ESCOLA SUPERIOR DE REDES

# Orquestração de Containers

## Agendamento no Kubernetes

# Tópicos abordados

- kube-scheduler
- Labels e selectors
- Taints, tolerations e affinities
- Requests e limits
- DaemonSets
- Static Pods

# Componentes do *control plane*

Como funciona o agendamento de PODs no kubernetes?

- *etcd*
- *kube-apiserver*
- *kube-controller-manager*
- *kube-scheduler*

# *kube-scheduler*

Processo do *control plane* responsável por assinalar *Pods* a *Nodes*

Determina quais *Nodes* são válidos com base em requerimentos e recursos disponíveis, informados pelo ***etcd***

Faz o rankeamento de *Nodes* válidos

Múltiplos *schedulers* podem user utilizados: o *kube-scheduler* é a implementação de referência

https://kubernetes.io/docs/tasks/extend-kubernetes/configure-multiple-schedulers/

# O que ocorre se não tivermos um *scheduler*?

```
root@s2-master-1:~# mv /etc/kubernetes/manifests/kube-scheduler.yaml /opt/
root@s2-master-1:~#
root@s2-master-1:~# k -n kube-system get pod
NAME                                   READY   STATUS        RESTARTS       AGE
calico-kube-controllers-674fff74c8-c4jrn   1/1     Running       6 (27h ago)    10d
calico-node-4n6kt                      1/1     Running       5 (27h ago)    10d
calico-node-t88wh                      0/1     Running       6 (27h ago)    10d
coredns-5d78c9869d-qdlnl               1/1     Running       5 (27h ago)    10d
coredns-5d78c9869d-smhkt               1/1     Running       5 (27h ago)    10d
etcd-s2-master-1                       1/1     Running       5 (27h ago)    10d
kube-apiserver-s2-master-1             1/1     Running       5 (27h ago)    10d
kube-controller-manager-s2-master-1    1/1     Running       7 (27h ago)    10d
kube-proxy-mnvjk                       1/1     Running       5 (27h ago)    10d
kube-proxy-x4n7t                       1/1     Running       5 (27h ago)    10d
metrics-server-75f45b4dd4-52ffw        1/1     Terminating   5 (27h ago)    31h
metrics-server-75f45b4dd4-xlwgf        0/1     Pending       0              21m
```

# O que ocorre se não tivermos um *scheduler*?

```
root@s2-master-1:~# kubens dev
Context "kubernetes-admin@kubernetes" modified.
Active namespace is "dev".
root@s2-master-1:~#
root@s2-master-1:~# k run curl --image=curlimages/curl -- sleep 360
pod/curl created
root@s2-master-1:~# k get pod
NAME   READY   STATUS    RESTARTS   AGE
curl   0/1     Pending   0          12s
```

```
root@s2-master-1:~# k describe pod curl | grep Events
Events:                          <none>
```

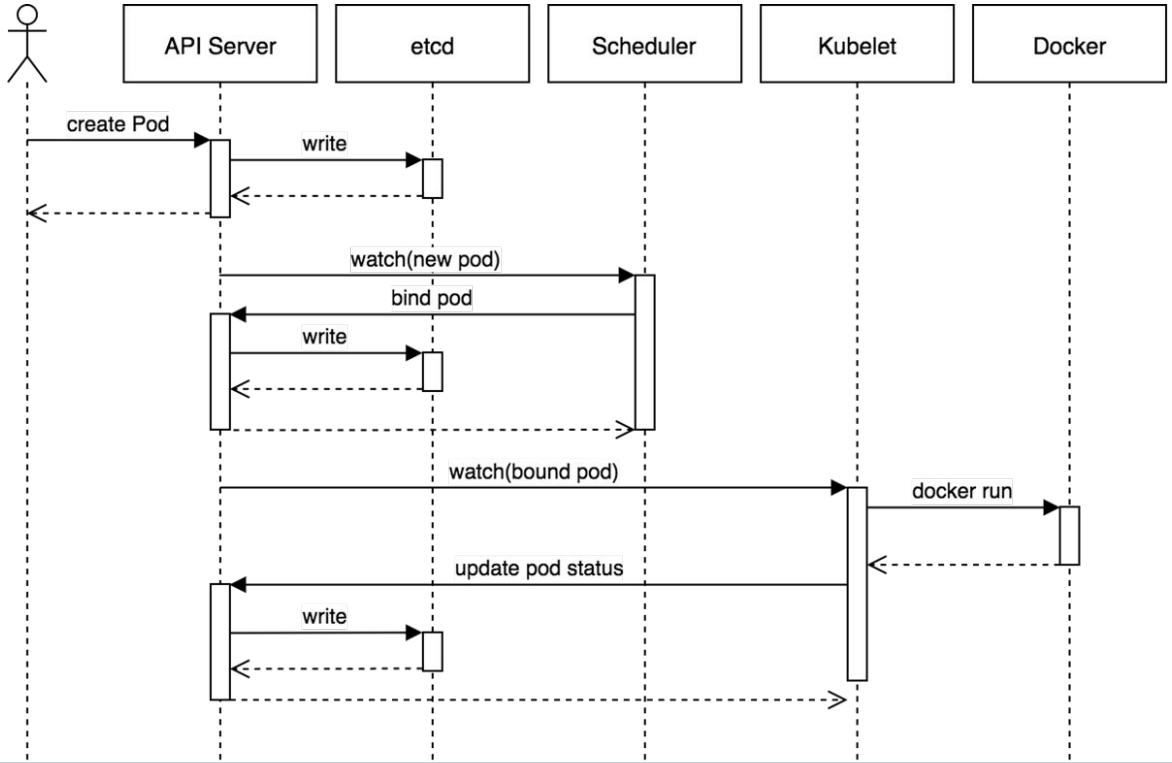
# Uma vez que ele é disponibilizado...

```
root@s2-master-1:~# mv /opt/kube-scheduler.yaml /etc/kubernetes/manifests/
root@s2-master-1:~#
root@s2-master-1:~# k -n kube-system get pod | grep scheduler
kube-scheduler-s2-master-1                 1/1     Running       0              30s
```

```
root@s2-master-1:~# k get pod curl
NAME   READY   STATUS    RESTARTS   AGE
curl   1/1     Running   0          13s
```

