Understanding Kubernetes Scheduling

2024/10 CNTUG Meetup / Ian Wu

吳翊瑋 Ian Wu

What I Do for a Living (Open to Work)

Kubernetes
DevSecOps
Platform Engineering

What I Do for Fun

BJJ
Guitar
Digital Wallet

Agenda

kube-scheduler

What is the role of kube-scheduler?

Scheduling Framework

What is the process in kube-scheduler?

kube-scheduler-simulator

What is the result of our scheduling strategy?

Control Plane Components

Control Plane

API-server

The central interface that manages and exposes the cluster’s data.

kube-controller-manager

Executes control loops to maintain the desired state of cluster components.

etcd

A distributed, reliable key-value store that provides persistent data storage for cluster state.

kube-scheduler

Determines which node for deploying pods.

Pod Manifest

What kind of properties would infect scheduling?

```yaml
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: my-nginx
  name: my-nginx
spec:
  containers:
  - image: nginx
    name: my-nginx
    resources:
      requests:
        cpu: 1
        memory: 500Mi
      limits:
        cpu: 1
        memory: 500Mi

```

```yaml
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: my-nginx
  name: my-nginx
spec:
  containers:
  - image: nginx
    name: my-nginx
    resources:
      requests:
        cpu: 1
        memory: 500Mi
      limits:
        cpu: 1
        memory: 500Mi
```

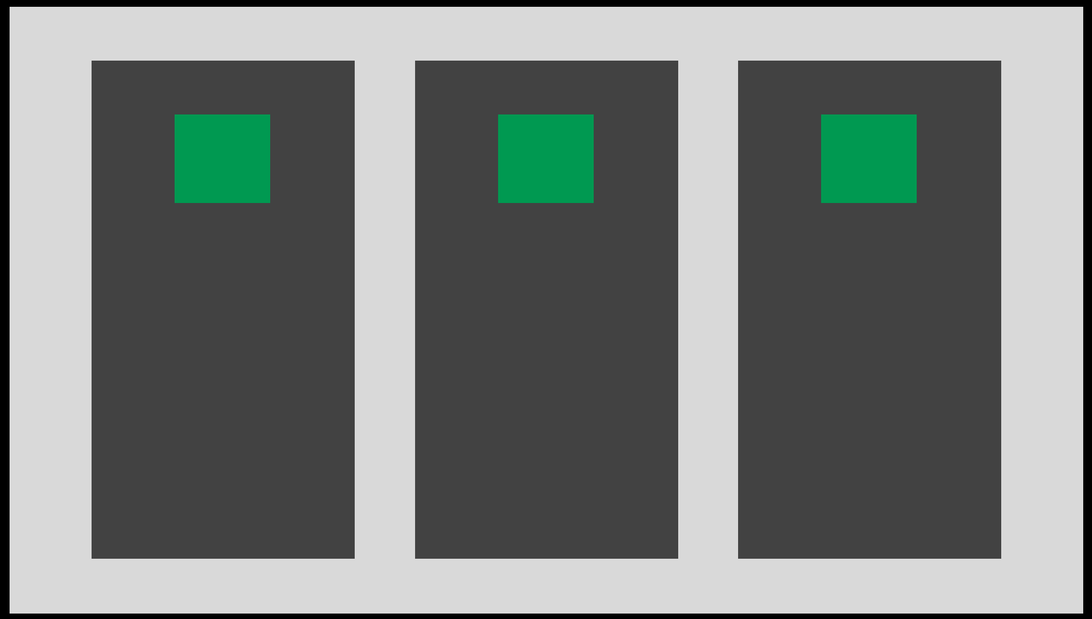
```yaml
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: my-nginx
  name: my-nginx
spec:
  schedulerName: default-scheduler
  nodeName: kind-control-plane
  containers:
  - image: nginx
    name: my-nginx
    resources:
      requests:
        cpu: 1
        memory: 500Mi
      limits:
        cpu: 1
        memory: 500Mi
```

The responsibility of kube-scheduler is find the Node and bind it into the Pod.

