Dynamically rebalancing workloads and optimizing resource utilization in Kubernetes

Chris Nesbitt-Smith

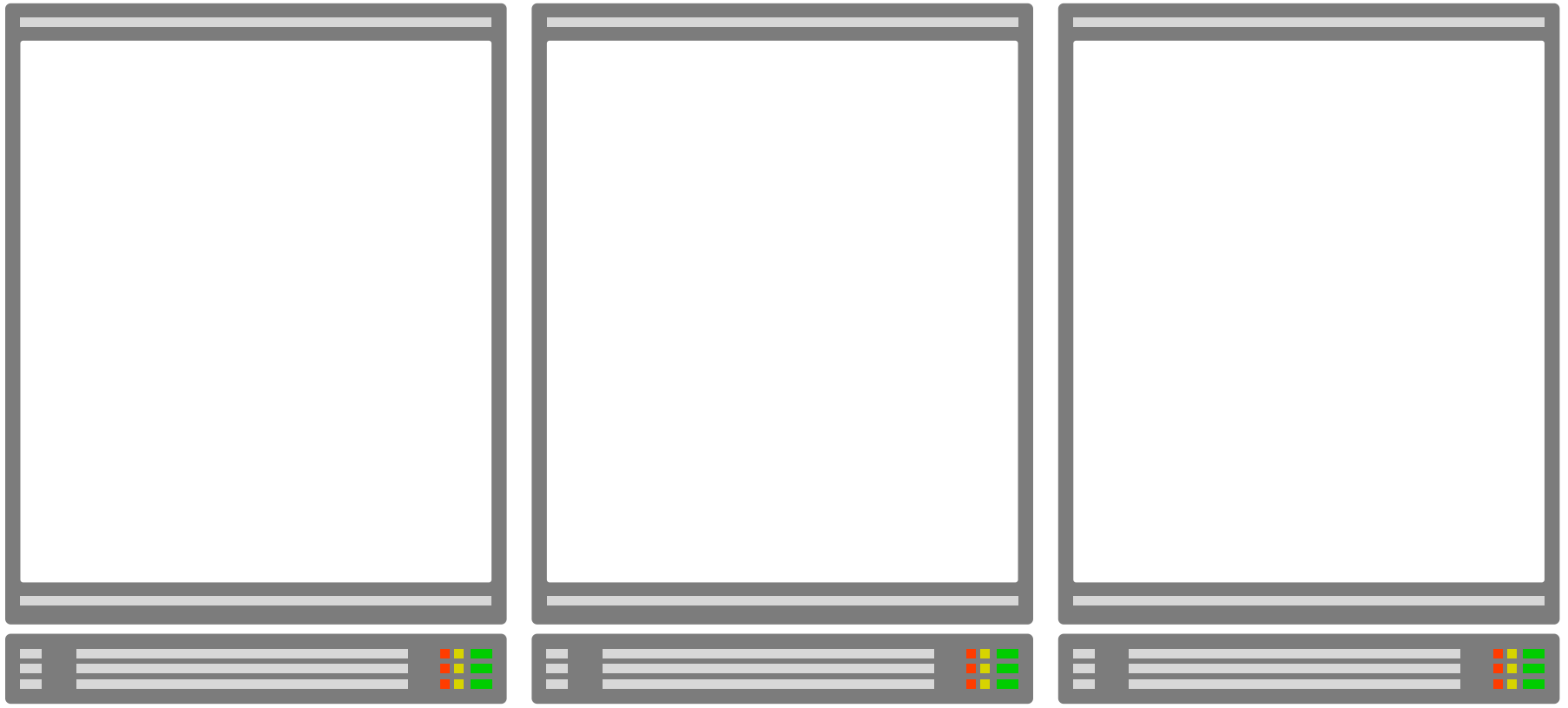
learnk8s

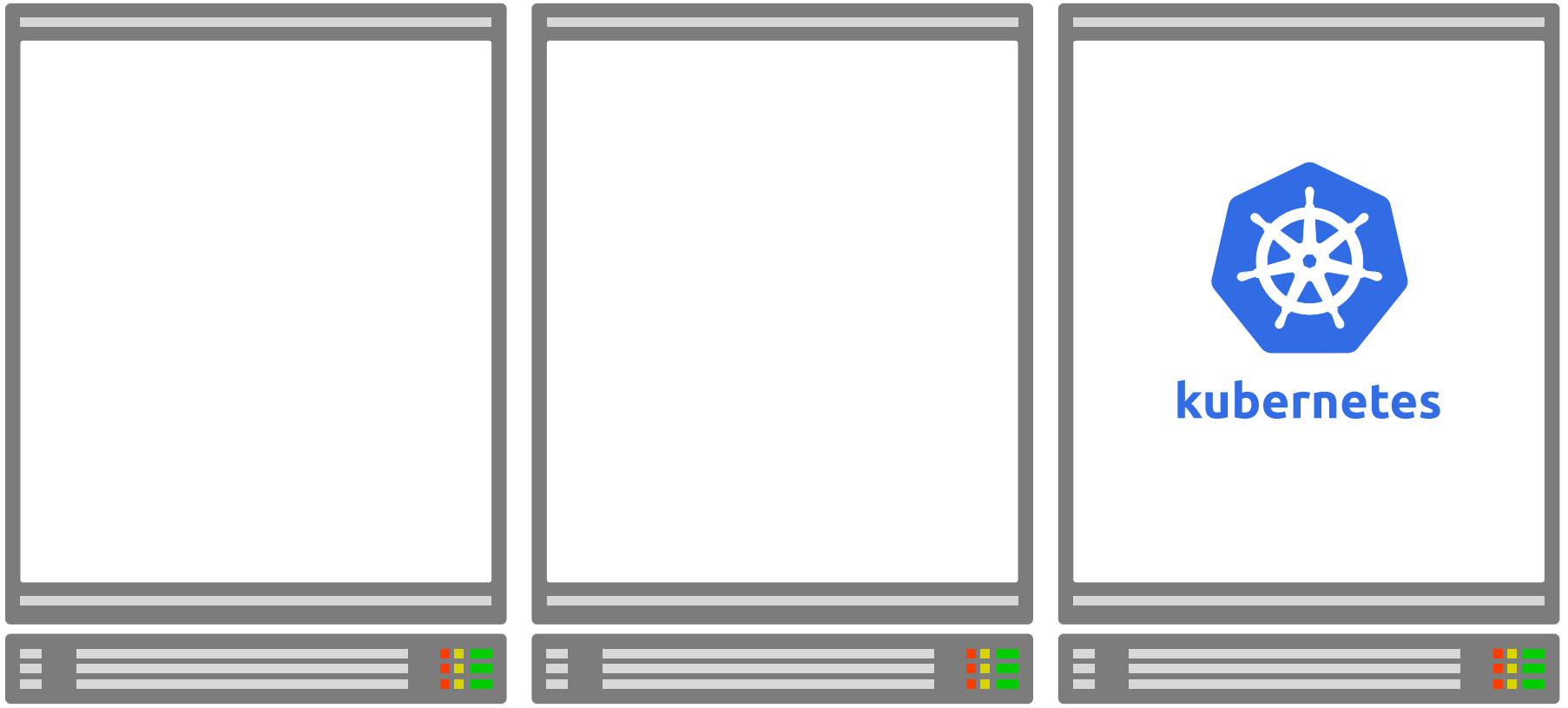
Chris Nesbitt-Smith

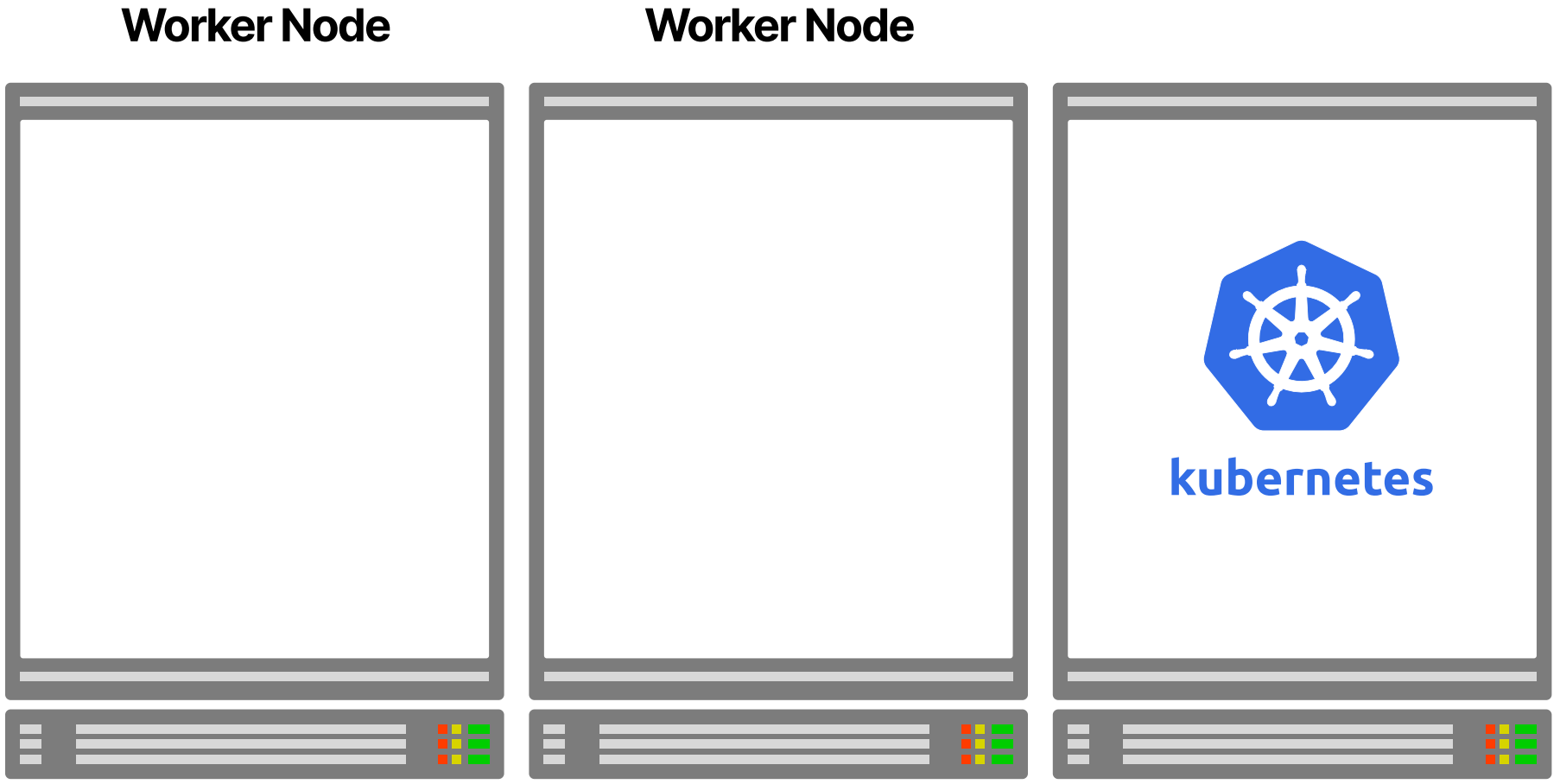
UK Gov | LearnK8s | Control Plane | lots of open source

