# Kubernetes: The Final Frontier

**Amanda Kay Moran**



#ossummit

@AmandaK_Data

# The Final Frontier….

*This is an introduction to Kubernetes tutorial.*

*The continuing mission: to explore the strange new worlds of microservices, containerization, and their management. To seek out new skills and new adventures. To boldly go where no one has gone before!*

# Agenda

- What is Kubernetes?
- Why is it so Popular?
- Why Should I learn about Kubernete?
- Kubernetes Architecture
- What is a Pod? What is a Deployment?
- Install MiniKube
- Deploy Simple Application
- Deploy Web Application
- How to have High Availability and Scalability
- Where to Learn more!

# Who is Amanda?

- Bay Area based Software Engineer/Solutions Architect
- Machine Learning, Analytics, Distributed Systems
- Variety of Companies, big and small!
- Apache Committer and PMC Member Apache Trafodion
- What Do I Love:
  - Dogs
  - Disneyland
  - Veggies
  - Teaching, training, helping others
  - Running and Exercise

# Hands On Lab: Prework

- Install a Hypervisor
- Personally, I use Virtualbox
- 2 minutes to give folks the ability to kick off their download
- We will also be using Minikube --if you already have virtualbox installed

# What is Kubernetes?

- Kubernetes (K8s) is an open-source system for automating deployment, scaling, and management of containerized applications.
- Donated to the CNCF foundation by Google and has been developed and used at Google for over 15 years
- First release was in June 2014 and is now 6 years old

# What is Kubernetes?

- **Benefits of Kubernetes**
  - Run Applications Anywhere
  - Easy cluster management
  - Service Discovery and load balancing
  - Storage management
  - Automated rollouts and rollbacks
  - Automatic bin packing -- placing containers by resources
  - Self Healing
  - Horizontal Scaling

# Why is it so Popular?

- Companies are moving away from monolithic applications

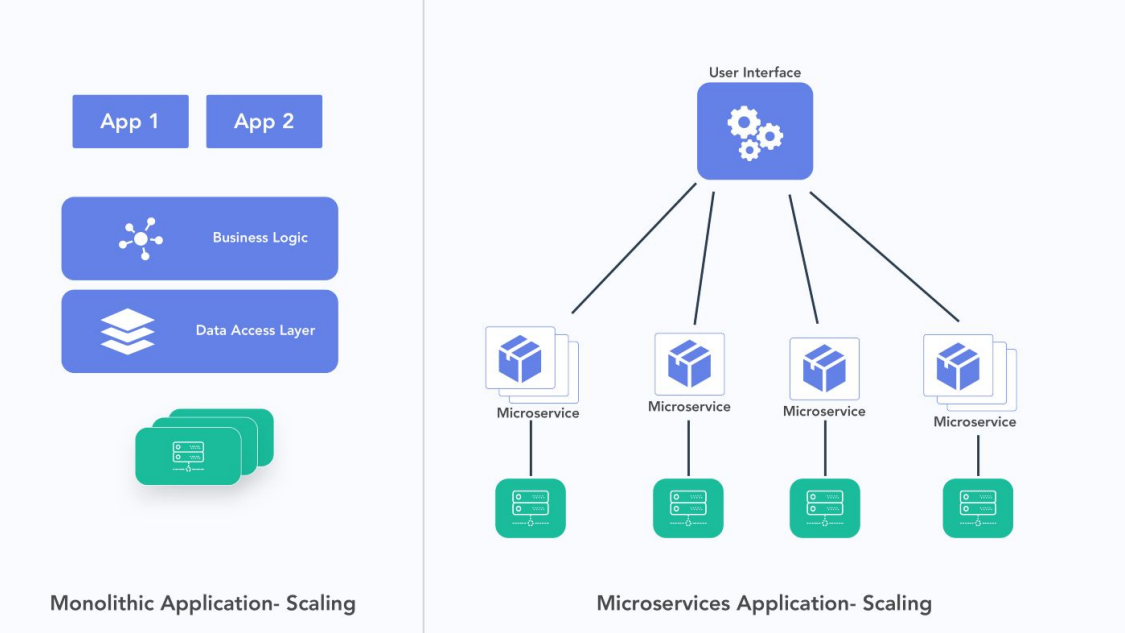
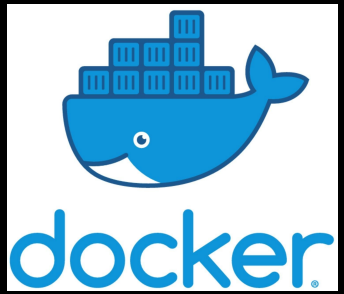



