



Kubernetes (k8s)

Application Orchestrator

Application Orchestrator

- Publica Aplicacoes (deploy)
- Responde dinamicamente a eventos
- Por exemplo:
 - Scale out/in dinamicamente de acordo com a demanda
 - Quando erros acontecem, o sistema se auto-recupera
 - Zero-downtime para atualizações e rollbacks
- Não precisa de interação humana para realizar seu trabalho
 - Uma vez configurado, funciona “sozinho”



Google and Cloud Native

- K8s foi criado pela Google
- A Google utilizava para suas próprias aplicações como Search e GMail
- Em 2014 a Google doou para a comunidade - CLOUD NATIVE Computing Foundation
- A palavra Kubernetes vem da palavra grega que significa Helmsman (Timoneiro)



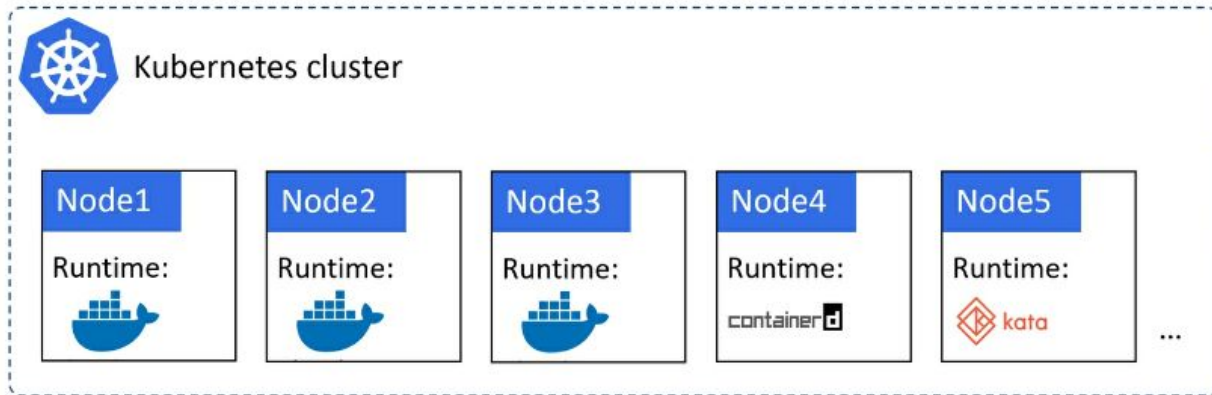
**CLOUD NATIVE
COMPUTING FOUNDATION**

<https://www.cncf.io>

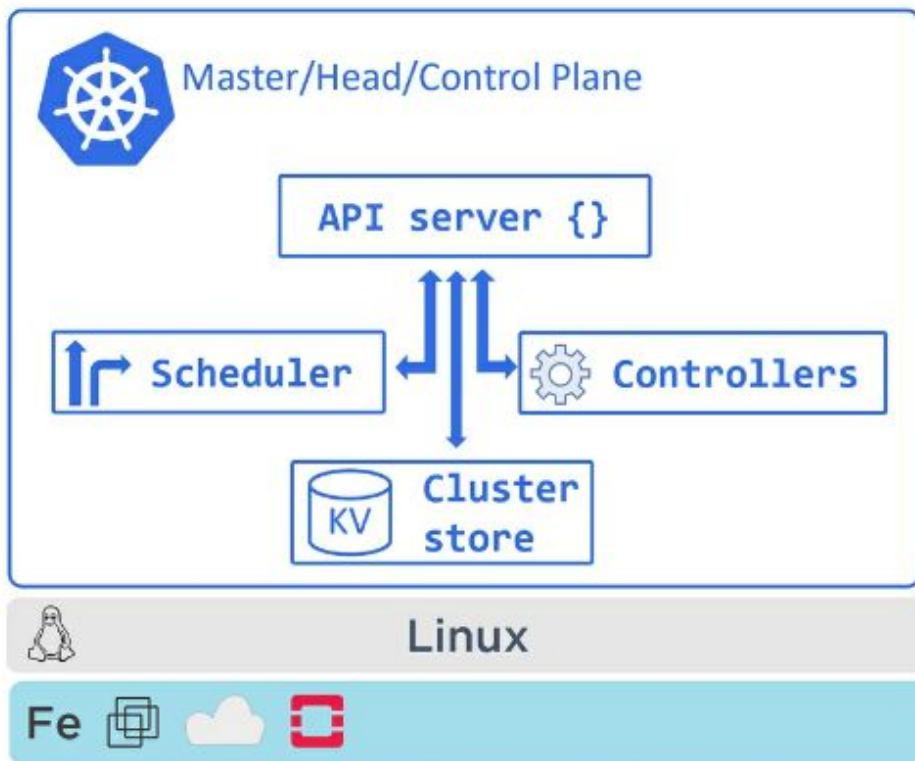


k8s and Docker

- Docker eh a tecnologia para Container
- k8s eh a tecnologia para orquestrar containers
- k8s pode utilizar outras solucoes para containers
- CRI - Container Runtime Interface