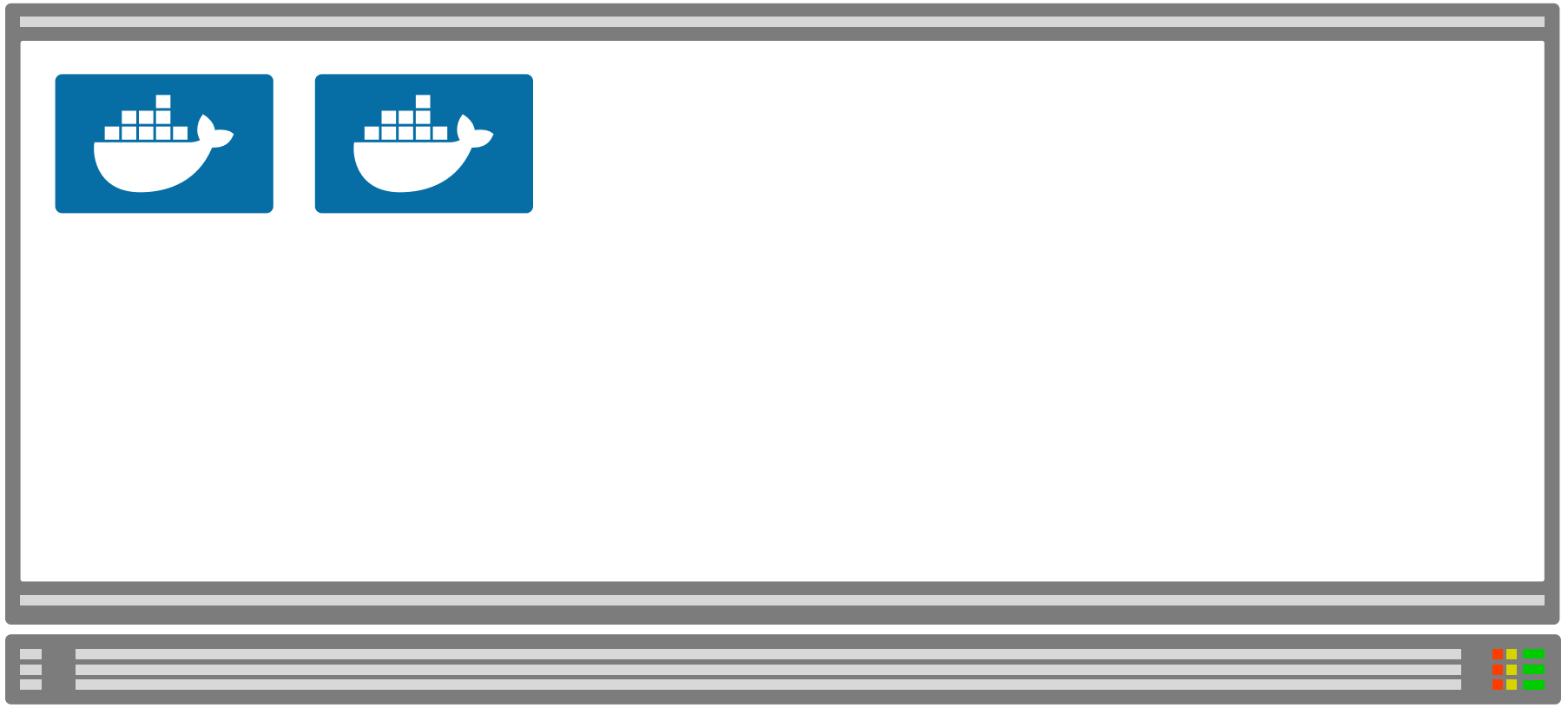
learnk8s

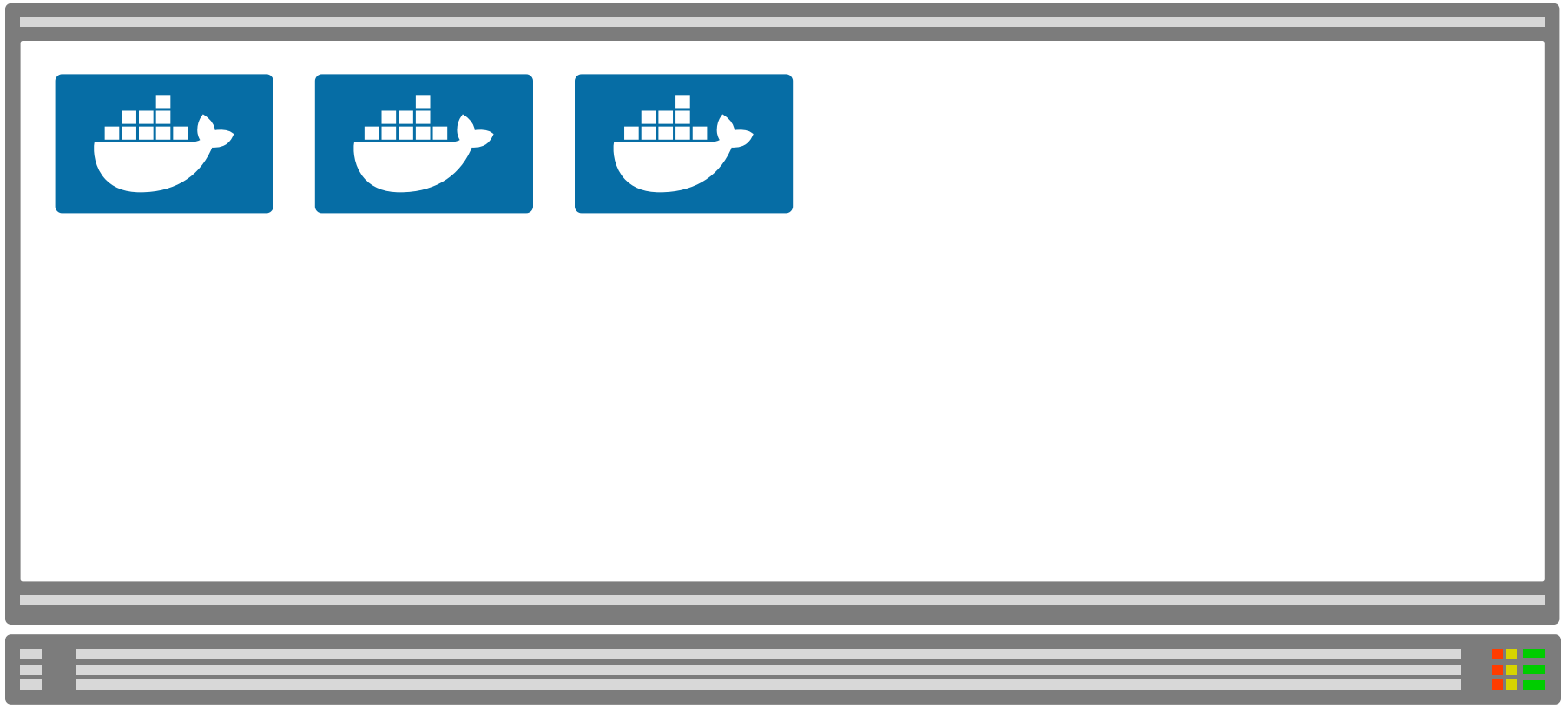
Datacentre as a single computer

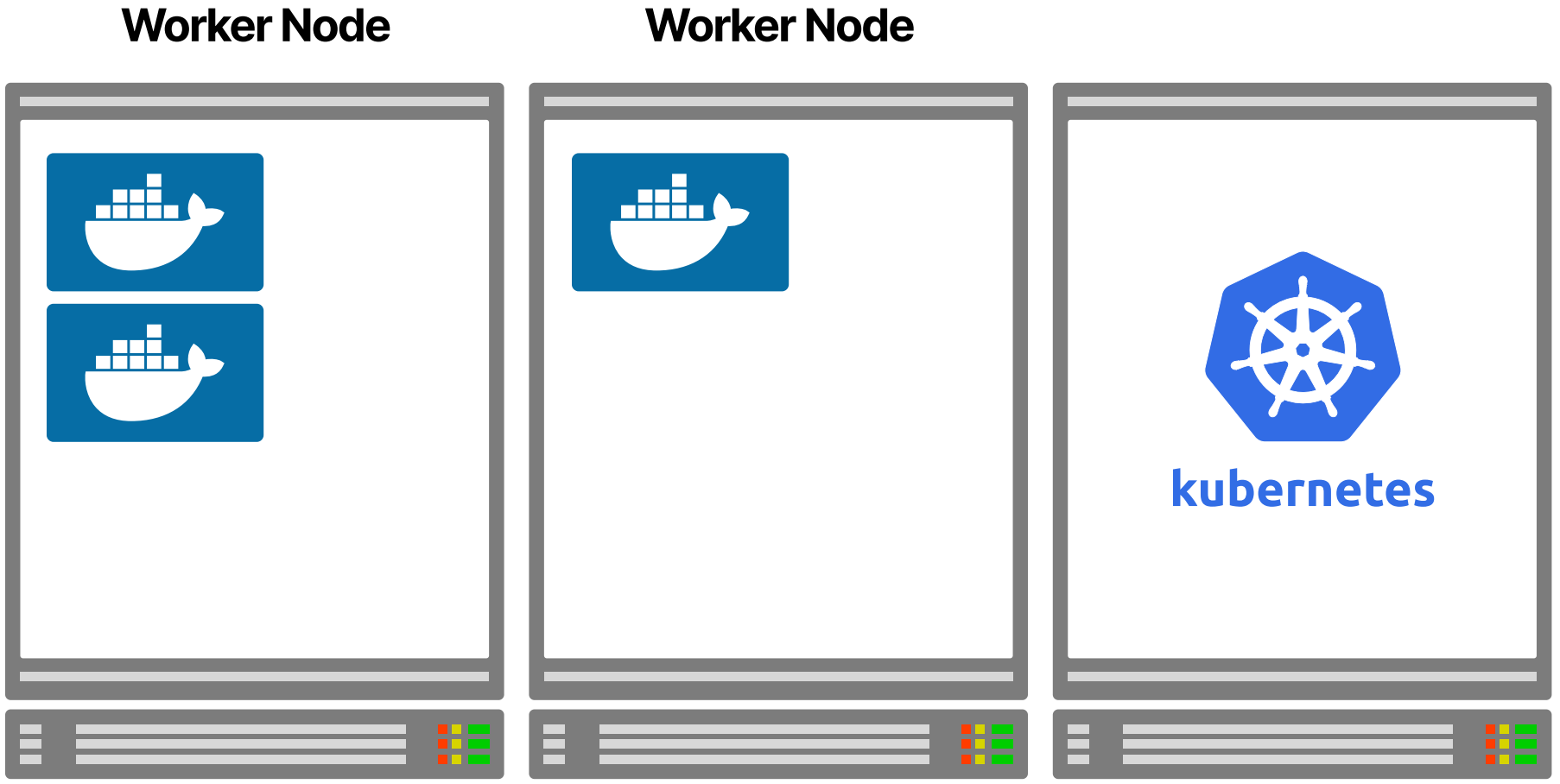



kubernetes

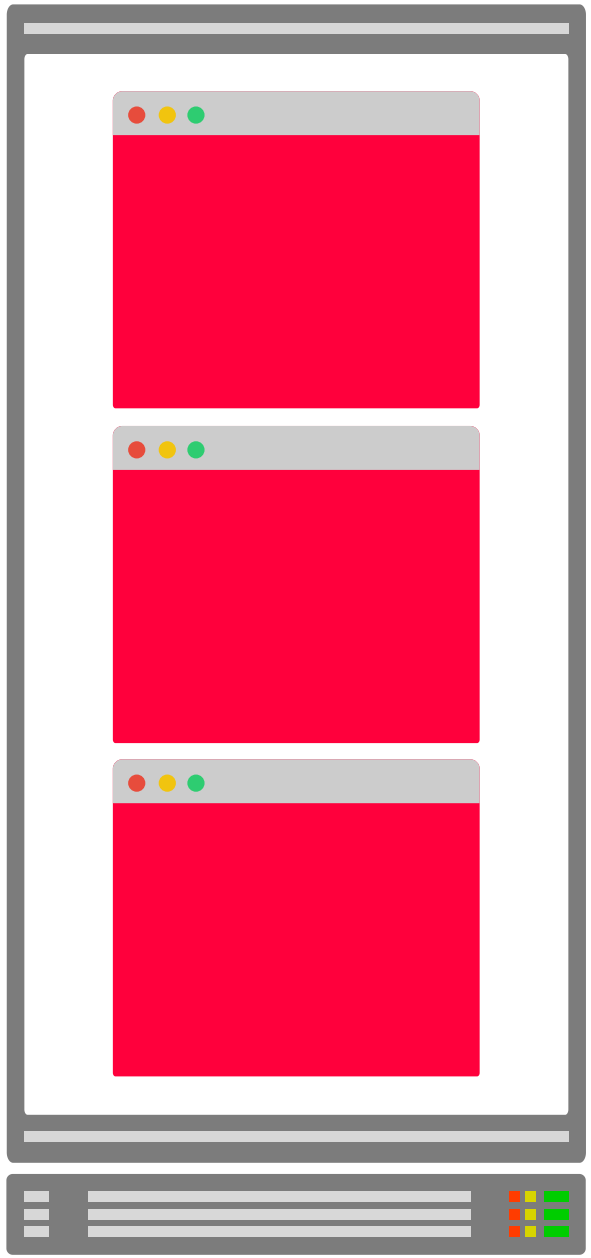

Worker Node
Worker Node
kubernetes

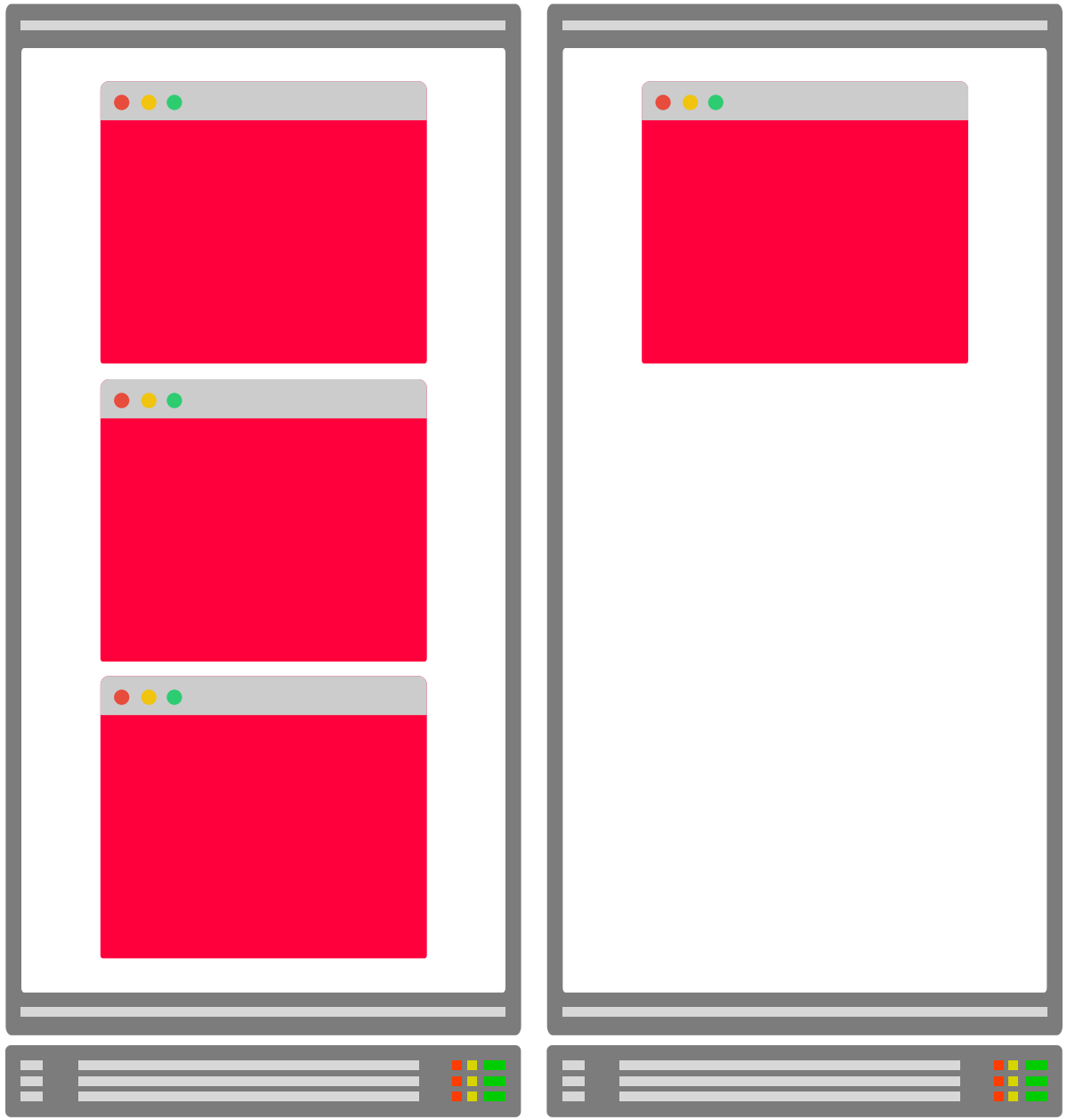


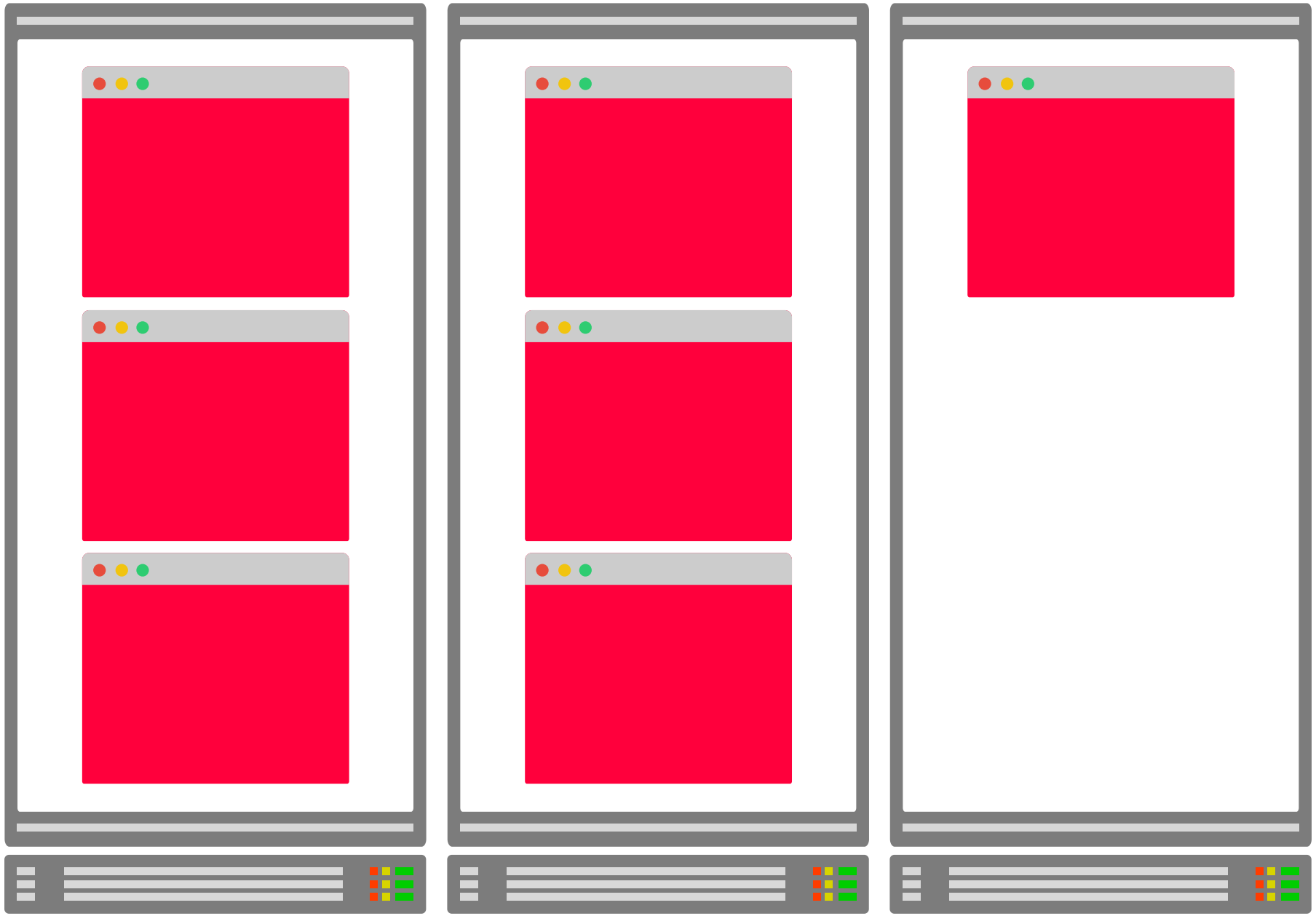



Worker Node
Worker Node
kubernetes

Combining autoscalers







COMBINING AUTOSCALERS FOR OPTIMAL RESOURCE ALLOCATIONS IN KUBERNETES

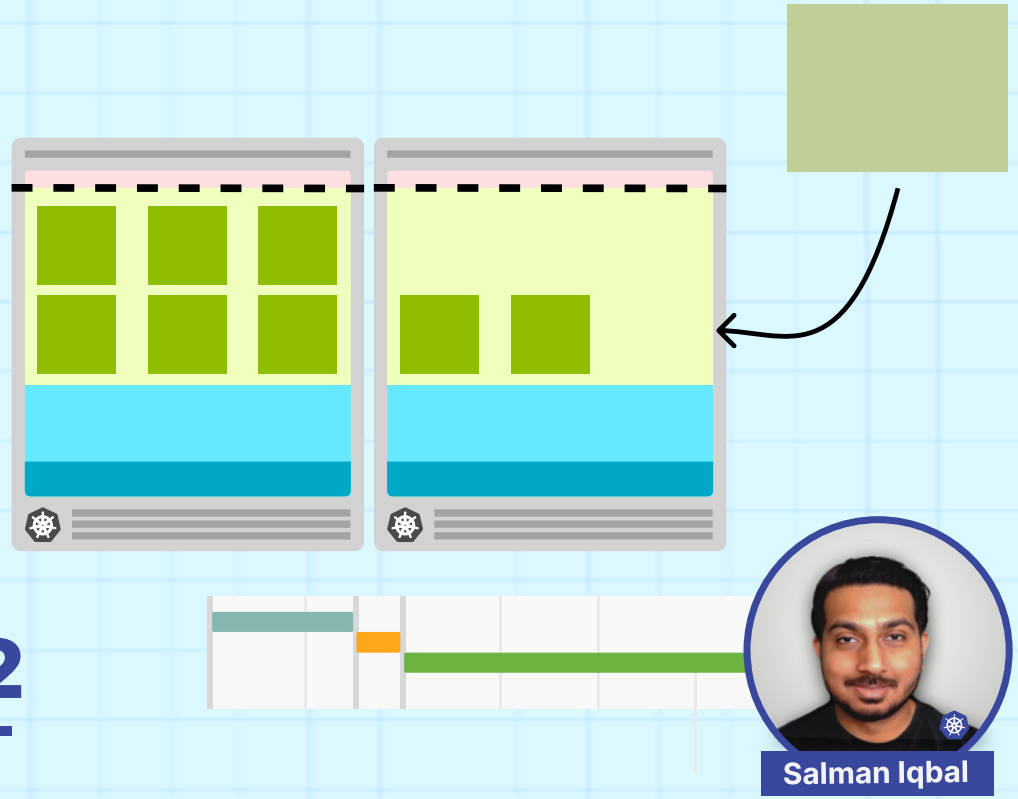
LIVE STREAMING

