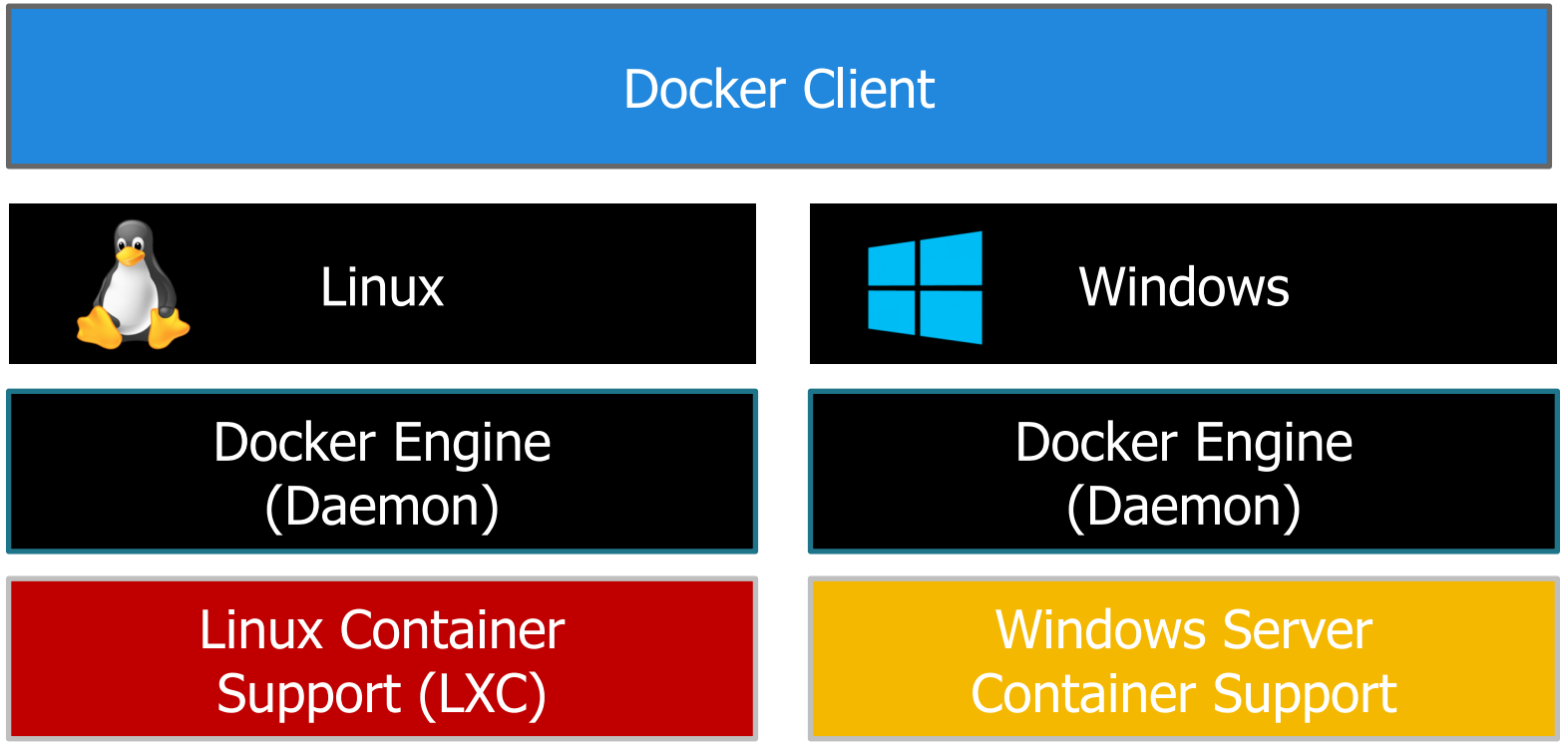
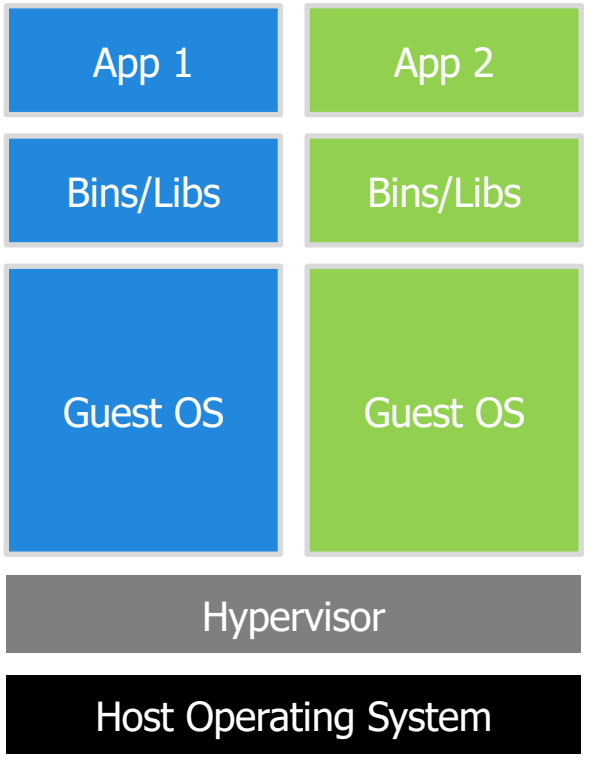
Getting Started with Docker

