Intro to Kubernetes

By: Tim Zabel

What is Kubernetes?

- Platform for managing containerized workloads and services
- Helps bring together many containers and orchestrates them together
- Allows for flexible auto-scaling of resources

Why Should I Use It?

- Provides a centric management environment
  - If you already have a lot of containers, k8s makes it easy to bring them under central management
  - Orchestrates networking, computing, storage

- Allows for segregation of containers
  - Namespaces offer different permissioning levels for dev/stage/prod environments and users

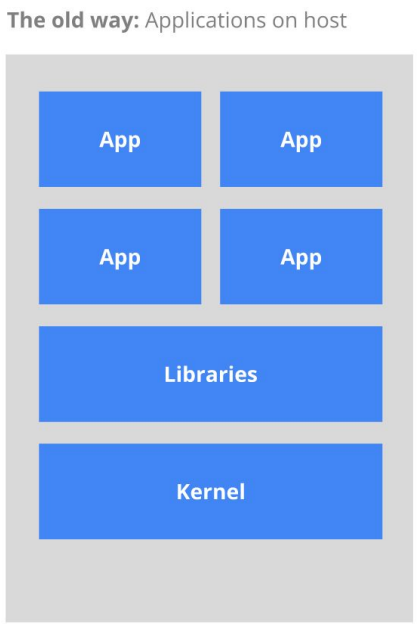
- Portability

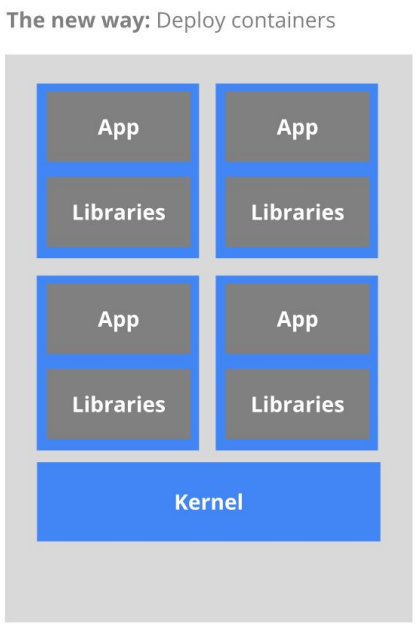
What Kubernetes is NOT

- Does NOT deploy source code or build your application
- Does NOT dictate logging, monitoring, or alerting
  - But provides mechanisms to do so
- Does NOT do CI/CD

What does K8s Look Like?

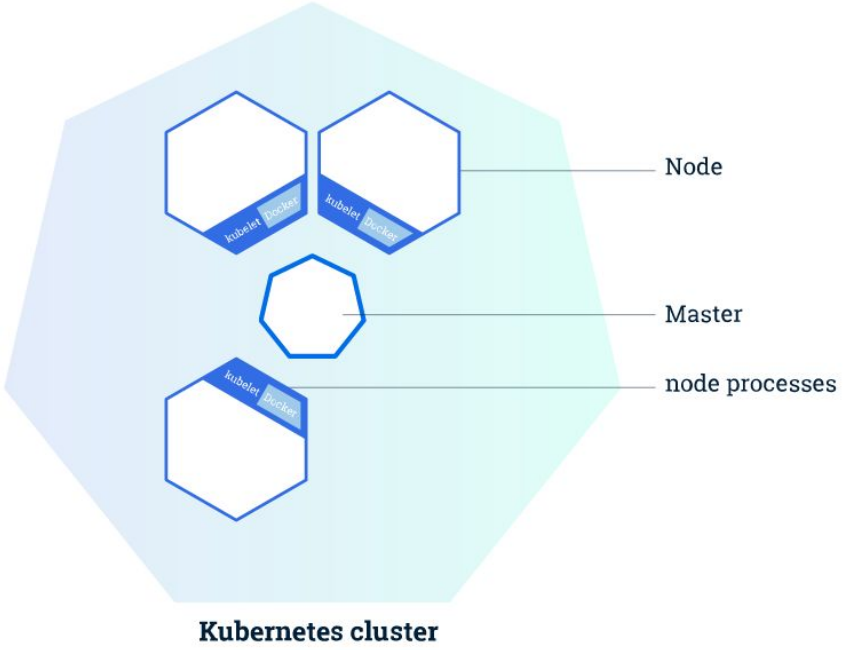
- Applications are containerized and portable
- All virtualized, agnostic of actual OS





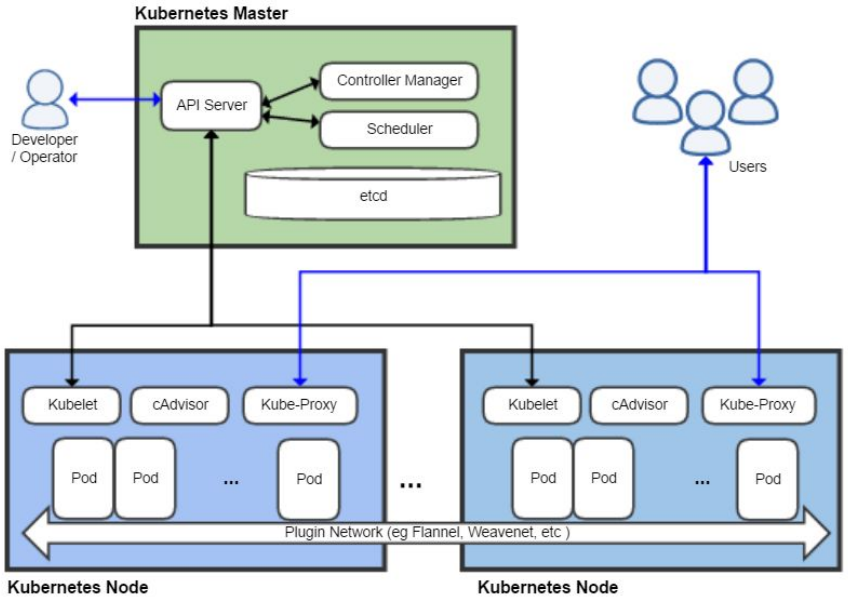
Higher Level Abstraction

- Master is responsible for managing the cluster
- Nodes are VMs or physical machines that serve as workers
- Container runtime (i.e. Docker) pulls in applications