Photo Credit: Microservices Orchestration with Kubernetes

# Why is it so Popular?

- Why use Microservices?
  - A microservice is a service that just does a single task and that is all
  - Rapid development
  - Ability to swap out components of an artitecture with ease
  - Easy to automate for CI/CD
  - Flexibility → easier to change course
- Microservices are easy to containerize
  - Container Services: Docker
- Kubernetes is the best place to manage containers

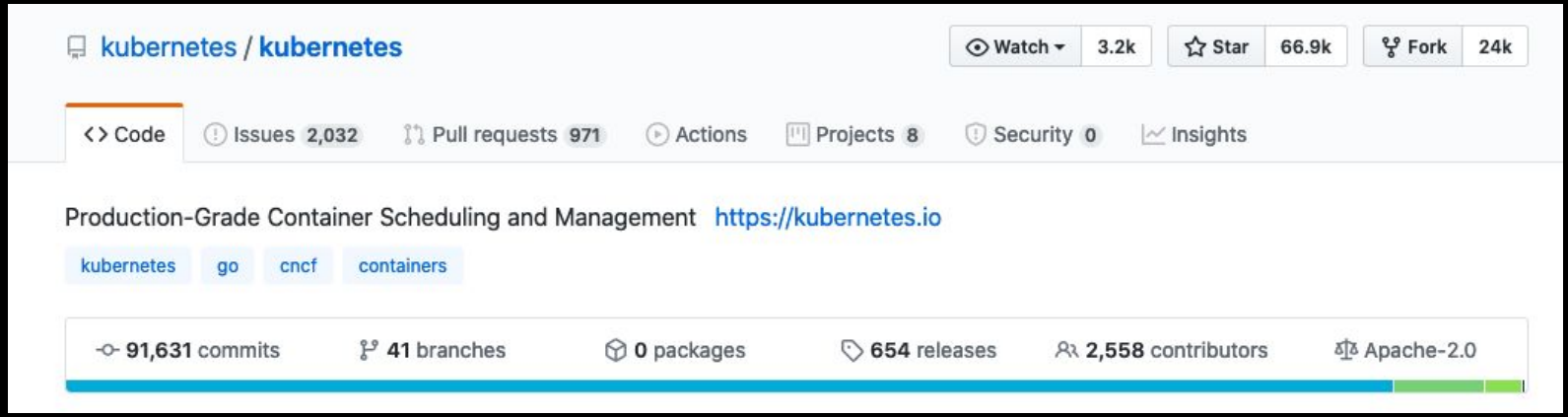


# Why is it so Popular?

- Many ways to install and run
- Manually
- Cloud Offering (Platform as a Service)
  - EKS
  - GKE
  - AKS
- Within minutes you can have a managed Kubernetes cluster
- 3 month release cycle

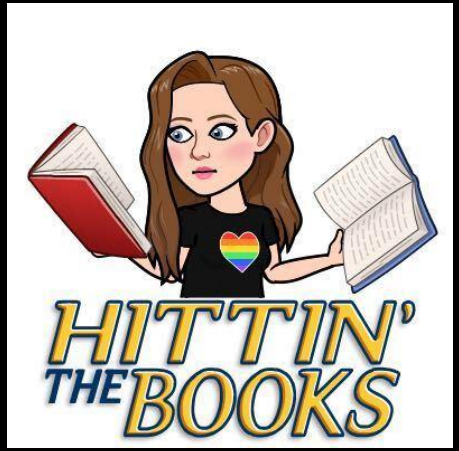
# Why is it so Popular?

