Kubernetes Deployment

What is Kubernetes?

The name Kubernetes originates from Greek, meaning "helmsman" or "pilot", and is the root of "governor" and "cybernetic".

K8s is an abbreviation derived by replacing the 8 letters "ubernete" with 8.

With Kubernetes you can deploy a full cluster of **multi-tiered** containers (frontend, backend, etc.) with a **single** configuration file and a **single** command (Ref).

Kubernetes is an open-source platform for **automating** deployment, scaling, and operations of **application containers** across **clusters** of hosts, providing container-centric infrastructure.

With Kubernetes, you are able to quickly and efficiently respond to customer demand:

- **Deploy** your applications quickly and predictably.
- **Scale** your applications on the fly.
- Seamlessly **roll out** new features.
- **Optimize** use of your hardware by using only the resources you need

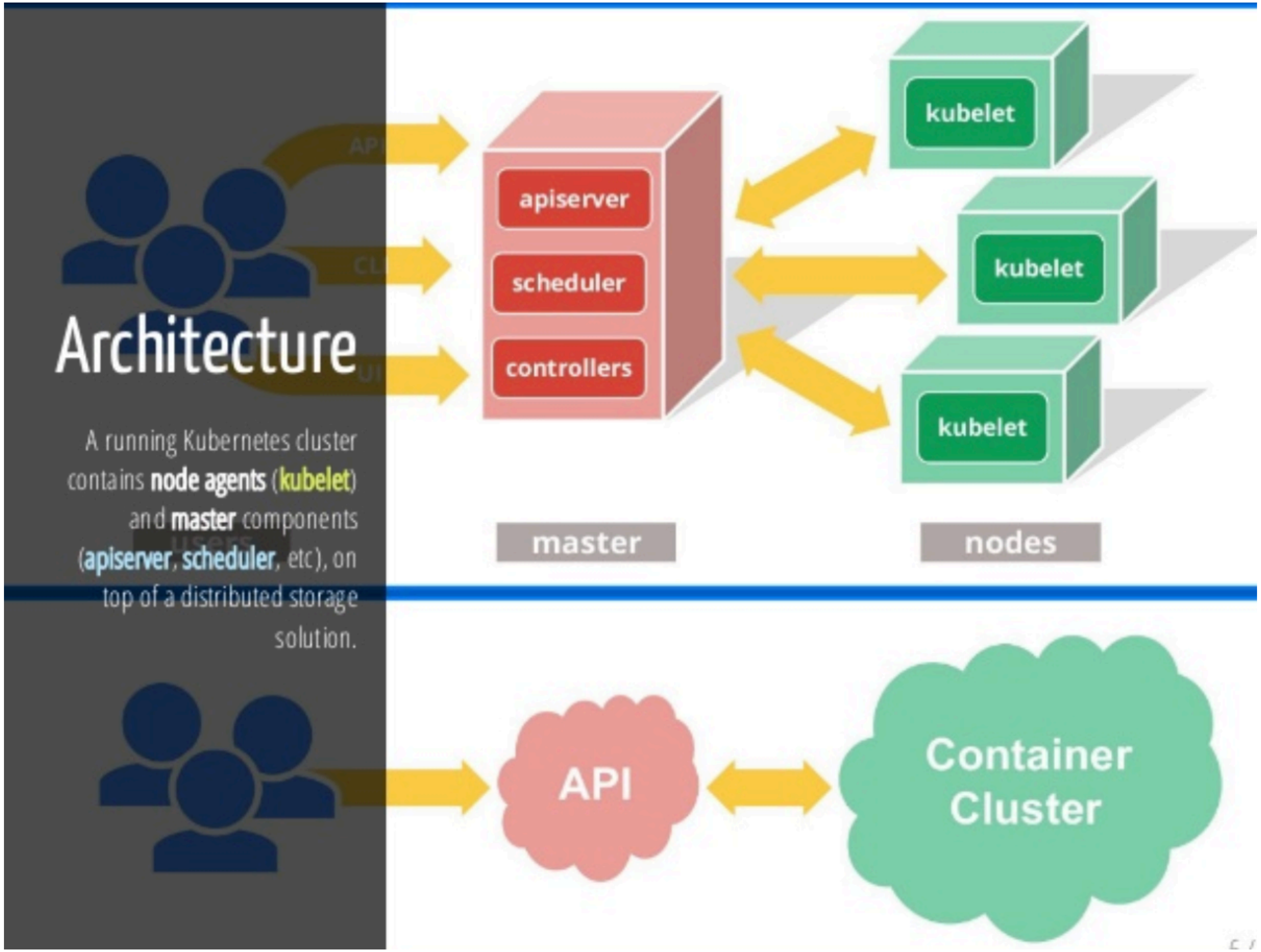
Kubernetes is:

