

<http://bit.ly/mpuck8s>



Kubernetes

An Introduction





Who am I?

michael palassis

- infrastructure architect (ops background)
- certified kubernetes administrator

mxnxpx at gmail * com

github: github.com/MxNxPx

venmo: @Michael-Palassis





Level setting

By show of hands...

- Who has heard of containers (Docker)?
- Who has heard of Kubernetes?
- Has anyone worked with Kubernetes before?



Pets vs Cattle

Pets = Legacy approach = VM or BM apps

- Require great attention
- Given actual names
- Unique and procured individually
- Sad if they die





Pets vs Cattle

Cattle = New Approach = Containers

- Look after themselves
- Don't have specific names - given numbers
- Can be simply replaced if they die
- Standardized

Containers are the atomic unit for cloud





Moooooooooooo!

But what if I have a herd-load of cattle?



A large, semi-transparent watermark of the Kubernetes logo is centered behind the text. It features a circular design composed of several overlapping blue and white triangles of varying sizes.

Kubernetes

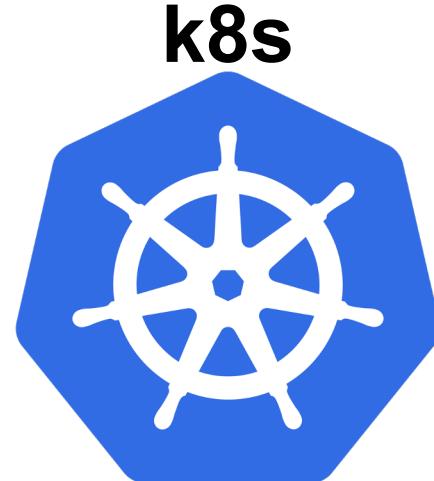
What Does “Kubernetes” Mean?



Greek for “pilot” or
“Helmsman of a ship”



Shorthand (numeronym)





What's the history of k8s?

- 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
- 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



What does Kubernetes do?

A Container Orchestration System





How does Kubernetes work?

- Kubernetes is a **declarative** way to **describe** your applications.
- Everything in Kubernetes is a **resource**.





What good is container orch?

- Resource efficiency and density with isolation
- Speed: start, create, replicate or destroy quickly
- Immense and smooth scaling
- Operational simplicity
 - Roll out updates
 - Consistency
 - Self-healing
- Improved developer productivity



Self Healing

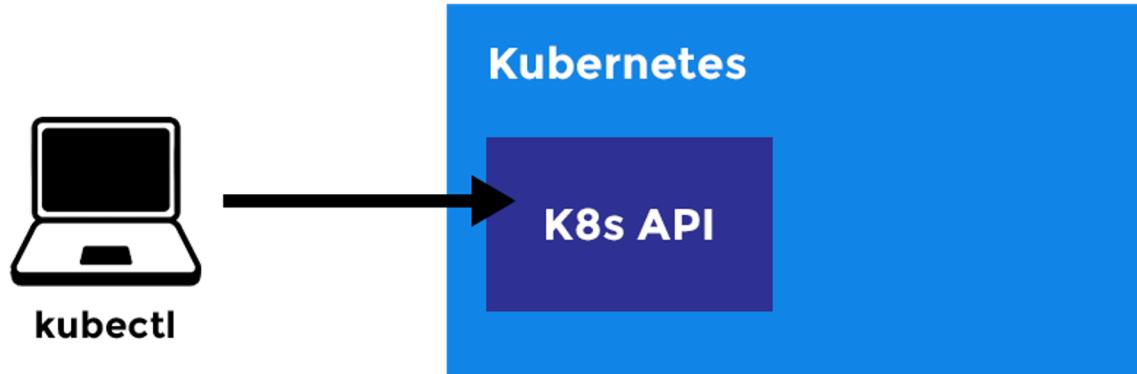
Kubernetes will **ALWAYS** try and steer the cluster to its desired state