- Kubernetes has a very active and supportive community
- Many different company contribute to the project
- Special Interest Groups -- that meet regularly
- Meetups and Conferences



# Why Should I learn about Kubernetes?

- If you work in DevOps or Infrastructure
  - No Brainer!
- Very popular new technology
- But what about Developers and Data Scientist
  - The infrastructure will affect how build/use applications
  - Will affect how you build models
  - Storage
  - Hardcoding
  - Bringing in new package → not persistent
- An Introduction!



# Kubernetes Architecture

- Terminology
- Control Plane
  - kube-api-server
  - kube-controller-manager
  - kube-scheduler
  - cloud-control-manager
  - Etcd (Key Value Database)
- Node
  - Kubelet
  - kube-proxy
  - Container Runtime

# Kubernetes Architecture

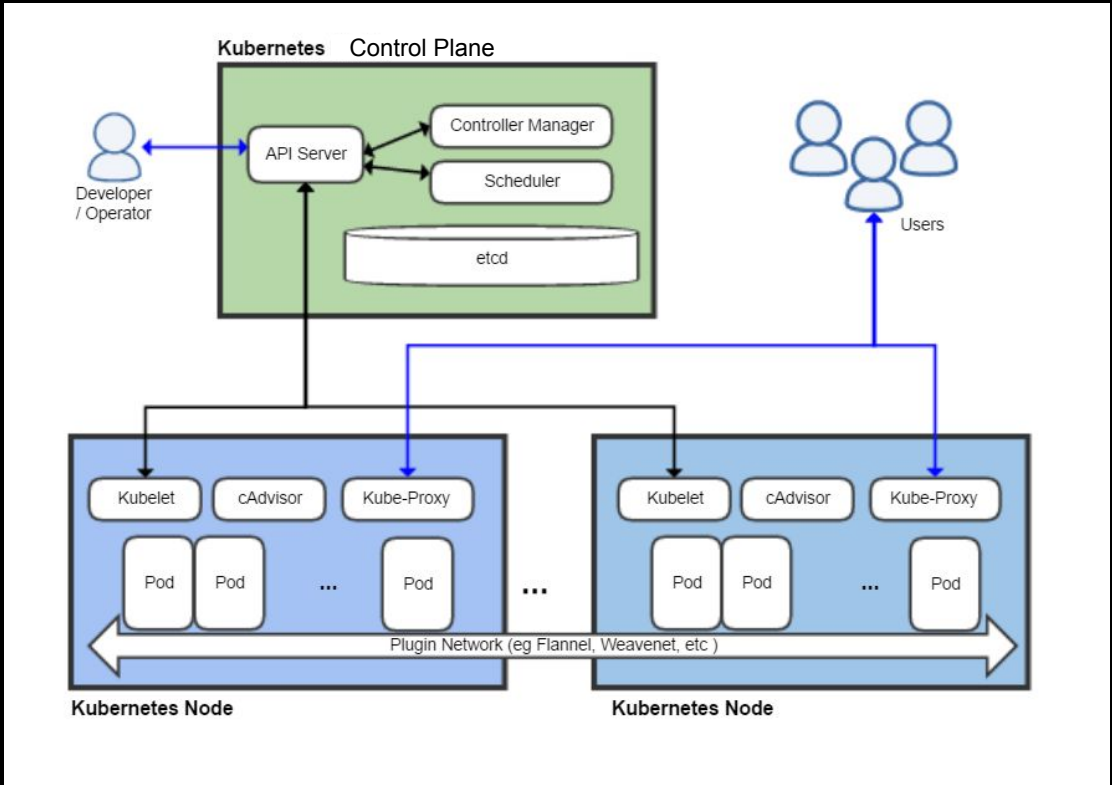



Photo Credit: Wikipedia: Kubernetes

# Kubernetes Architecture

- Control Plane
  - kube-api-server
  - kube-controller-manager
  - kube-scheduler
  - cloud-control-manager
  - Etcd (Key Value Database)