Painel de Controle



k8s as a cluster

- Cluster: N Nodos + Painel de Controle
- Painel de Controle (cerebro do cluster)
 - Expoem uma API
 - Possui um *scheduler* para designar trabalho aos nodos
 - Registra o status dos nodos num *persistent store*
 - **Os nodos master (head nodes) estão no comando do cluster (decisoões de schedule, monitoramento da performance, aplicar mudancas, responder a eventos, etc)**
- Nodos (musculos do cluster)
 - Onde as aplicações são executadas
 - Possuem uma comunicação direta com os *head nodes*
 - Constantemente “monitoram” por novos trabalhos



Publicando apps num cluster k8s

- Escrever a aplicacao como *microservices* independentes
- Empacotar cada *microservice* em seu proprio container
- Encapsular cada container em seu próprio Pod
- Publicar os Pods no cluster via controles de alto nivel tais como:
Deployments, DaemonSets, StatefulSets, CronJobs, etc
 - Deployment oferece escalabilidade e *rolling update*
 - DaemonSet executa uma instância de um serviço em cada nodo do cluster
 - StatefulSet são usados para componentes da aplicação stateful
 - CronJob são usados para executar tarefas de curta duração que precisam rodar em determinados horários



k8s way

- Modo Imperativo
- Modo Declarativo (recomendado)



API Server

- Toda a comunicação entre os componentes são feitas através da API Server
- Expõe uma API RESTful que permite que seja enviado (POST) as configurações em formato YAML
- A configuração em formato YAML é o **estado desejado** da aplicação
- Este **estado desejado** contém por exemplo:
 - Qual container imagem sera utilizada
 - Qual porta sera exposta
 - Quantas réplicas de Pod executaram



Cluster Store

- Unica parte *stateful* do Painel de Controle
- Persiste toda a configuracao e estado do cluster
- *No cluster store - no cluster*
- **etcd** - Banco de dados distribuido



Controller Manager

- Implementa todos controles que rodam em background que monitoram o cluster e respondem a eventos
- Controller of controllers
 - Node controller
 - Endpoint controller
 - ReplicaSet Controller
- Todos os controladores executam em background (watch-loop) constantemente observando o API Server por mudancas
 - “Regra do jogo: garantir que **estado atual** do cluster e o **estado desejado** sejam iguais”




Control Loop

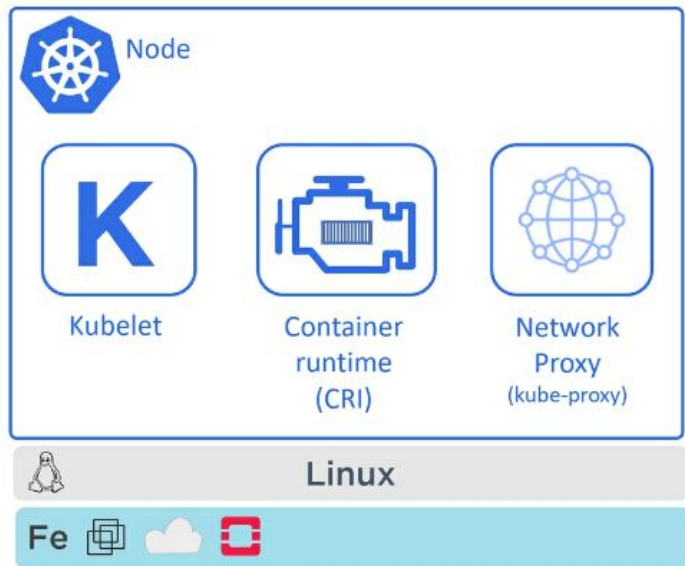
1. Obtém o estado desejado
2. Observa o estado atual
3. Determina as diferenças
4. Reconcilia as diferenças



Scheduler

- Observa o API Server por novas tarefas e atribui estas aos nodos saudáveis
 - Algoritmo complexo
 - Filtra os nodos “incapazes”
 - Ranqueia os nodos “capazes”
 - O nodo com maior score é selecionado para executar a tarefa
 - Identificar os nodos capazes
 - O nodo esta saudavel?
 - Existem regras de afinidade ou anti-afinidade?
 - A porta de rede esta disponivel para este nodo?
 - O nodo possui recursos suficientes?
 - Nodos selecionados
 - O nodo tem a imagem requerida?
 - Quando de recurso livre o nodo possui?
 - Quantas tarefas já estão executando no nodo?
 - Se o Scheduler não pode encontrar um Nodo a tarefa não pode ser agendada e fica como pendente
- 

Nodos



- Observam o API Server por novas tarefas
- Executam as tarefas
- Reportam de volta ao Painel de Controle (via API Server)

Kubelet

- Agente k8s
- Executa em cada nodo
- Observa o API Server por novas tasks
- Reportam de volta ao Painel de Controle (via API Server)
- Se não consegue executar a tarefa, então reporta ao API Server