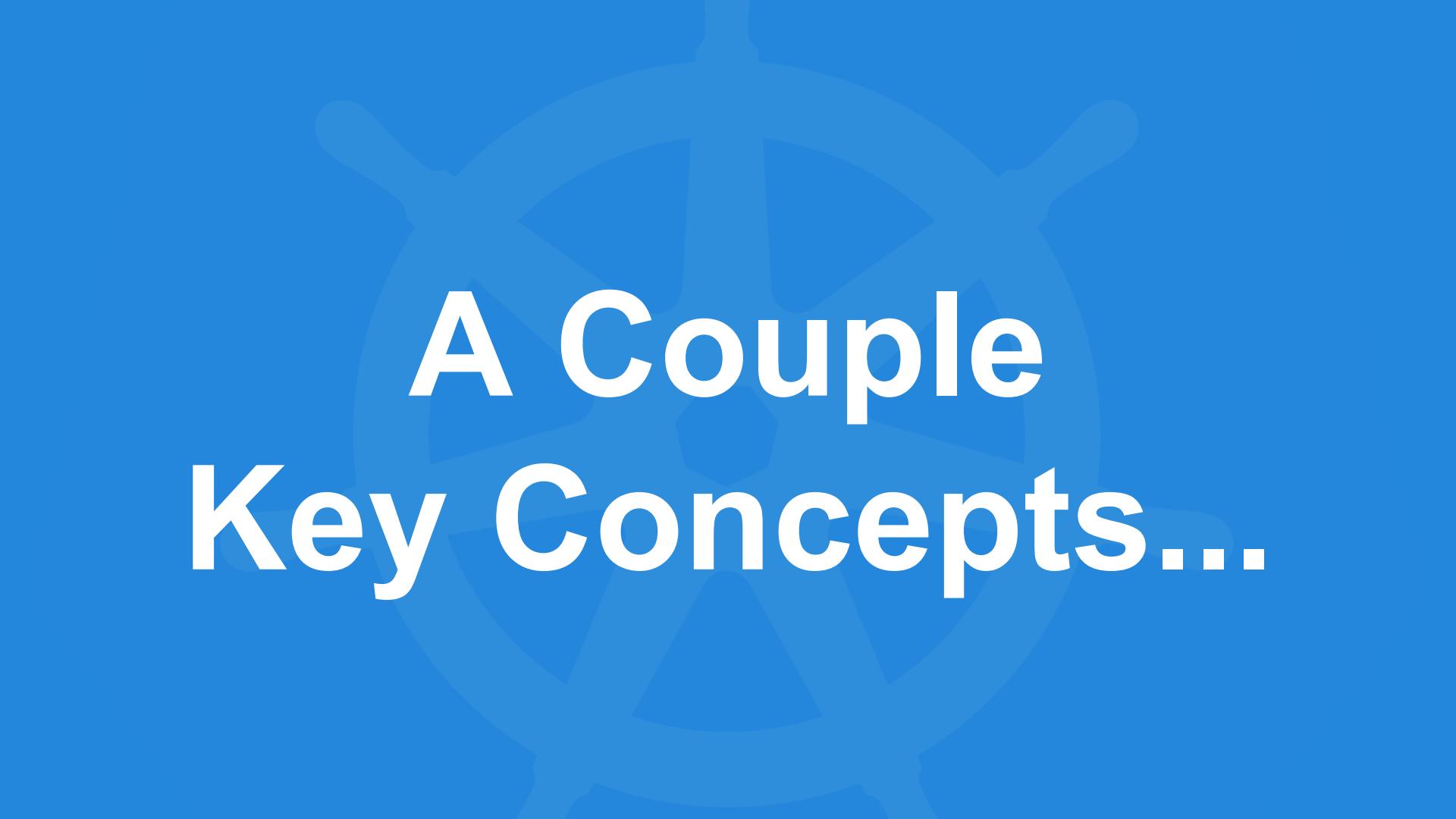
- **Me:** “I want 3 healthy instances of redis to always be running”
- **Kubernetes:** “Okay, I’ll ensure there are always 3 instances up and running”
- **(some time later...) Kubernetes:** “Oh look, one has died... I’m going to attempt to spin up a new one”



How to interface with k8s?

- API calls
 - Resources = endpoints in the Kubernetes API
- Kubectl (CLI)
 - Makes the API calls to Kubernetes



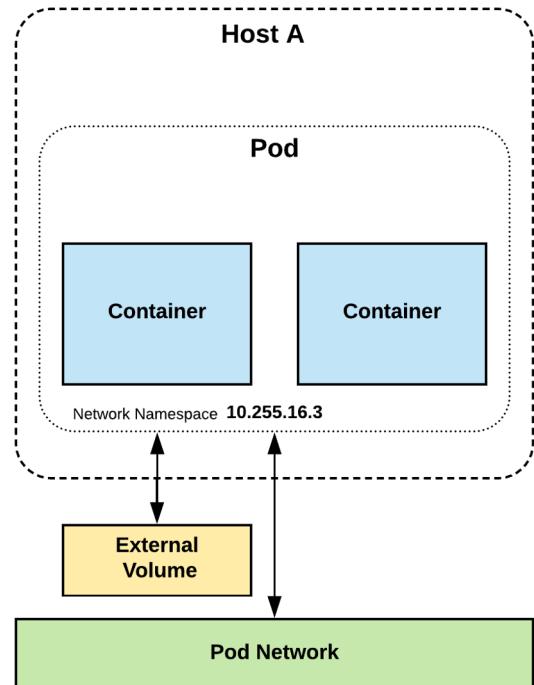


A Couple
Key Concepts...