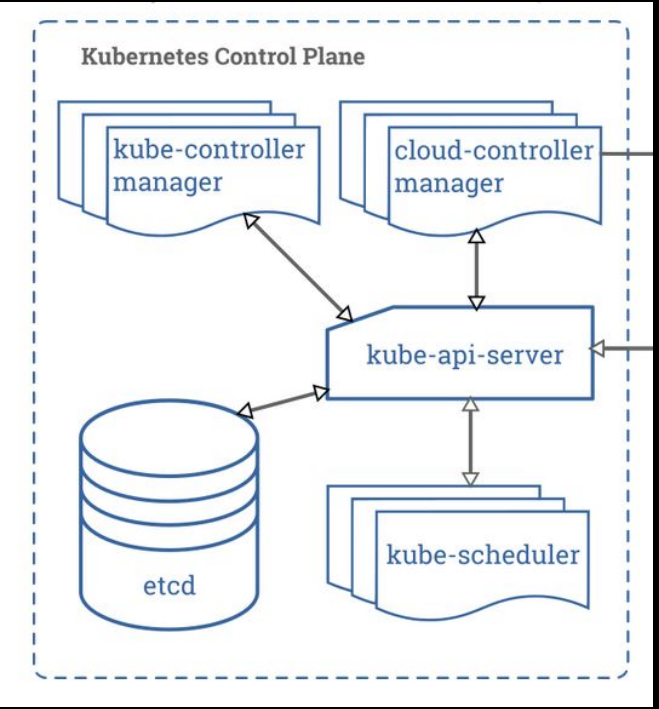
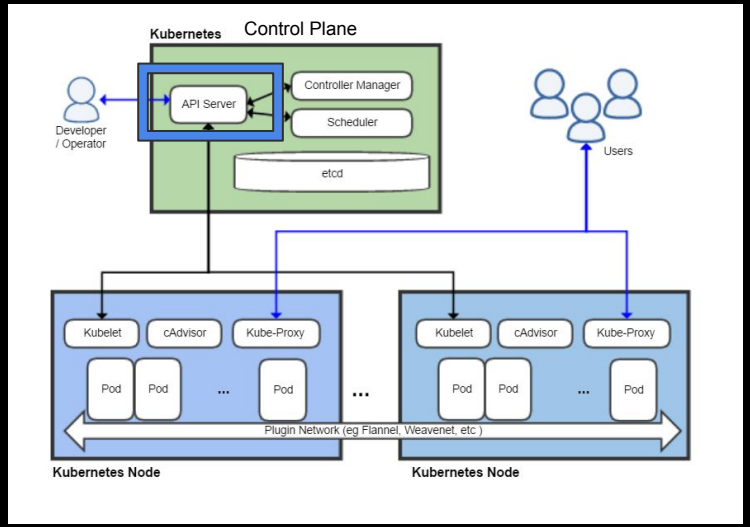



Photo Credit: Kubernetes Documentation

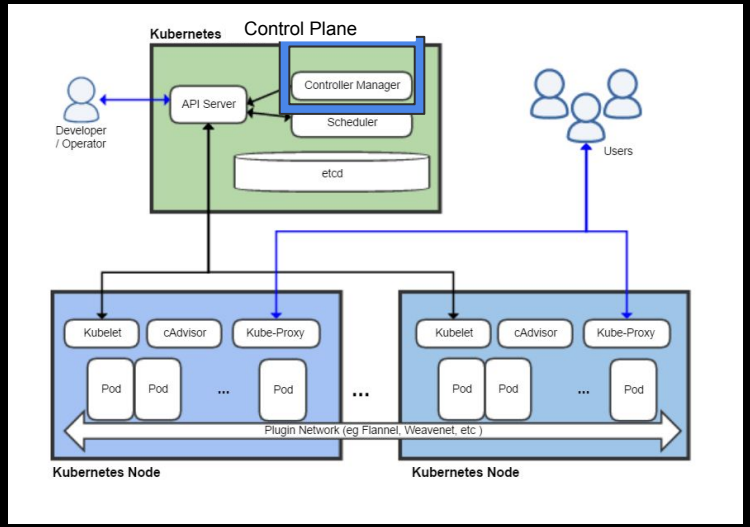
# Kubernetes Architecture

- kube-api-server
  - Responsible for the Kubernetes API
  - How you will interact with the k8s cluster
  - Will use a command line tool kubectl to interact
  - Interacts with etcd, scheduler, controllers