Container

- Kubelet precisa do container para executar as atividades relacionadas, tais como pulling imagens, start | stop container
- Inicialmente o k8s tinha suporte nativo para alguns containers como Docker
- Atualmente existe o CRI - Container Runtime Interface (plugin model)
- O container mais popular atualmente: cri-containerd
 - containerd foi retirado do Docker Engine e doado pela Docker Inc ao CNCF



kube-proxy

- Este componente executa em cada nodo do cluster e é responsável pela rede
 - Garantir que o nodo possua um endereço de IP único
 - Implementa IPTABLE ou IPVS para roteamento, balanceamento do tráfego na rede do Pod



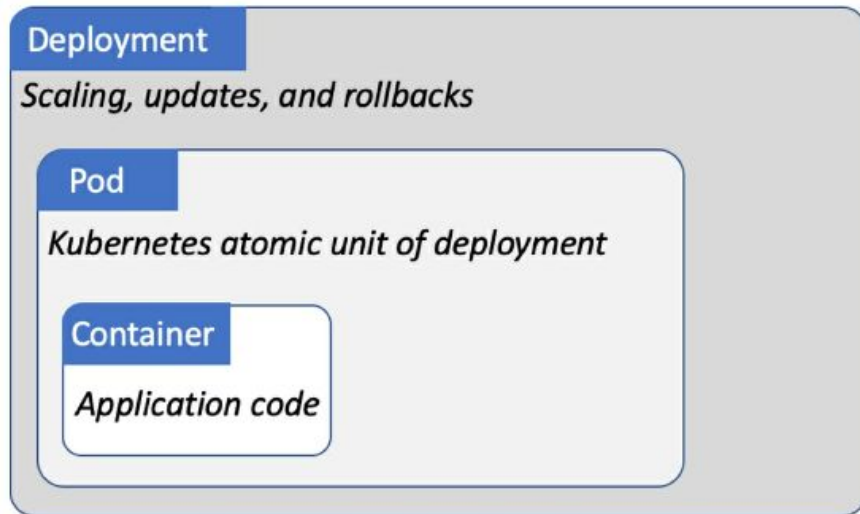
k8s DNS

- O cluster possui um servidor DNS (um ou mais nodes kube-dns)
- Possui um IP estático que fica armazenado em cada Pod, garantindo assim que cada container seja capaz de encontrar o servidor DNS
- Cada novo serviço é automaticamente registrado no servidor DNS, portanto todos os componentes do cluster podem encontrar o serviço por nome
- Baseado no projeto CoreDNS (<https://coredns.io/>)



Publicando Aplicacoes num Cluster k8s

1. Empacotar a aplicacao (or microservice) num Container
2. Publicar num Container Registry
3. Empacotar o container num Pod
4. Publicar via arquivo manifest



Declarative Mode

1. Declarar o estado desejado do microservice num arquivo de manifest (formato YAML)
2. Submeter o manifest ao API Server
3. k8s armazena o estado desejado no cluster store
4. k8s implementa o estado desejado no cluster
5. k8s implementa o *watch-loop* - garante que o estado atual seja igual ao estado desejado



Arquivo Manifest

- Imagem de container que sera utilizada
- Quantas réplicas serao executadas
- Quais portas serão expostas
- Como (estrategia) será feito as atualizações (updates)



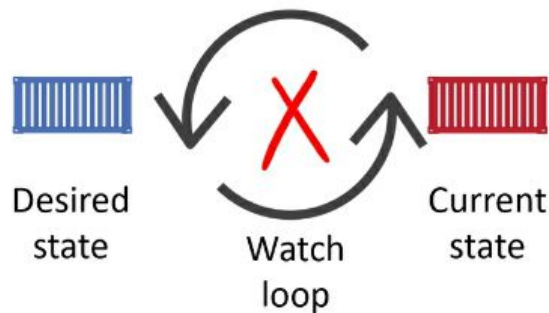
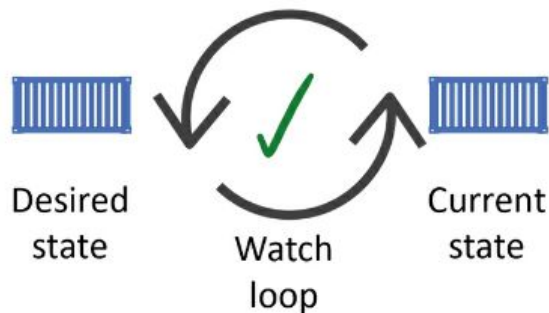
Enviar (POST) manifest para o API Server

- **kubectl**
 - Ferramenta de linha de comando
 - Envia o arquivo de manifest através de HTTPS POST usualmente na porta 443
- **k8s**
 - Inspecciona o arquivo de manifest
 - Identifica quais controller deve tratar a requisicao (Deployment Controller, etc)
 - Registra a configuracao (estado desejado) no cluster store
 - Passos adicionais
 - Pulling images
 - Iniciar containers
 - Construir redes
 - Iniciar aplicacoes
 - Reconciliation Loops