In-Depth: K8s Master and Layout

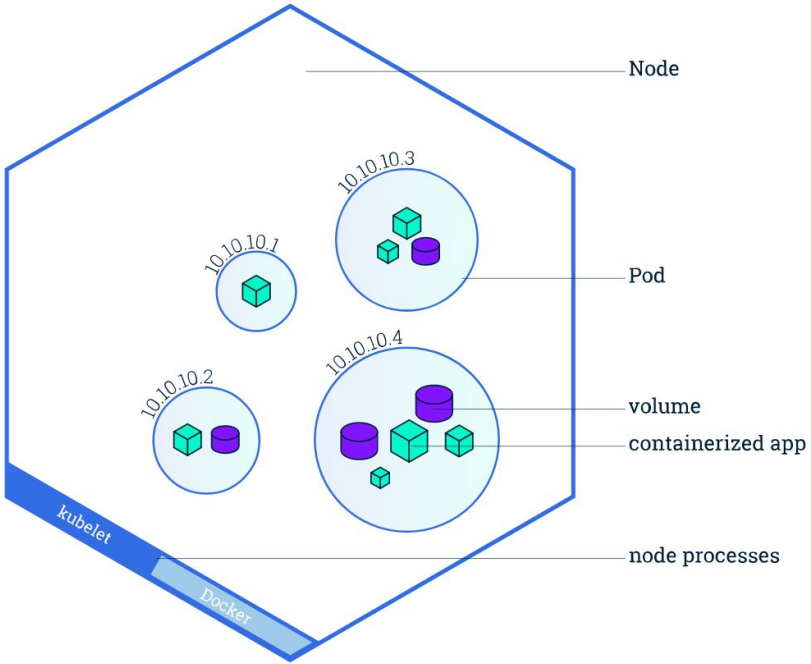
- Cluster communicates to the master via the API Server
- Each Node runs kubelet process, which receives API Server requests



Credit: Khatan66 on Wikipedia

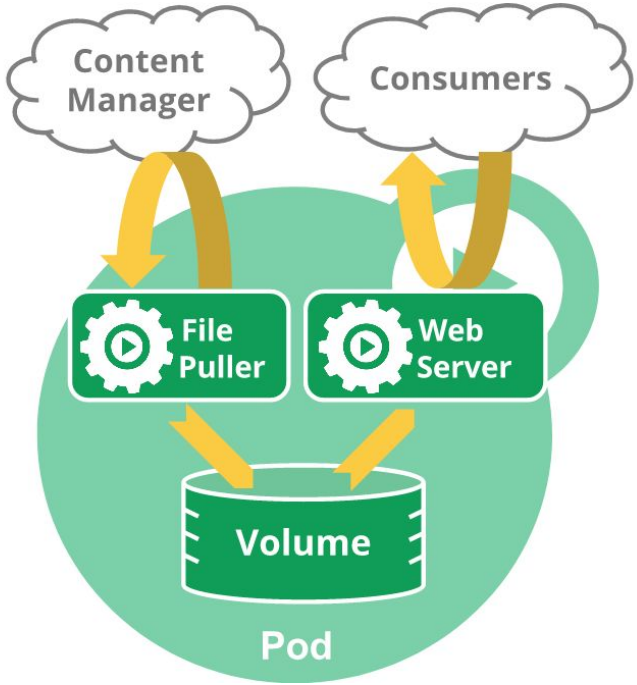
In-Depth: Nodes

- Pods always run on Nodes
- Nodes can have multiple pods
- Master takes care of scheduling of pods between Nodes
- Kubelet handles communication



Pods Overview

- Group of one or more containers
- Each pod serves a specific purpose
  - i.e. one pod per application
- Containers talk over localhost



Pod Controllers

- StatefulSet
  - Guarantees uniqueness of pods
  - Not interchangeable
- Deployment
  - Used to specify *desired* state
  - Naming of pods doesn't matter
- DaemonSet
  - Run daemons
  - Used for storage (ceph), monitoring (fluentd, logstash)

Well that’s great and all, but how do I actually create my application?

Containerize Application

- Application must be containerized and available on a hub
  - i.e. Build a Dockerfile → Docker registry

- Most of the work is in the containerization process

Create a Deployment

- YAML (or JSON)
- Specify deployment name, number of replicas (pods)

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.7.9
        ports:
        - containerPort: 80
```

Run Application With Kubectl

- `Kubectl create deployment -f <path_to_yaml>`

- Kubectl is the command line interface for interacting with clusters

Expose Deployment with Services

- `kubectl expose deployment <deployment_name> --type=LoadBalancer --port=<port>

- Kubernetes deployments are not inherently available to outside public
- Services are needed to expose applications to the public

Done!

- Basic application is now running and available on the web
- Application will restart if it fails
- Will always try to keep desired state after this point

Questions?