Pods



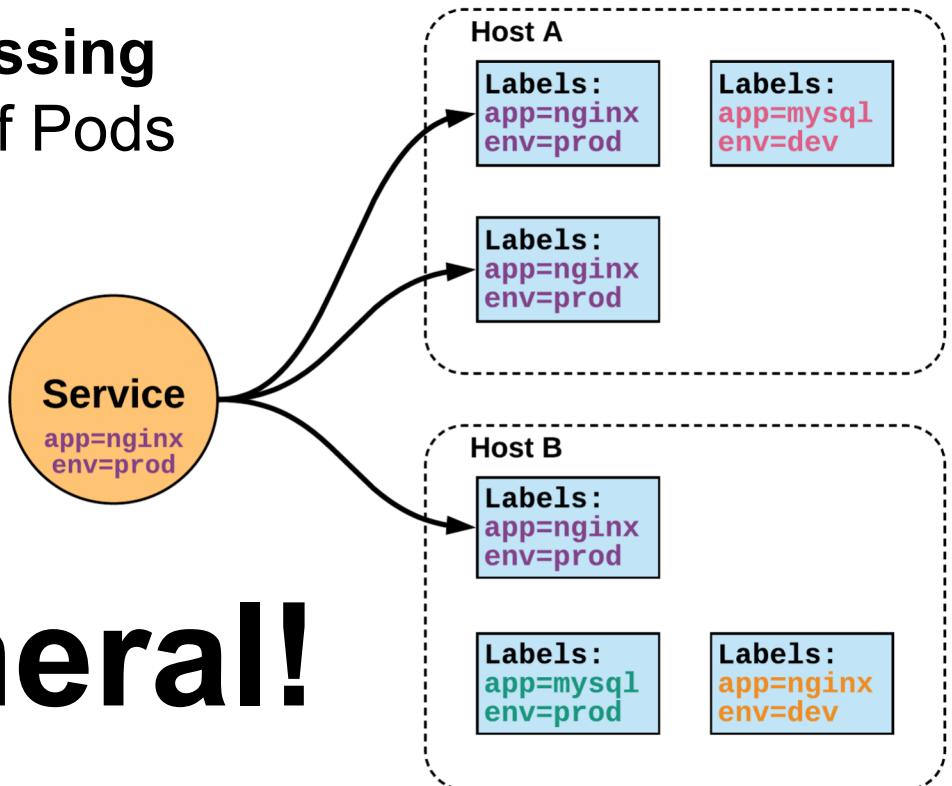
- **Atomic unit** or smallest “*unit of work*” of Kubernetes
- Pods are **one or MORE containers** that share volumes and namespace
- **They are ephemeral!**