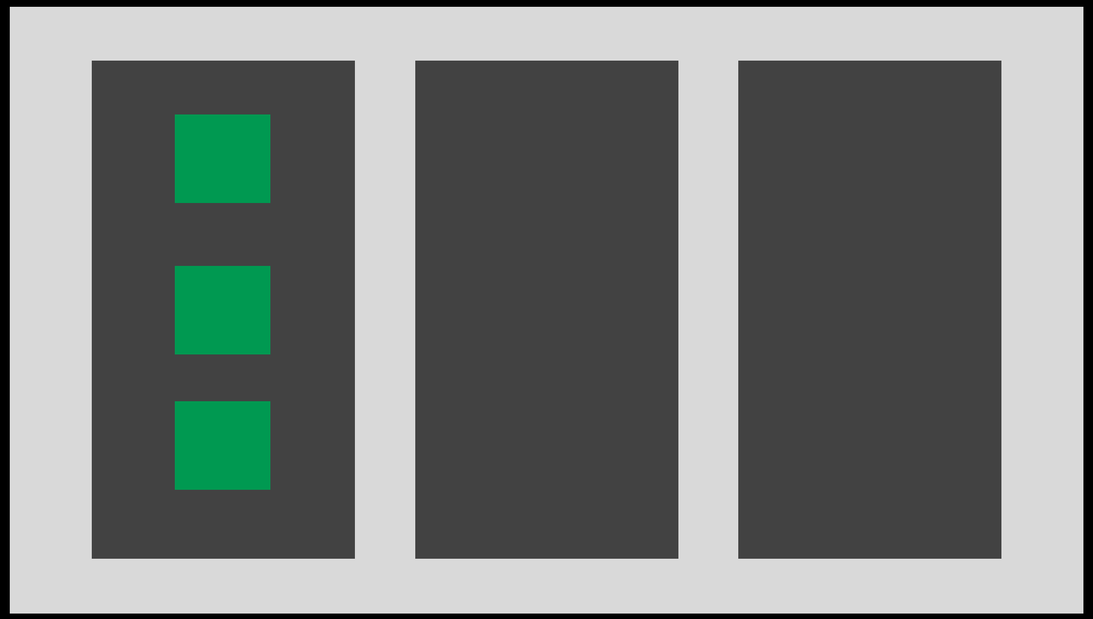
“How does Kubernetes schedules workloads evenly across nodes in a cluster?”

Adam · No Where

Scheduling



Balanced Scheduling



Imbalanced Scheduling

Pod Manifest

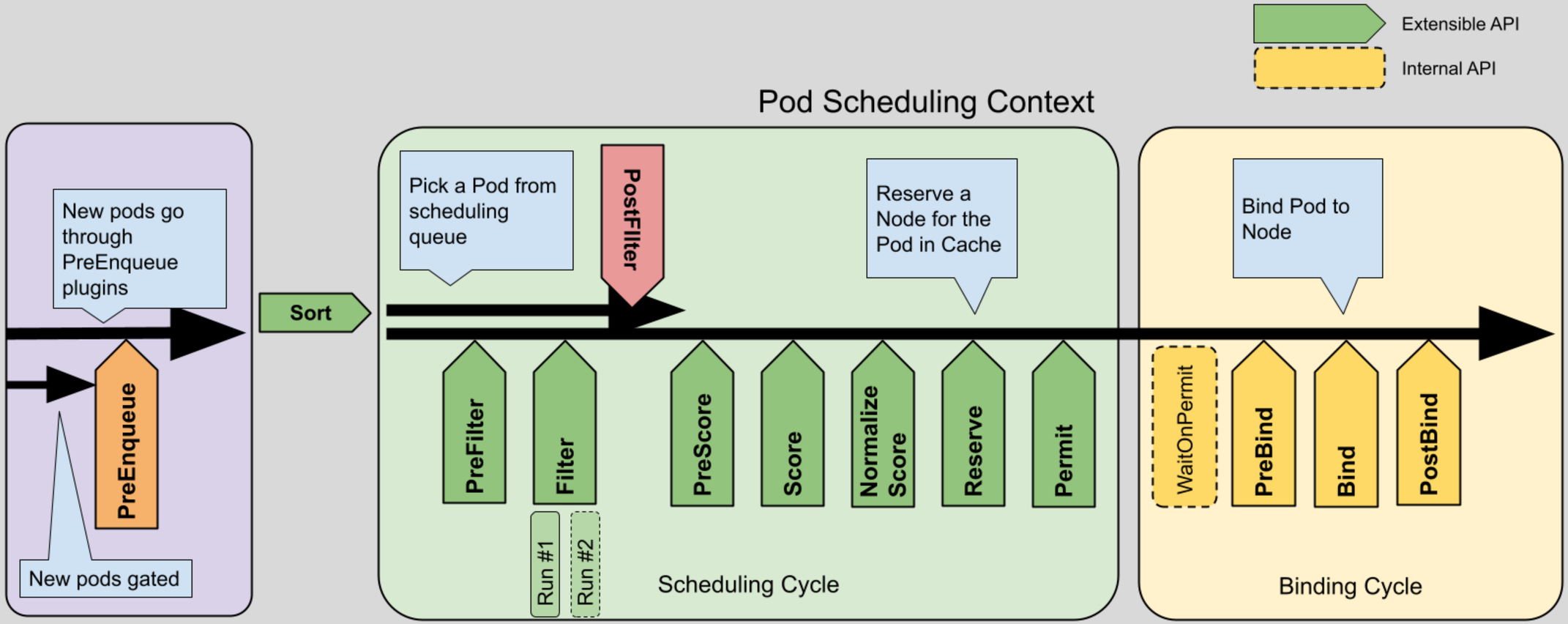
What kind of properties would infect scheduling?

```yaml
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: my-nginx
  name: my-nginx
spec:
  containers:
  - image: nginx
    name: my-nginx
    resources:
      requests:
        cpu: 1
        memory: 500Mi
      limits:
        cpu: 1
        memory: 500Mi