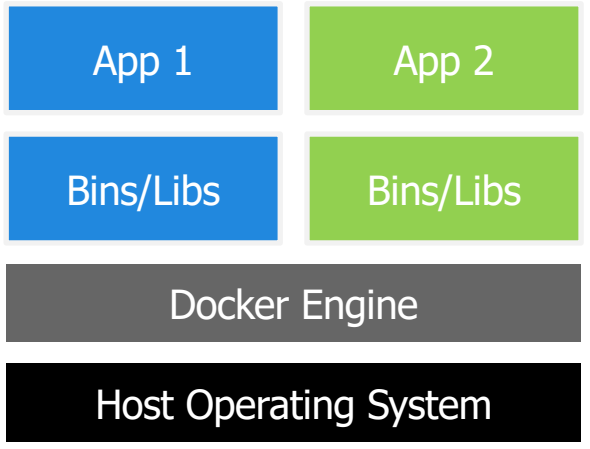
Where Does Docker Run?



Docker Containers Versus Virtual Machines



Virtual Machines

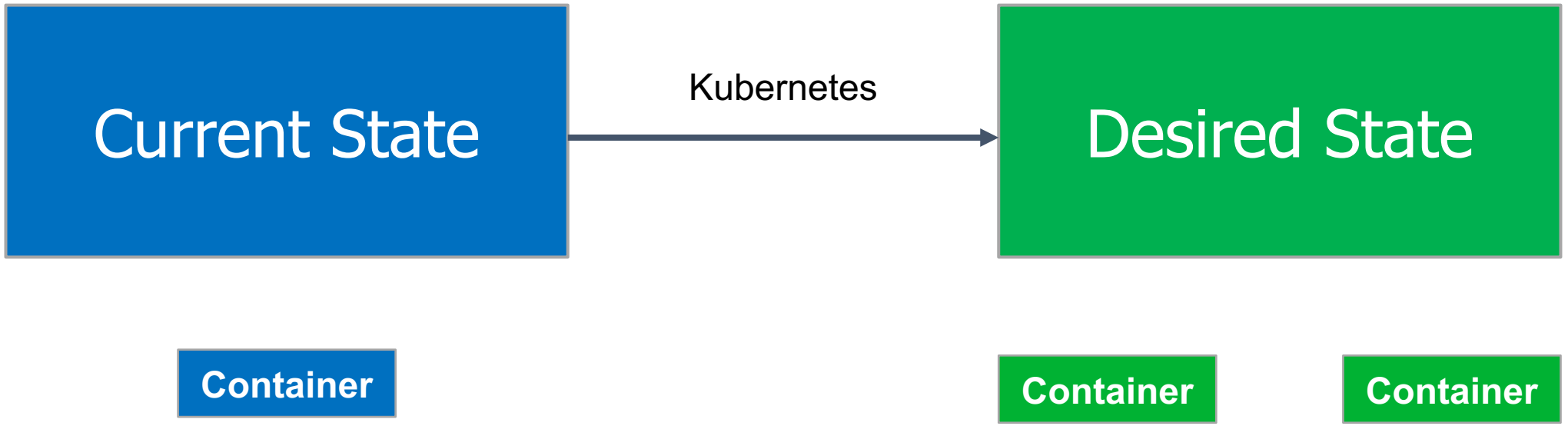


Docker Containers

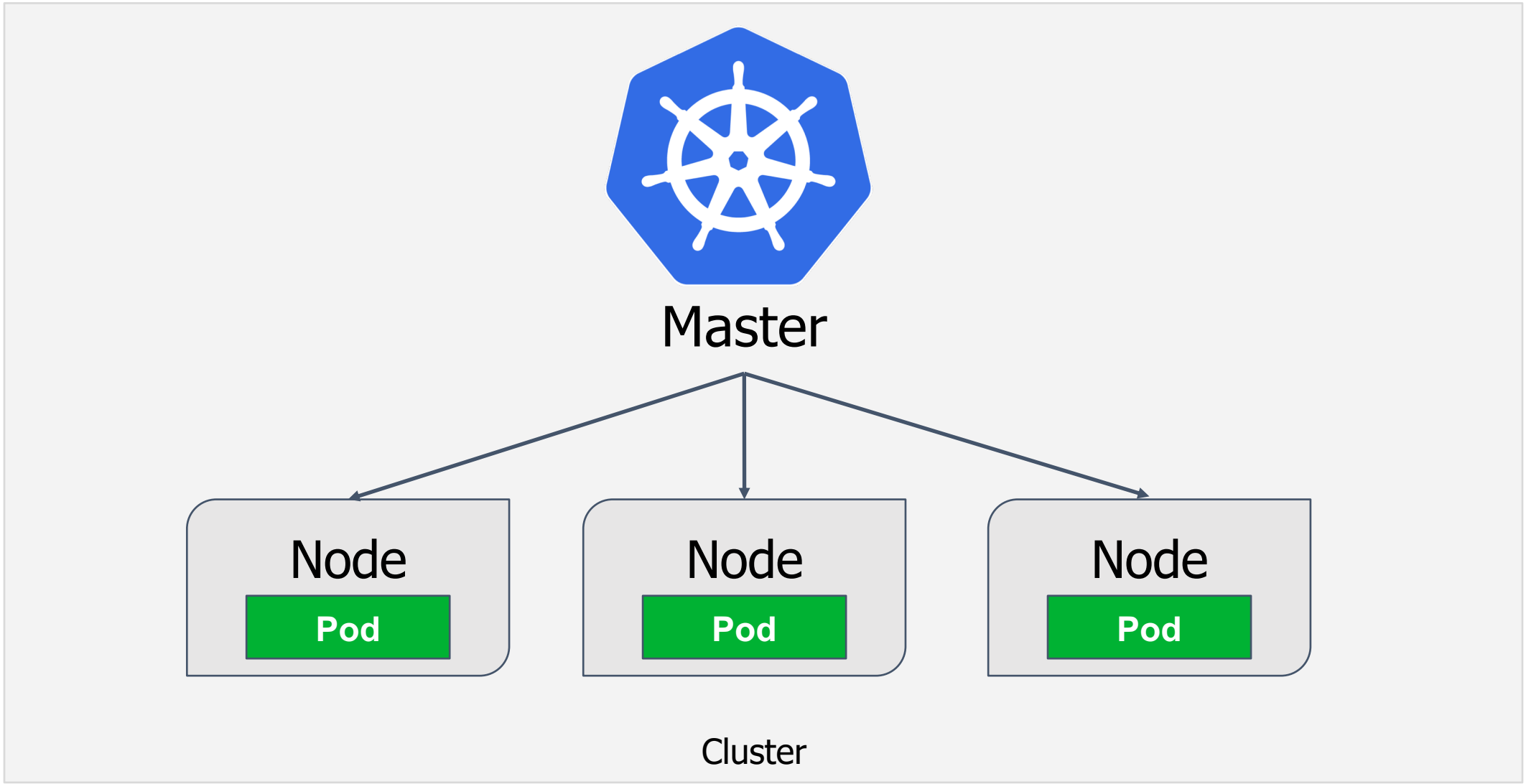
Kubernetes Overview

- Container and cluster management
- Open source project
- Used internally by Google for 15+ years and donated to the Cloud Native Computing Foundation
- Supported by all major cloud platforms
- Provides a "declarative" way to define a cluster's state