28th of Sep

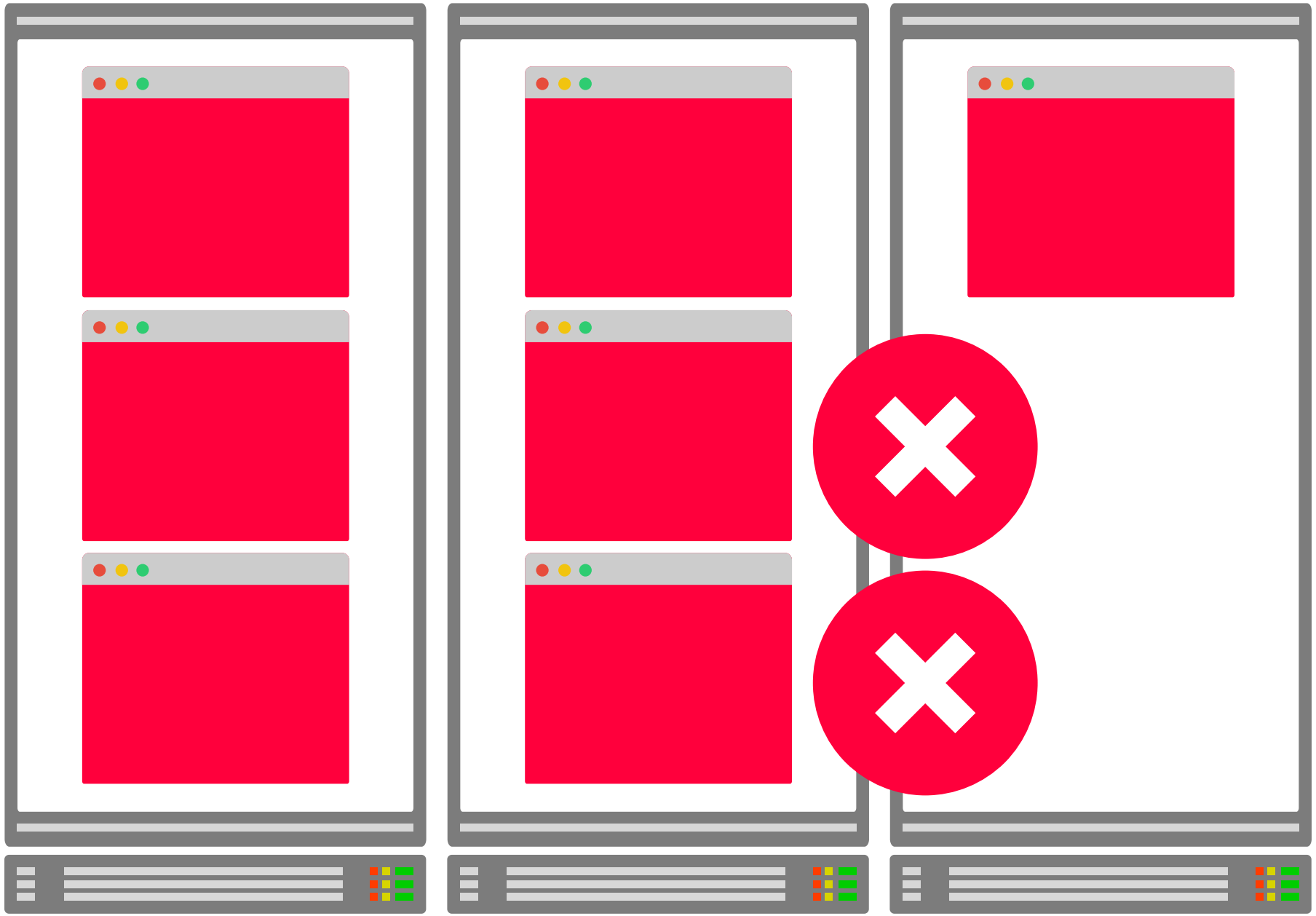
8am PT | 5pm CET

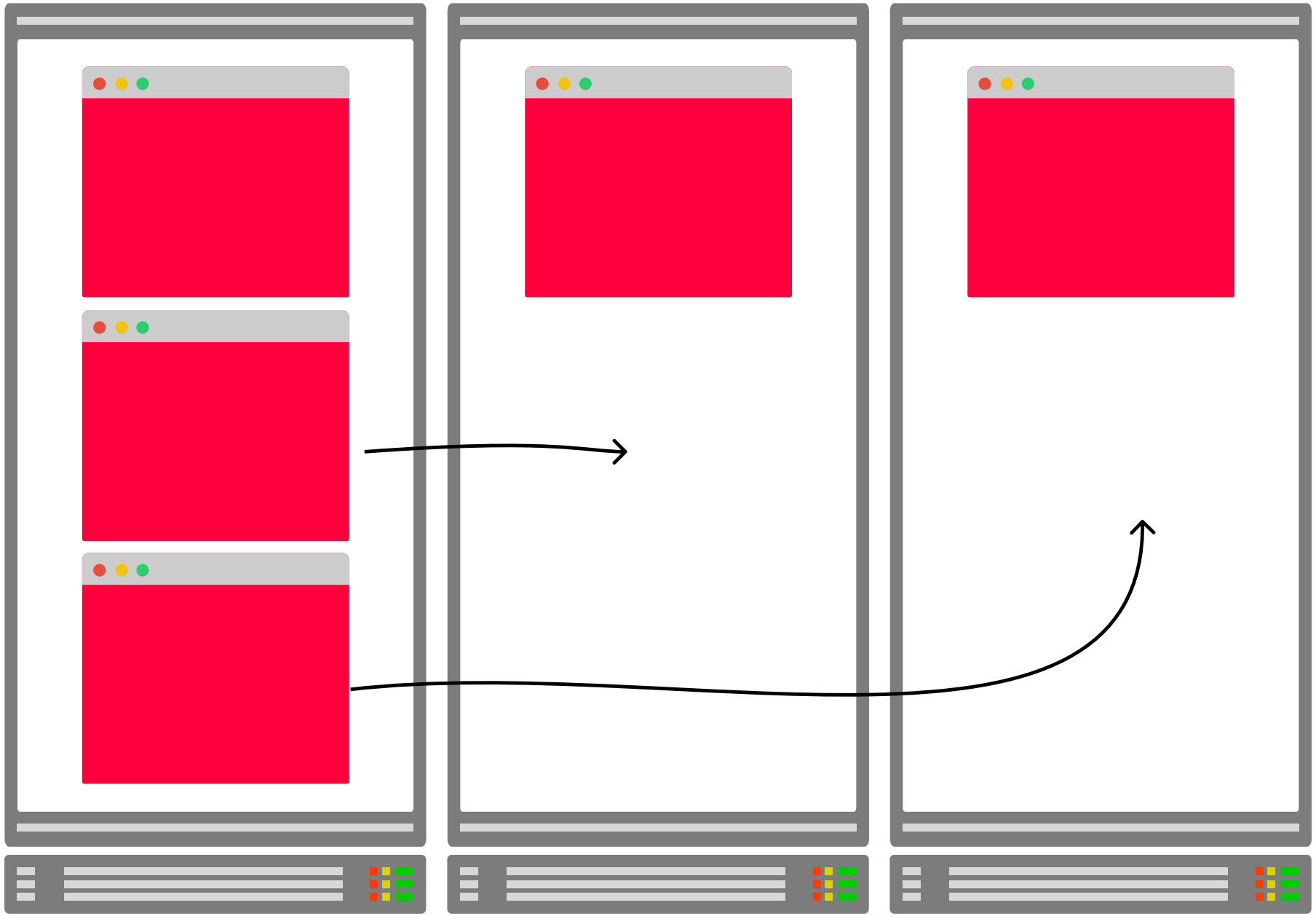
REGISTER HERE

bit.ly/k8s-optimize-2



Downscaling and fragmentation



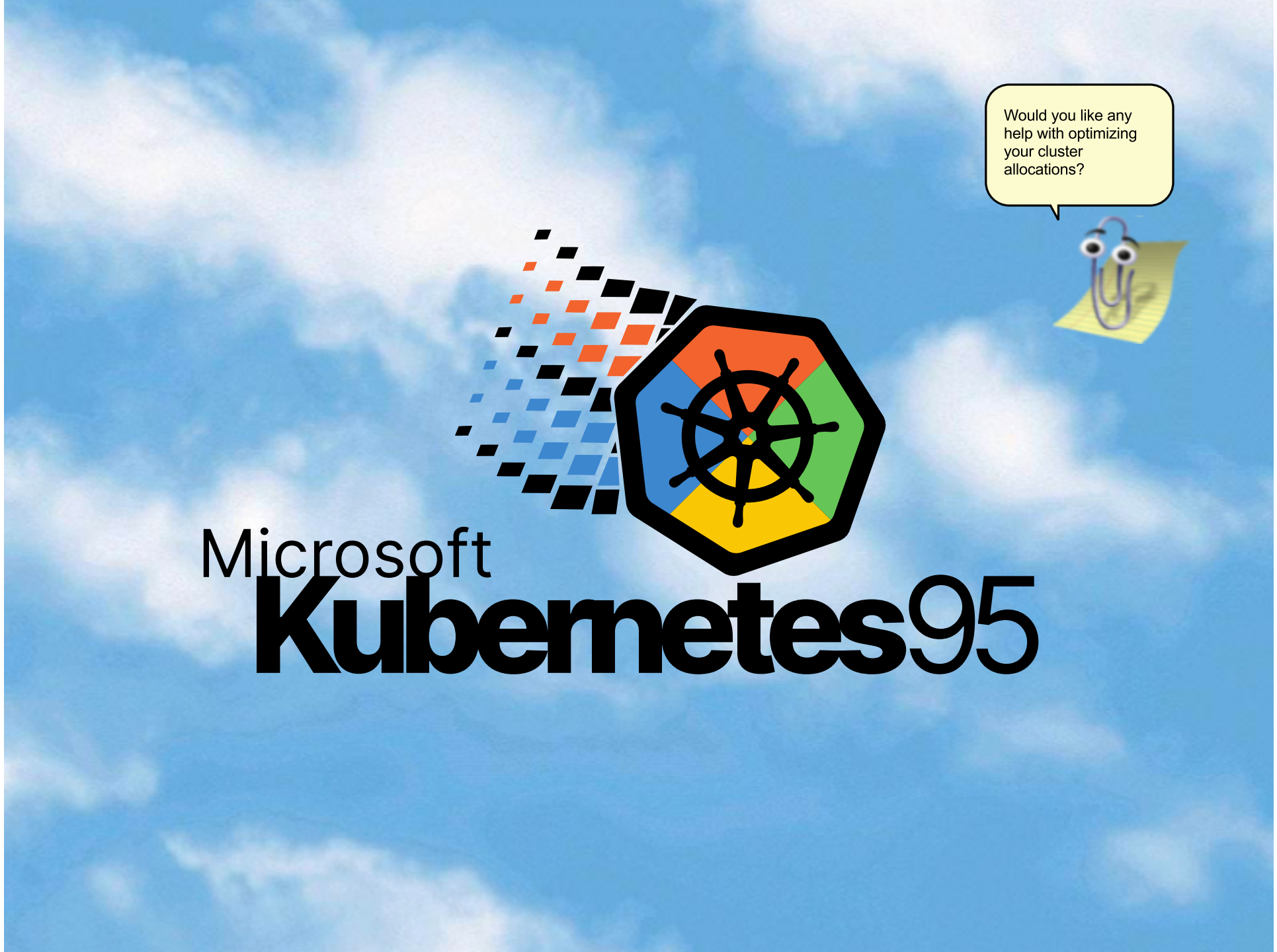


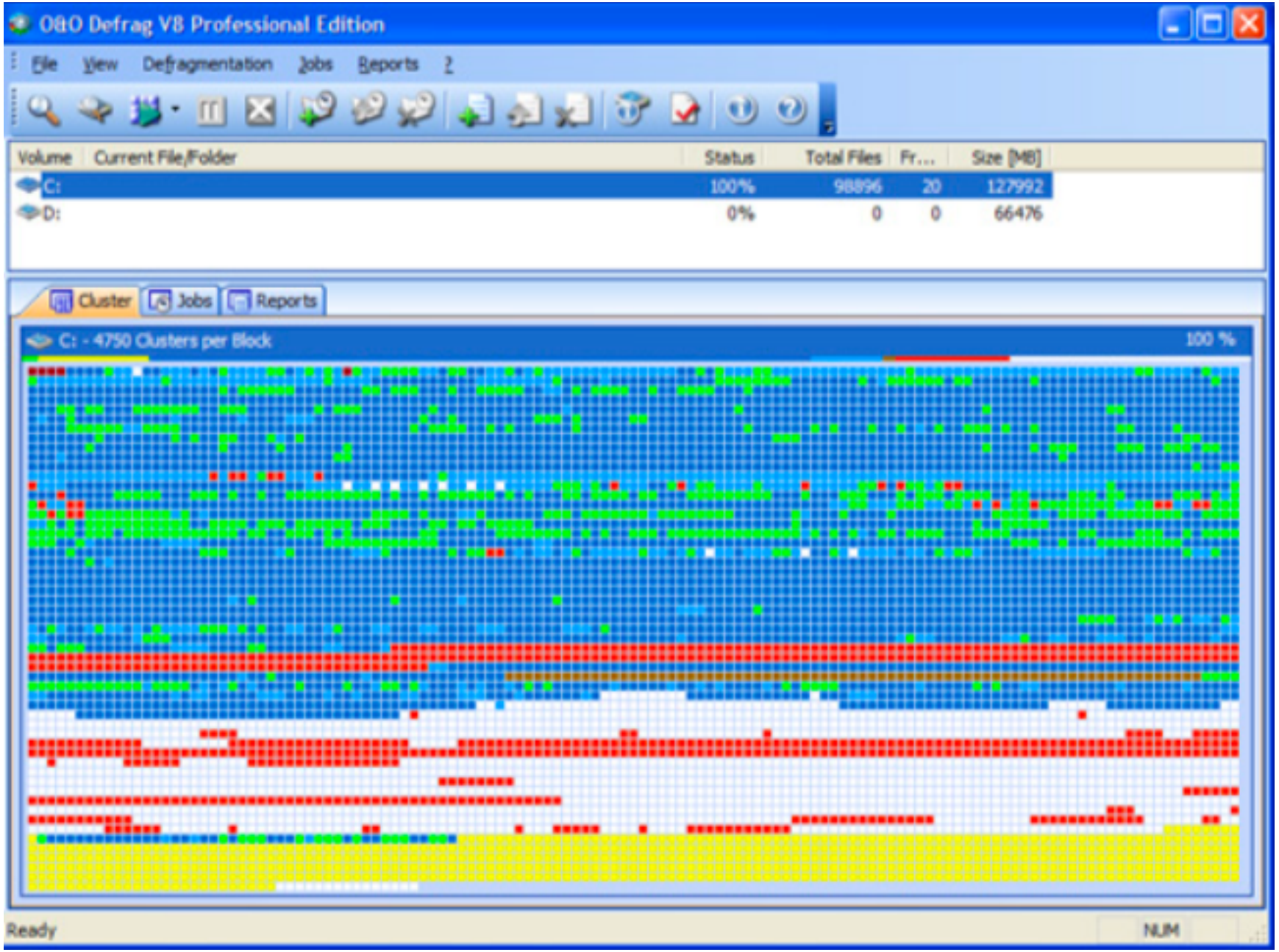
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 5
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
```

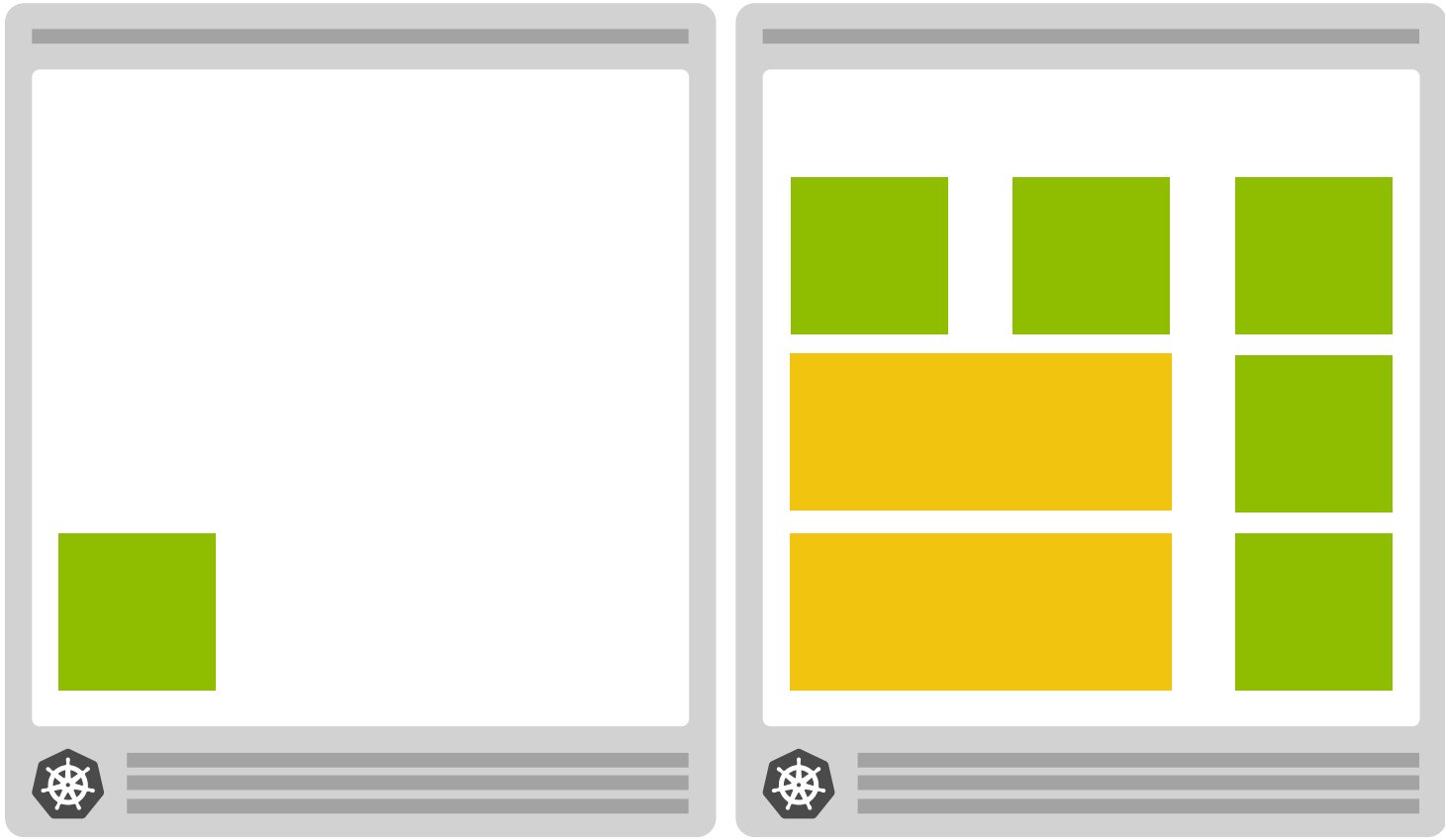
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 5
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:1.14.2
```