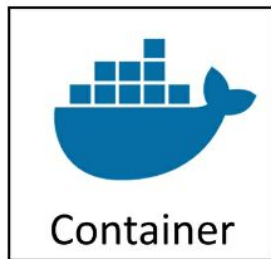
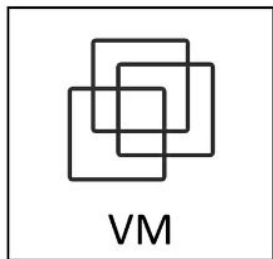
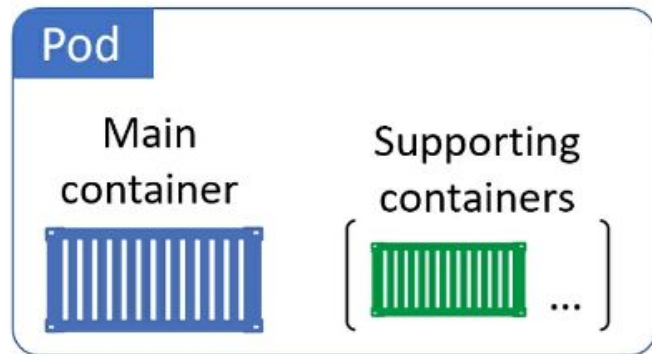
Reconciliation Loops

- Constantemente monitora o estado do cluster
- Se o estado atual varia em comparação ao estado desejado então o k8s irá realizar as tarefas necessárias para reconciliar esta diferença



Pods

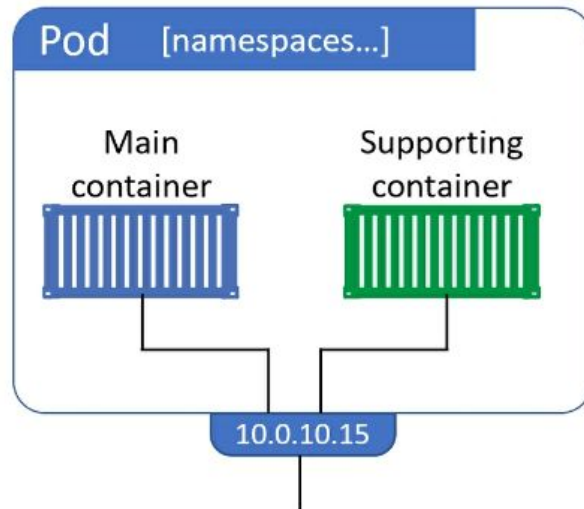
- Unidade de trabalho para o k8s
- *“Pod of whales” - bando de baleias*
- Pod esta para o K8s
 - Container esta para o Docker
 - VM esta para o VMWare
- k8s não pode executar diretamente o container, mas sim o Pod



← Atomic units of scheduling →

Pod Anatomy

- Sandbox para isolar o(s) container(s)
- Área isolada que mantém a estrutura para executar os containers
 - Host OS
 - Network stack
 - Kernel namespaces
- Todos os containers dentro de um mesmo Pod compartilham o mesmo IP