Kubernetes Moves You to a Desired State



The Big Picture



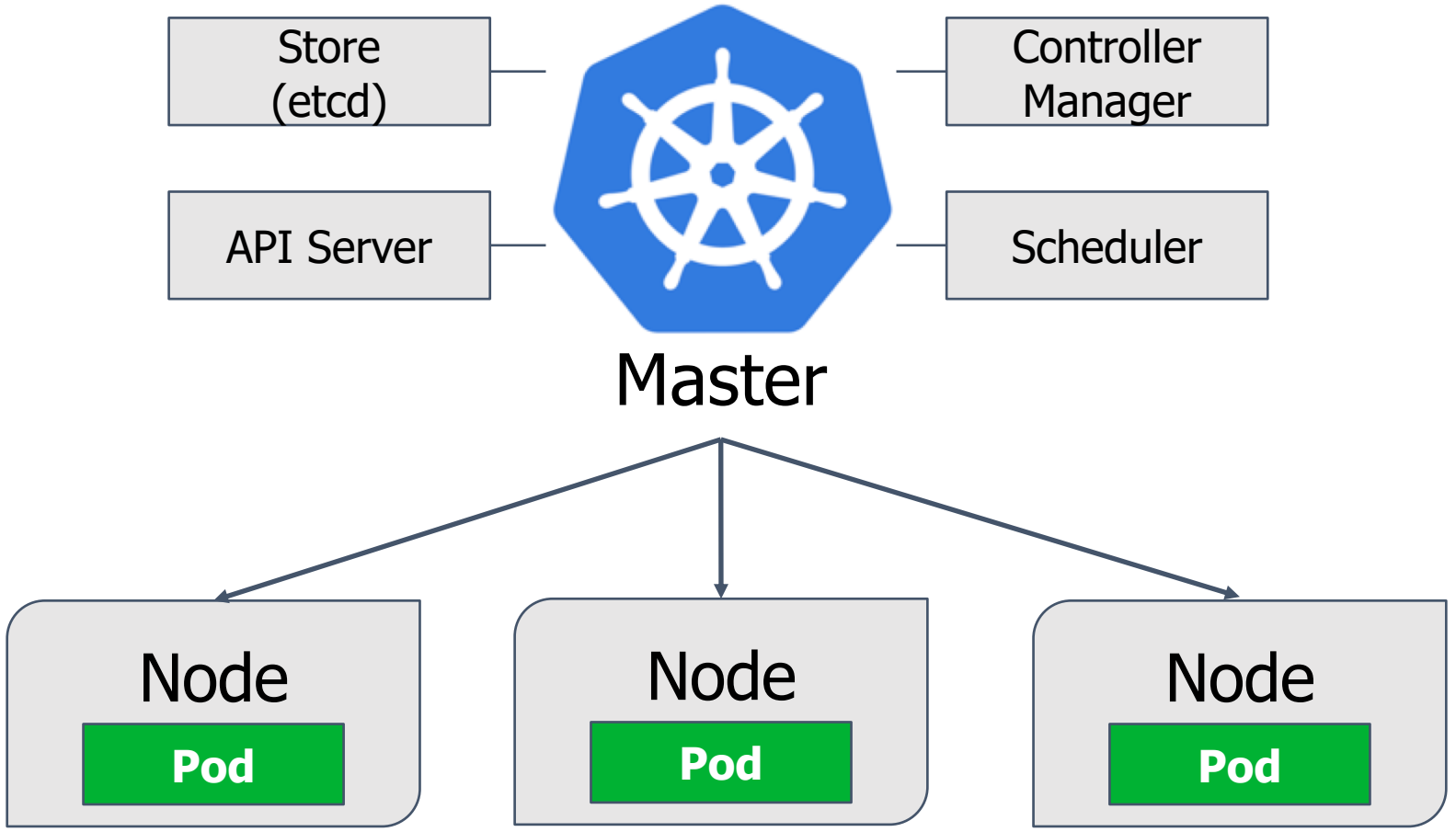
The Master Node

The Master

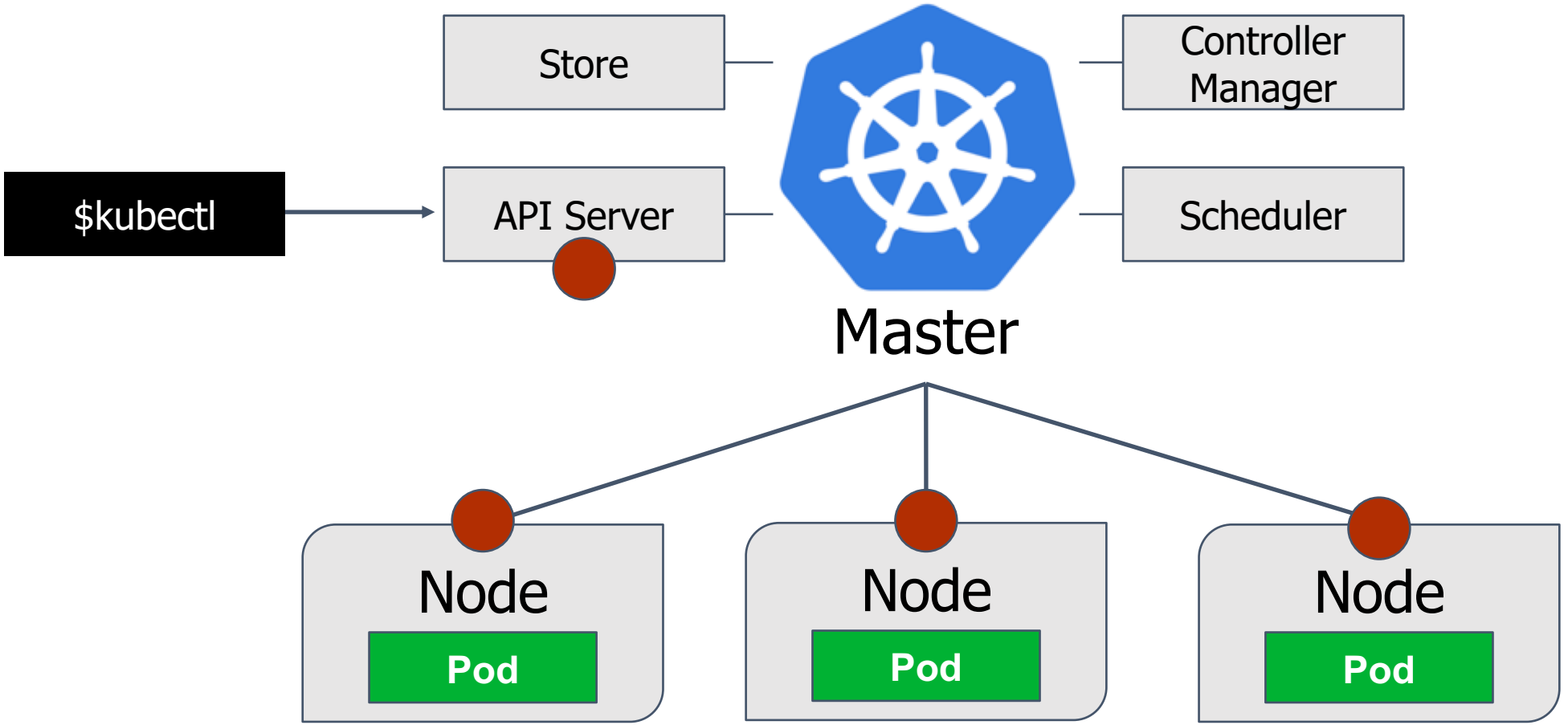
- Gets us to a desired end state through declarative manifest files
- When you interact with Kubernetes you're interacting with the master