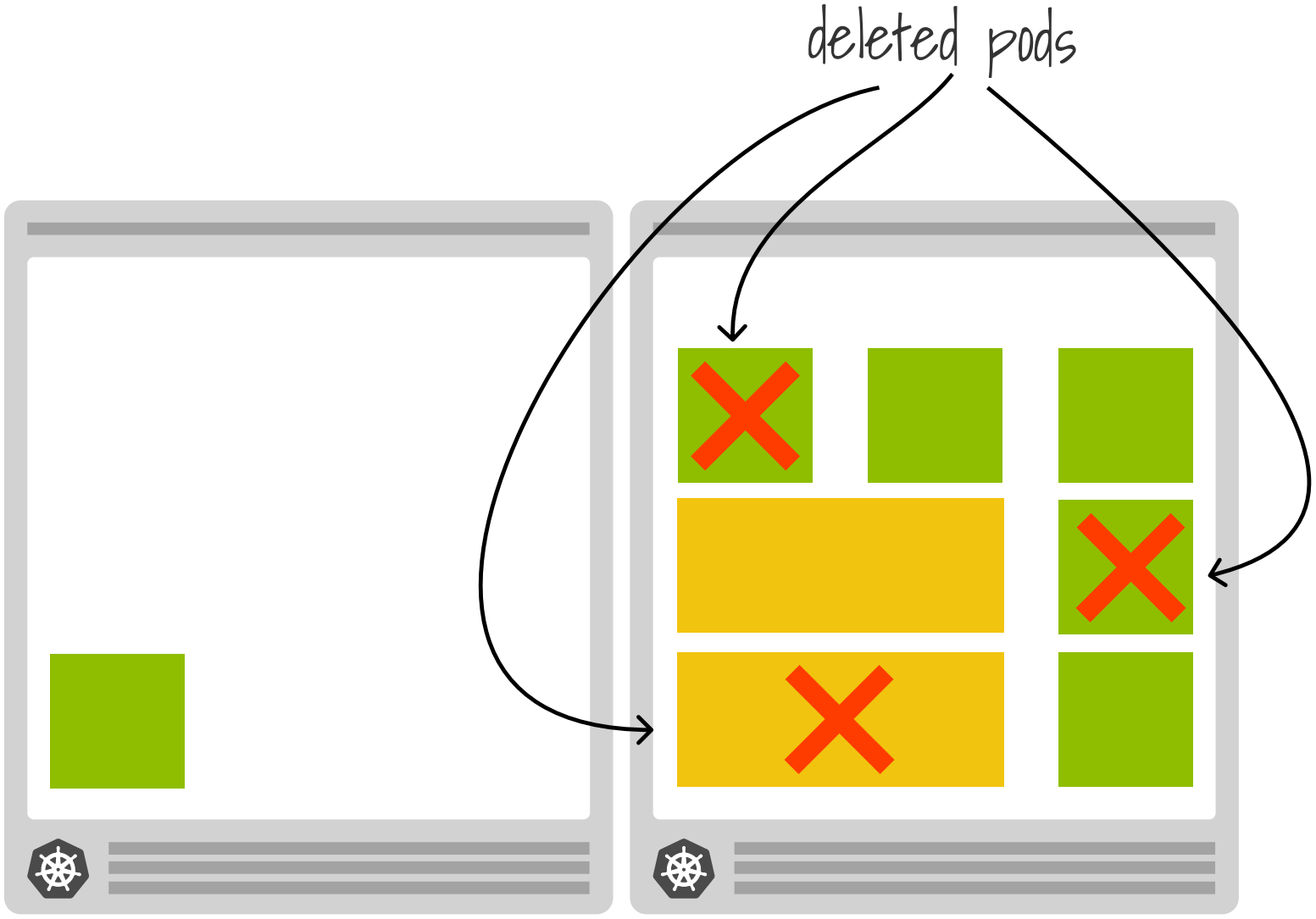
There's no field for "rebalance"!

Descheduler

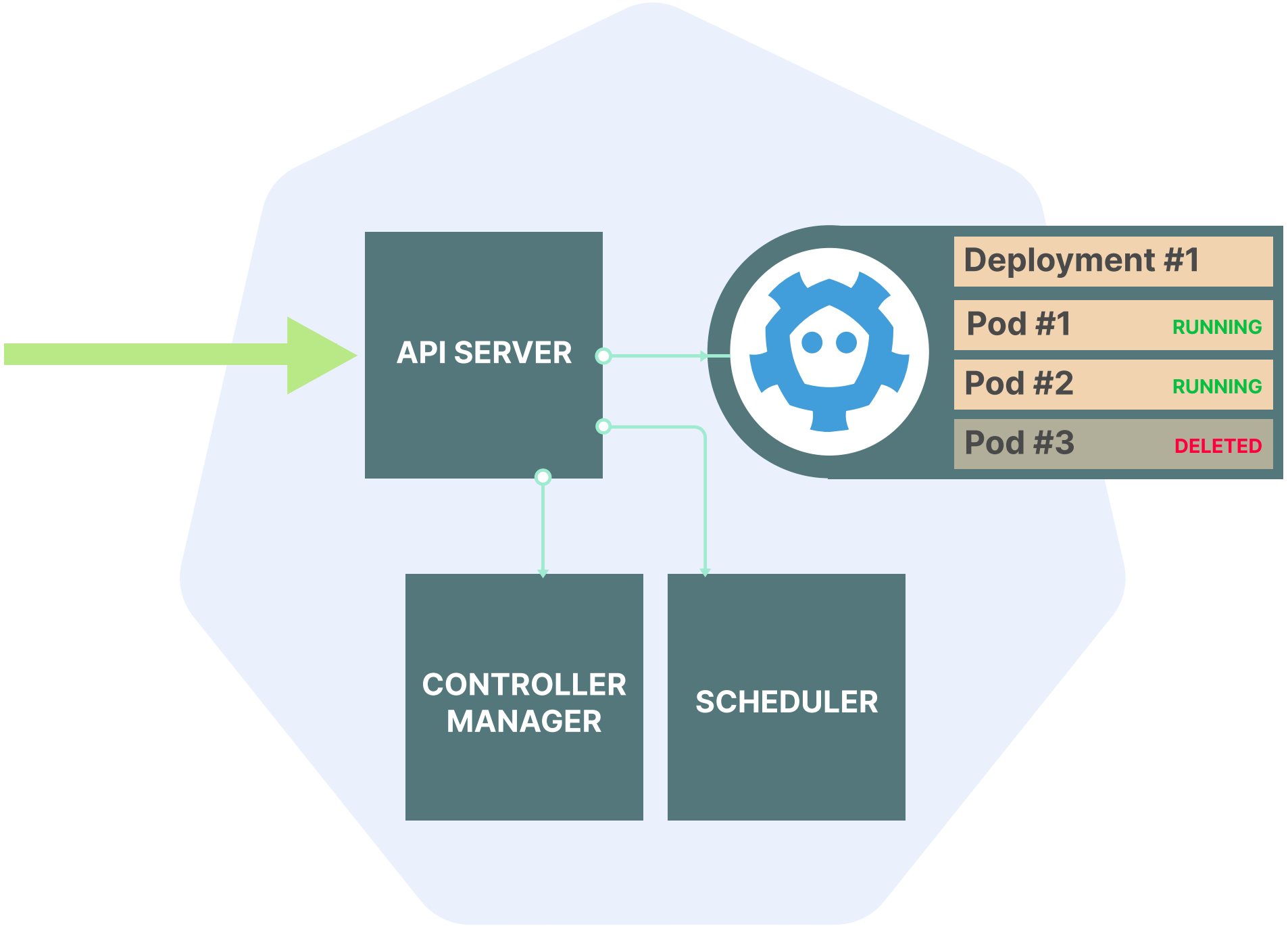

Would you like any help with optimizing your cluster allocations?
Microsoft
Kubernetes95

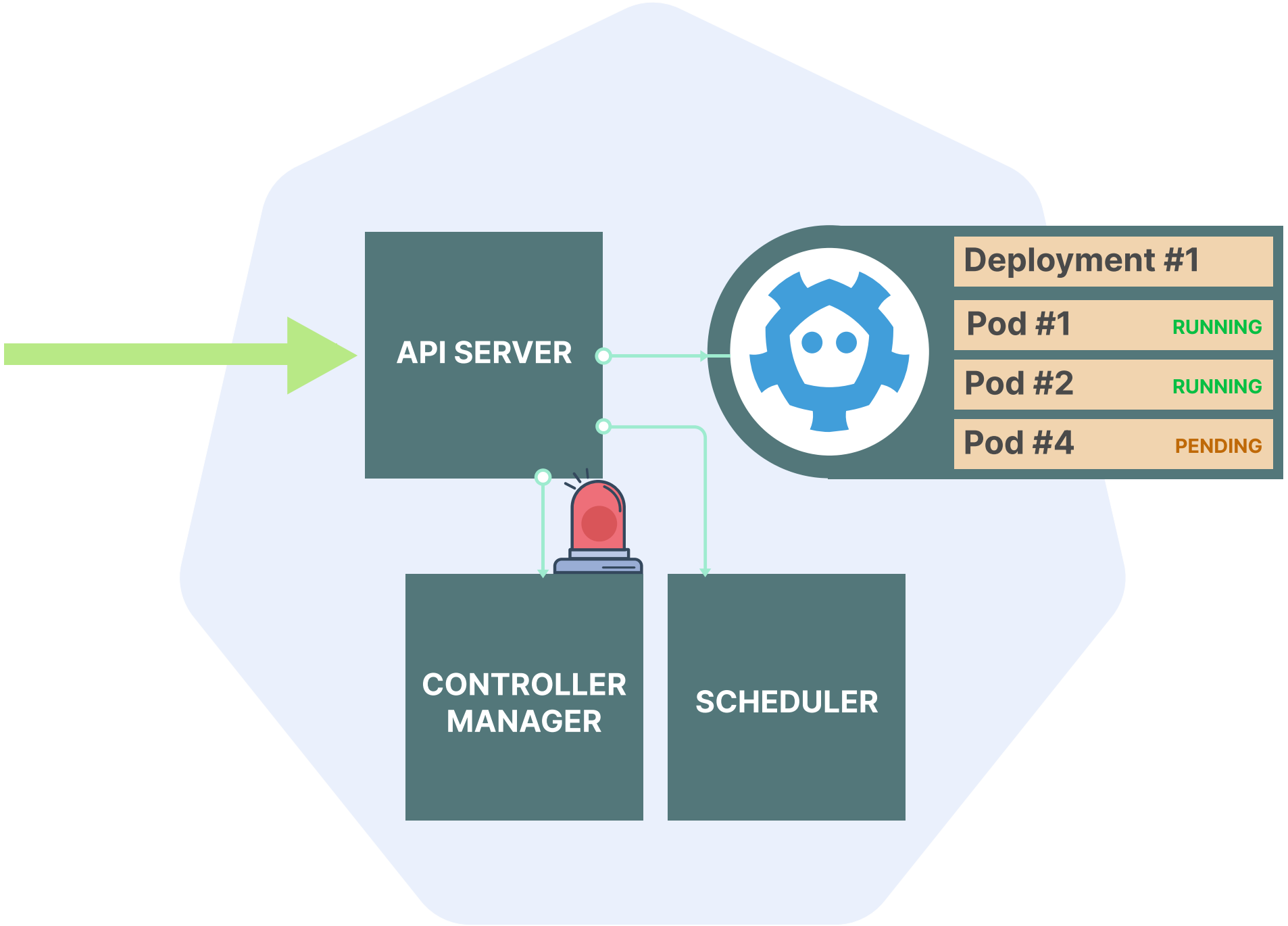

O&O Defrag V8 Professional Edition
File View Defragmentation Jobs Reports ?
Volume Current File/Folder Status Total Files Fr... Size [MB]
C: 100% 98896 20 127992
D: 0% 0 0 66476
Cluster Jobs Reports
C: - 4750 Clusters per Block 100 %
Ready
NUM

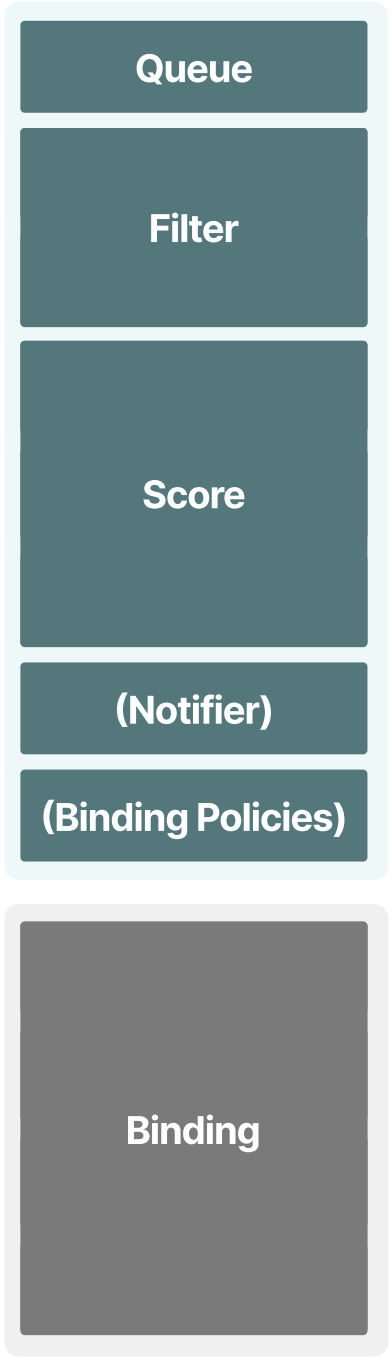



deleted pods

Kubernetes scheduler

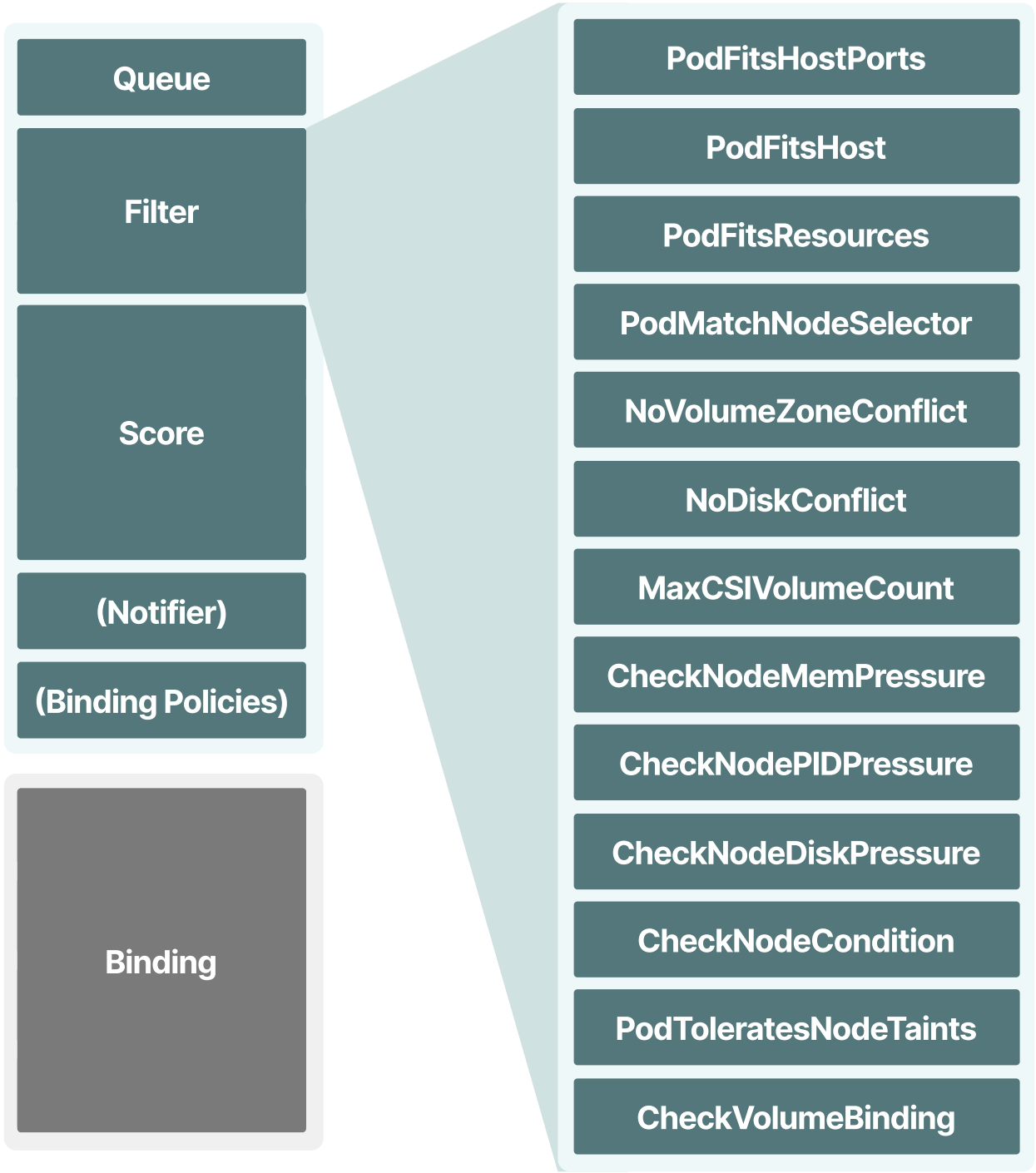

API SERVER
Deployment #1
Pod #1
RUNNING
Pod #2
RUNNING
Pod #3
DELETED
CONTROLLER MANAGER
SCHEDULER