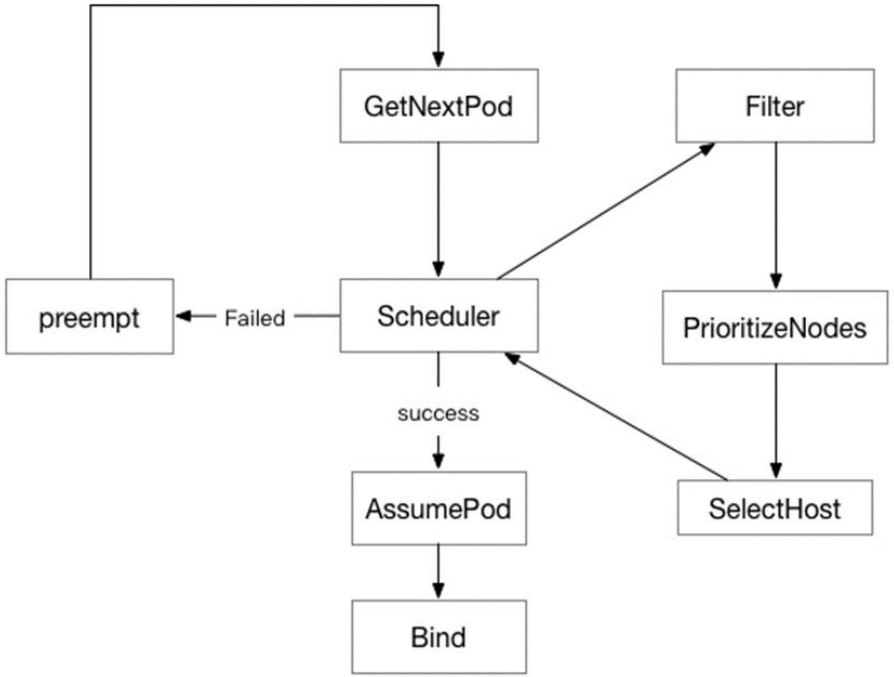
```
root@s2-master-1:~# k describe pod curl | grep Events -A 10
Events:
  Type    Reason     Age    From               Message
  ----    ------     ----   ----               -------
  Normal  Scheduled  2m35s  default-scheduler  Successfully assigned dev/curl to s2-node-1
  Normal  Pulling    2m37s  kubelet            Pulling image "curlimages/curl"
  Normal  Pulled     2m35s  kubelet            Successfully pulled image "curlimages/curl" in 1.917656333s (1.917668779s
  Normal  Created    2m35s  kubelet            Created container curl
  Normal  Started    2m35s  kubelet            Started container curl
```

# Kube-scheduler




https://www.alibabacloud.com/blog/a-brief-analysis-on-the-implementation-of-the-kubernetes-scheduler_595083

# Tarefa de agendamento



# *Labels*

*Labels* são pares chave-valor associados a objetos (p.ex. *Pods*)

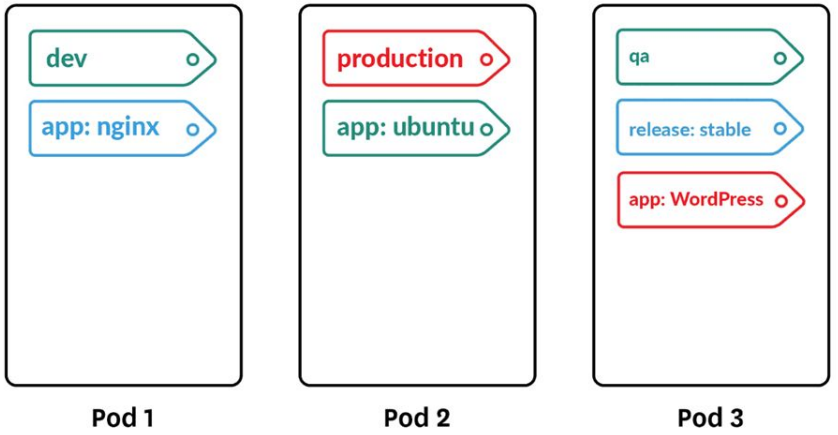
Utilizados para identificar atributos relevantes para usuários e administradores do Kubernetes