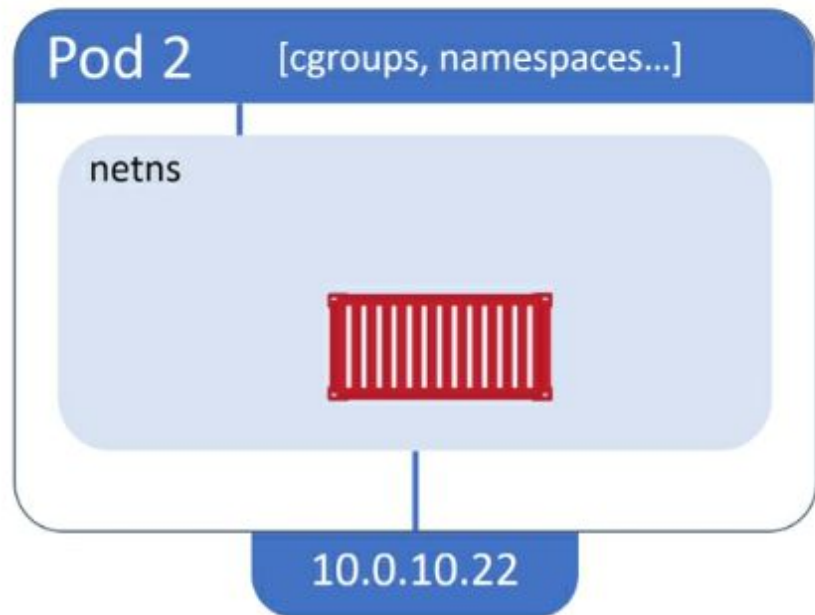
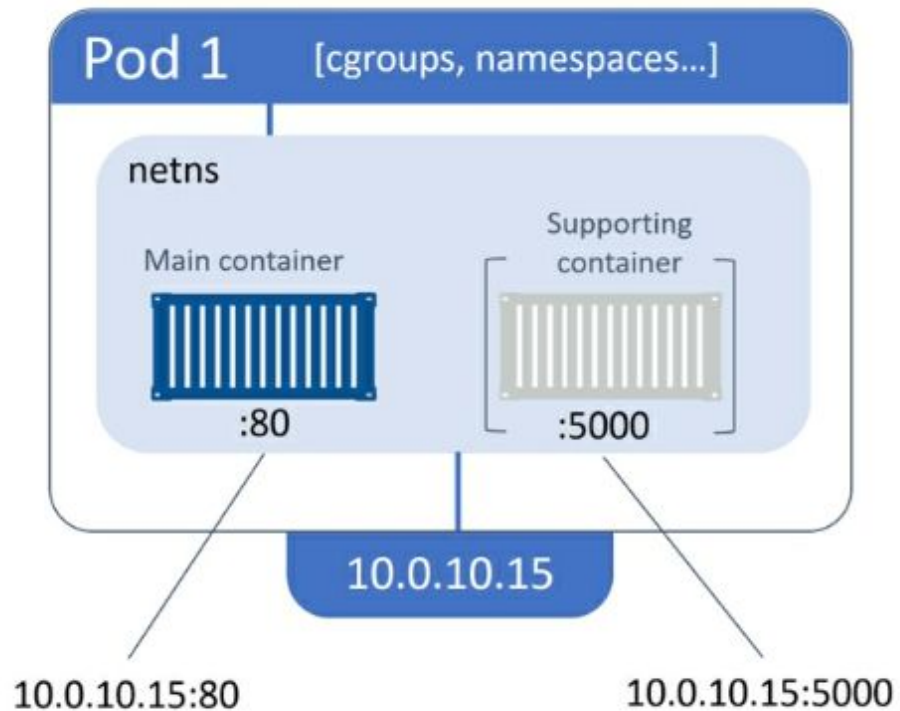


Pod Multi-Container

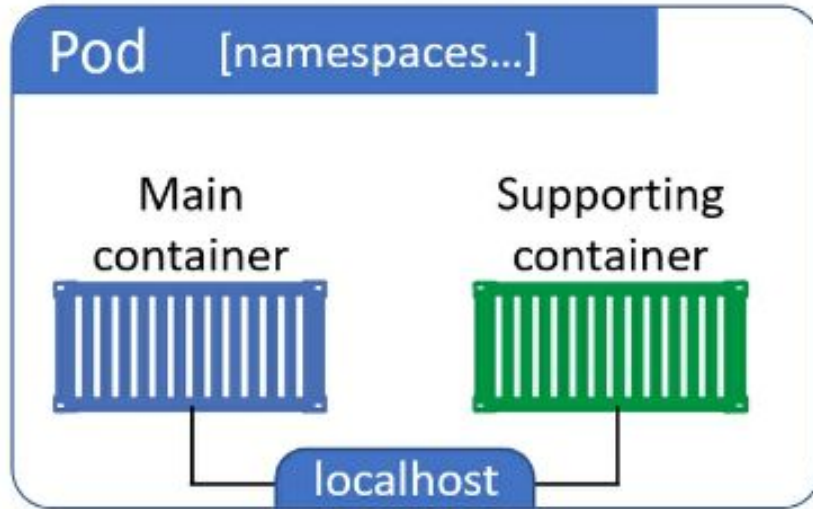
- Requisito onde containers devem executar de forma mais “acoplada”
 - Compartilham
 - Memória
 - Volume
 - IPC Sockets
 - Network namespace (IP, localhost adapter, port range, routing table, etc)
- O mais comum é o cenário de cada Container no seu Pod