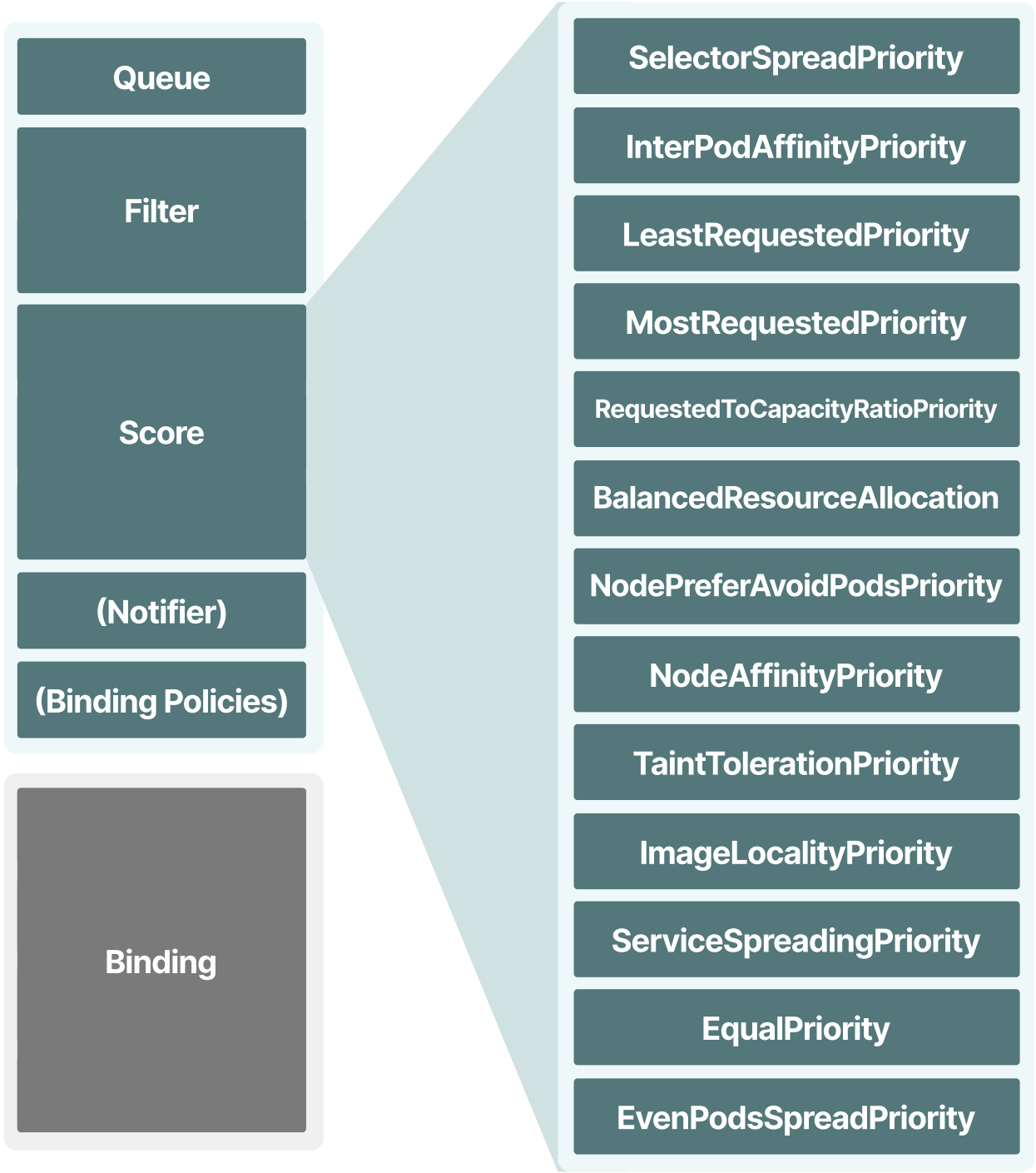

API SERVER
Deployment #1
Pod #1
RUNNING
Pod #2
RUNNING
Pod #4
PENDING
CONTROLLER MANAGER
SCHEDULER

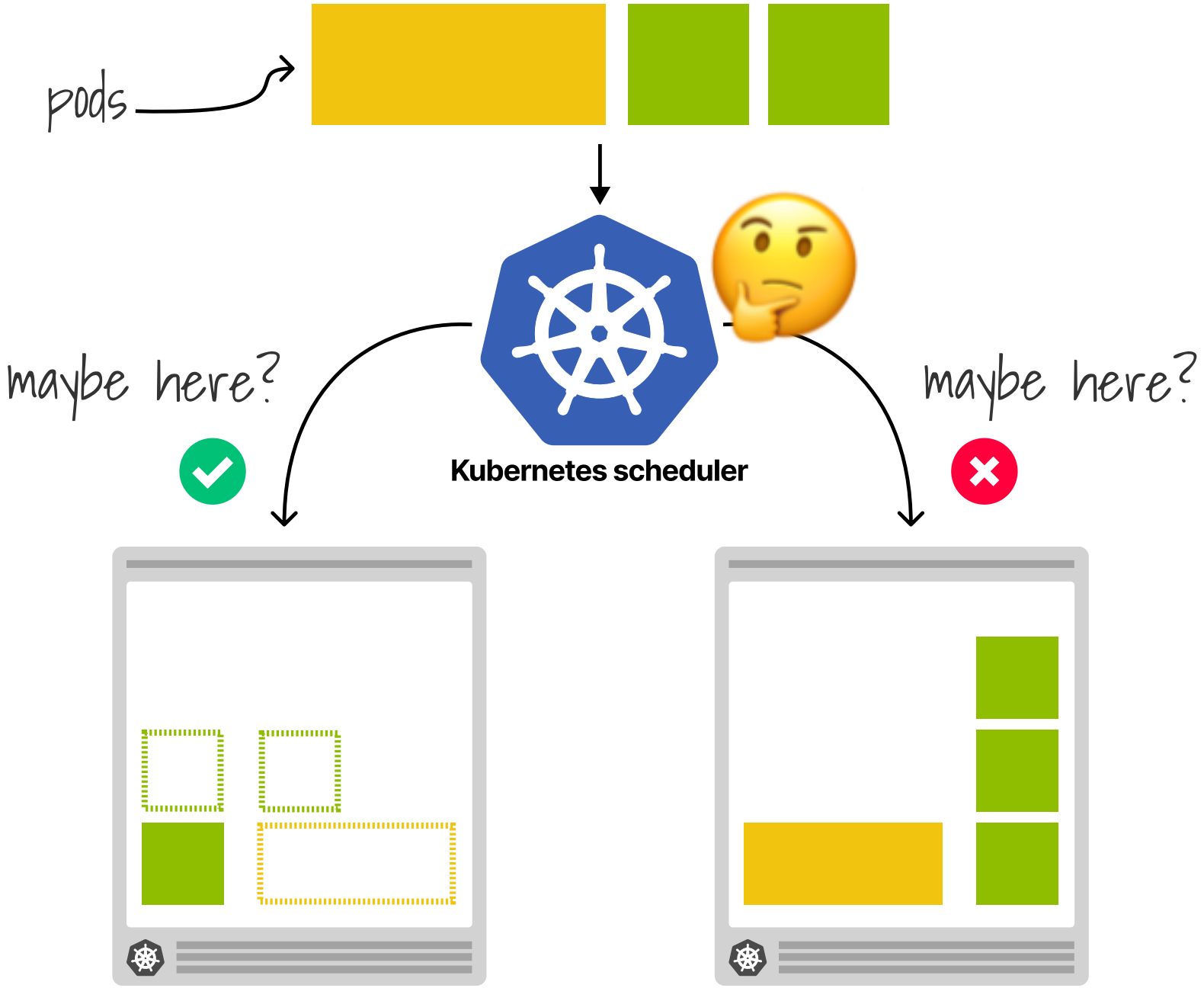


Scheduling

Binding


Queue
Filter
Score
(Notifier)
(Binding Policies)
Binding
PodFitsHostPorts
PodFitsHost
PodFitsResources
PodMatchNodeSelector
NoVolumeZoneConflict
NoDiskConflict
MaxCSIVolumeCount
CheckNodeMemPressure
CheckNodePIDPressure
CheckNodeDiskPressure
CheckNodeCondition
PodToleratesNodeTaints
CheckVolumeBinding


Queue
Filter
Score
(Notifier)
(Binding Policies)
Binding
SelectorSpreadPriority
InterPodAffinityPriority
LeastRequestedPriority
MostRequestedPriority
RequestedToCapacityRatioPriority
BalancedResourceAllocation
NodePreferAvoidPodsPriority
NodeAffinityPriority
TaintTolerationPriority
ImageLocalityPriority
ServiceSpreadingPriority
EqualPriority
EvenPodsSpreadPriority


pods
maybe here?
Kubernetes scheduler
maybe here?

Descheduler policies

Descheduler plugins

Name	Extension Point Implemented	Description
RemoveDuplicates	Balance	Spreads replicas
LowNodeUtilization	Balance	Spreads pods according to pods resource requests and node resources available
HighNodeUtilization	Balance	Spreads pods according to pods resource requests and node resources available
RemovePodsViolatingInterPodAntiAffinity	Deschedule	Evicts pods violating pod anti affinity
RemovePodsViolatingNodeAffinity	Deschedule	Evicts pods violating node affinity
RemovePodsViolatingNodeTaints	Deschedule	Evicts pods violating node taints
RemovePodsViolatingTopologySpreadConstraint	Balance	Evicts pods violating TopologySpreadConstraints
RemovePodsHavingTooManyRestarts	Deschedule	Evicts pods having too many restarts
PodLifeTime	Deschedule	Evicts pods that have exceeded a specified age limit
RemoveFailedPods	Deschedule	Evicts pods with certain failed reasons

```yaml
apiVersion: "descheduler/v1alpha2"
kind: "DeschedulerPolicy"
profiles:
  - name: ProfileName
    pluginConfig:
    - name: "RemoveDuplicates"
    - name: "RemovePodsHavingTooManyRestarts"
      args:
        podRestartThreshold: 100
        includingInitContainers: true
    plugins:
      deschedule:
        enabled:
          - "RemovePodsHavingTooManyRestarts"
      balance:
        enabled:
          - "RemoveDuplicates"
```

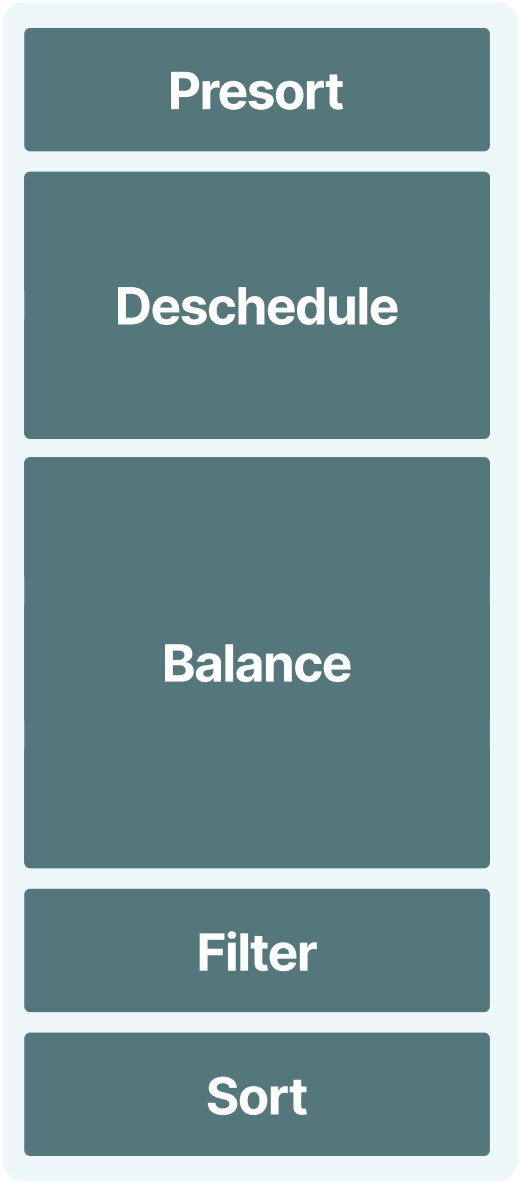
```yaml
apiVersion: "descheduler/v1alpha2"
kind: "DeschedulerPolicy"
profiles:
  - name: ProfileName
    pluginConfig:
    - name: "RemoveDuplicates"
    - name: "RemovePodsHavingTooManyRestarts"
      args:
        podRestartThreshold: 100
        includingInitContainers: true
    plugins:
      deschedule:
        enabled:
          - "RemovePodsHavingTooManyRestarts"
      balance:
        enabled:
          - "RemoveDuplicates"
```

Plugin configs

```yaml
apiVersion: "descheduler/v1alpha2"
kind: "DeschedulerPolicy"
profiles:
  - name: ProfileName
    pluginConfig:
    - name: "RemoveDuplicates"
    - name: "RemovePodsHavingTooManyRestarts"
      args:
        podRestartThreshold: 100
        includingInitContainers: true
    plugins:
      deschedule:
        enabled:
          - "RemovePodsHavingTooManyRestarts"
      balance:
        enabled:
          - "RemoveDuplicates"
```

plugins

```yaml
apiVersion: "descheduler/v1alpha2"
kind: "DeschedulerPolicy"
profiles:
  - name: ProfileName
    pluginConfig:
    - name: "RemoveDuplicates"
    - name: "RemovePodsHavingTooManyRestarts"
      args:
        podRestartThreshold: 100
        includingInitContainers: true
    plugins:
      deschedule:
        enabled:
          - "RemovePodsHavingTooManyRestarts"
      balance:
        enabled:
          - "RemoveDuplicates"
```



Extension points

Restart policy

```yaml
apiVersion: "descheduler/v1alpha2"
kind: "DeschedulerPolicy"
profiles:
  - name: ProfileName
    pluginConfig:
    - name: "PodLifeTime"
      args:
        maxPodLifeTimeSeconds: 10
    plugins:
      deschedule:
        enabled:
          - "PodLifeTime"
```