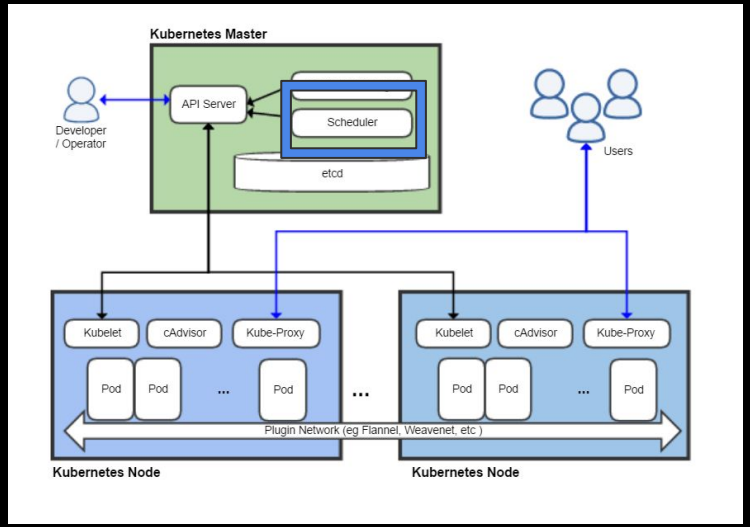
# Kubernetes Architecture

- kube-controller-manager
  - Runs multiple controllers
  - Node controller
  - Replication Controller
  - Service Account and Token Controller



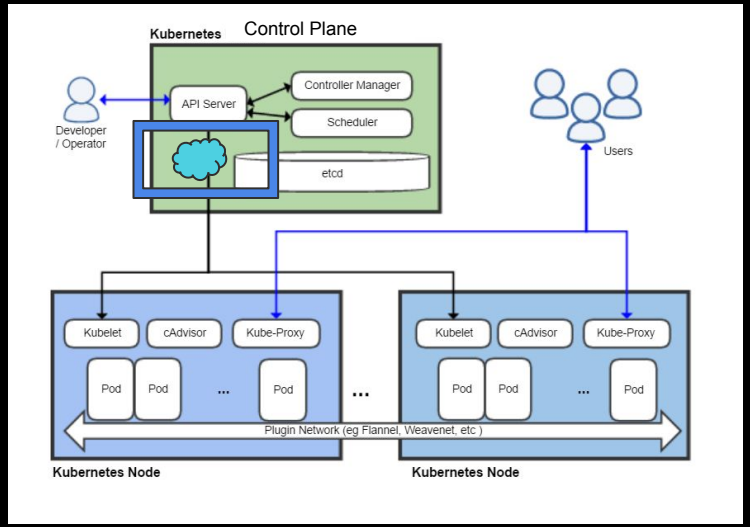
# Kubernetes Architecture

- kube-scheduler
  - Scheduler for the pods on the cluster
    - Resources needed
    - Affinity
    - Data locality
  - Uses different algorithms (configurable) to place
  - Supply and demand