```

containers.resources.requests

The Node must have such allocatable resources.

```yaml
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: my-nginx
  name: my-nginx
spec:
  containers:
  - image: nginx
    name: my-nginx
    resources:
      requests:
        cpu: 1
        memory: 500Mi
      limits:
        cpu: 1
        memory: 500Mi

```



Scheduling Framework

Scheduling Cycle

Filter

Select the feasible Nodes.

Score

Rank the feasible Nodes.

containers.resources.requests

The Node must have such allocatable resources.

```yaml
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: my-nginx
  name: my-nginx
spec:
  containers:
  - image: nginx
    name: my-nginx
    resources:
      requests:
        cpu: 1
        memory: 500Mi
      limits:
        cpu: 1
        memory: 500Mi

```

NodeResourcesFit in Filter and Score

Filter

Select nodes with sufficient resources for the pod.

Score

Rank the feasible nodes based on resource availability.

NodeResourcesFit in Filter and Score

Filter

```yaml
resources:
  requests:
    cpu: 1
    memory: 500Mi
```

Select nodes with sufficient resources for the pod.

Score

$$\text{score} = 100 - \left(\frac{\text{used CPU}}{\text{total CPU}} + \frac{\text{used memory}}{\text{total memory}}\right)$$

Rank the feasible nodes based on resource availability.

Cloud charge cross-zone Traffic

Cloud charges the traffic between different Zone.

“The primary Database deploy on Zone A.
In Kubernetes, how to schedule the workload to Zone-A?”

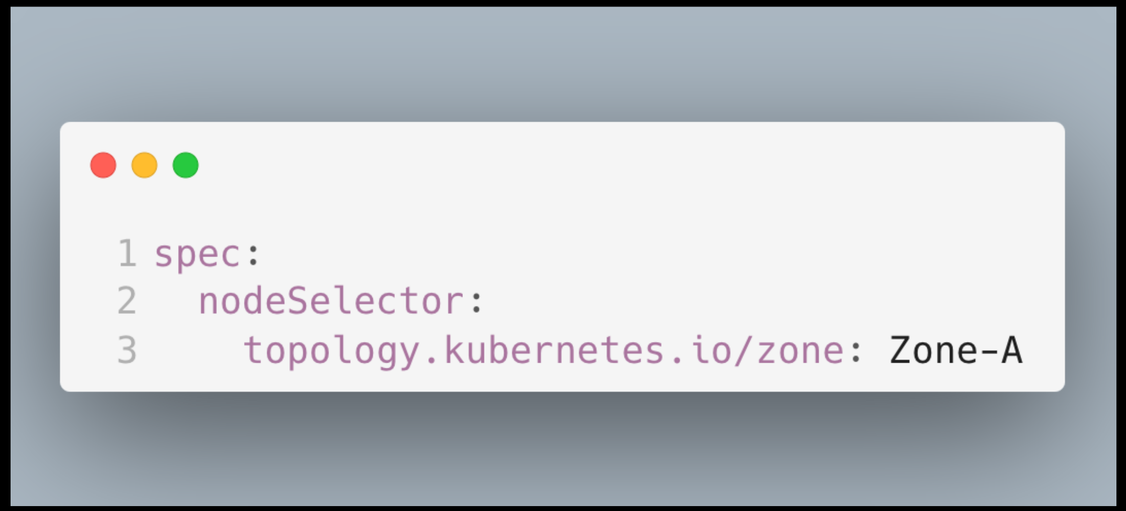
Adam · No Where

“NodeSelector!”

