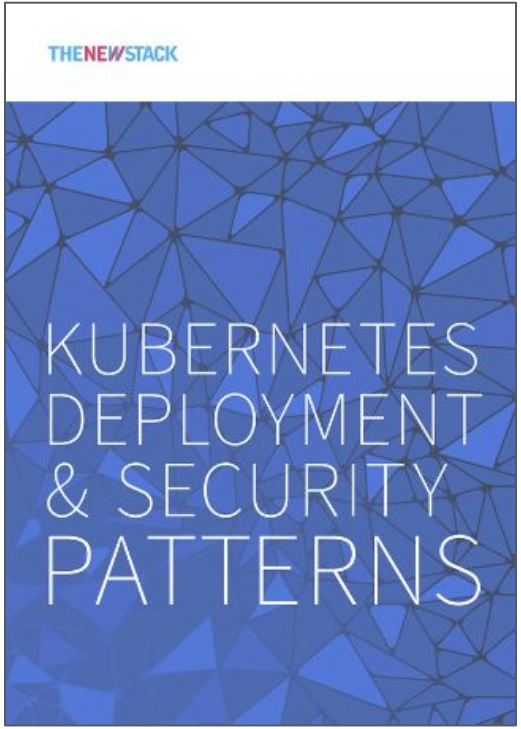
2 5 17

Free ebooks from The New Stack to dive into



THENEWSTACK

THE STATE OF THE KUBERNETES ECOSYSTEM

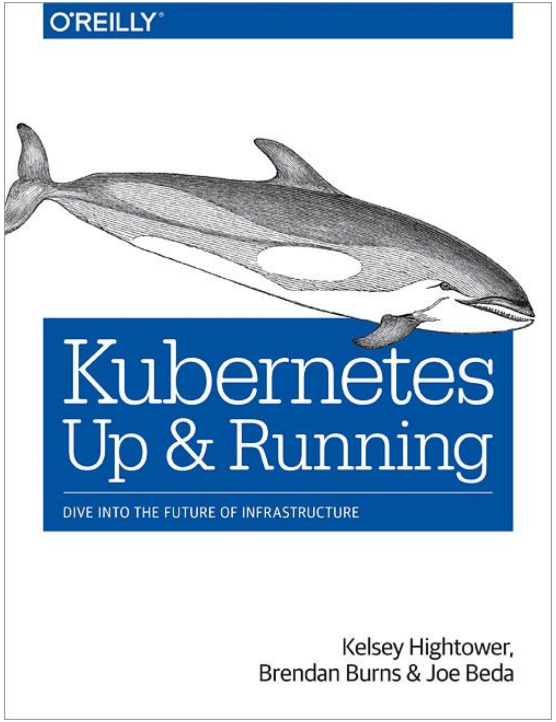


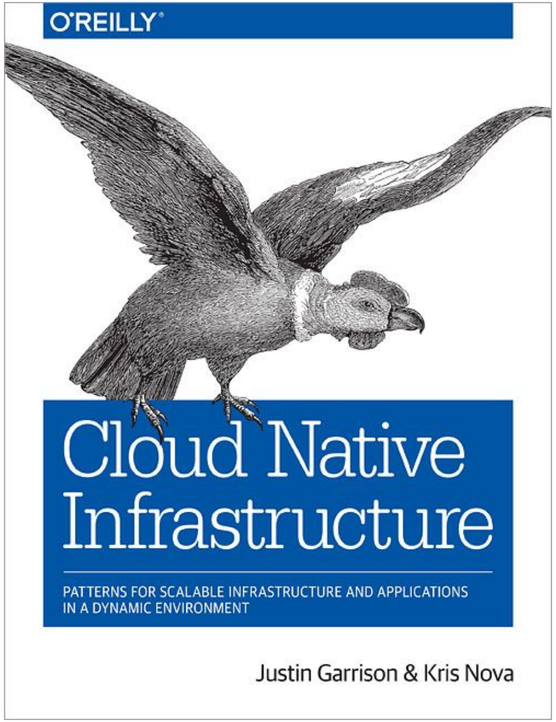
THENEWSTACK

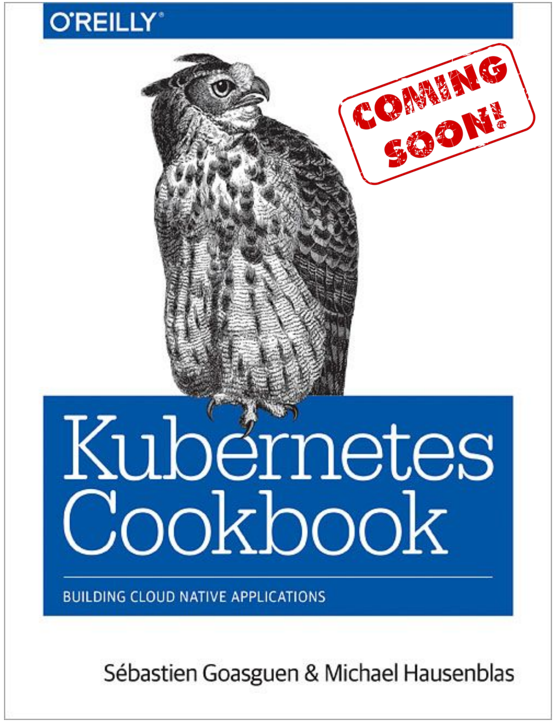
KUBERNETES DEPLOYMENT & SECURITY PATTERNS

The State of the Kubernetes Ecosystem & Kubernetes Deployment and Security Patterns

Excellent Kubernetes books




O'REILLY
Cloud Native
Infrastructure
PATTERNS FOR SCALABLE INFRASTRUCTURE AND APPLICATIONS
IN A DYNAMIC ENVIRONMENT
Justin Garrison & Kris Nova


O'REILLY
COMING
SOON!
Kubernetes
Cookbook
BUILDING CLOUD NATIVE APPLICATIONS
Sébastien Goasguen & Michael Hausenblas

Kubernetes: Up and Running & Cloud Native Infrastructure & Kubernetes Cookbook

The core primitive: A Pod

The basic, atomically deployable unit in Kubernetes.

A Pod consists of one or many co-located containers.

The containers in a Pod share the loopback interface (localhost) and can share mounted directories.

A Pod represents *a single instance of an application.*