Pod Multi-Container

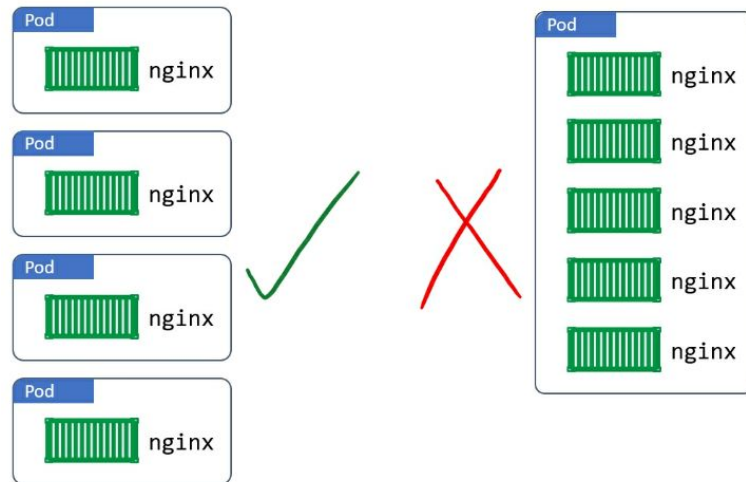


Pod Multi-Container



Pod Unidade de Escala

- Escalabilidade | Elasticidade
 - Adicionar ou remover Pods



Pod Atomic Operation

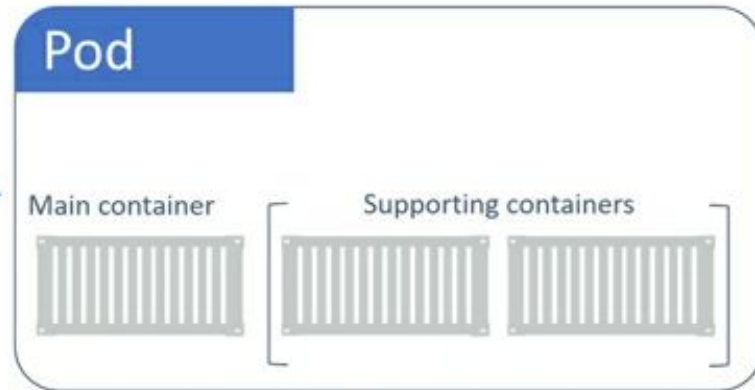
- A publicação de um novo Pod é uma operação atômica
 - O Pod é considerado pronto quando todos os seus containers estão rodando
 - Ou o Pod com todos os seus container são colocados em serviço, ou não - neste caso o deploy falhou



Pod Deploy

Define Pod in a manifest → POST manifest to API server → Schedule Pod on cluster

```
apiVersion: v1
kind: Pod
metadata:
  name: hello-pod
spec:
  containers:
  - name: hello-ctr
    image:...
    ports:
    - containerPort: 8080
```

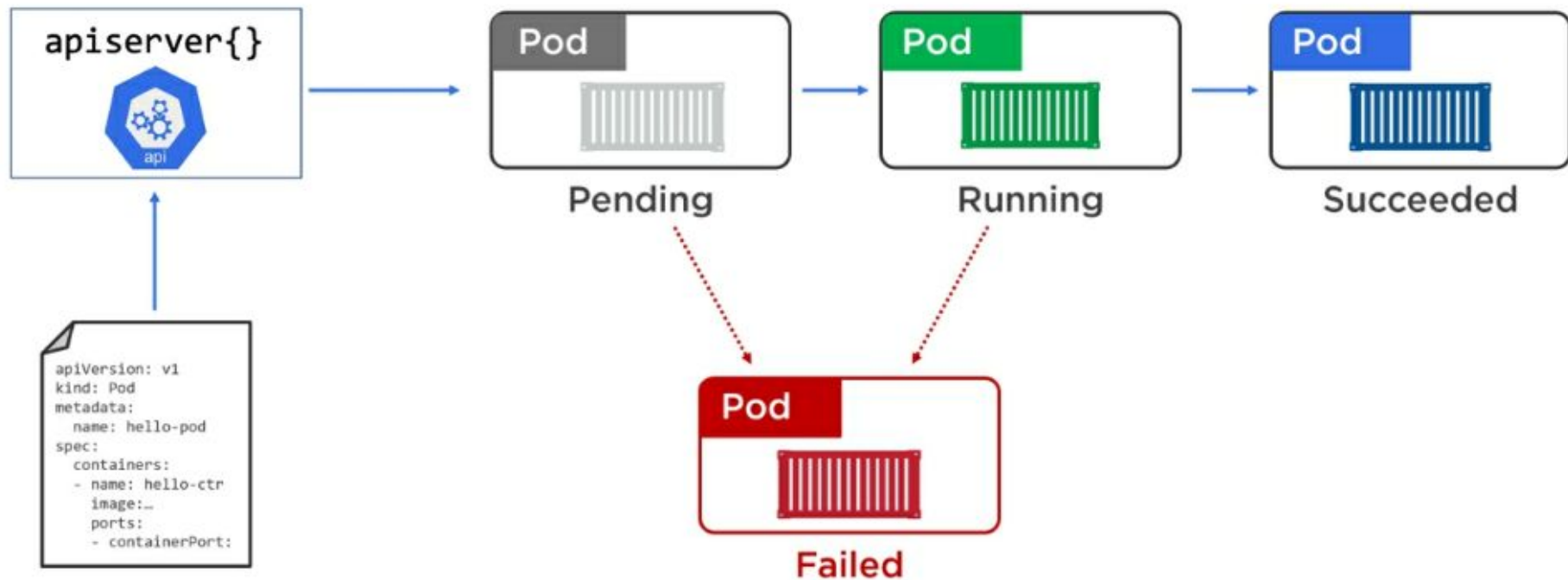


Pod Lifecycle

- Pods são mortais
 - São criados, vivem e morrem
 - Se um Pod morre inesperadamente, o k8s simplesmente cria outro e coloca o novo no lugar do antigo
 - O novo terá outro ID e outro endereço de IP
- Design da Aplicação
 - A aplicação não deve estar acoplada a um Pod (ou ao IP do Pod)
 - Projete a aplicação considerando que se o Pod falhar, outro completamente novo será criado