pods older than 10 seconds are removed

Demo

Descheduler deployment

Descheduler deployment

1. Job
2. CronJob
3. Deployment

Descheduler deployment

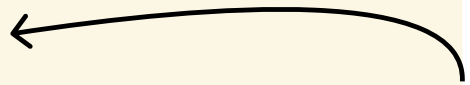
1. Job
2. CronJob
3. Deployment

Descheduler deployment

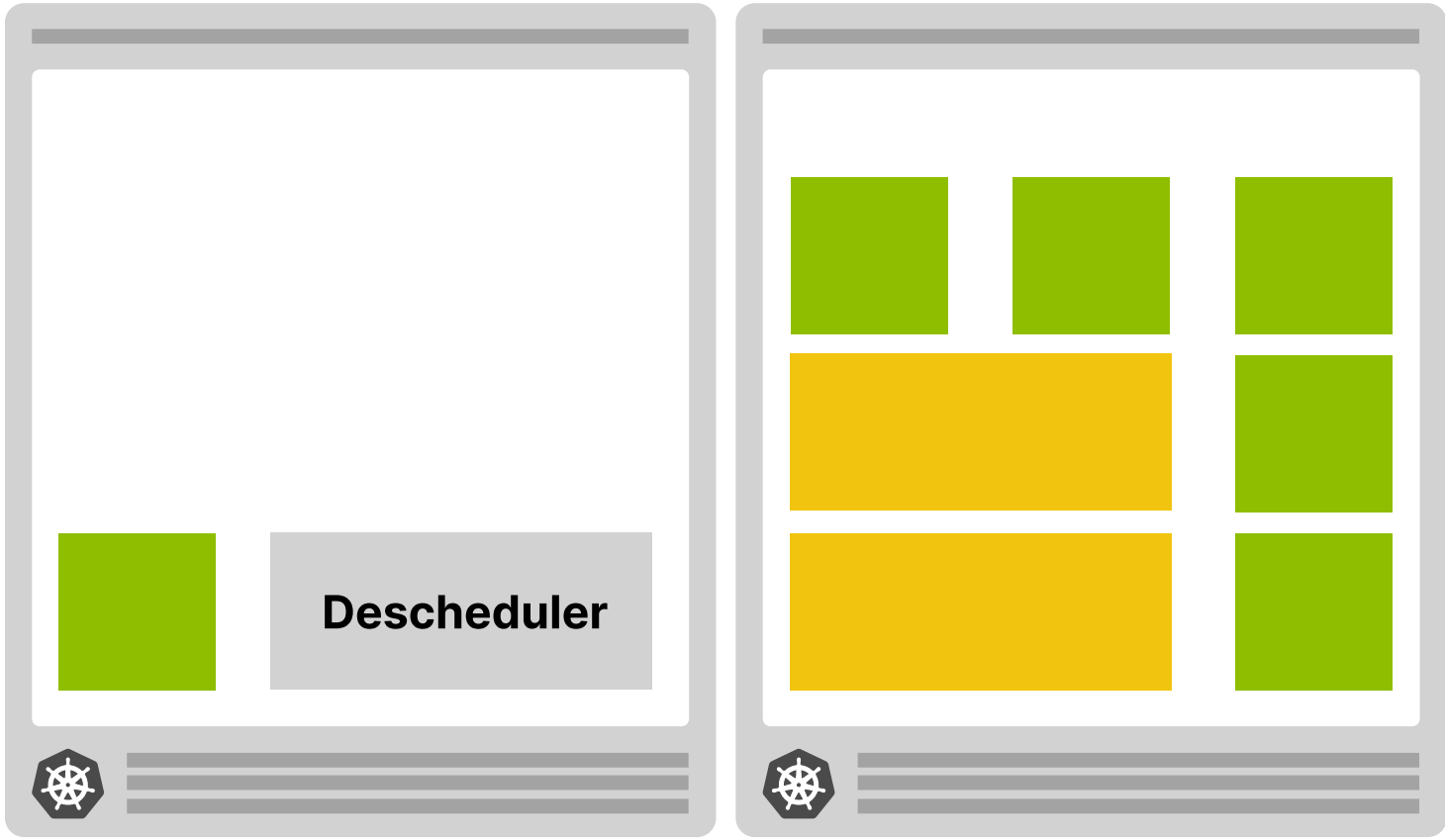
1. Job
2. CronJob
3. **Deployment**

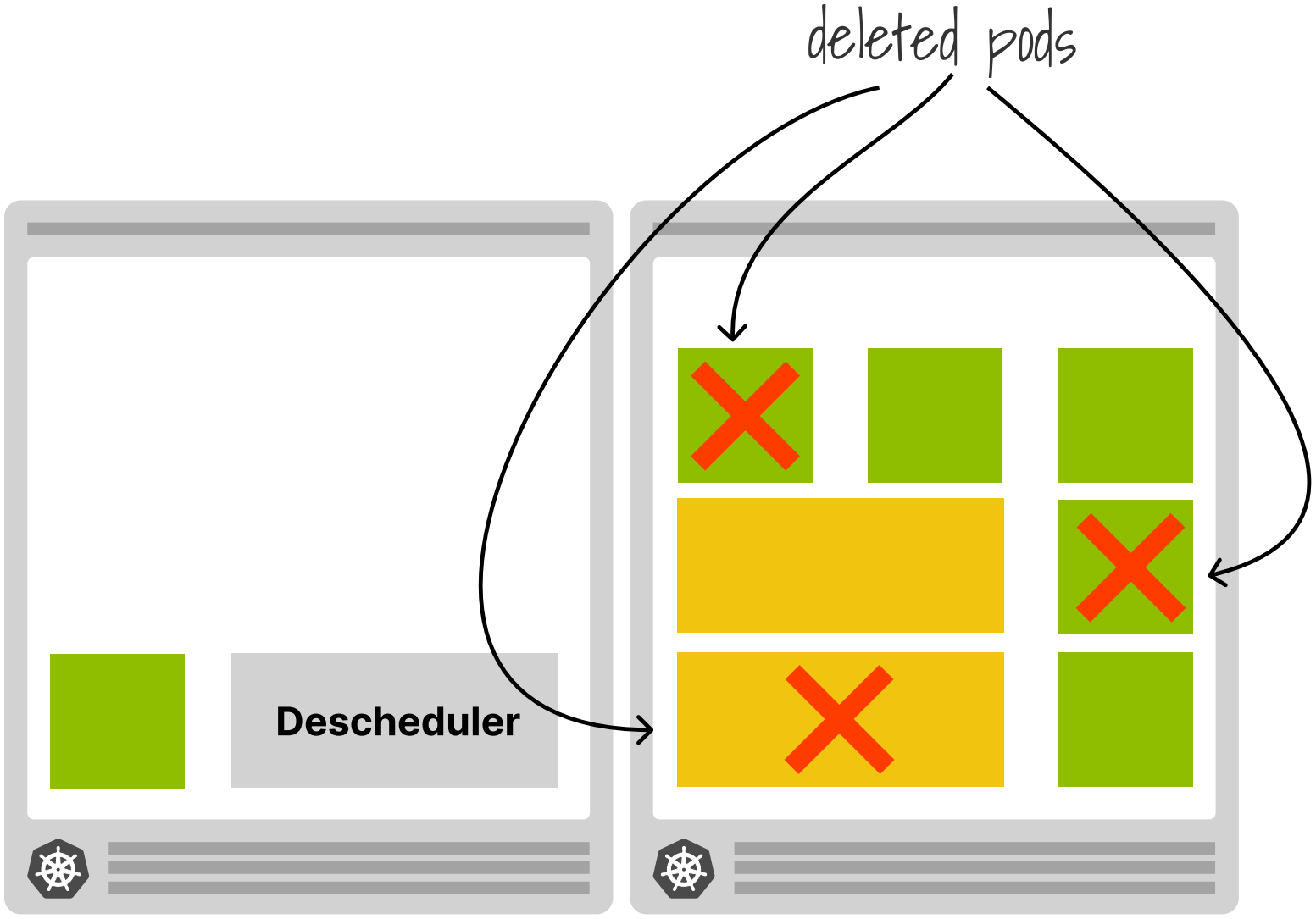
CronJob

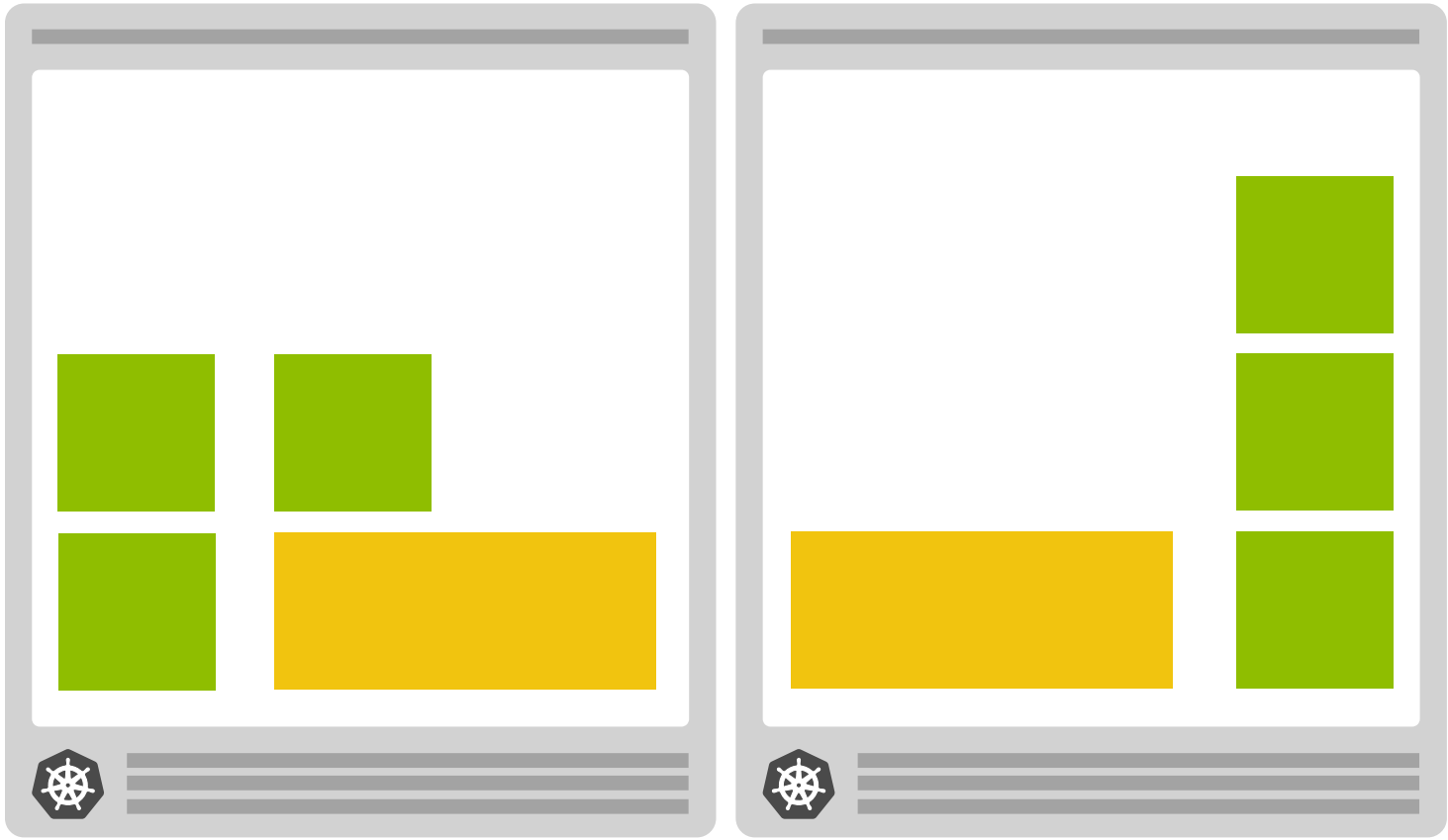
```yaml
apiVersion: batch/v1
kind: CronJob
metadata:
  name: descheduler-cronjob
  namespace: kube-system
spec:
  schedule: "*/1 * * * *"
  concurrencyPolicy: "Forbid"
  jobTemplate:
    spec:
      template:
        metadata:
          name: descheduler-pod
        spec:
          containers:
            - name: descheduler
              image: registry.k8s.io/desch…
```