Each Pod has it’s own, uniquely assigned and internal IP.

Pods are **mortal**, which means that if the node the Pod runs on becomes unavailable, the workload also goes

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  namespace: default
  labels:
    app: nginx
spec:
  containers:
  - image: nginx:1.13.9
    name: nginx
    ports:
    - name: http
      containerPort: 80
```

A replicated, upgradeable set of Pods: A Deployment

With a Deployment, you can manage Pods in a declarative and upgradable manner.

Note the *replicas* field. Kubernetes will make sure that amount of Pods created from the template always are running.

When the Deployment is updated, Kubernetes will perform an rolling update of the Pods running in the cluster. Kubernetes will create one new Pod, and remove an old until all Pods are new.

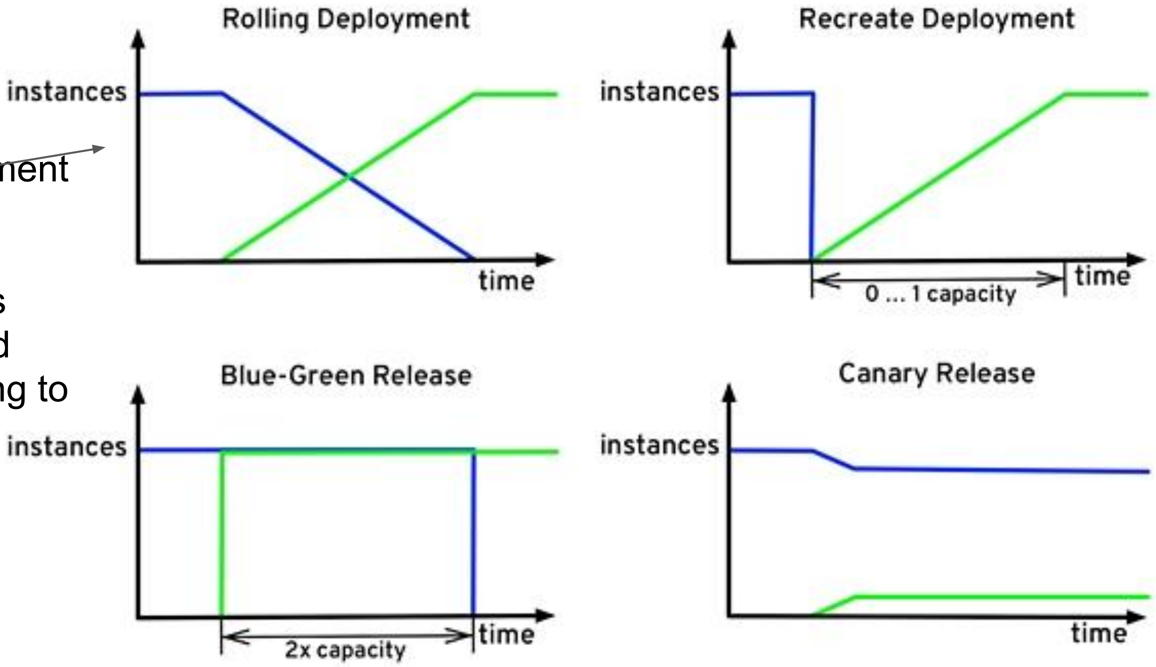
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: nginx
  name: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - image: nginx:1.13.9-alpine
        name: nginx
        ports:
        - name: http
          containerPort: 80
```