Pod Lifecycle



Pod cgroup

- cgroup é uma tecnologia Linux que permite impor restrições aos containers para que não utilize toda a CPU, RAM, IOPS de um nó
- Isto permite, por exemplo, que num mesmo Pod cada container possa ter sua própria definição de cgroups

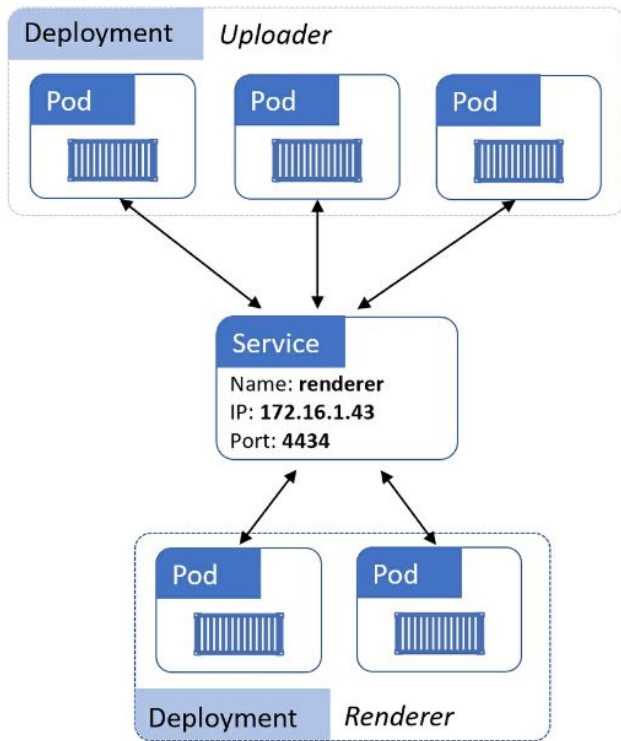


Deployment

- Objeto dentro do ecossistema do k8s que adiciona ao Pod as características
 - Escalabilidade
 - Zero downtime updates
 - Versioned rollback
- Deployments, DaemonSets e StatefulSets implementam um controlador e *watch loop* que monitoram o cluster para garantir que o estado atual seja igual ao estado desejado



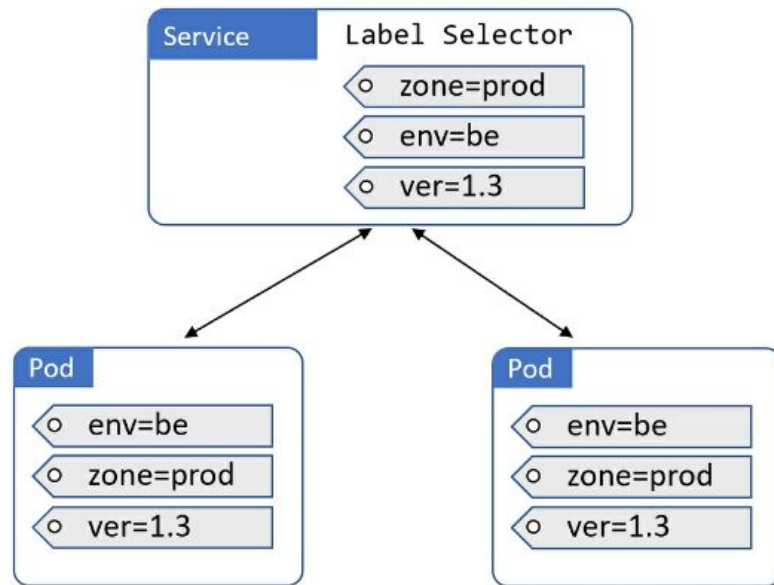
Service



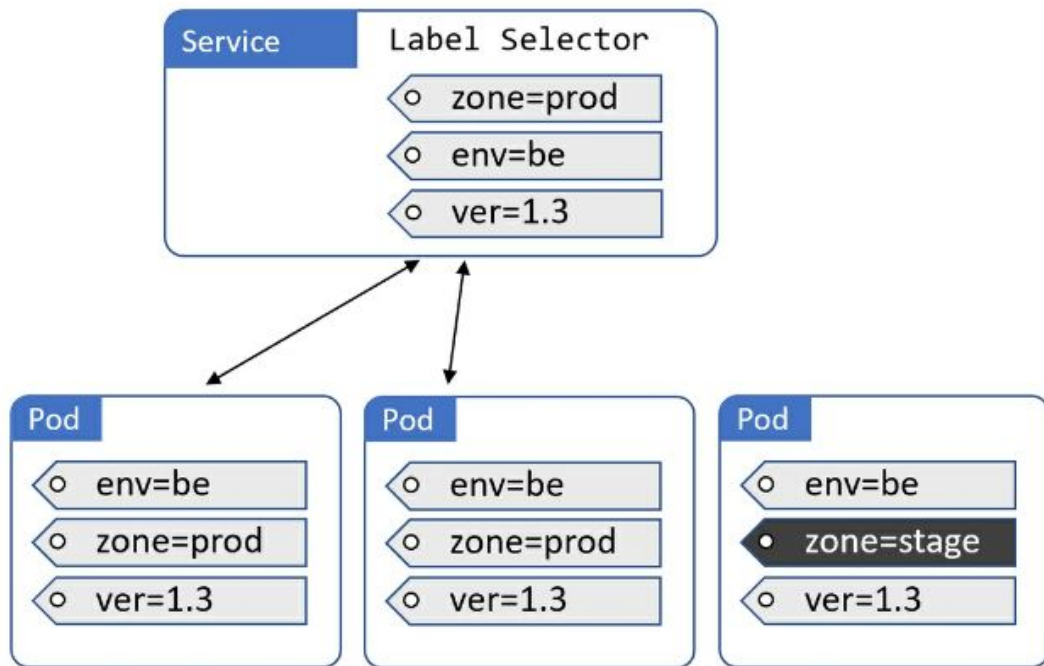
- Provê uma camada de rede confiável
 - DNS Name
 - IP Address
 - Port
- Load-Balancing
- Enviam tráfego de rede para Pods saudáveis (healthy check)