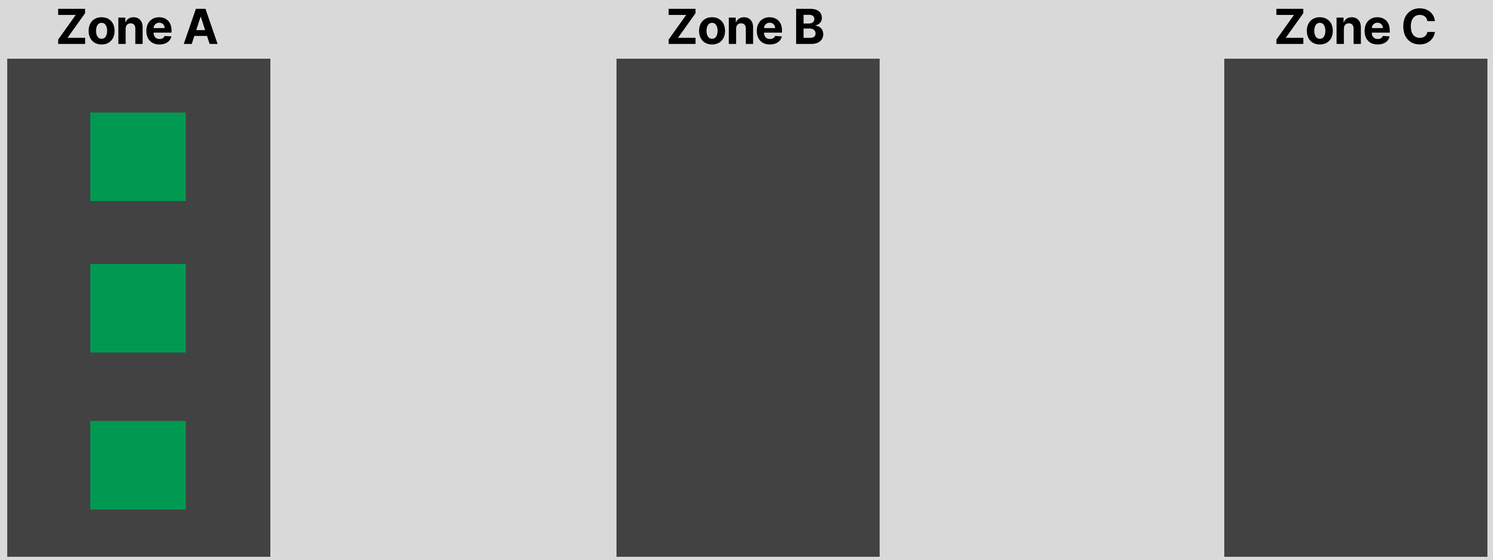
Adam · No Where

NodeSelector

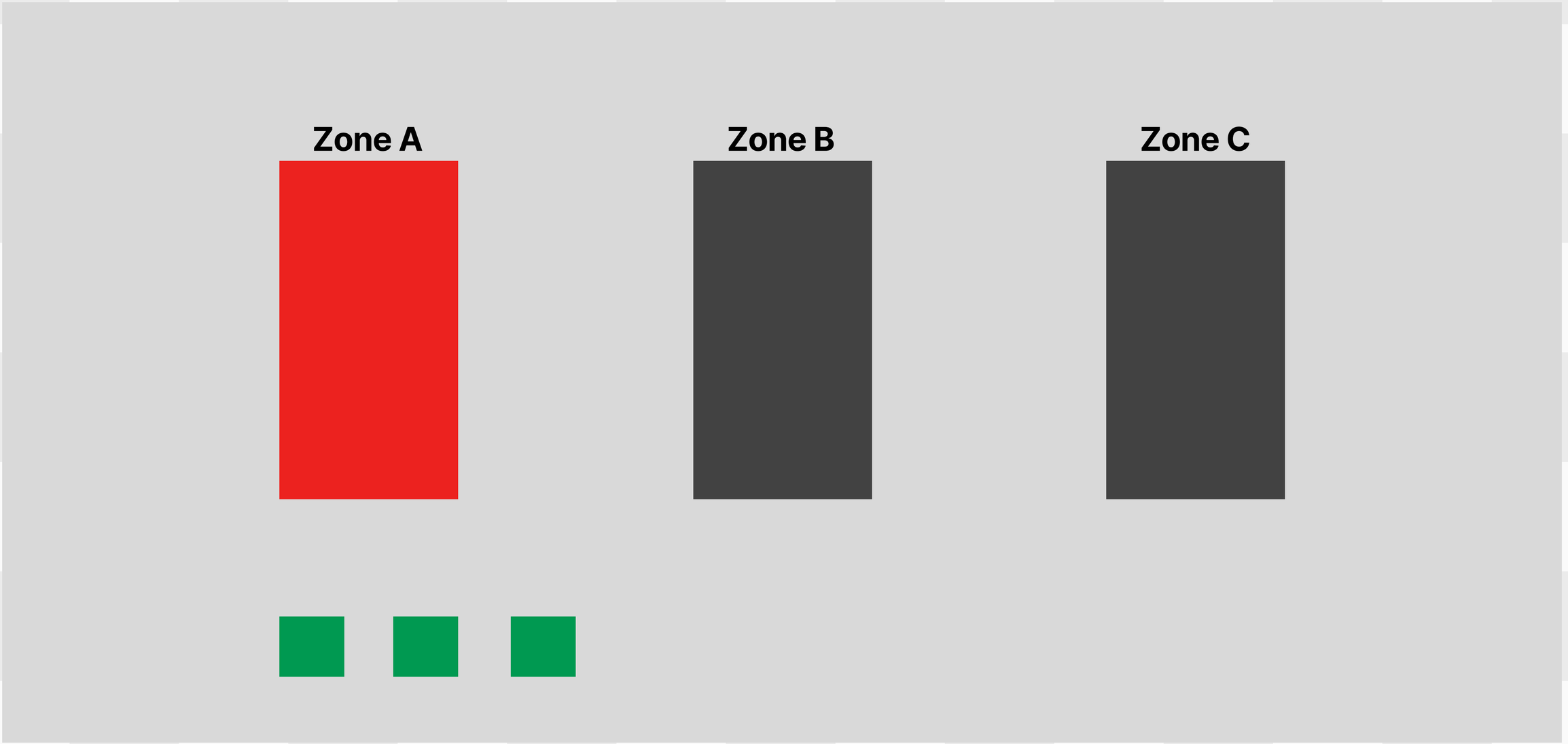
Kubernetes MUST scheudules the workload into Zone A.



```
spec:
  nodeSelector:
    topology.kubernetes.io/zone: Zone-A
```