The Pod Template

Various possible Deployment upgrade strategies

The built-in Deployment behavior

The other strategies can be implemented fairly easily by talking to the API.



Picture source: [Kubernetes effect by Bilgin Ibryam]

Access your replicated Pods by creating a Service

A Service exposes one or many Pods via a stable, immortal, internal IP address in the cluster, a ClusterIP. The ClusterIP can be declaratively specified, or dynamically allocated.

The service is also reachable via cluster-internal DNS: `{service-name}.{namespace}.svc.cluster.local` or `nginx.default.svc.cluster.local`

The Service selects Pods based on the label key-value selectors (here `app=nginx`)

A Service can expose multiple ports.

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
  namespace: default
  labels:
    app: nginx
spec:
  type: ClusterIP
  ports:
  - name: http
    port: 80
    targetPort: 80
  selector:
    app: nginx
```

The Pod Selector

Expose your Service to the world with an Ingress

A Service is by default only reachable inside of the cluster.

In order to expose the Service to the internet, you must deploy an Ingress controller, like Traefik, and create an Ingress Rule

The Ingress rule is the Kubernetes-way of mapping hostnames and paths from internet requests to cluster-internal Services.

The Ingress controller is a loadbalancer that looks at the API when creating the rules.

```yaml
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: nginx
  namespace: default
  labels:
    app: nginx
spec:
  rules:
  - host: nginx.demo.kubernetesfinland.com
    http:
      paths:
      - path: /
        backend:
          serviceName: nginx