Service connect to Pod

- Service usa um mecanismo de labels e label selector para selecionar os Pods que devera estar conectado

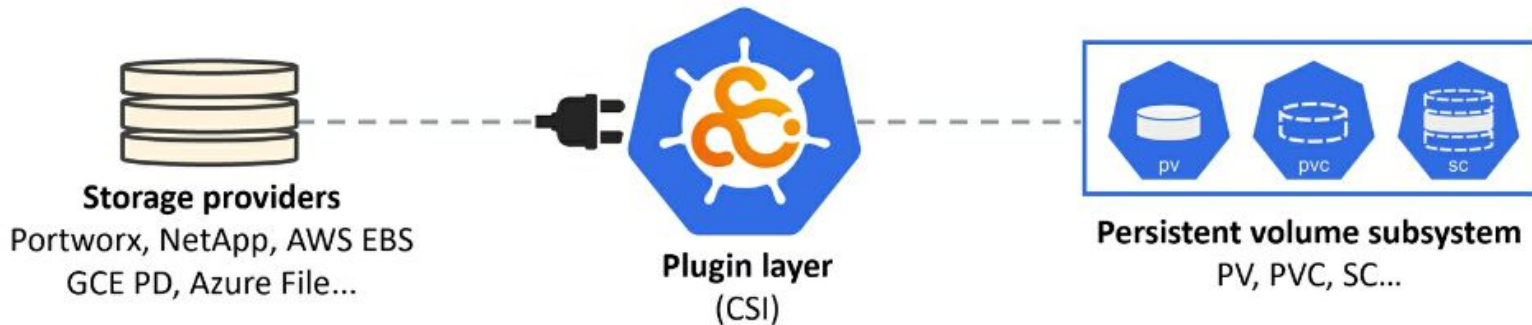


Service connect to Pod



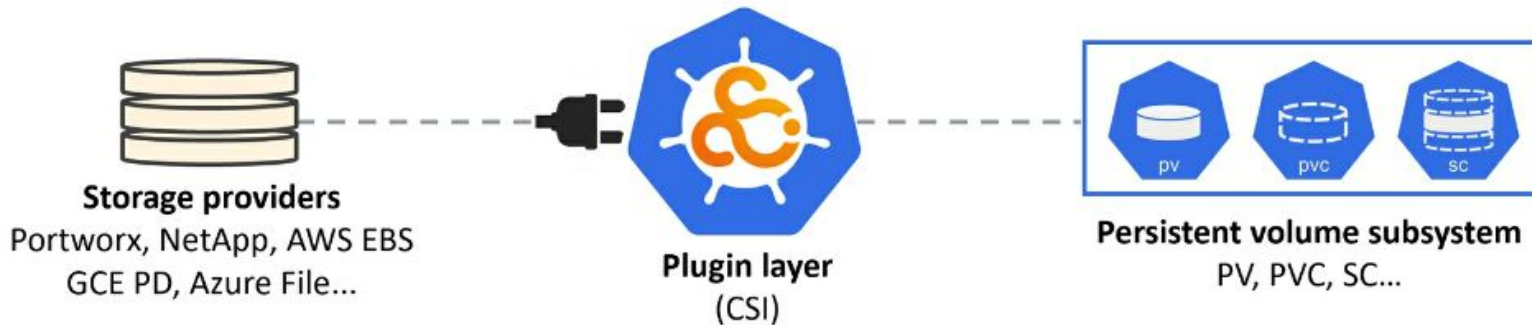
k8s Storage

- K8s suporta varios tipos de storage
 - iSCSI
 - SMB
 - NFS
- Na Cloud ou on-premises data centers



k8s Storage

- PV - Persistence Volume
 - Como mapear storage externo para o cluster
- PVC - Persistence Volume Claim
 - Sao como tickets para autorizar as aplicacoes para utilizar o volume



k8s Storage