Podem ser utilizados para criar subgrupos de objetos

Podem ser criados e/ou adicionados a qualquer momento

# *Labels* frequentemente utilizados

```
"release" : "stable", "release" : "canary"
"environment" : "dev", "environment" : "qa", "environment" : "production"
"tier" : "frontend", "tier" : "backend", "tier" : "cache"
"partition" : "customerA", "partition" : "customerB"
"track" : "daily", "track" : "weekly"
```



# *Selectors*

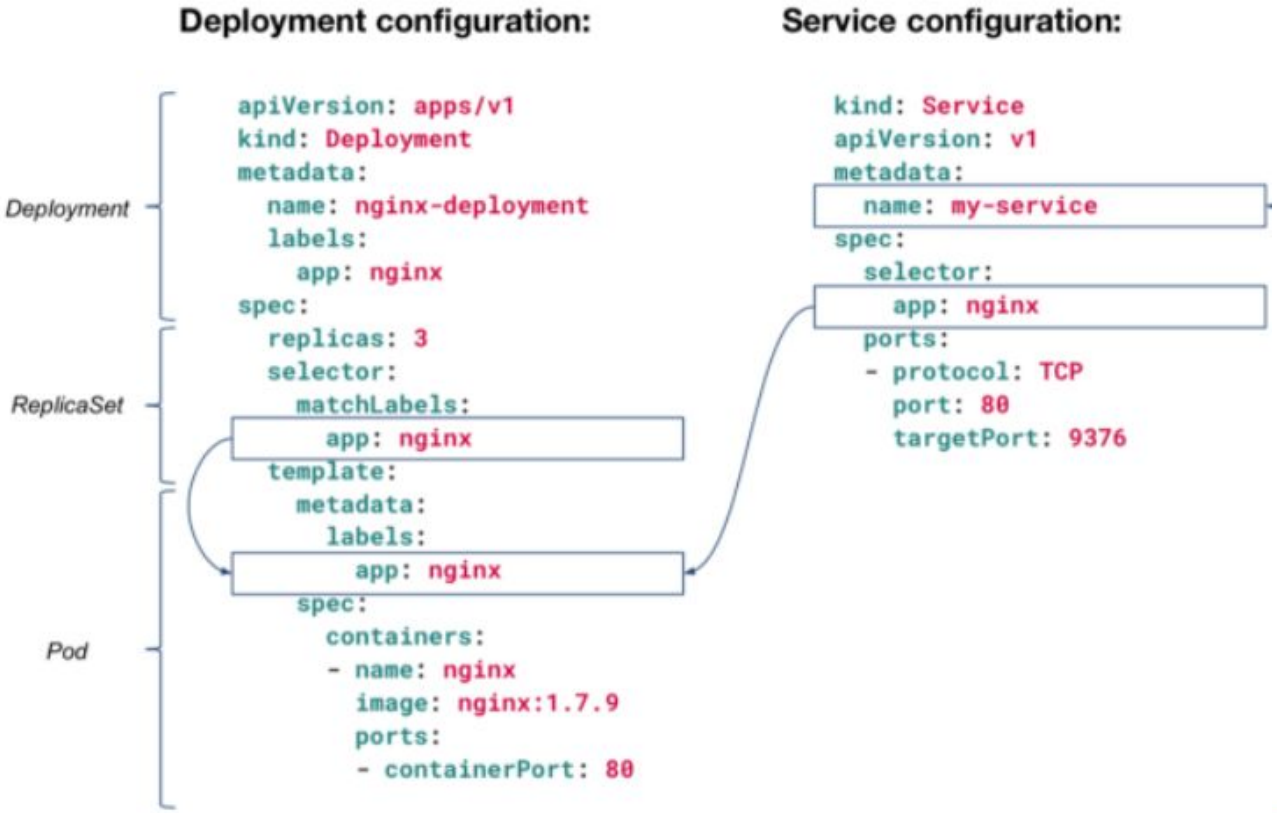
Label *selectors* são métodos de identificação de conjuntos de objetos

Primitiva básica de agrupamento no Kubernetes

Podem ser baseados em igualdade (*equality*) ou conjunto (set)

Podem ser exigidos múltiplos requerimentos em um mesmo *selector*, agrupados via AND lógico

# Labels e Selectors



# *Equality-based*

```
environment = production
tier != frontend
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: cuda-test
spec:
  containers:
    - name: cuda-test
      image: "k8s.gcr.io/cuda-vector-add:v0.1"
      resources:
        limits:
          nvidia.com/gpu: 1
  nodeSelector:
    accelerator: nvidia-tesla-p100
```

# *Set-based*

```
environment in (production, qa)
tier notin (frontend, backend)
partition
!partition
```

```
# kubectl get pod -l 'genre in (racing, fighting)'
NAME             READY   STATUS    RESTARTS   AGE
granturismo      1/1     Running   0          10m
roadrash         1/1     Running   0          10m
streetfighter    1/1     Running   0          10m
```

# *Equality-based vs Set-based*

## 2) Etiquetas e seletores

a. Antes de iniciar, execute o comando abaixo:

```
# lab-3.2.1
```

b. Diversos pods foram publicados, com os *labels* `sys`, `pub` e `genre`. Quantos pods existem no sistema `nes`?

▶ Visualizar resposta

c. Quantos pods possuem o gênero `platform`?

▶ Visualizar resposta

d. Quantos objetos existem no *publisher* `konami`, incluindo pods, deployments e demais?