- **portable**: public, private, hybrid, multi-cloud
- **extensible**: modular, pluggable, hookable, composable
- **self-healing**: auto-placement, auto-restart, auto-replication, auto-scaling

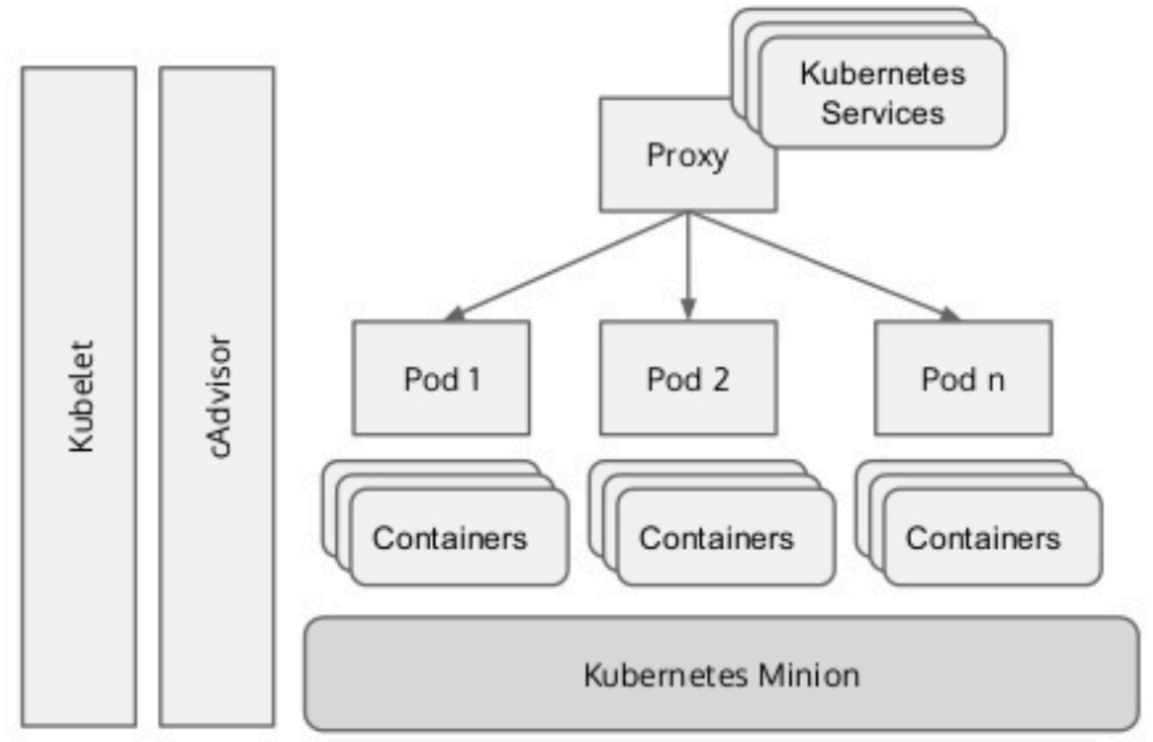
Ref: kubernetes.io

Architecture

A running Kubernetes cluster contains **node agents** (kubelet) and **master** components (apiserver, scheduler, etc), on top of a distributed storage solution.



Kubernetes Minion (Worker Node)



https://medium.com/google-cloud/kubernetes-101-pods-nodes-containers-and-clusters-c1509e409e16

Deployment/Updates

- Create a New Image
- Upload the Image
- Update Deployment
- Notify Kubernetes

Istio

- Language-agnostic Service Mesh on a Kubernetes Cluster
  - Istio uses the Envoy proxy as its sidecar
  - Routing
  - Tracing/Metrics
- Setup