Services

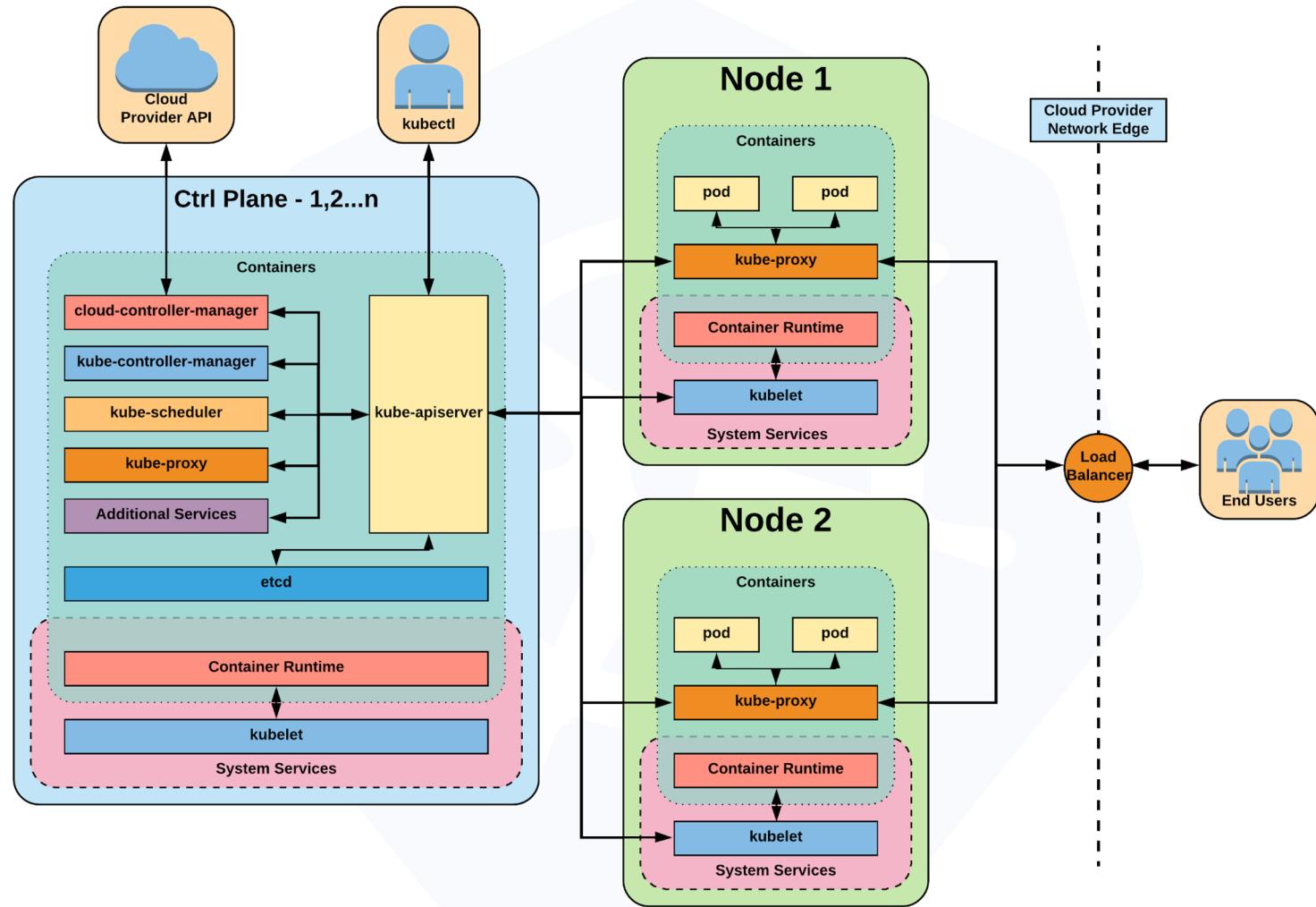


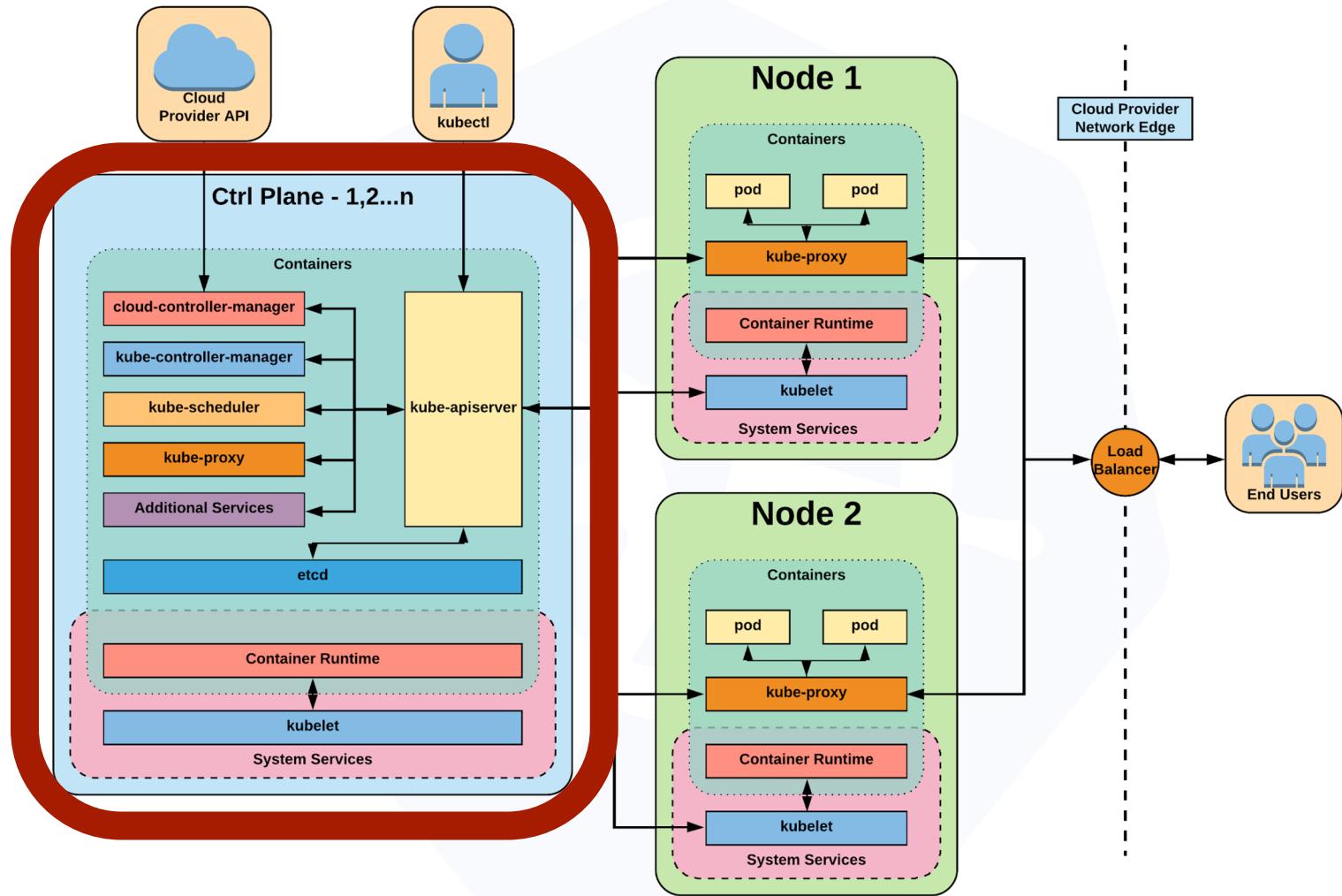
- **Unified method of accessing** the exposed workloads of Pods
- **Durable resource**
 - static cluster IP
 - static namespaced DNS name



NOT Ephemerall!