Schedule Pod into Zone A



Zone A outage

Schedule with Nodeselector

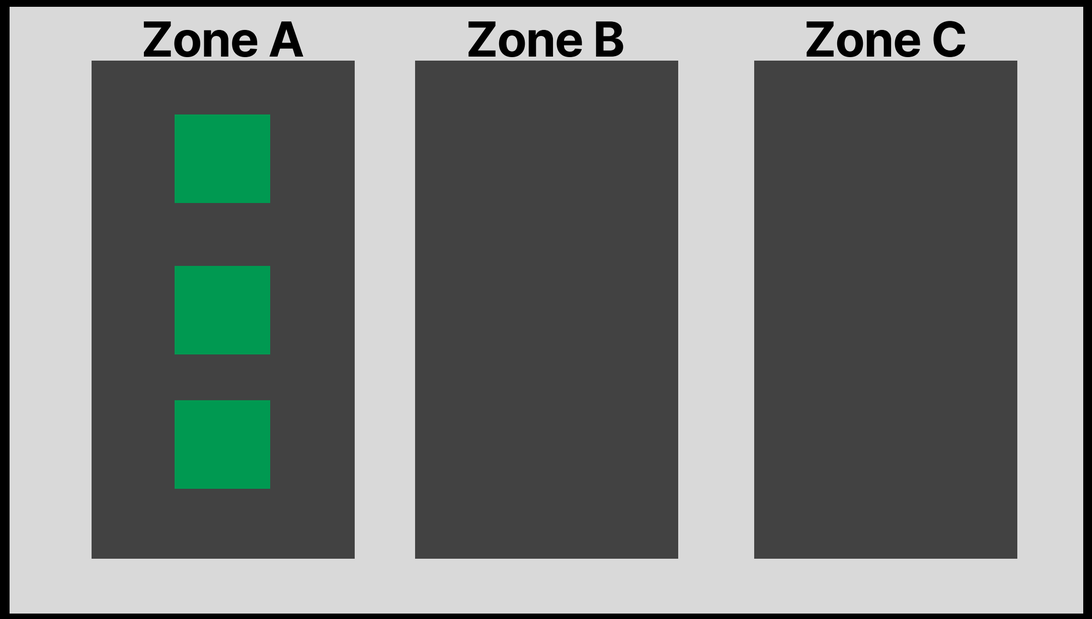
Filter

Select the feasible Nodes which is in Zone A.