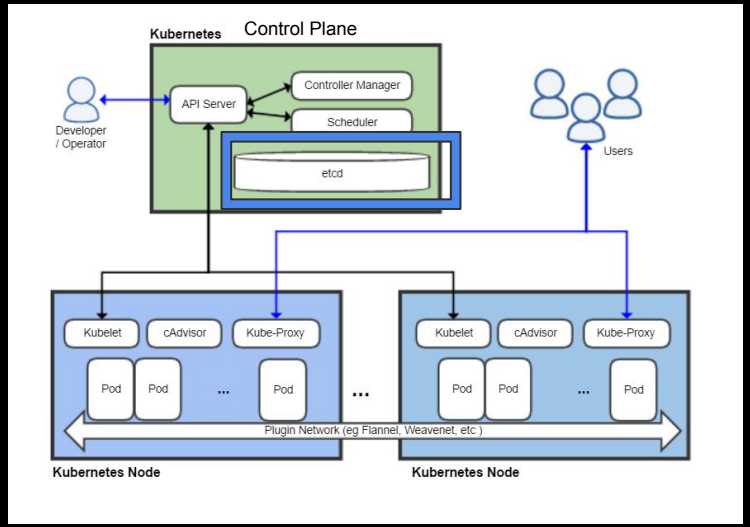
# Kubernetes Architecture

- cloud-control-manager
  - Ability to use your Cloud providers API
  - Only have in the cloud (not on perm or Minikube)



# Kubernetes Architecture

- etcd
  - Key value consistent database
    - Consistent over Available → CAP Theorem
  - Stores all activity on the cluster
  - Combined with API Server to perform actions
  - The API Server used watch API on etcd to monitor



# Kubernetes Architecture

- Node
  - Kubelet
  - kube-proxy
  - Container Runtime

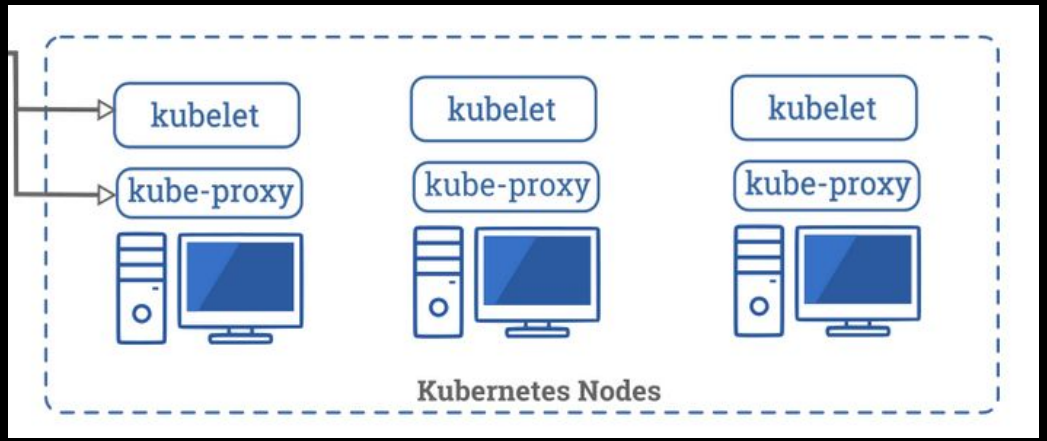



Photo Credit: Kubernetes Documentation

# Kubernetes Architecture

- Kubelet
  - A process (agent) that runs on each kubernetes node
  - Uses Pod Specifications to understand which pods and containers should run
  - Monitors node it is responsible for
  - Control plane and kubelet work closely together

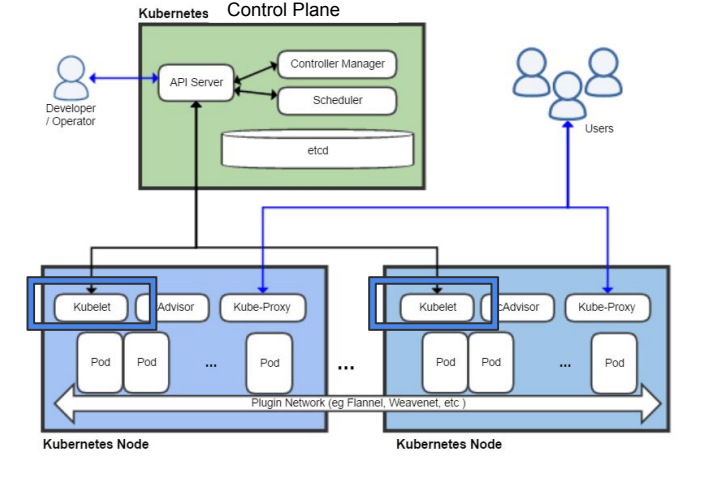



Photo Credit: [Kubernetes Documentation]