# Taints e Tolerations

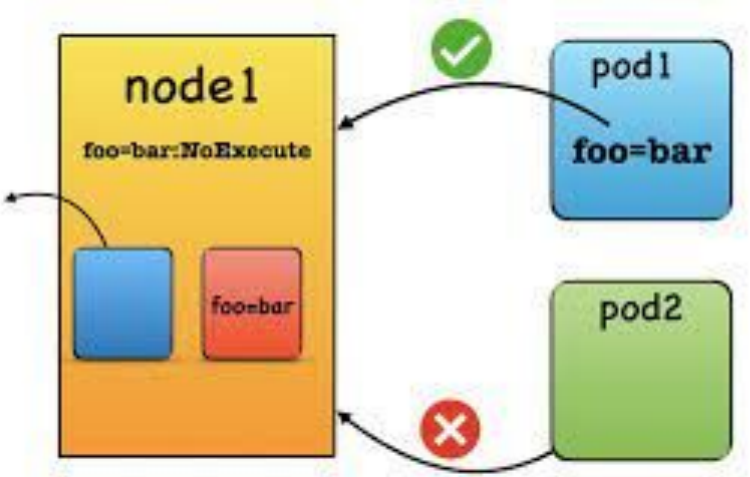
***Affinity*** é uma propriedade que atrai *pods* a um conjunto de *nodes*

***Taints*** fazem o oposto: repele pods de um conjunto de nodes

***Tolerations*** são aplicados a *pods*, e permitem *scheduling* em *nodes* compatíveis

***Ambos*** podem ser usados em conjunto para garantir que *pods* sejam alocados em *nodes* apropriados

**TAINT**
key=value:effect
key:effect




**TOLARATION**
key=value:effect
key:effect

# Por que utilizar?

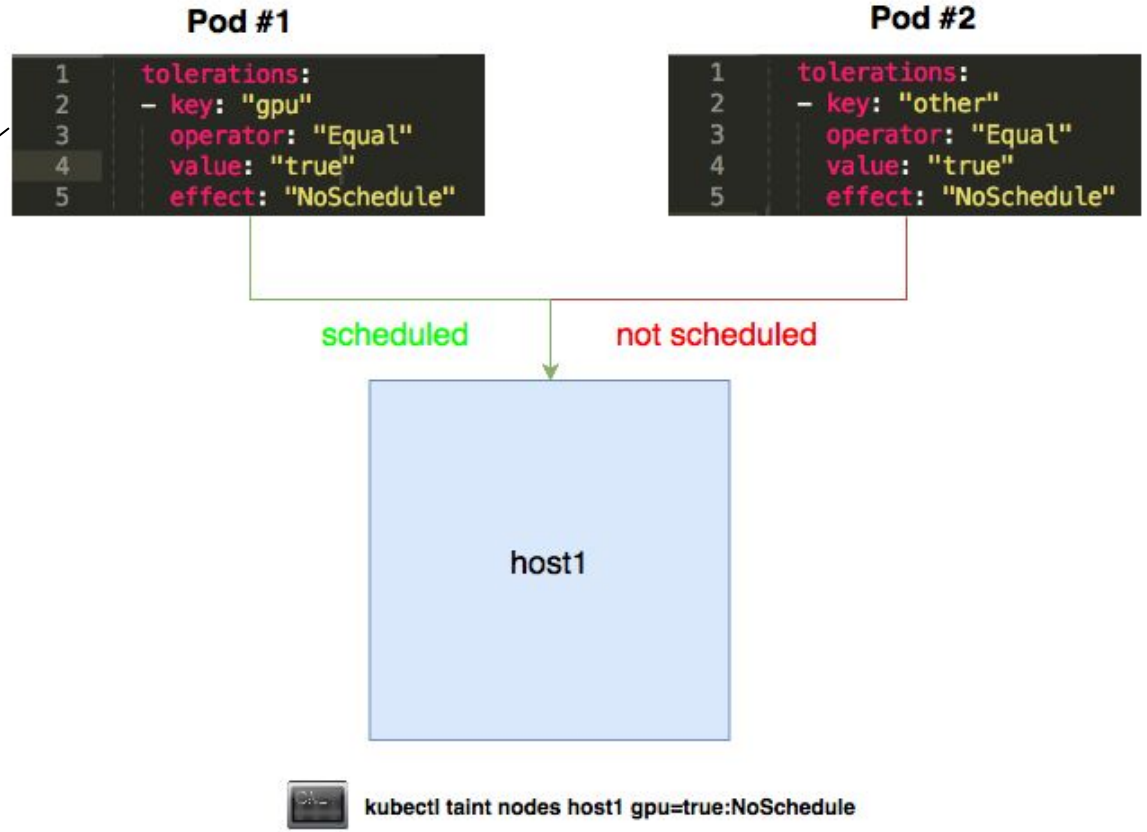
*Nodes* dedicados a um grupo de usuários ou função

*Nodes* com hardware especializado

Remoção de *pods* baseada em *taints* (p.ex., em caso de problemas)

# Aplicando *taints* e *tolerations*

gpu=true:NoSchedule



# Efeitos de *taints*

| | |
|---|---|
| *NoSchedule* | k8s não agendará (schedule) PODs exceto se tive toleration equivalente |
| *PreferNoSchedule* | k8s tentará evitar agendar (schedule) PODs exceto se tive toleration equivalente |
| *NoExecute* | k8s removerá PODs em execução no NODE exceto se tive toleration equivalente |

# Perguntas:

Há algum ***taint*** aplicado a algum dos *nodes* do *cluster*?

```
kubectl describe node <node> | grep Taints
kubectl get nodes -o json | jq '.items[].spec.taints'
```

Há algum *toleration* aplicado a algum *POD* do *cluster*?

```
kubectl describe pod <pod> | grep Tolarations
kubectl get pod -o json | jq '.items[].spec.tolerations'
```

# Taint no master node

Por padrão, o k8s restringe o uso dos master nodes para execução de PODs.