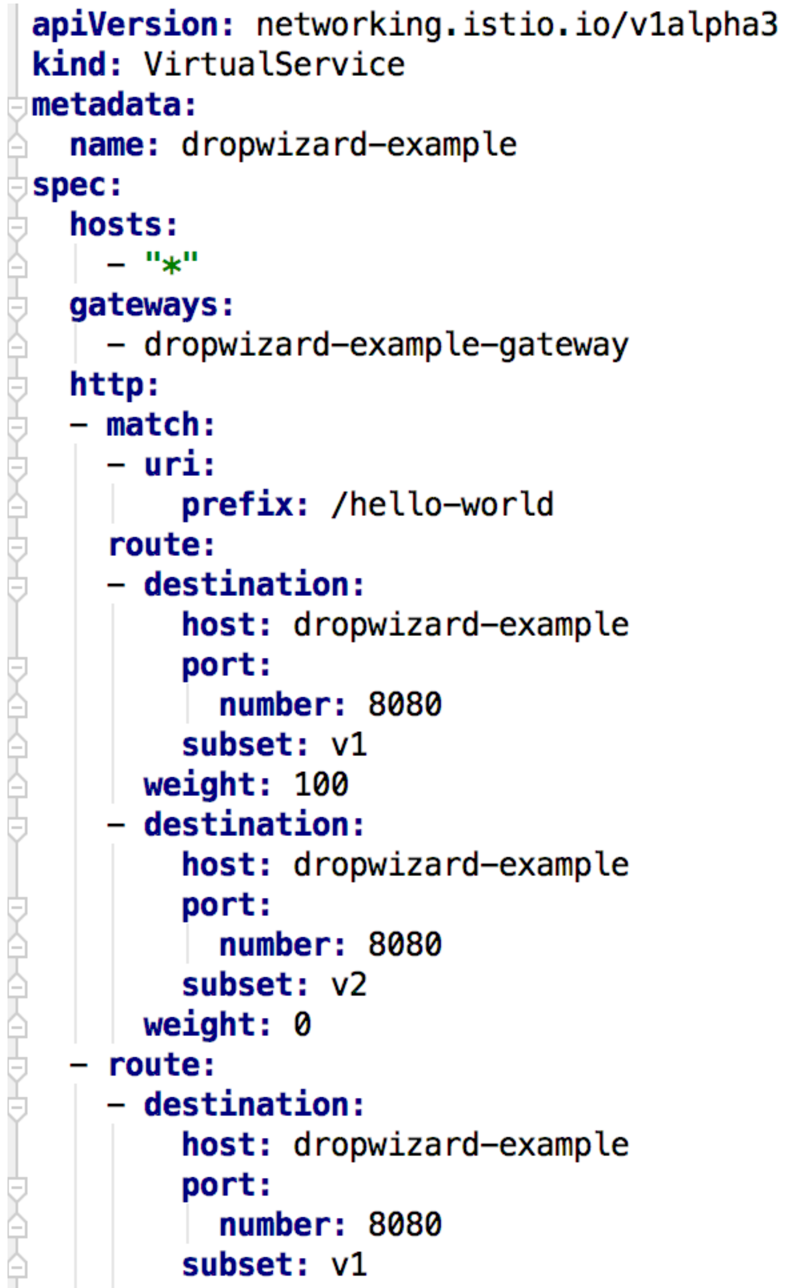
```
# Installing Istio on Kubernetes Engine
$ kubectl apply -f install/kubernetes/istio-auth.yaml
# Enable sidecar injection
$ kubectl label namespace default istio-injection=enabled
```

In production, use Helm and Tiller to manage the lifecycle of Istio.

Canary Deployment

- https://istio.io/blog/2017/0.1-canary/

Istio’s service mesh provides the control necessary to manage traffic distribution with complete independence from deployment scaling. This allows for a simpler, yet significantly more functional, way to do canary test and rollout.



```yaml
apiVersion: networking.istio.io/v1alpha3
kind: VirtualService
metadata:
  name: dropwizard-example
spec:
  hosts:
    - "*"
  gateways:
    - dropwizard-example-gateway
  http:
  - match:
    - uri:
        prefix: /hello-world
    route:
    - destination:
        host: dropwizard-example
        port:
          number: 8080
        subset: v1
      weight: 100
    - destination:
        host: dropwizard-example
        port:
          number: 8080
        subset: v2
      weight: 0
  - route:
    - destination:
        host: dropwizard-example
        port:
          number: 8080
        subset: v1
```

Telemetry

- Demonstrates how to collect telemetry information from the mesh
- https://istio.io/docs/tasks/telemetry/
  - Jaeger
  - Prometheus
  - Grafana
  - Fluentd, Elasticsearch, Kibana Stack