Score

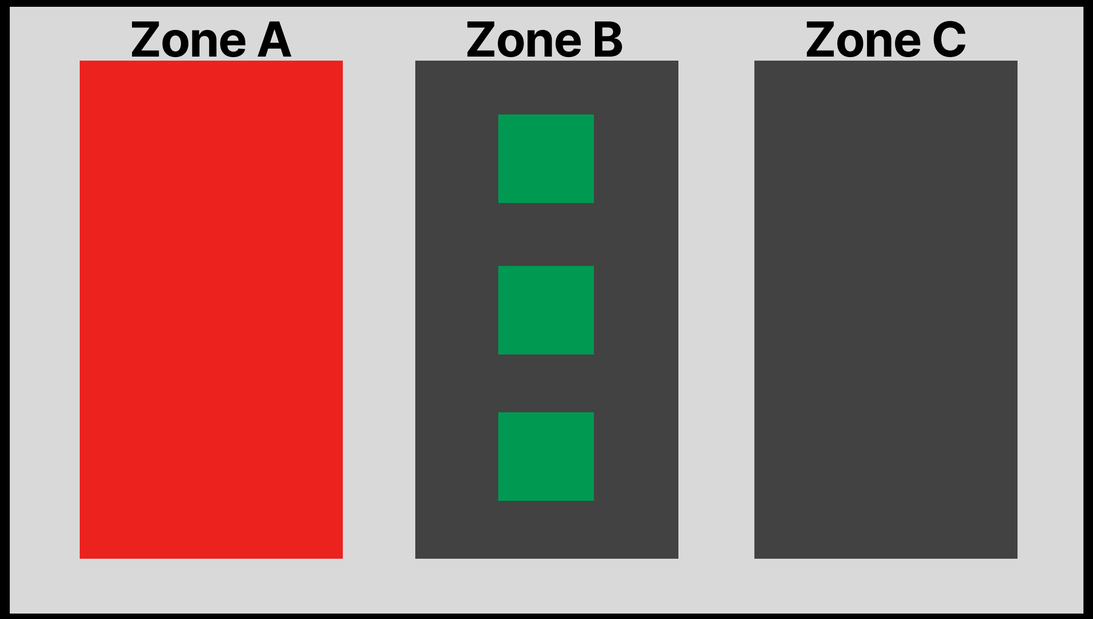
Rank the feasible Nodes.

Failover



When anything works fine

Schedule the workload into Zone A.



When something got wrong.

Schedule the workload into Zone B.

NodeAffinity

Kubernetes PREFER schedules the workload into Zone A.

```yaml
spec:
  affinity:
    nodeAffinity:
      preferredDuringSchedulingIgnoredDuringExecu
      - weight: 100
        preference:
          matchExpressions:
          - key: topology.kubernetes.io/zone
            operator: In
            values:
            - Zone-A  # 優先排程在 Zone-A
      - weight: 50
        preference:
          matchExpressions:
          - key: topology.kubernetes.io/zone
            operator: In
            values:
            - Zone-B  # failover 時排程在 Zone-A
```

Schedule with NodeAffinity

Filter

Select the feasible Nodes

Score

Rank the feasible Nodes, prefer Zone A, but Zone B is fine

Schedule with NodeAffinity

Filter

Select the feasible Nodes **in Zone A**

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: with-node-affinity
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
          matchExpressions:
          - key: topology.kubernetes.io/zone
            operator: In
            values:
            - A-zone      # 只能在 A-zone 調度