Kubernetes Cluster Architecture Overview

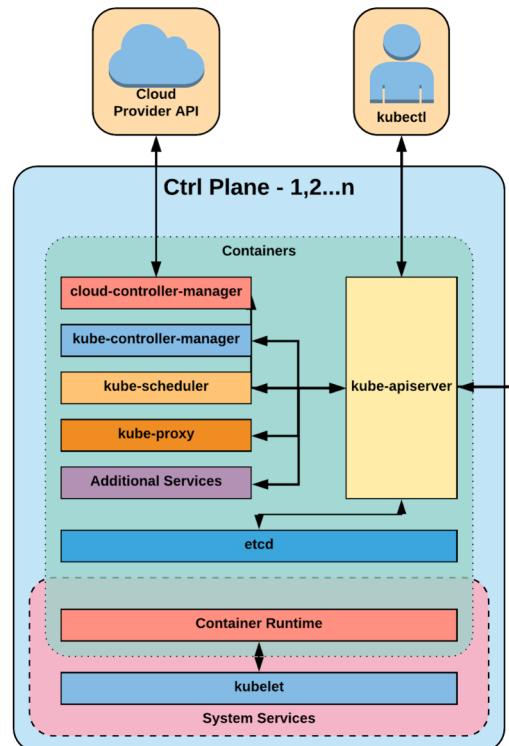


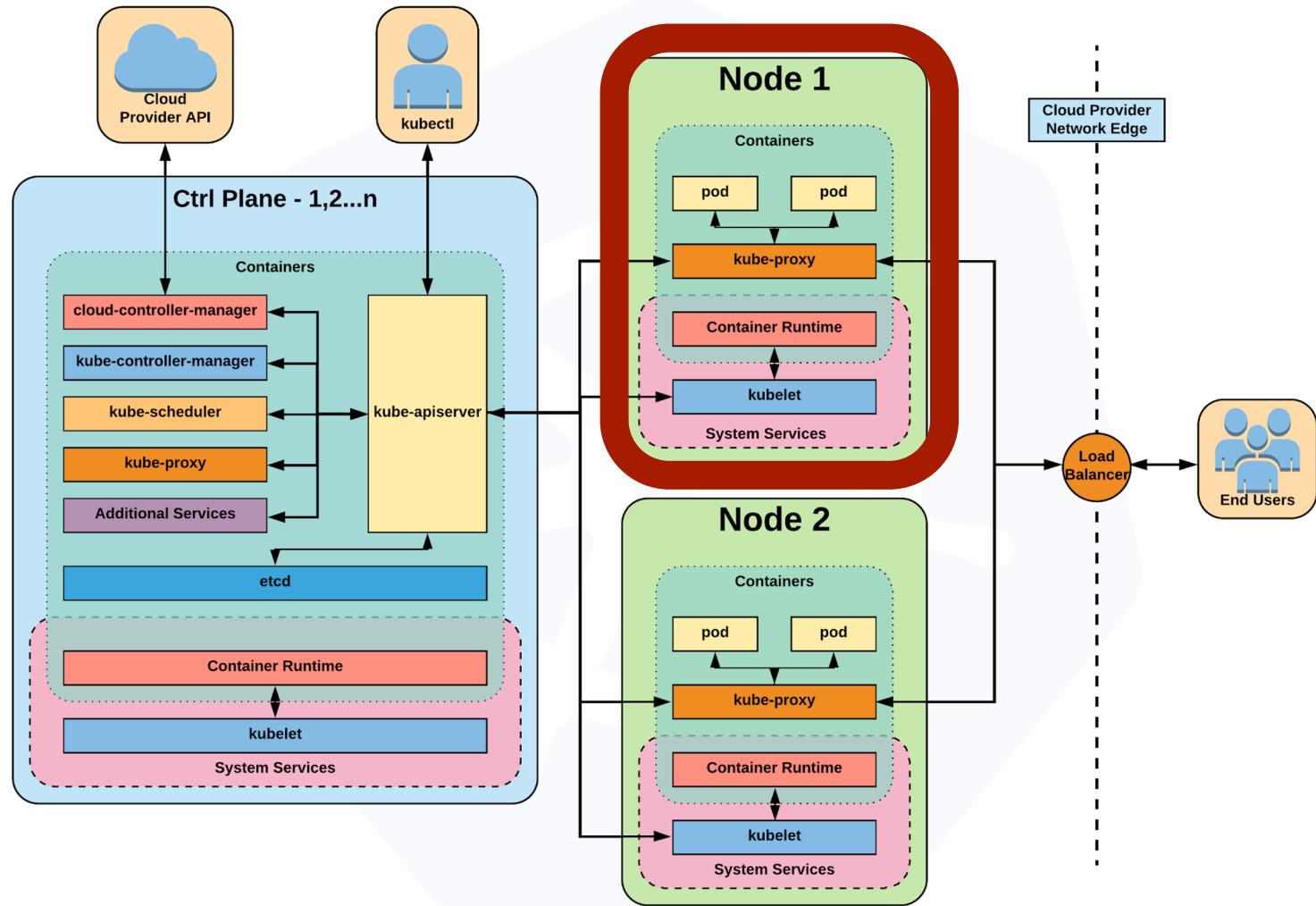


Control Plane Components



- kube-apiserver
- etcd
- kube-controller-manager
- kube-scheduler
- cloud-controller-manager