Essa restrição é imposta pela taint:
**node-role.kubernetes.io/control-plane:NoSchedule**

https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/create-cluster-kubeadm/#control-plane-node-isolation

# Assinalando *pods* a *nodes*

| | |
|---|---|
| *nodeName* | *nodeSelector* |
| Afinidade e anti-afinidade com *nodes* | Afinidade e anti-afinidade com *pods* |

# *nodeName:* simples, mas limitado

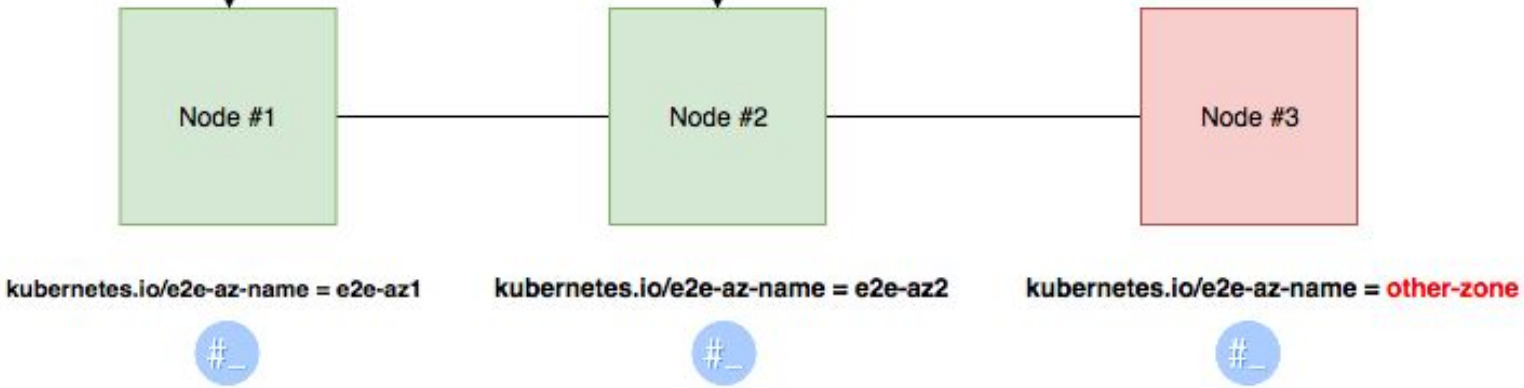
```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - name: nginx
    image: nginx
  nodeName: kube-01
```

# *nodeSelector:* selecionando por características

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx
  labels:
    env: test
spec:
  containers:
  - name: nginx
    image: nginx
    imagePullPolicy: IfNotPresent
  nodeSelector:
    disktype: ssd
```

# Afinidade com relação a *nodes*

```yaml
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: kubernetes.io/e2e-az-name
            operator: In
            values:
            - e2e-az1
            - e2e-az2
```



# Afinidade e anti-afinidade com relação a *nodes*

É conceitualmente similar ao NodeSelector, mas é mais versátil

*requiredDuringSchedulingIgnoredDuringExecution*

*preferredDuringSchedulingIgnoredDuringExecution*

https://kubernetes.io/docs/concepts/scheduling-eviction/assign-pod-node/#node-affinity

## Afinidade e anti-afinidade

NodeAffinity → Usado geralmente para selecionar nodes com características específicas
Ex: OS, CPU, MEM, AZ, REGION, etc

PodAffinity → Usado geralmente para executar pods juntos no mesmo node
Ex: app + cache

# Afinidade e anti-afinidade com relação a *pods*

**Deployment**
web-server
replicas: 3
labels:
  app: web-store

**Deployment**
redis-cache
replicas: 3
labels:
  app: store

| node-1 | node-2 | node-3 |
|---|---|---|
| *webserver-1* | *webserver-2* | *webserver-3* |
| *cache-1* | *cache-2* | *cache-3* |

# Afinidade e anti-afinidade com relação a *pods*

**Deployment**
web-server

```yaml
spec:
  affinity:
    podAntiAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
      - labelSelector:
          matchExpressions:
          - key: app
            operator: In
            values:
            - web-store
        topologyKey: "kubernetes.io/hostname"
    podAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
      - labelSelector:
          matchExpressions:
          - key: app
            operator: In
            values:
            - store
        topologyKey: "kubernetes.io/hostname"
```

**Deployment**
redis-cache

```yaml
spec:
  affinity:
    podAntiAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
      - labelSelector:
          matchExpressions:
          - key: app
            operator: In
            values:
            - store
        topologyKey: "kubernetes.io/hostname"
```

Label do Node

https://kubernetes.io/docs/reference/labels-annotations-taints/

# Exemplo prático

## 5) Afinidade em relação a pods

a. Antes de iniciar, execute o comando abaixo:

```
# lab-3.5.1
```

b. Em uma aplicação web típica, pode ser interessante fazer com que réplicas de atendimento ao usuário (*front-end*) não operem no mesmo *node*, para fins de redundância. De igual forma, é crítico que pods auxiliares a esses serviços (p.ex. um serviço de cache em memória como o Redis) estejam no mesmo *node* que a réplica de *front-end*. Para esse fim, podemos utilizar o recurso de afinidade e anti-afinidade entre pods.

Crie um deployment de nome `cache` usando a imagem `redis:alpine`, com 2 réplicas. Usando anti-afinidade, garanta que esses pods não executem no mesmo *host*.

▸ Visualizar resposta

c. Continuando a atividade anterior, crie agora o deployment de uma aplicação web fictícia com o nome `webapp`, usando a imagem `nginx:alpine`, com 2 réplicas. Garanta que os pods de cada réplica não executem no mesmo *host*, usando anti-afinidade, e garanta **também** que cada um desses pods executem conjuntamente com um dos pods do deployment `cache`.

https://kubernetes.io/docs/concepts/scheduling-eviction/assign-pod-node/#more-practical-use-cases

# Requests e Limits

Pode-se opcionalmente especificar recursos ao criar *Pods*

Requests dizem respeito aos recursos mínimos requeridos

*Limits* indicam limites máximos que não devem ser ultrapassados

Normalmente referem-se a CPU e RAM, mas há outros recursos

# Requests e Limits

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: frontend
spec:
  containers:
  - name: app
    image: images.my-company.example/app:v4
    resources:
      requests:
        memory: "64Mi"
        cpu: "250m"
      limits:
        memory: "128Mi"
        cpu: "500m"
  - name: log-aggregator
    image: images.my-company.example/log-aggregator:v6
    resources:
      requests:
        memory: "64Mi"
        cpu: "250m"
      limits:
        memory: "128Mi"
        cpu: "500m"
```

Fundamental ajustar recursos corretamente em *clusters* de produção

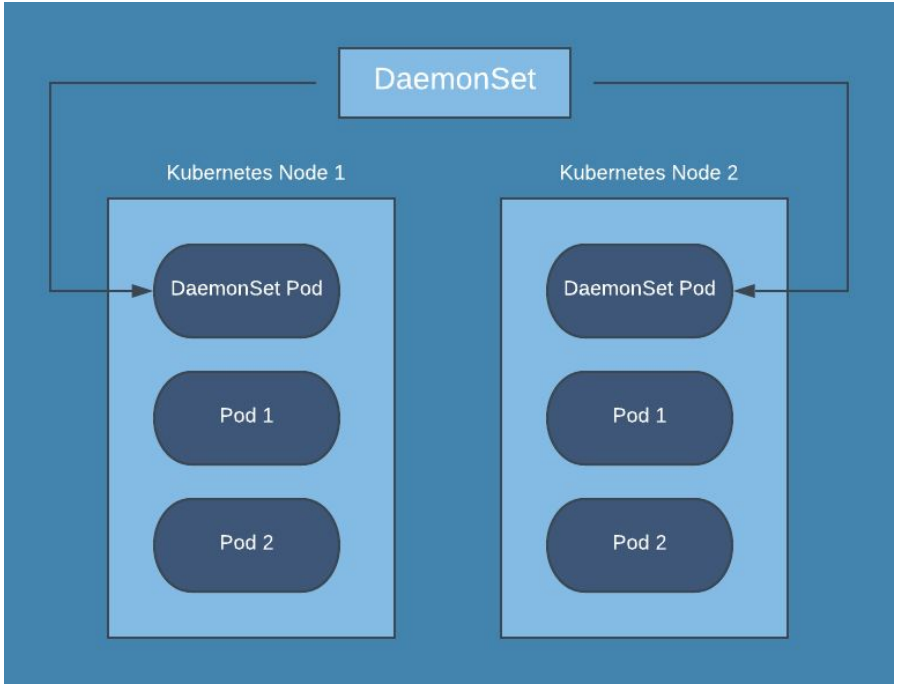
Monitoramento com *metrics* API

Limites de recursos podem ser configurados para *namespaces*

https://kubernetes.io/docs/concepts/configuration/manage-resources-containers

# *DaemonSets*

Construto que permite garantir que todos (ou parte) dos *nodes* executem uma cópia de um *pod*



# Alguns casos típicos

Executar um *daemon* de gestão de armazenamento em todos os *nodes*

Executar um coletor de logs em todos os *nodes*

Executar um *daemon* de monitoramento em todos os *nodes*

# Sintaxe YAML

```yaml
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: fluentd-elasticsearch
  namespace: kube-system
spec:
  selector:
    matchLabels:
      name: fluentd-elasticsearch
  template:
    metadata:
      labels:
        name: fluentd-elasticsearch
    spec:
      tolerations:
      - key: node-role.kubernetes.io/master
        effect: NoSchedule
      containers:
      - name: fluentd-elasticsearch
        image: quay.io/fluentd_elasticsearch/fluentd:v4.3.8
```