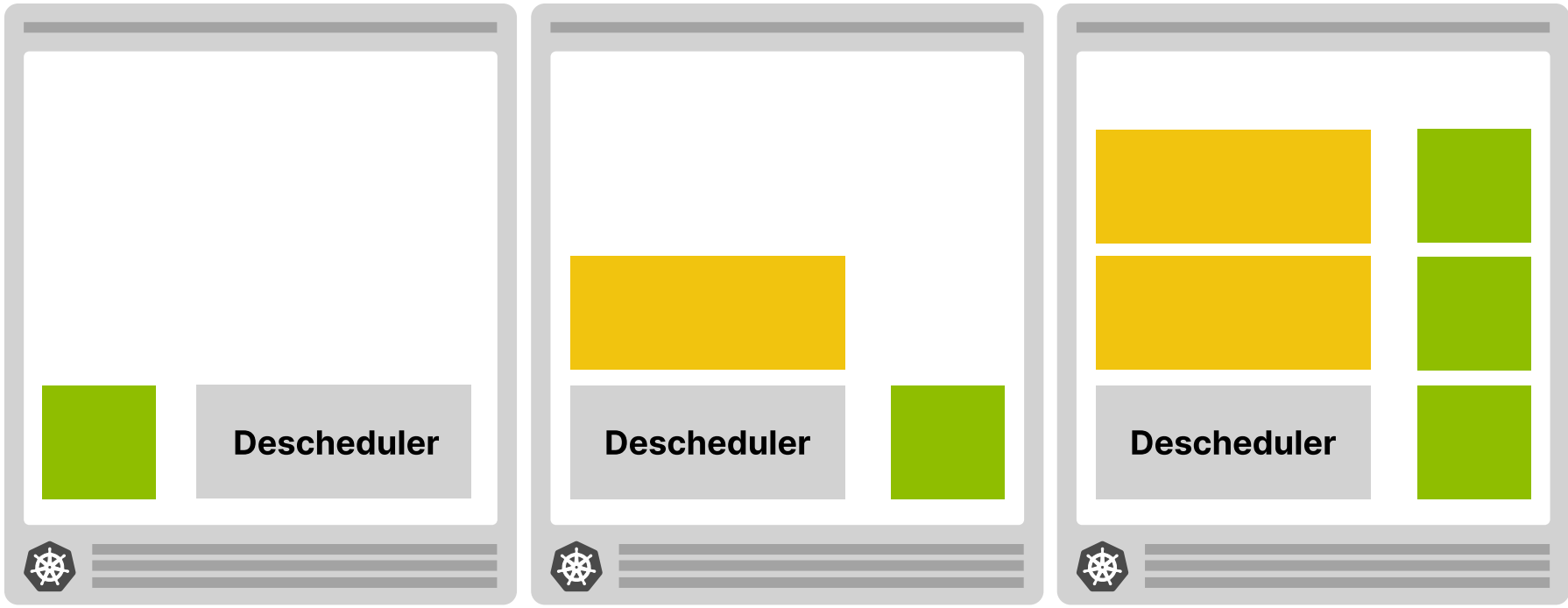
Frequency

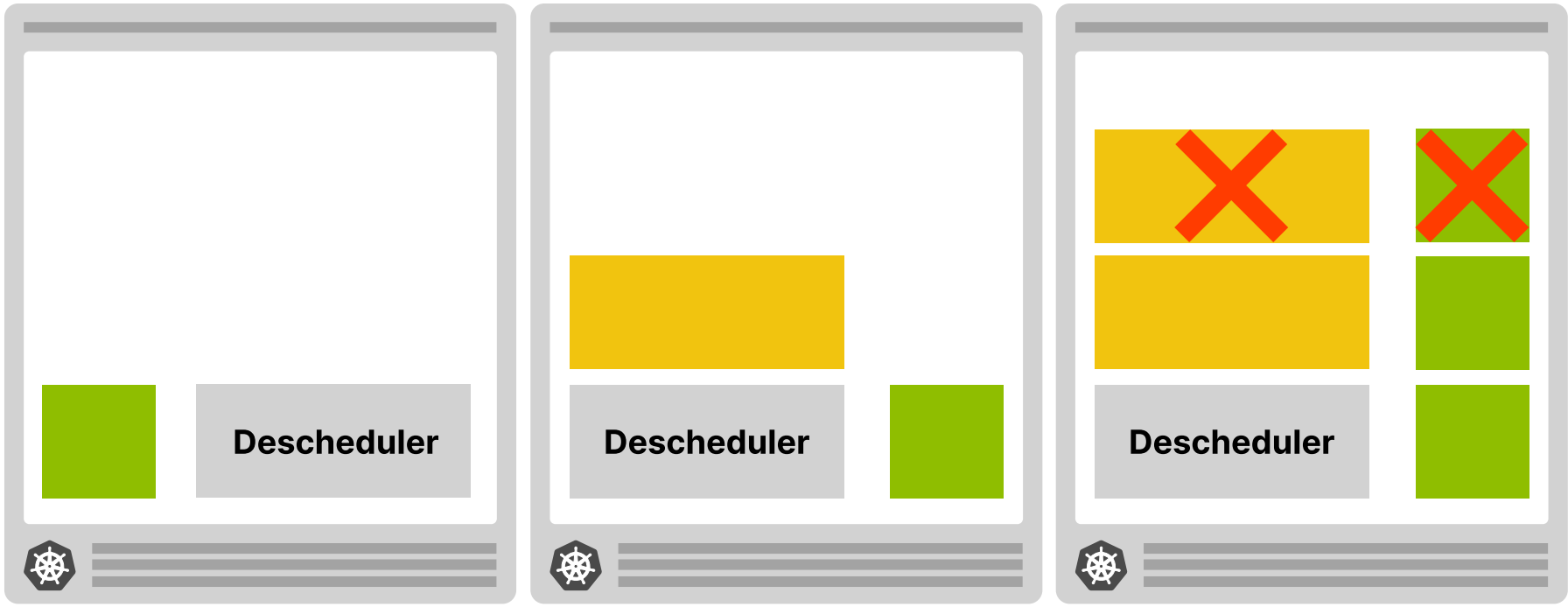

Descheduler

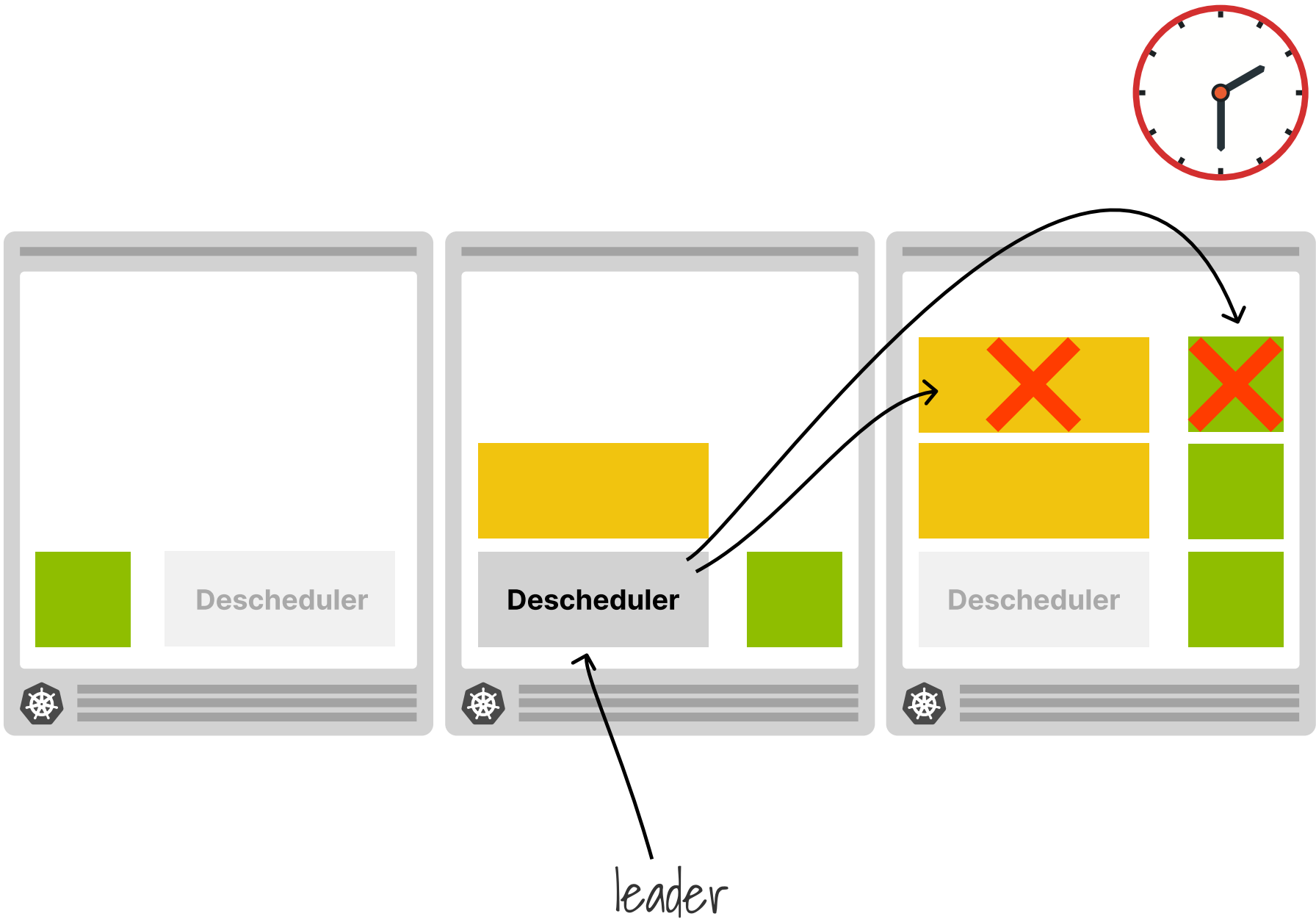

deleted pods
Descheduler



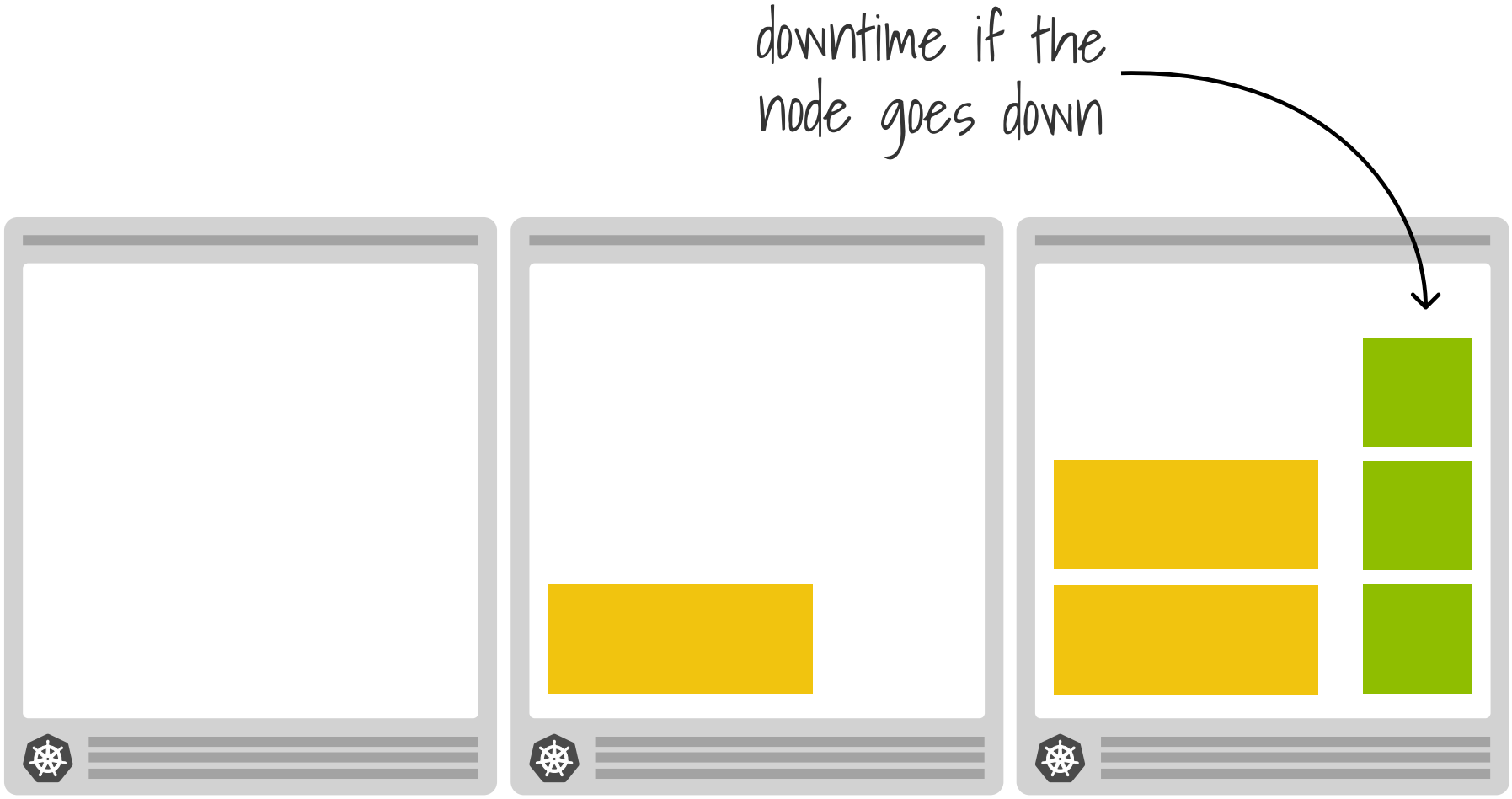
Deployment

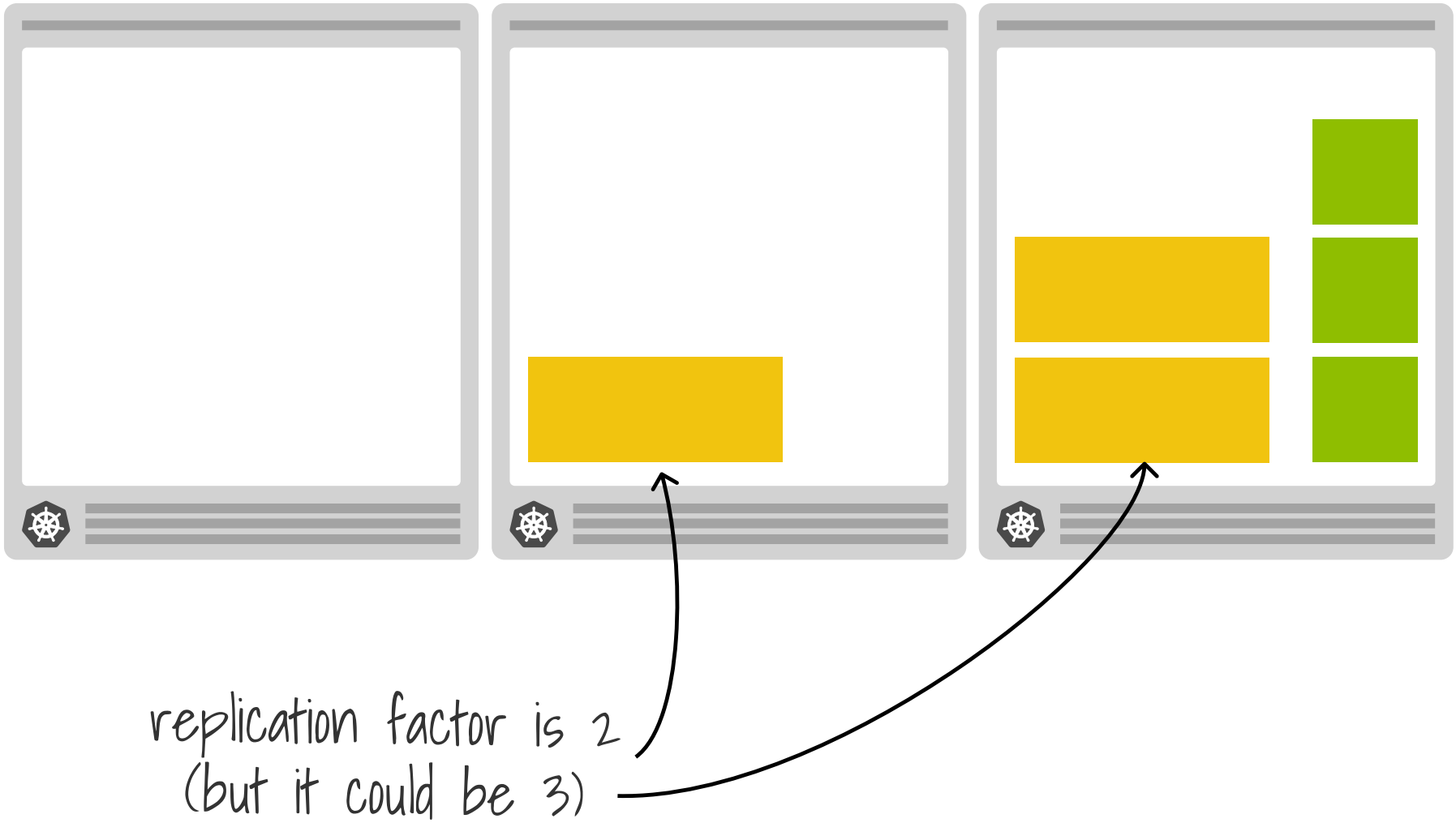

Descheduler
Descheduler
Descheduler


Descheduler
Descheduler
Descheduler

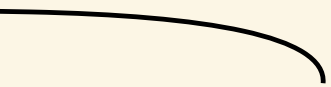

Descheduler
Descheduler
Descheduler
leader

Duplicate policy

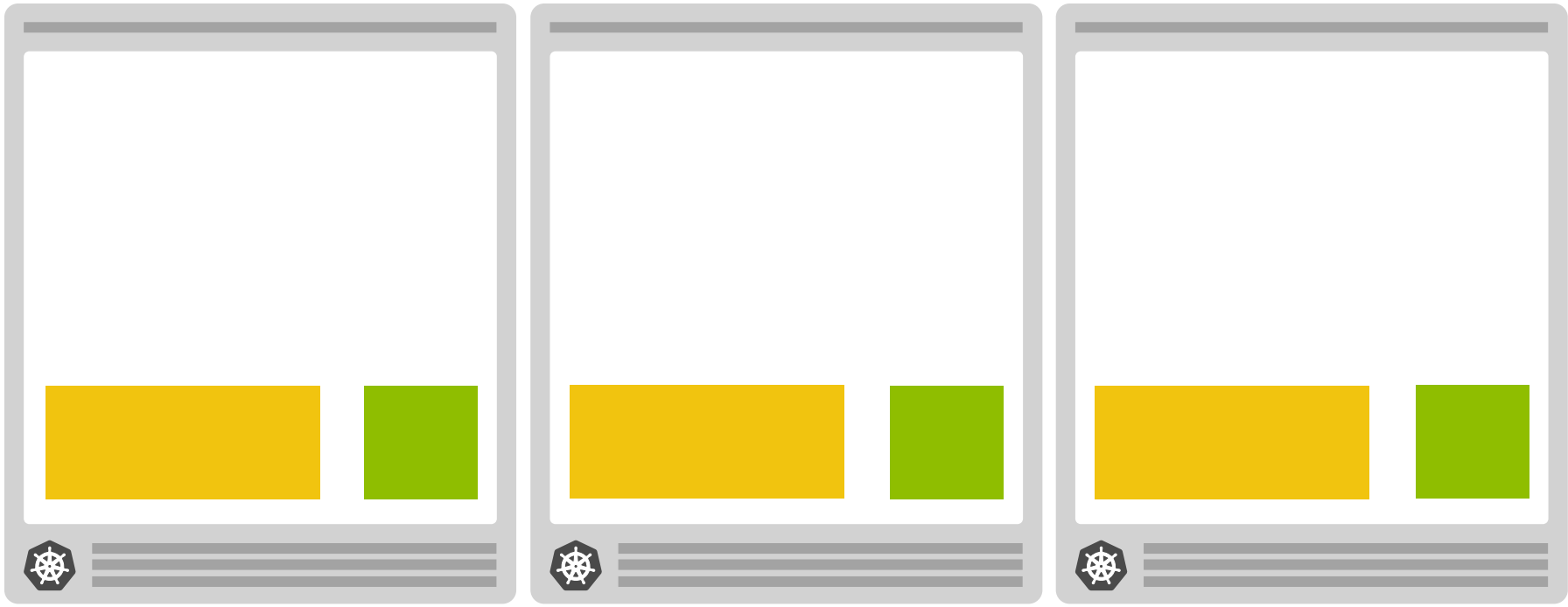

downtime if the
node goes down


replication factor is 2
(but it could be 3)

```yaml
apiVersion: "descheduler/v1alpha1"
kind: "DeschedulerPolicy"
profiles:
  - name: ProfileName
    pluginConfig:
    - name: "RemoveDuplicates"
  plugins:
    balance:
      enabled:
        - "RemoveDuplicates"
```