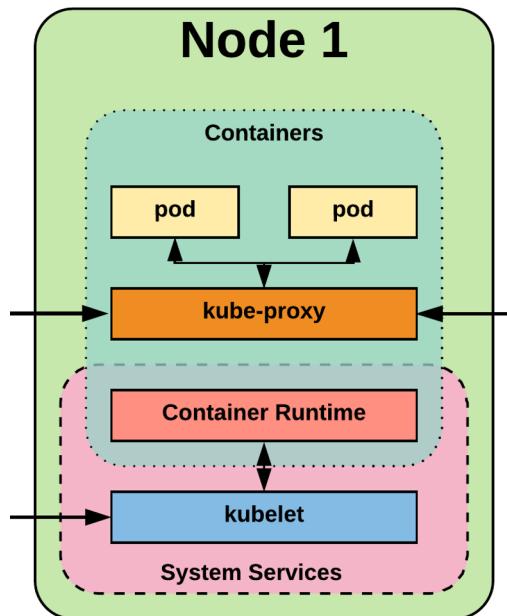


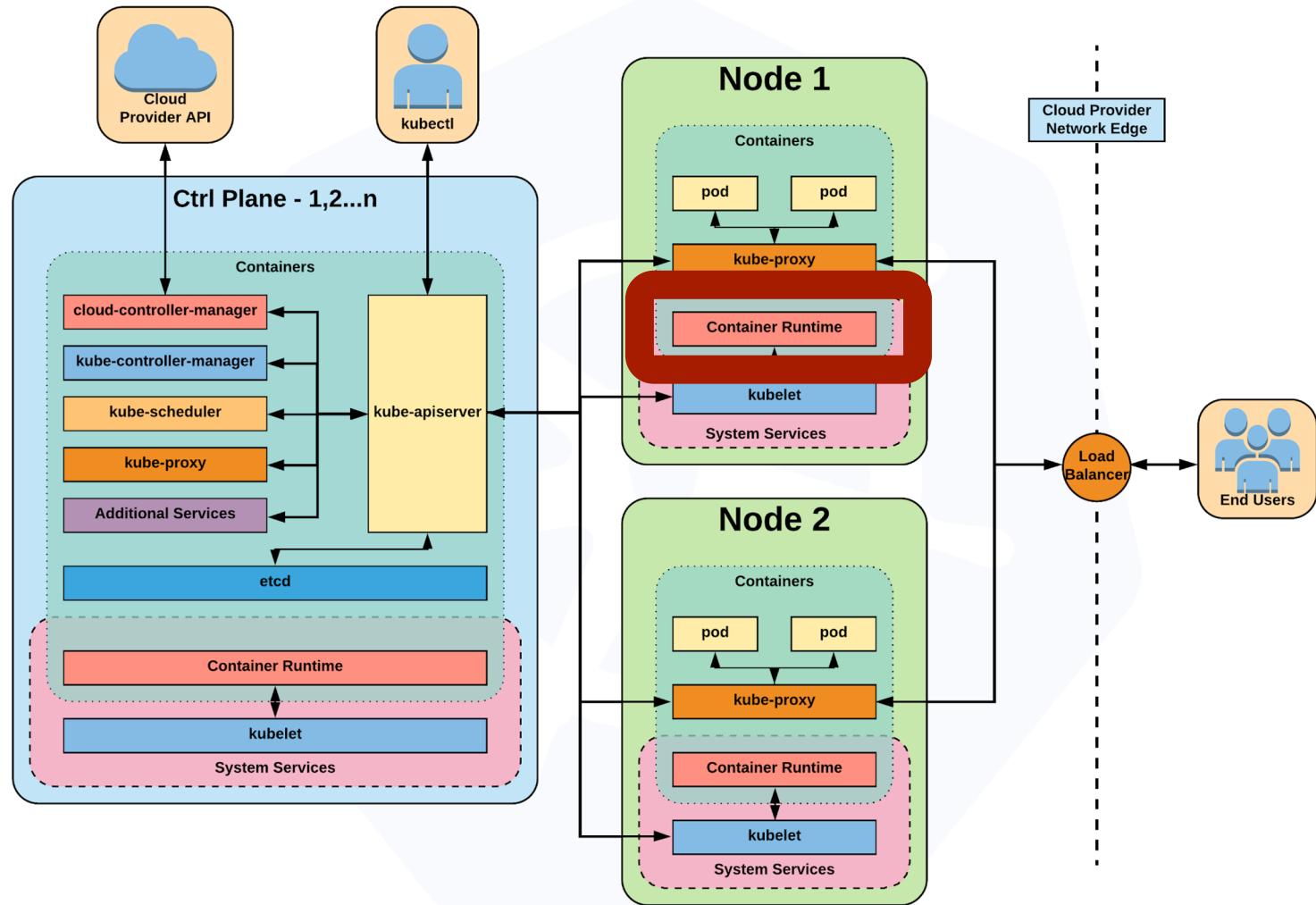


Node Components



- kubelet
- kube-proxy
- Container Runtime Engine







Container Runtime Engine

- A container runtime is a CRI (Container Runtime Interface) compatible application that executes and manages containers.
 - Containerd (docker)
 - Cri-o
 - Rkt
 - Kata (formerly clear and hyper)
 - Virtlet (VM CRI compatible runtime)

Core Resources

- Namespaces
- Pods
- Labels
- Deployments

Namespaces



Namespaces are a logical cluster or environment, and are the primary method of partitioning a cluster or scoping access

```
apiVersion: v1
kind: Namespace
metadata:
  name: prod
  labels:
    app: MyBigWebApp
```

```
$ kubectl get ns --show-labels
NAME      STATUS   AGE     LABELS
default   Active   11h    <none>
kube-public   Active   11h    <none>
kube-system   Active   11h    <none>
prod       Active   6s     app=MyBigWebApp
```