# Explorando alguns exemplos

*kube-proxy*

Calico

# Static PODs

Pods gerenciados diretamente pelo *kubelet*, sem gestão pelo *apiserver*

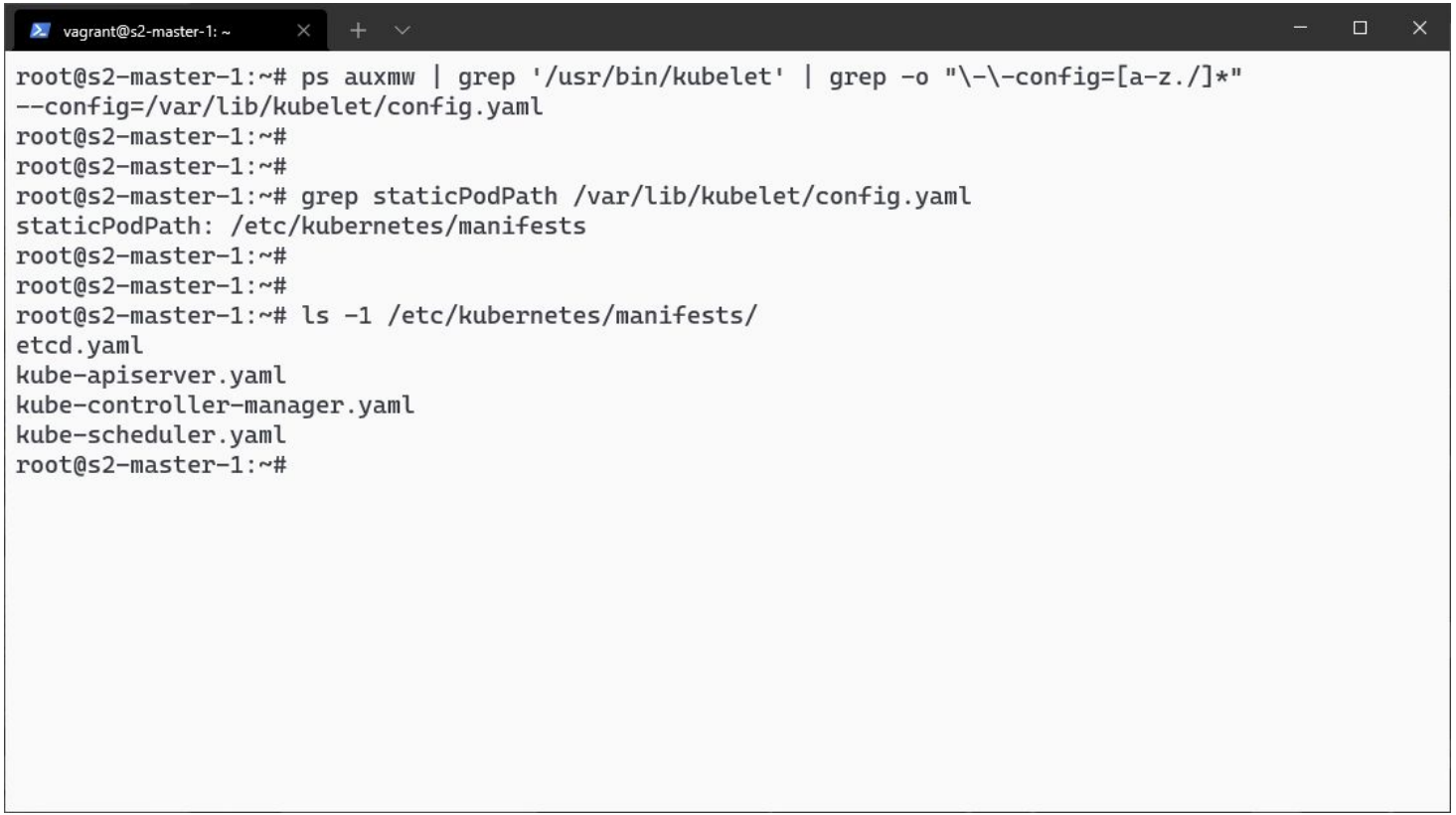
Pods espelho são criados no *apiserver* para visualização (não edição)