- Composed of a collection of processes:
  - API server
  - Store (etcd)
  - Controller manager
  - Scheduler

Master

The Master

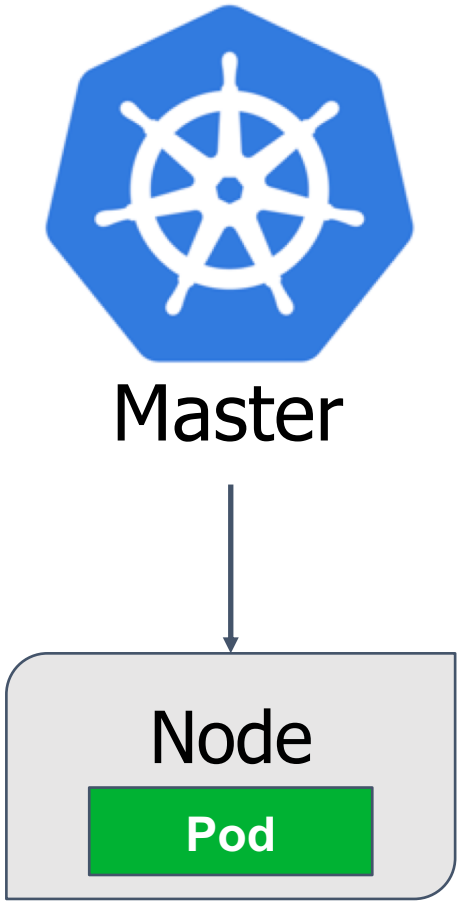

Store
(etcd)
Controller
Manager
API Server
Scheduler
Master
Node
Pod
Node
Pod
Node
Pod