Pod Example

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-example
spec:
  containers:
  - name: nginx
    image: docker.io/nginx:stable-alpine
    ports:
    - containerPort: 80
```

Key Pod Container Attributes



- `name` - The name of the container
- `image` - The container image
- `ports` - array of ports to expose -- can be granted a friendly name and protocol may be specified
- `env` - array of environment variables
- `command` - Entrypoint array (equiv to Docker `ENTRYPOINT`)
- `args` - Arguments to pass to the command (equiv to Docker `CMD`)

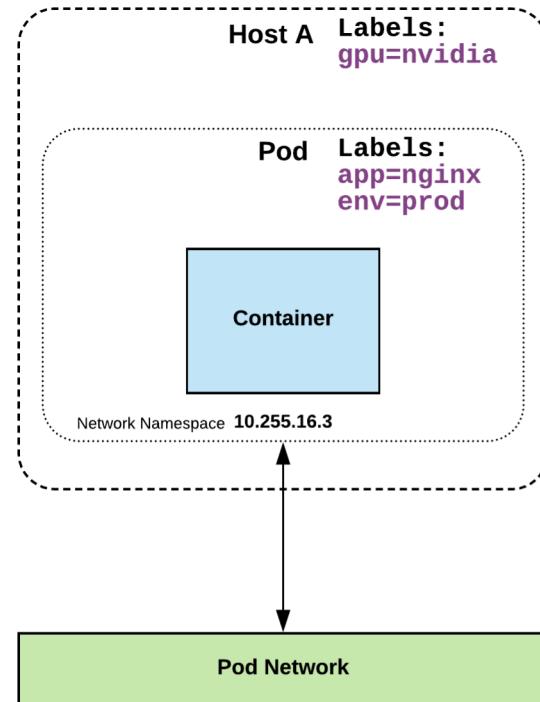
Container