      preferredDuringSchedulingIgnoredDuringExecution:
          - weight: 100
            preference:
              matchExpressions:
              - key: topology.kubernetes.io/zone
                operator: In
                values:
                - A-zone  # 優先在 A-zone 調度
  containers:
  - name: with-node-affinity
    image: registry.k8s.io/pause:2.0
```

Score

Rank the feasible Nodes, **prefer Zone A.**

Happy Disaster

Some product go viral, increase the loading of workload rapidly.

Problem

Even we deploy the HPA and clusterautoscaler, but the scaled node might takes minutes to join cluster.

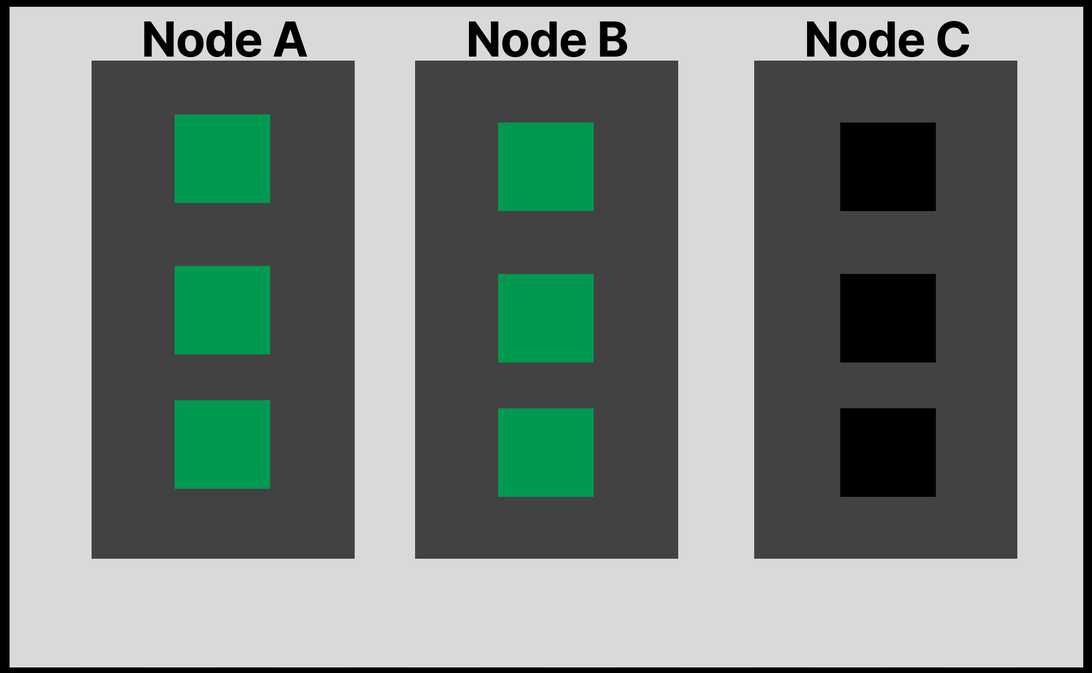
“How to scale workload ASAP?”

Adam · No Where

Solution

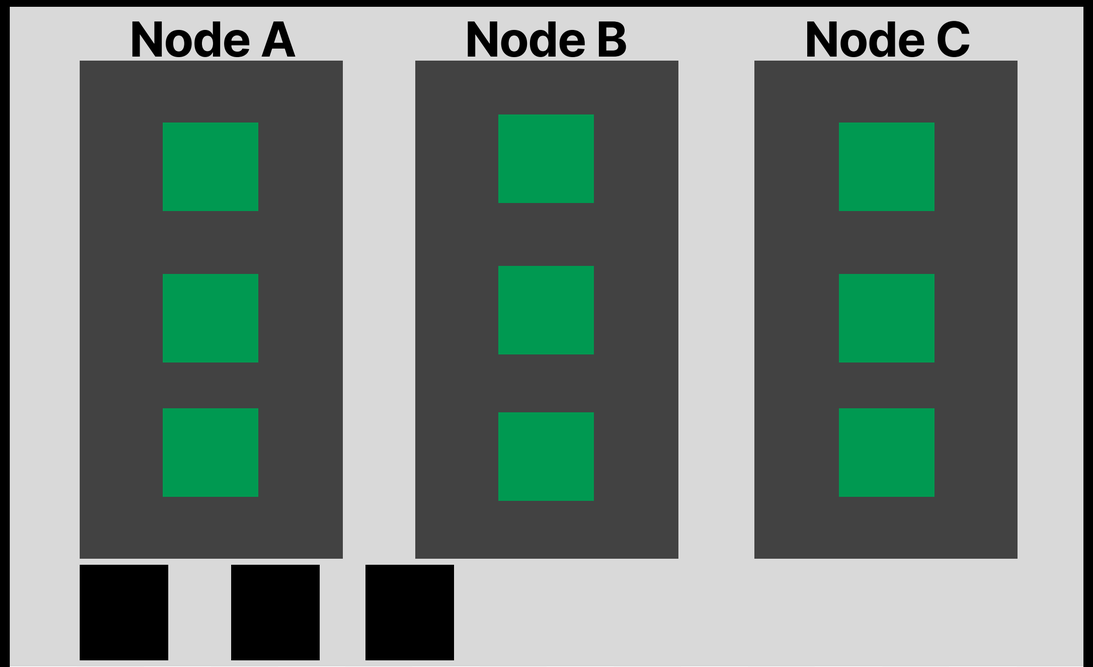
Overprovision

Over-provisoning



Proactive Node standby

Schedule low priority Pod on Proactive Node



Proactive Node services

High priority Pod preempts low priority Pod on Proactive Node

High Priority Pod

Log Priority Pod

Scheduling Cycle

Filter

Select the feasible Nodes.

Score

Rank the feasible Nodes.

Scheduling Cycle

Filter

Select the feasible Nodes.

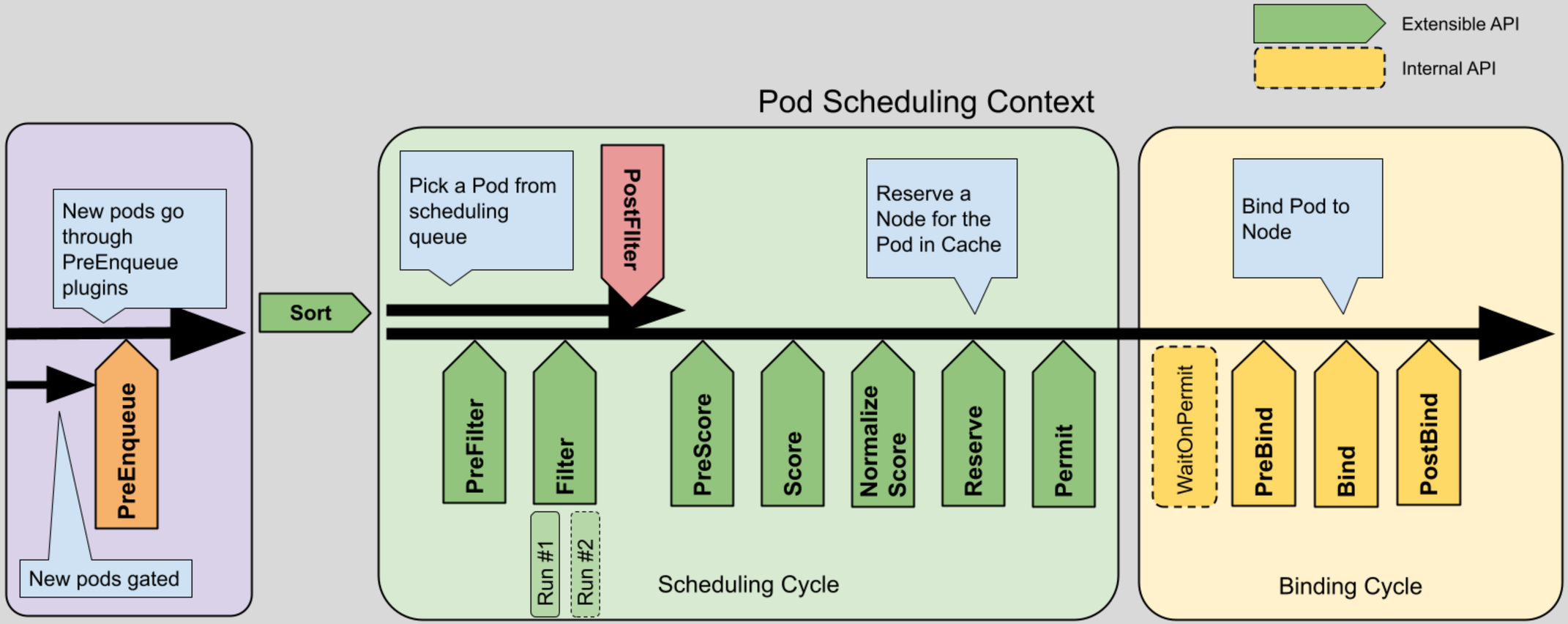
If **NO** feasible Nodes

Post-filter

Preemption runs here.

Score

Rank the feasible Nodes.



Scheduling Framework

“Sounds wonderful, but how to test it?”

Adam · No Where

Three problem about test scheduling

Cost

If you want to build a test environment for testing scheduling strategy, it might be expensive.

Test at Large Scale

Private Cloud might not have enough machines, even in Public Cloud, you might need some time to build a large cluster.

Trouble shooting

The scheduling would like to know the all nodes in the Cluster, complicated Scheduling strategy is hard to test.

kube-scheduler-simulator

Three problem about test scheduling

Cost

kss can test scheduling without resources.

Test at Large Scale

kss can test scheduling at Large Scale easy.

Trouble shooting

kss provides debuggable scheduler, it logs all result in stages about Scheduling Framework.