Communicating with with kubectl

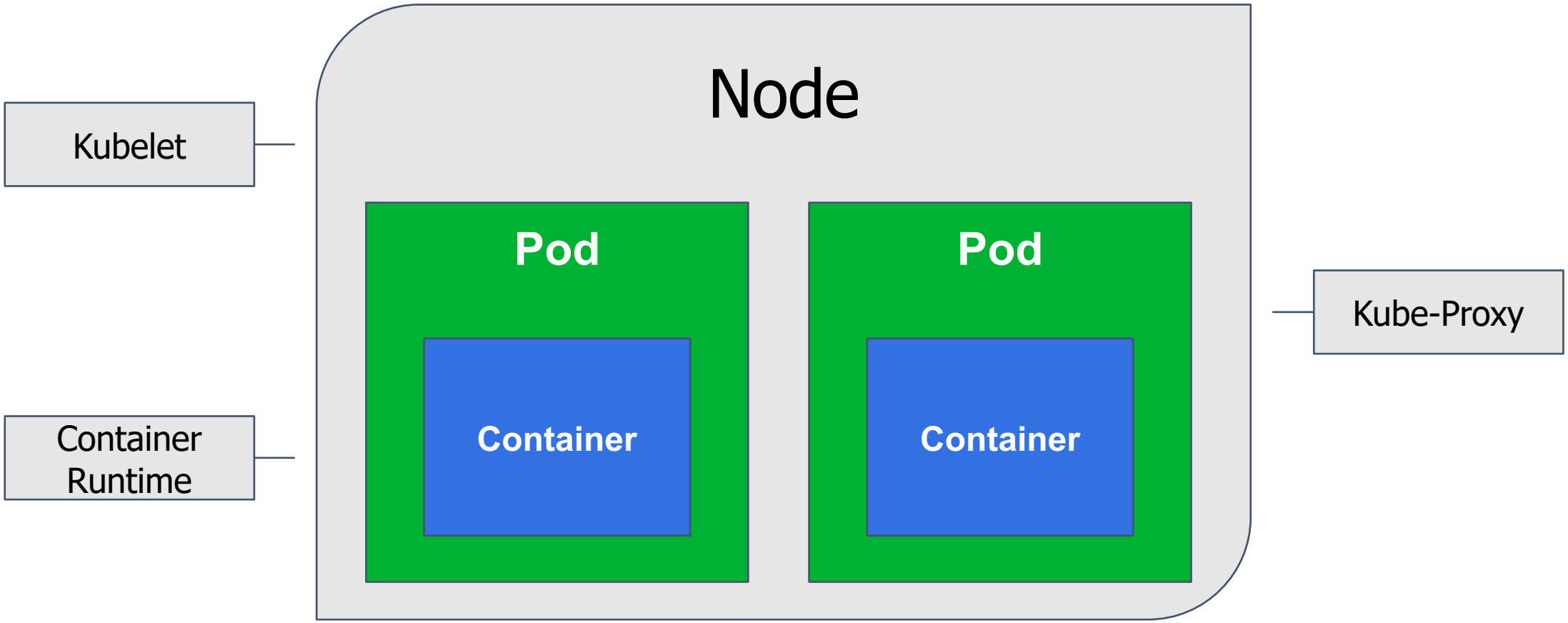


Worker Nodes and Pods

Nodes and Pods

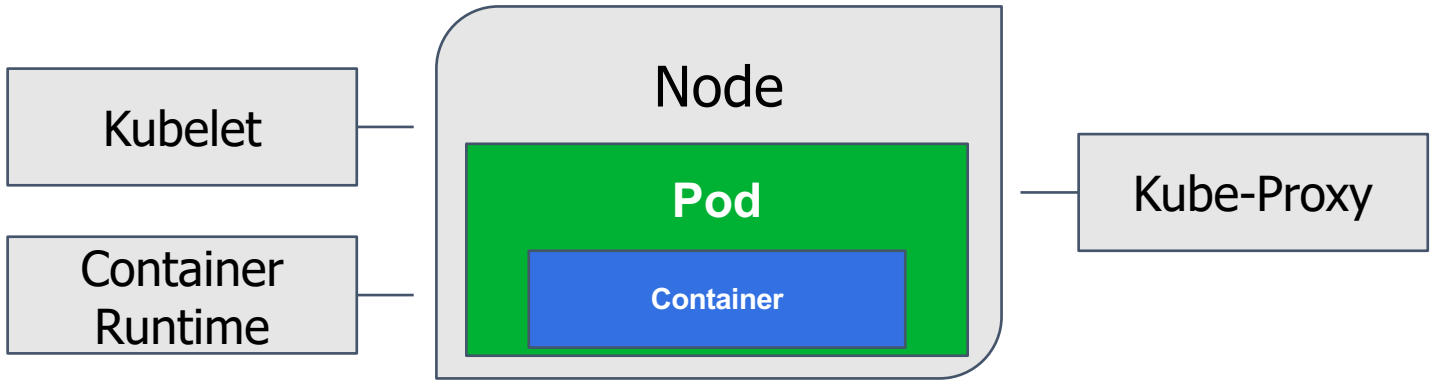


Structure of a Node



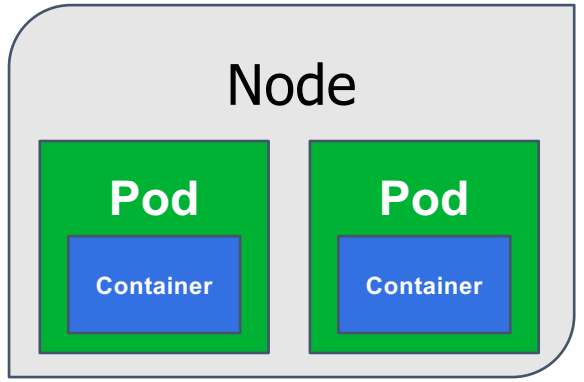
Node

- A single VM or physical server in the cluster
- Contains one or more pods
- Composed of:
  - Kubelet
  - Container runtime
  - Kube-proxy