# Kubernetes Architecture

- kube-proxy
  - Network proxy that runs on each node
  - Uses network rules to allow for pods to talk to each other inside and outside the cluster

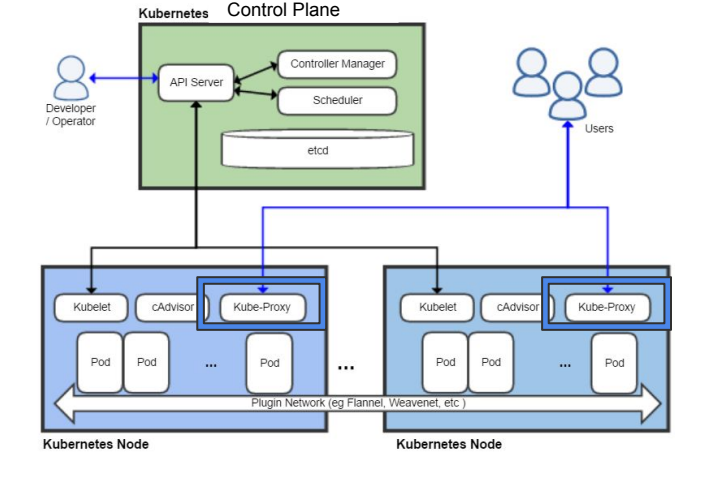



Photo Credit: [Kubernetes Documentation]

# Kubernetes Architecture

- Container Runtime
  - Containers live within pods and package our application
  - Must be installed on each node
  - Doesn’t have to just be Docker
    - Containerd
    - Crio

# Workloads: What is a Pod?

Photo Credit: Science News

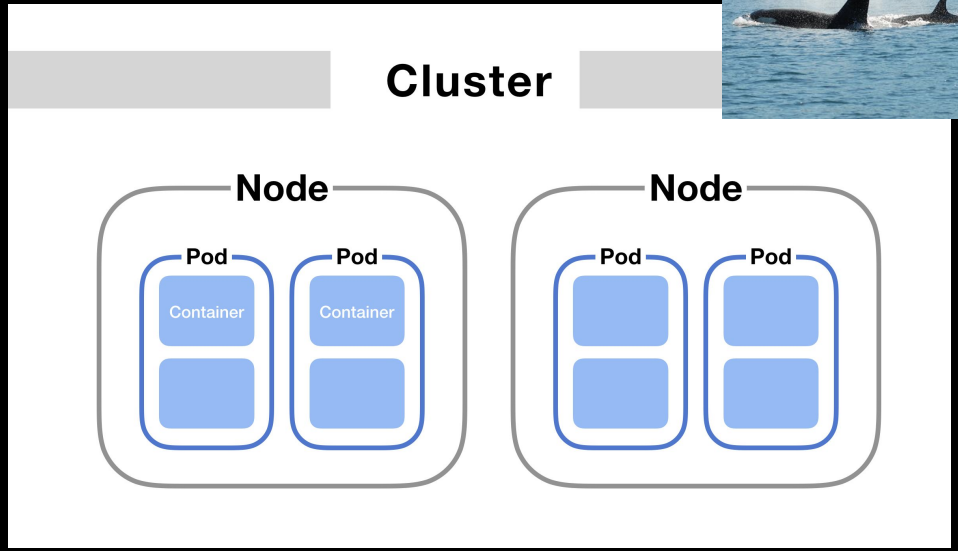



Photo Credit: MatthewPalmer.net

# Workloads: What is a Pod?

- Smallest unit within managed Kubernetes
  - Containers run outside Kubernetes not managed
- Pods configured by a yaml file
- Will pull a container  image from a source
- Containers in a pod are always colocated
- Unique internal ip address per pod
- Containers talk to each other via localhost
- All containers use common volume storage
- Managed by the kube api or controller

```
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    creationTimestamp: null
5    labels:
6      run: pod
7    name: pod
8  spec:
9    containers:
10   - image: amoran06/simple
11     name: pod
12     resources: {}
13   dnsPolicy: ClusterFirst
14   restartPolicy: Always
15 status: {}
```