Pode ser criado com um arquivo *manifest* em diretório específico

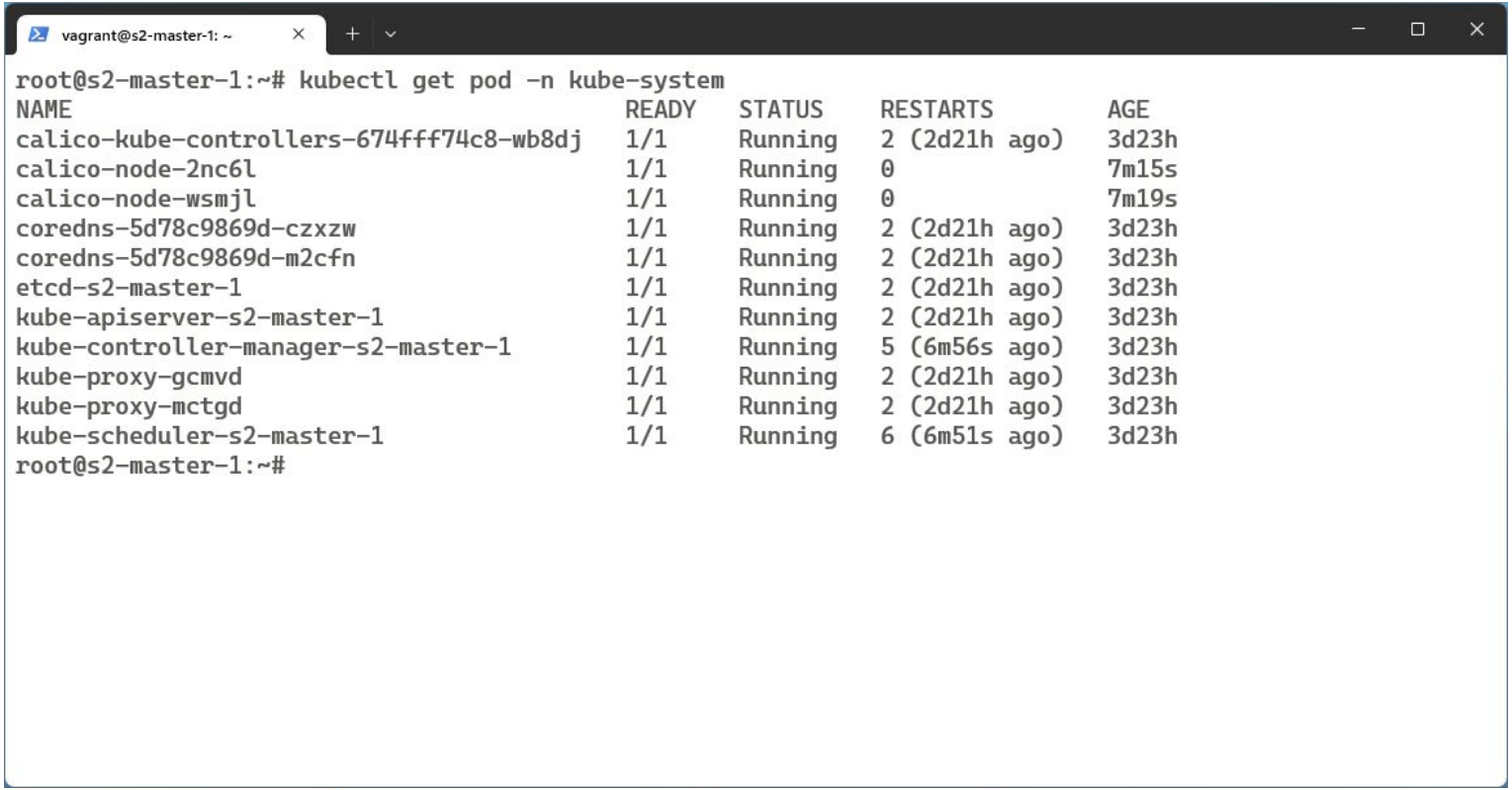
Outra alternativa é o download de um arquivo *manifest* via URL

https://kubernetes.io/docs/tasks/configure-pod-container/static-pod

# Static PODs


vagrant@s2-master-1: ~
root@s2-master-1:~# ps auxmw | grep '/usr/bin/kubelet' | grep -o "\-\-config=[a-z./]*"
--config=/var/lib/kubelet/config.yaml
root@s2-master-1:~#
root@s2-master-1:~#
root@s2-master-1:~# grep staticPodPath /var/lib/kubelet/config.yaml
staticPodPath: /etc/kubernetes/manifests
root@s2-master-1:~#
root@s2-master-1:~#
root@s2-master-1:~# ls -1 /etc/kubernetes/manifests/
etcd.yaml
kube-apiserver.yaml
kube-controller-manager.yaml
kube-scheduler.yaml
root@s2-master-1:~#

# Static PODs


vagrant@s2-master-1: ~
root@s2-master-1:~# kubectl get pod -n kube-system
NAME READY STATUS RESTARTS AGE
calico-kube-controllers-674fff74c8-wb8dj 1/1 Running 2 (2d21h ago) 3d23h
calico-node-2nc6l 1/1 Running 0 7m15s
calico-node-wsmjl 1/1 Running 0 7m19s
coredns-5d78c9869d-czxzw 1/1 Running 2 (2d21h ago) 3d23h
coredns-5d78c9869d-m2cfn 1/1 Running 2 (2d21h ago) 3d23h
etcd-s2-master-1 1/1 Running 2 (2d21h ago) 3d23h
kube-apiserver-s2-master-1 1/1 Running 2 (2d21h ago) 3d23h
kube-controller-manager-s2-master-1 1/1 Running 5 (6m56s ago) 3d23h
kube-proxy-gcmvd 1/1 Running 2 (2d21h ago) 3d23h
kube-proxy-mctgd 1/1 Running 2 (2d21h ago) 3d23h
kube-scheduler-s2-master-1 1/1 Running 6 (6m51s ago) 3d23h
root@s2-master-1:~#

# Tarefa 3

As atividades práticas desta sessão podem ser obtidas em formato HTML via:

https://bit.ly/ads19-tarefas-s3

ESCOLA SUPERIOR DE REDES

# Agendamento no Kubernetes

RNP ORGANIZAÇÃO SOCIAL DO MCTI

MINISTÉRIO DO TURISMO

MINISTÉRIO DA DEFESA

MINISTÉRIO DA SAÚDE

MINISTÉRIO DA EDUCAÇÃO

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E INOVAÇÕES

PÁTRIA AMADA BRASIL GOVERNO FEDERAL