Demo

Using kube-scheduler-simulator to verify the over-provisioning pattern.

Scale out Node → 10

Deploy Backend x 40

Deploy low-priority-class and low-priority-pod x 12

Observe LPP Pending

Simulate the behavior of clusterautoscaler：Scale out Node → 13

Observe LPP, it should be deployed to the Proactive Node

Simulate the behavior of HPA：Scale out backend 40 → 52

Observe Backend preempt LPP

“I heard every stages in Scheduling Framework is an extension point, how can we extend it?”

Adam · No Where

Extend your scheduler

Now about extending scheduler

Scheduler Profile

If default scheduler fits your needs, but you want some custom.
For example, the parameter of some Plugin, or high weight of specific Plugin.

Scheduler Extender

bad performance, bad error handling.

Plugins

You need to build your scheduler with the custom plugins.

Demo

Using kube-scheduler-simulator to verify the custom plugin.

Enable the custom plugin in KubeSchedulerConfiguration
Deploy new Pod
Verify the score of custom plugin

Trend about scheduler

Descheduler

Obey the scheduling policy after deployment.
Evict the pod before the resources usage hit the limits.

Batch Job for ML

Batch Job for ML is different from normal.
Kueue and QueueHint.

kube-scheduler-wasm-extension

Extend kube-scheduler with WASM.

Takeaway

Understand Scheduling gives us more choices about actions.

Bonus

Three Common Misconception

First

pod.containers.resources.requests only works during scheduling.

Second

Usage above pod.containers.resources.limits won’t cause any problem.

Third

Only Memory would cause OOMKilled.

Three Common Misconception

First

pod.containers.resources.requests only works during scheduling.

The requests.cpu would affect the cgroup.

Second

Usage above pod.containers.resources.limits won't cause any problem.

The limits.cpu would cause the CPU throttling.

Third

Only Memory would cause OOMKilled.

The GC Thread might not GC completely because the CPU throttling, it might trigger OOMKilled.