# Controllers: What is a Controller?

- A controller is a control loop that manages the environment
  - Never terminates and continues to monitor the situation
- kube-control-manager helps with managing each controller type
- It works on the current state and gets you to the desired state
  - I have two pods and I want 3
- Think of an Oven senor

# Controllers: What is a Deployment?

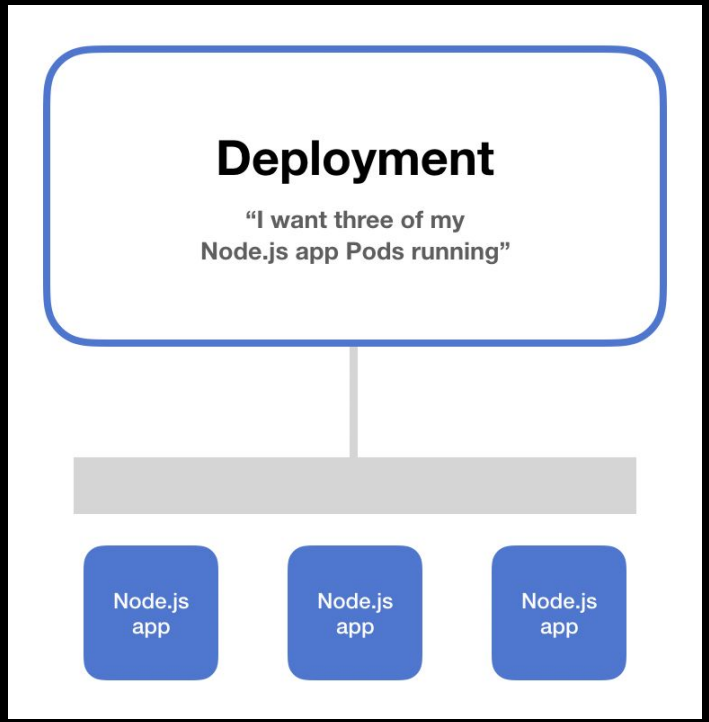



Photo Credit:
MatthewPalmer.net

# Controllers: What is a Deployment?

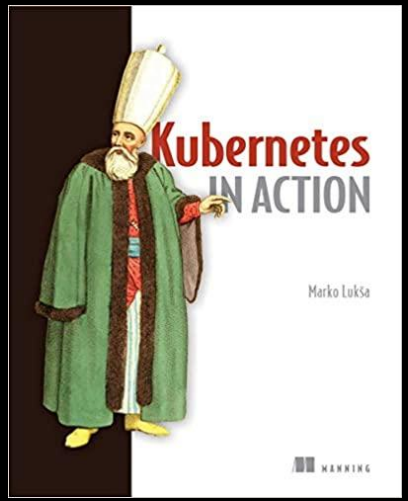
- A deployment is setting a desired state for your pods
- Delete and Add Pods
- Add a ReplicaSet → How many I want
- Do I want a restart?
  - Batch job that just runs once
- Ability to easily update and upgrade

# Lab Time

Go to: https://github.com/amandamoran/opensourceSummit.git

# Continue Learning…

- Reading Kubernetes Docs
- Udacity course: Introducing Scalable Microservices Kubernetes
- Kubernetes in Action by Marko Luksa
- Linux Foundation Training: Introduction to Kubernetes



# Become Apart of the Community!

- Join a Meetup
- Make a Pull Request
- Keep attending Conferences
- Answer Questions
- Write docs
- Everyone is welcome!

# References

- Kubernetes Docs
- Kubernetes Wikipeida

# Thank you!