          servicePort: 80
```

The Service reference

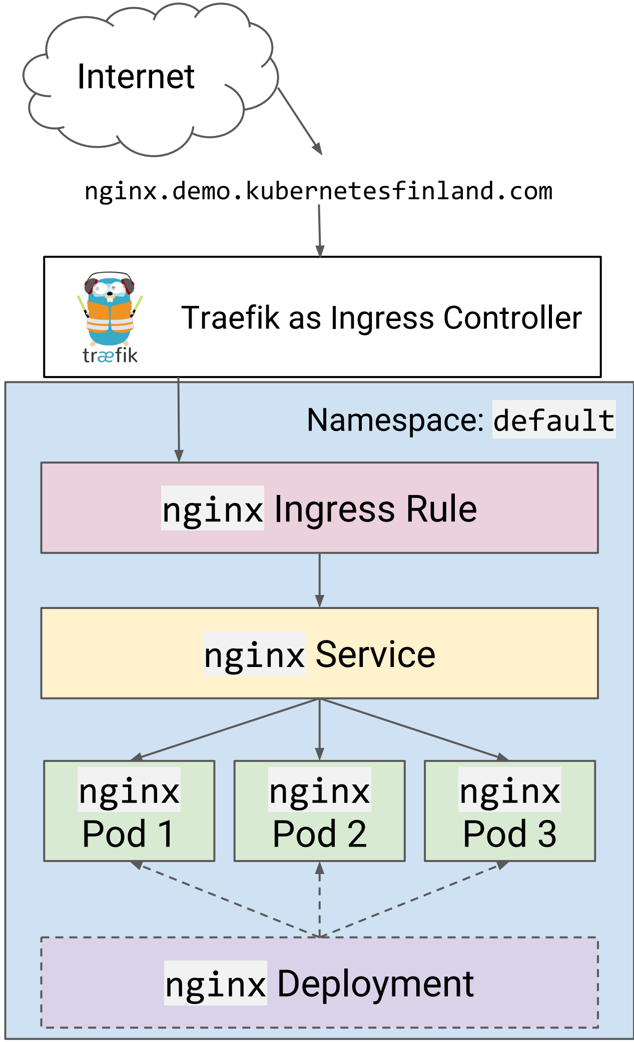
Put all your stuff in a Namespace

A Namespace is a logical isolation method, most resources are namespace-scoped.

You can group logically similar workloads in one namespace and enforce different policies.

You can e.g. have one namespace per team, and let them play in their own virtual environment.

Role Based Access Control (RBAC) can be used to control what Kubernetes users can do, and what resources in what namespaces an user can access is one of the parameters to play with there.



How do I kick the tires with Kubernetes?

Play with Kubernetes right away in your browser!

Create a single-node cluster on your laptop or workstation with minikube

Create a real cluster with only a couple of commands with kubeadm

Create a production-ready cluster on AWS with kops

Create a Kubernetes cluster on GCE with GKE (Google Kubernetes Engine)

kubicorn is a Kubernetes installer project which has gained some traction

Create a cluster with kubeadm

1. Provision a Linux machine with Ubuntu, Debian, RHEL, CentOS or Fedora
2. Install kubeadm:

```
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -
echo “deb http://apt.kubernetes.io/ kubernetes-xenial main” > /etc/apt/sources.list.d/kubernetes.list
apt-get update && apt-get install -y kubeadm docker.io
```

3. Make kubeadm set up a master node for you:

```
kubeadm init
```

4. Install a Pod Network solution from a third-party provider:

```
kubectl apply -f https://git.io/weave-kube-1.6
```

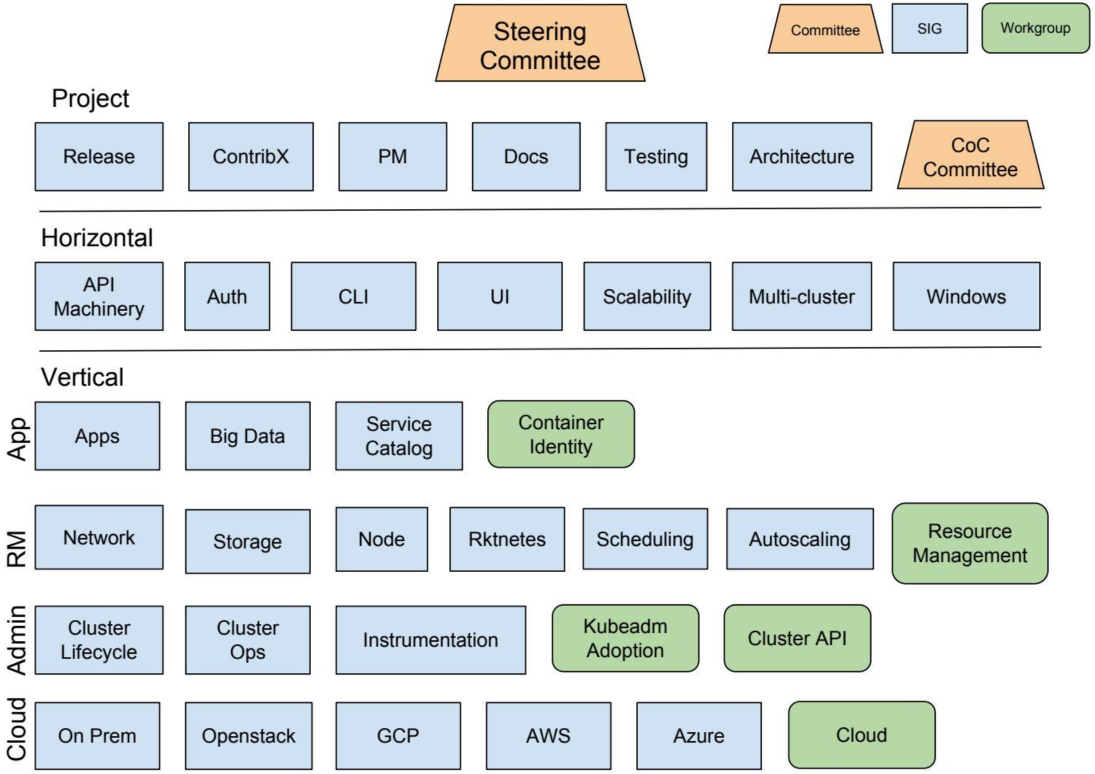
5. Repeat step 1 & 2 on an other node and join the cluster:

```
kubeadm join --token <token> <master-ip>:6443
```

A couple of core Kubernetes features...

- **Self-healing**: Restarts containers that fail, replaces and reschedules containers when nodes die, kills containers that don't respond to your user-defined health check, and doesn't advertise them to clients until they are ready to serve
- **Automatic binpacking**: Automatically places containers based on their resource requirements and other constraints, while not sacrificing availability. Mix critical and best-effort workloads in order to drive up utilization and save even more resources.
- **Horizontal scaling and autoscaling**: Scale your application up and down with a simple command, with a UI, or automatically based on CPU usage or custom metrics
- **Automated rollouts and rollbacks**: Kubernetes progressively rolls out changes to your application or its configuration, while monitoring application health to ensure it doesn't kill all your instances at the same time. If something goes wrong, Kubernetes will rollback the change for you.
- **Service Discovery and Load Balancing**: No need to modify your application to use an unfamiliar service discovery mechanism. Kubernetes gives containers their own IP addresses and a single DNS name for a set of containers, and can load-balance across them
- **Secret and configuration management**: Deploy and update secrets and application configuration without rebuilding your image and without exposing secrets in your stack configuration
- **Storage Orchestration**: Automatically mount the storage system of your choice, whether from local storage, a public cloud provider such as GCP or AWS, or a network storage system such as NFS, iSCSI, Gluster, Ceph, Cinder, or Flocker
- **Batch Execution**: In addition to services, Kubernetes can manage your batch and CI workloads, replacing containers that fail, if desired

Everything is done in SIGs (Special Interest Groups)



Steering Committee

Committee SIG Workgroup

Project

Release ContribX PM Docs Testing Architecture CoC Committee

Horizontal

API Machinery Auth CLI UI Scalability Multi-cluster Windows

Vertical

App: Apps Big Data Service Catalog Container Identity

RM: Network Storage Node Rktnetes Scheduling Autoscaling Resource Management

Admin: Cluster Lifecycle Cluster Ops Instrumentation Kubeadm Adoption Cluster API

Cloud: On Prem Openstack GCP AWS Azure Cloud

Special Interest Groups manage Kubernetes’ various components and features.

All code in the Kubernetes Github organization should be owned by one or more SIGs; with directory-level granularity.

SIGs have regular (often weekly) video meetings where the attendees discuss design decisions, new features, bugs, testing, onboarding or whatever else that is relevant to the group. Attending these meetings is the best way to get to know the project

Image source

Next steps?

Follow the Kubernetes blog, YouTube channel & Twitter feed

Do as 24 000+ others and take the free edX "Introduction to Kubernetes" course

Join 39 000+ others in the Kubernetes Slack: http://slack.k8s.io

Prep for and take the Certified Kubernetes Administrator or Certified Kubernetes Application Developer exam

Join a Special Interest Group and attend the weekly meetings

Kick the tires with Kubernetes on your machines with minikube or kubeadm

Check out the weekly Kubernetes Community Meeting or Kubernetes Office Hours on Zoom

Join the community in Shanghai & Seattle!

Thank you!

@luxas on Github
@kubernetesonarm on Twitter
lucas@luxaslabs.com