```
name: nginx
image: nginx:stable-alpine
ports:
- containerPort: 80
  name: http
  protocol: TCP
env:
- name: MYVAR
  value: isAwesome
command: ["/bin/sh", "-c"]
args: ["echo ${MYVAR}"]
```

Labels



- key-value pairs that are used to identify, describe and group together related sets of objects or resources
- Have a strict syntax with a slightly limited character set*



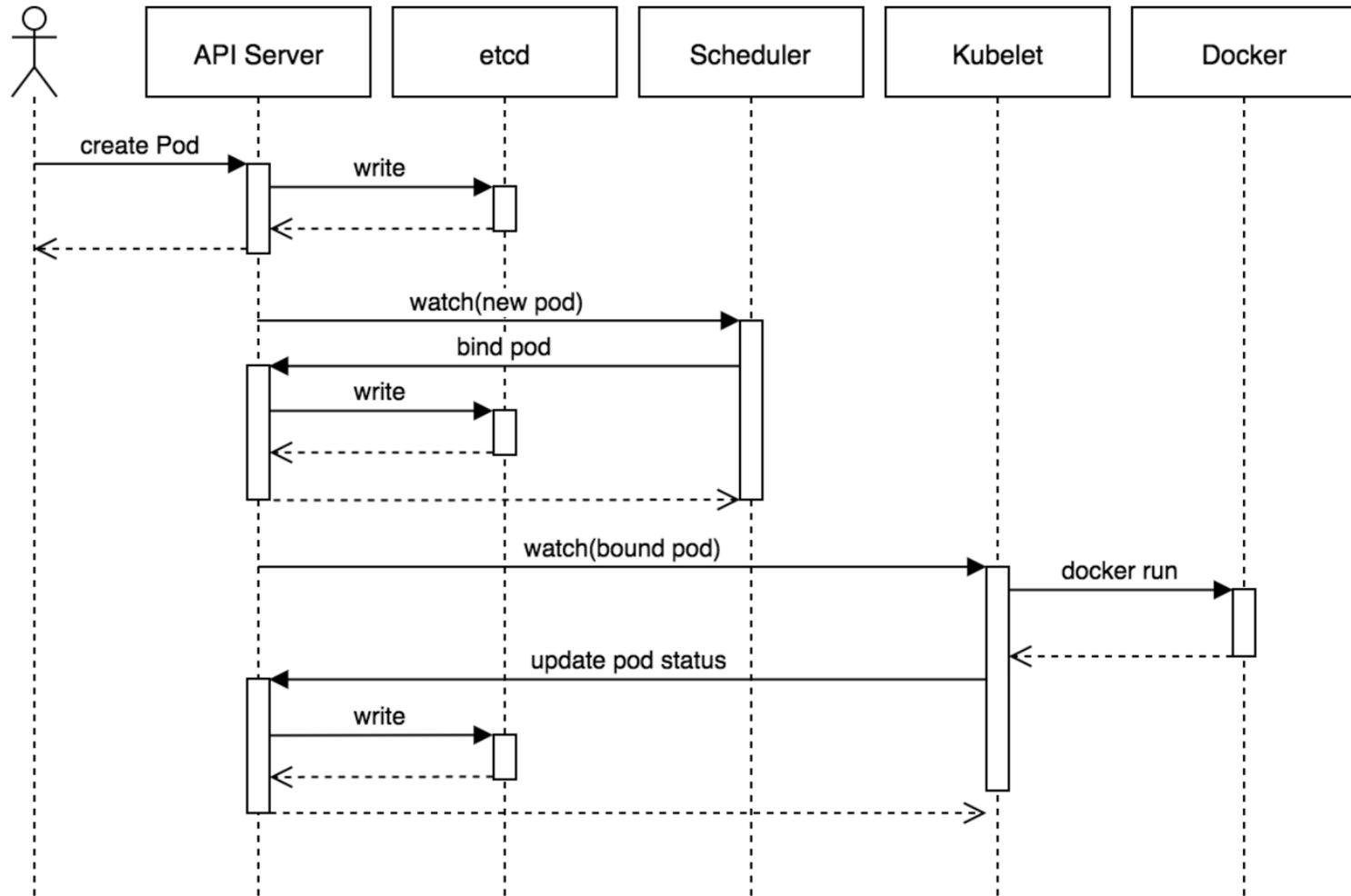
* <https://kubernetes.io/docs/concepts/overview/working-with-objects/labels/#syntax-and-character-set>



Deployment

- Way of managing Pods via **ReplicaSets**
- Provide rollback functionality and update control
- Updates are managed through the **pod-template-hash** label
- Each iteration creates a unique label that is assigned to both the **ReplicaSet** and subsequent Pods





Live demo

Play along in cloud env: <http://bit.ly/mpuck8sd> (requires github account)

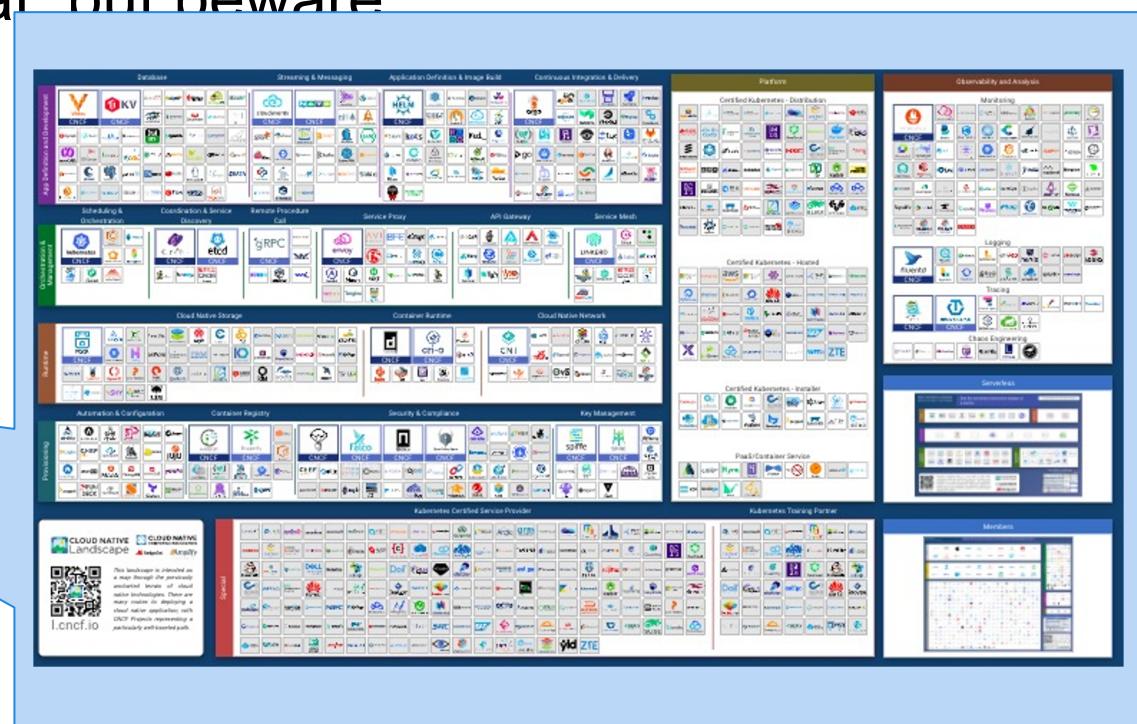
Code: <https://github.com/MxNxPx/kind-kubethanos>



Final thoughts



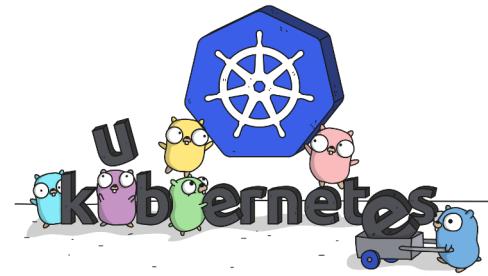
- Kubernetes is great, but beware



Links



- **Free Kubernetes Courses**
<https://www.edx.org/>
- **Interactive Kubernetes Tutorials**
<https://www.katacoda.com/courses/kubernetes>
- **Learn Kubernetes the Hard Way**
<https://github.com/kelseyhightower/kubernetes-the-hard-way>
- **Official Kubernetes Youtube Channel**
<https://www.youtube.com/c/KubernetesCommunity>
- **Official CNCF Youtube Channel**
<https://www.youtube.com/c/cloudnativefdn>
- **Track to becoming a CKA/CKAD (Certified Kubernetes Administrator/Application Developer)**
<https://www.cncf.io/certification/expert/>
- **Awesome Kubernetes**
<https://www.gitbook.com/book/ramitsurana/awesome-kubernetes/details>



Any Questions?