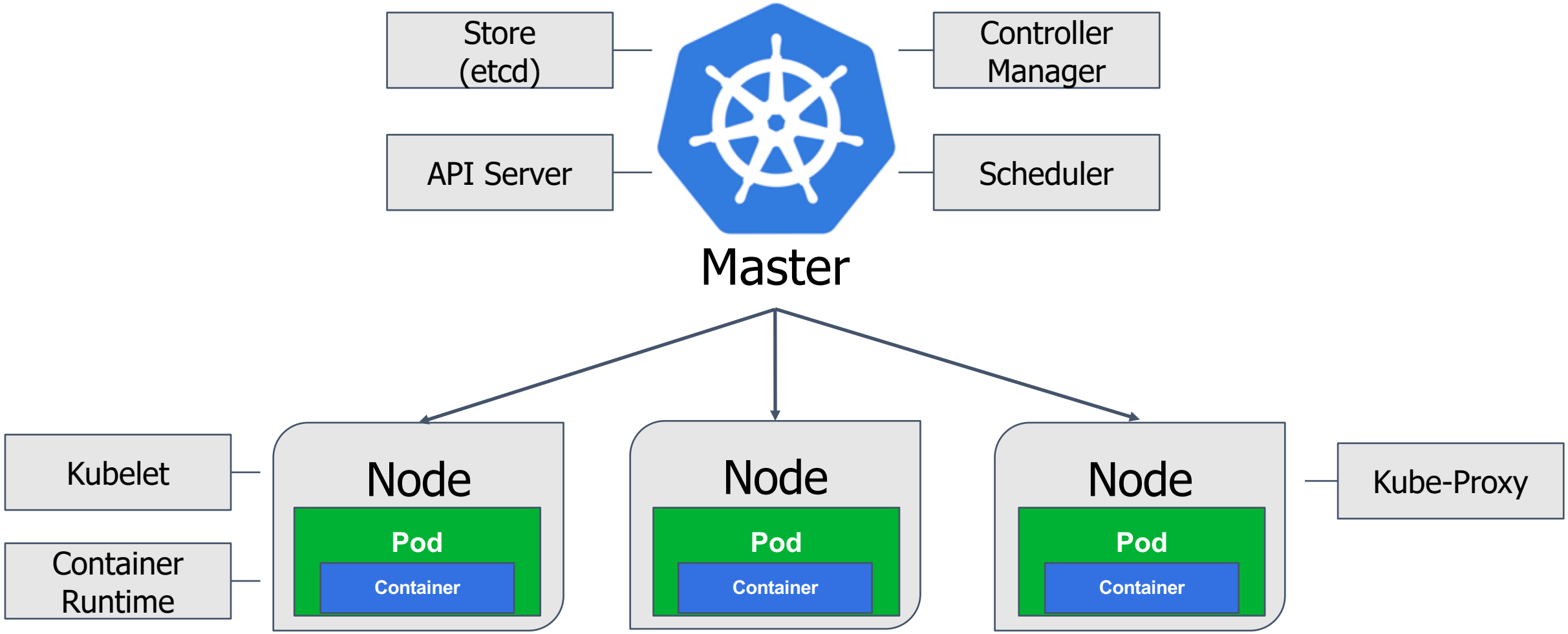


Pod

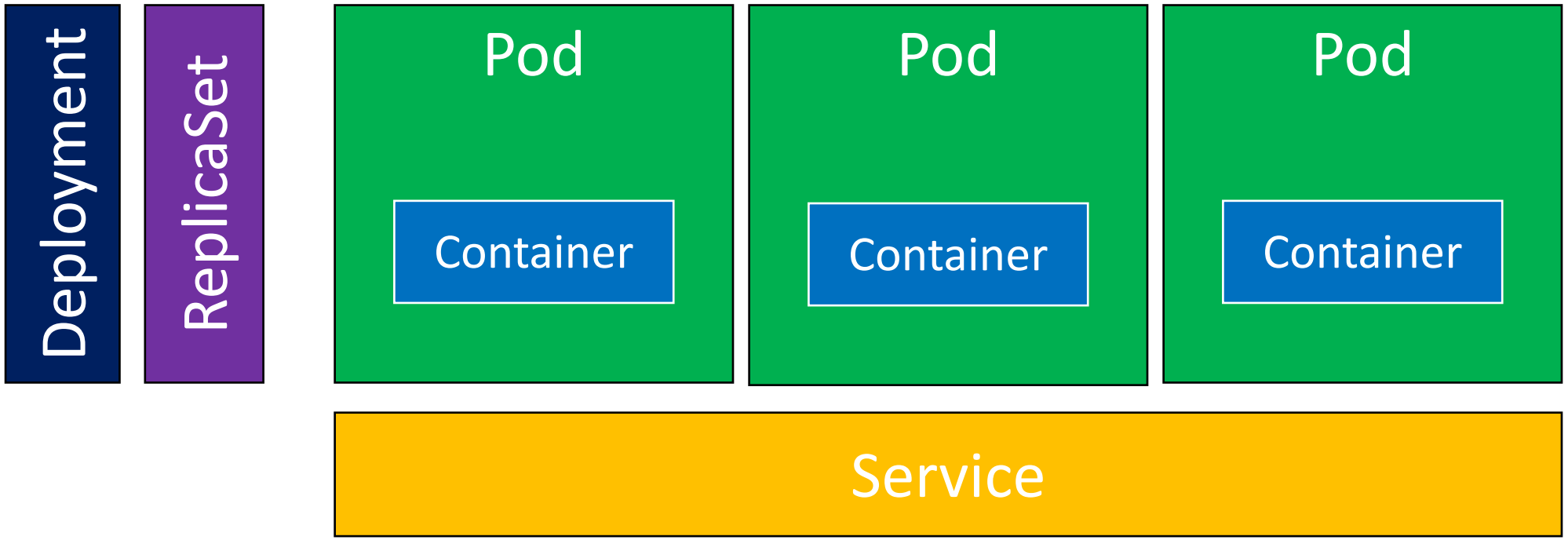
- Environment for containers
- Smallest object of the Kubernetes object model
- Pods live and die (but never come back to life)



Putting it All Together



Key Kubernetes Resources



Deployment

```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:alpine
       ports:
       - containerPort: 80
```

Service

```yaml
kind: Service
apiVersion: v1
metadata:
 name: nginx-service
spec:
 selector:
   app: nginx-app
 ports:
 - protocol: TCP
   port: 80
   targetPort: 9376
```

Basic Commands

Check Kubernetes version

```
kubectl version
```

Get cluster information

```
kubectl cluster-info
```

Get deployments

```
kubectl get [deployments | services | pods | more…]
```

Get details about a deployment

```
kubectl describe [deployment | service | pod]  [name]
```

Get all (in any namespace)

```
kubectl get [deployments | services | pods] --all-namespaces
```

Services and Deployments Commands

Create a simple deployment (good to create a pod quickly)

```
kubectl run nginx-server --image=nginx:alpine
```

Forward the port

```
kubectl port-forward [name-of-pod] 8080:80
```

Create a simple service

```
kubectl expose deploy nginx-server --port 8080 --type NodePort
```

Create a deployment from a file

```
kubectl apply -f file.yml
```

Aliasing kubectl (to save on typing)

Aliasing kubectl on Windows Powershell

```
Set-Alias -Name k -Value kubectl
```

Aliasing kubectl on Mac/Linux

```
alias k="kubectl"
```

K8s Summary

- Kubernetes provides master, node, and pod functionality to run containers
- Deployments define a target state and can be used for updates
- Services act as entry points for pods
- Deployments, services and more can be defined using YAML files
- The kubectl command can be used to interact with the master API server