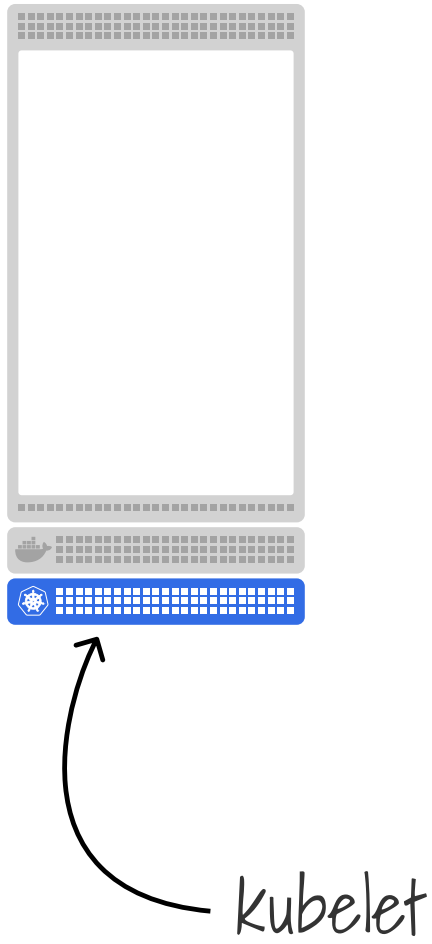


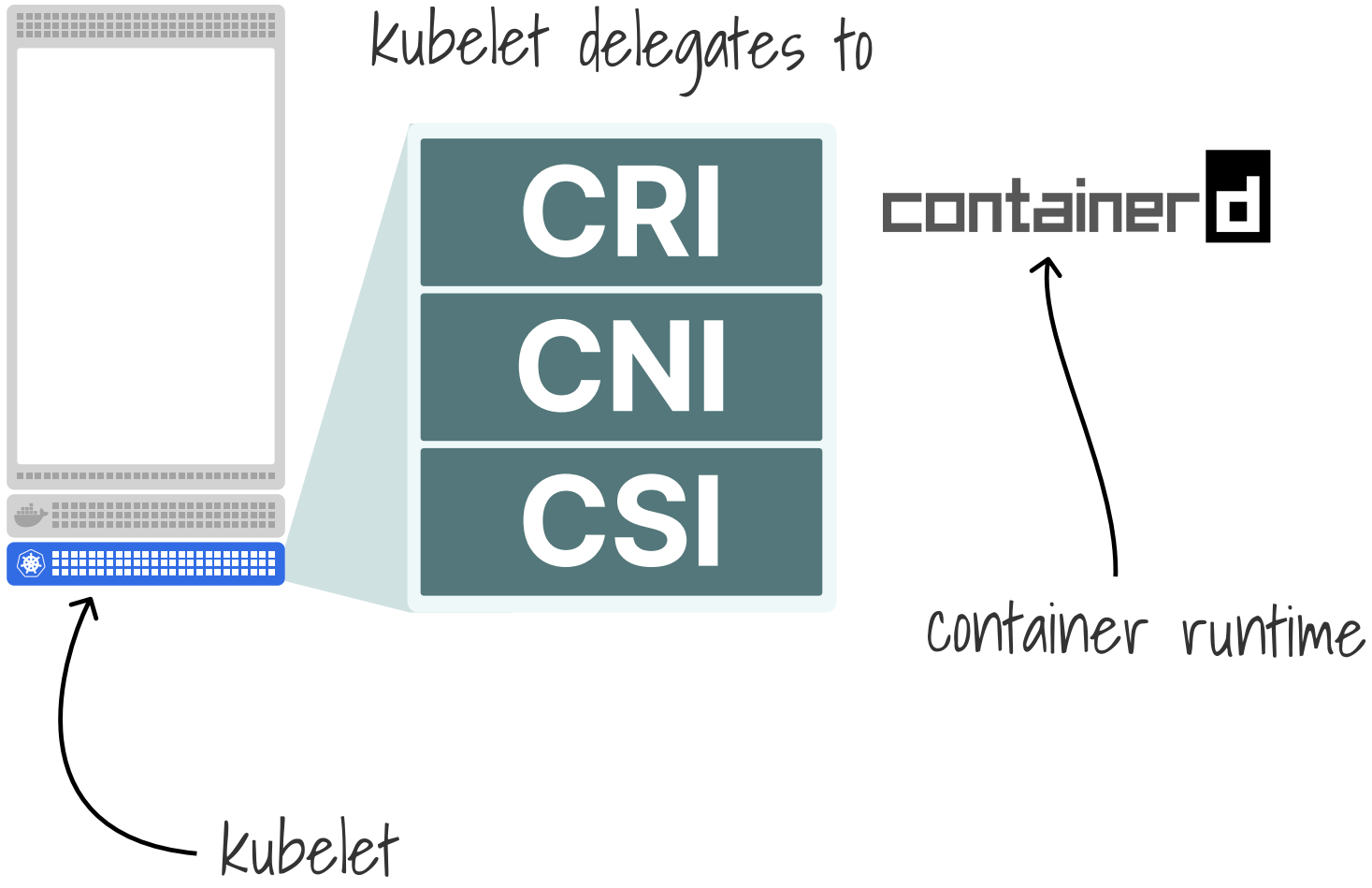
consider more than one pod at once

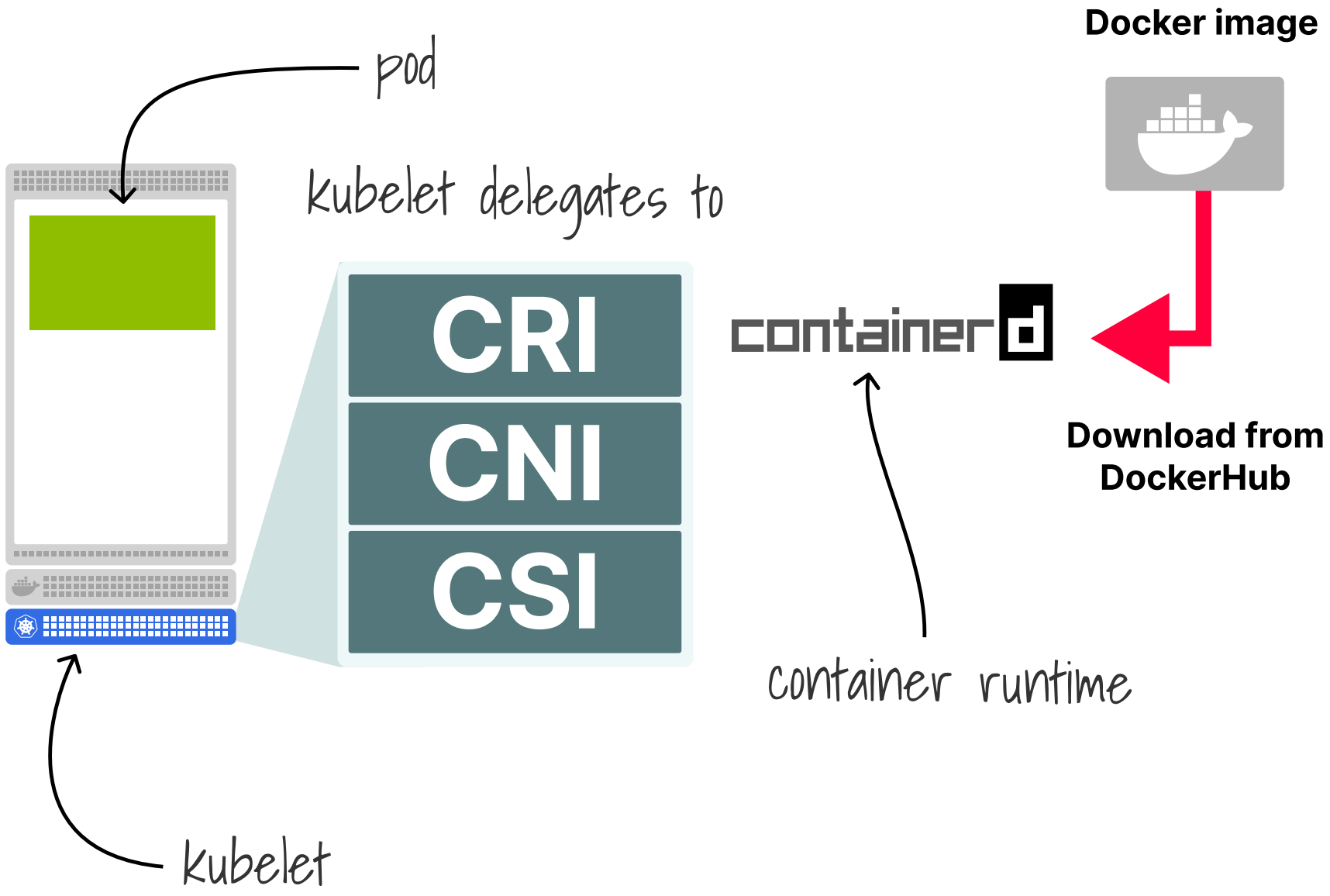


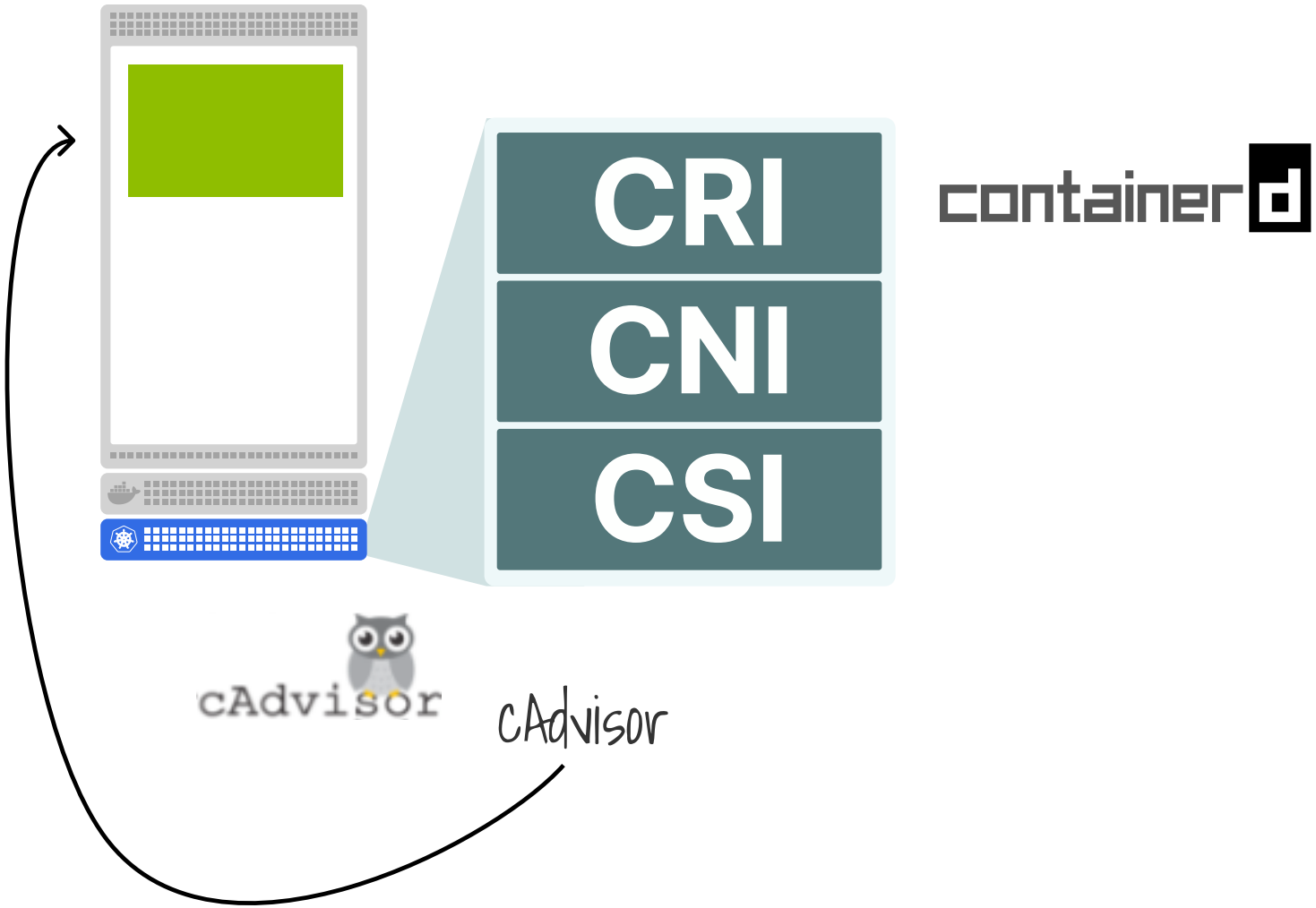
Demo

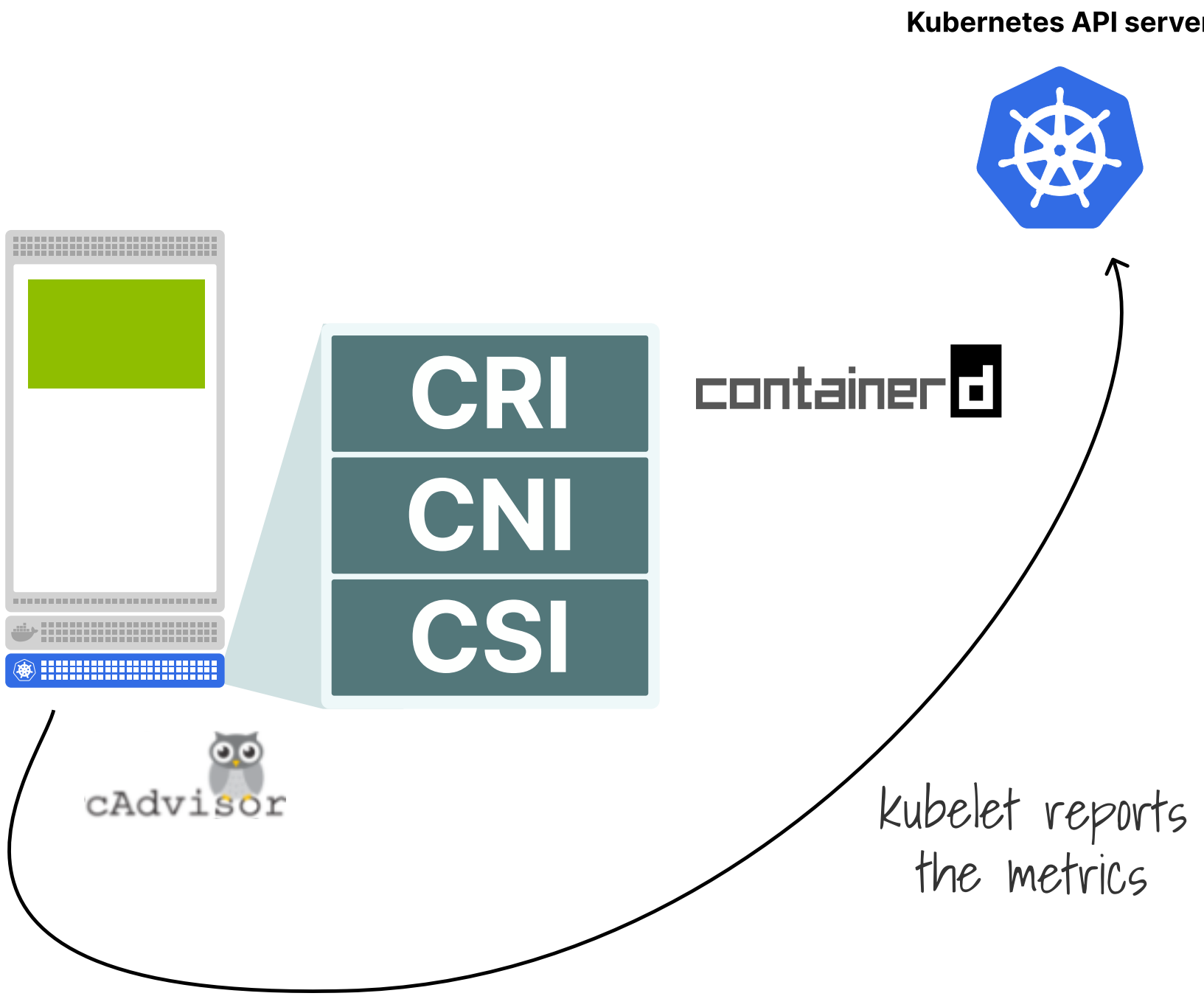
Collecting metrics


kubelet


Kubelet delegates to
CRI
CNI
CSI
containerd
container runtime
kubelet

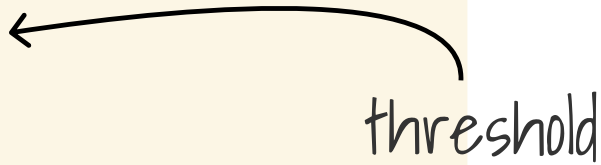

Docker image
pod
Kubelet delegates to
CRI
CNI
CSI
containerd
Download from DockerHub
container runtime
kubelet

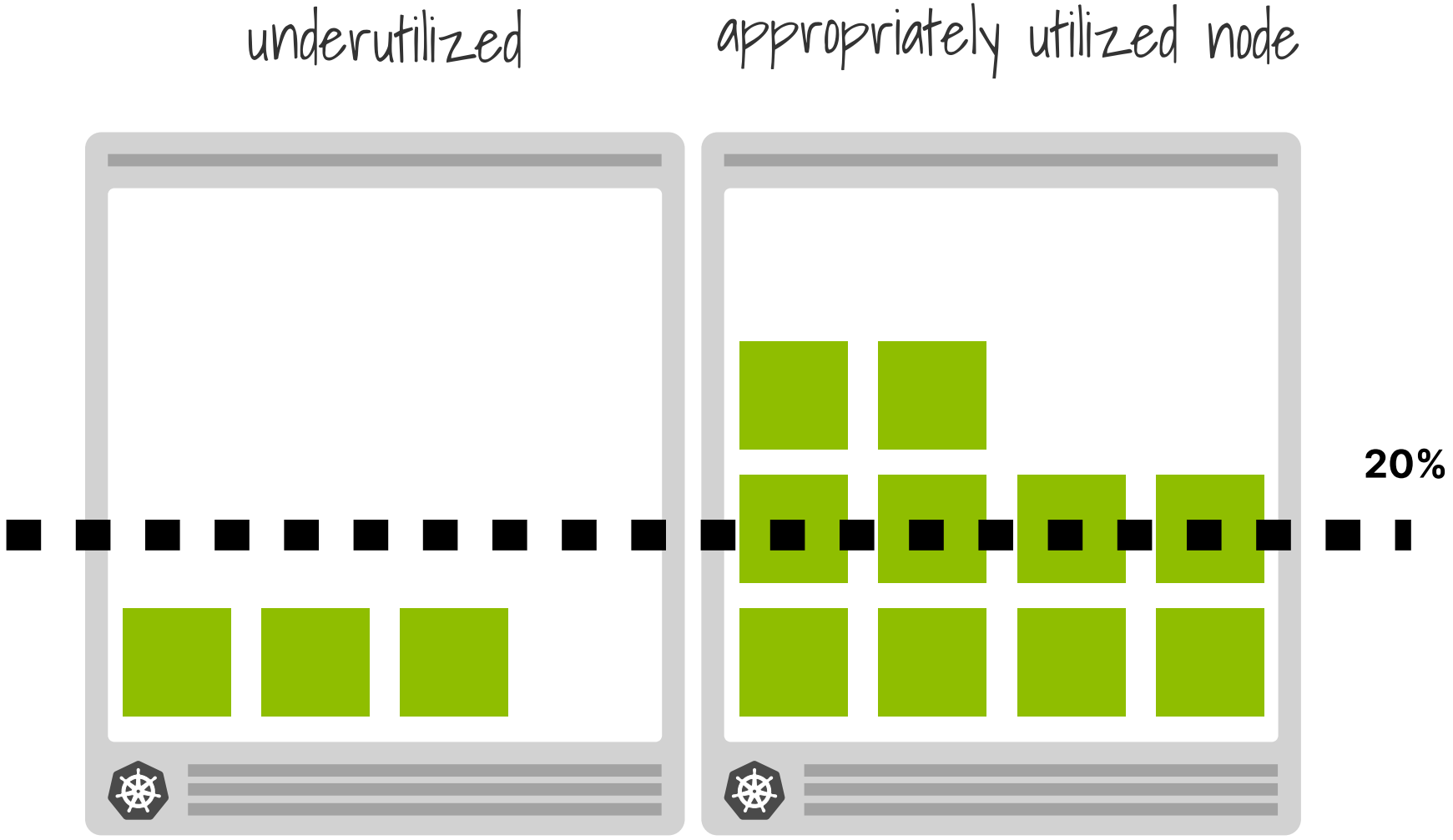

CRI
CNI
CSI
containerd
cAdvisor
cAdvisor

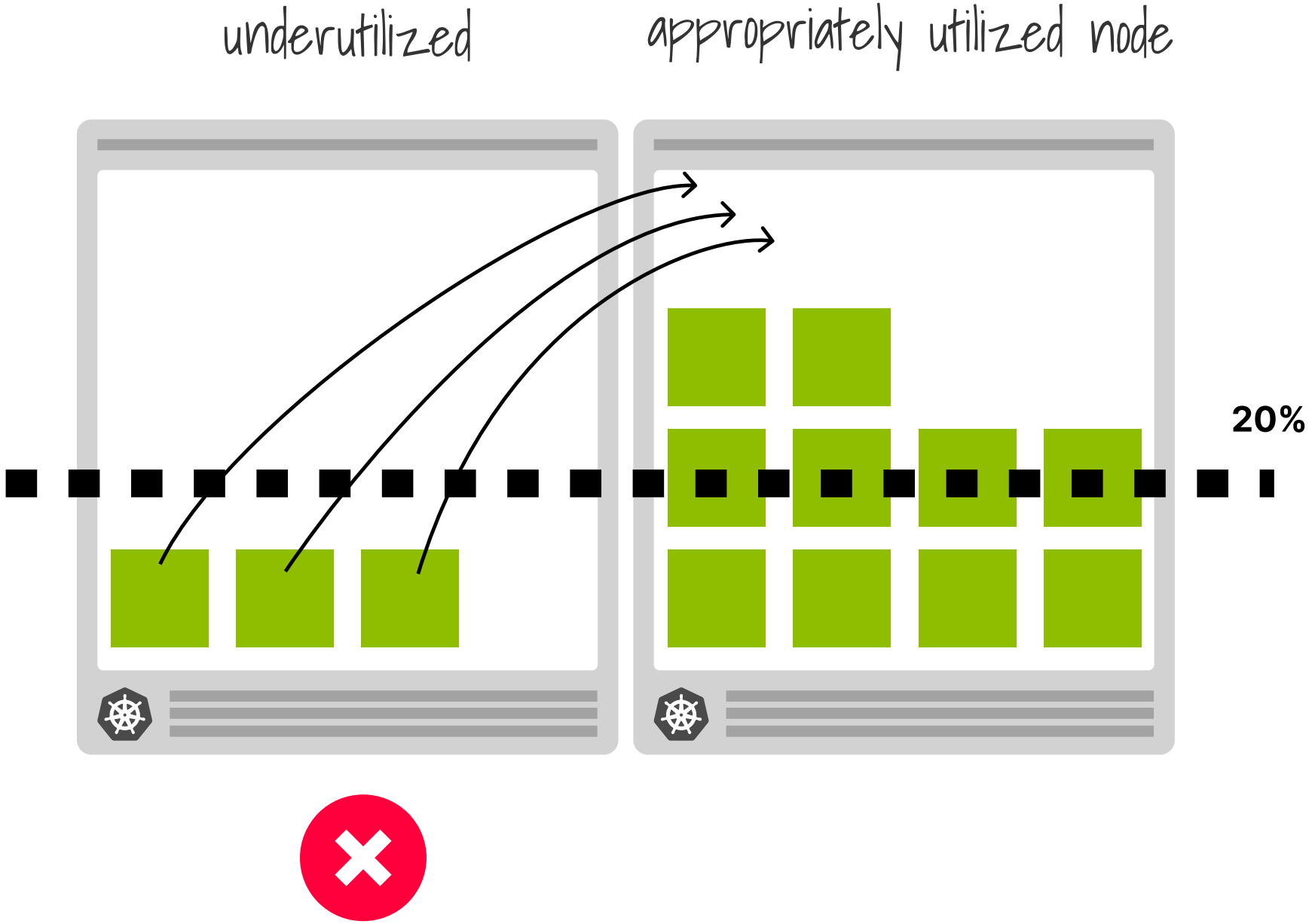

Kubernetes API server
CRI
CNI
CSI
containerd
cAdvisor
kubelet reports
the metrics

High utilization policy

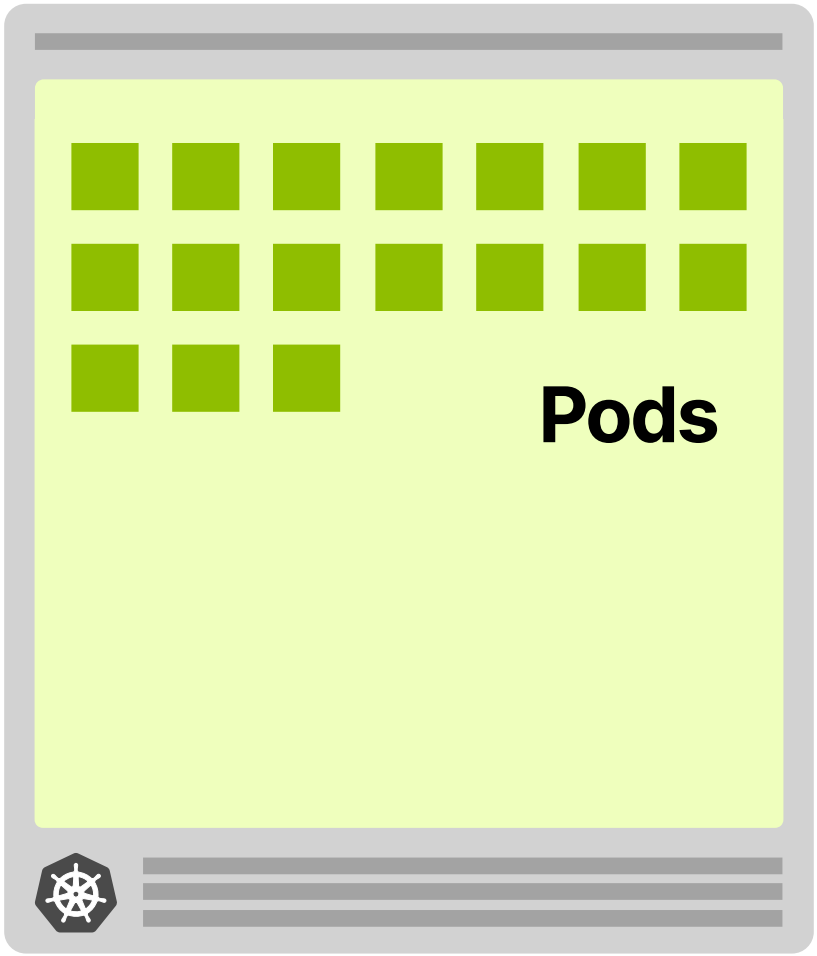
```yaml
apiVersion: "descheduler/v1alpha2"
kind: "DeschedulerPolicy"
profiles:
  - name: ProfileName
    pluginConfig:
    - name: "HighNodeUtilization"
      args:
        thresholds:
          "memory": 20
    plugins:
      balance:
        enabled:
          - "HighNodeUtilization"
```

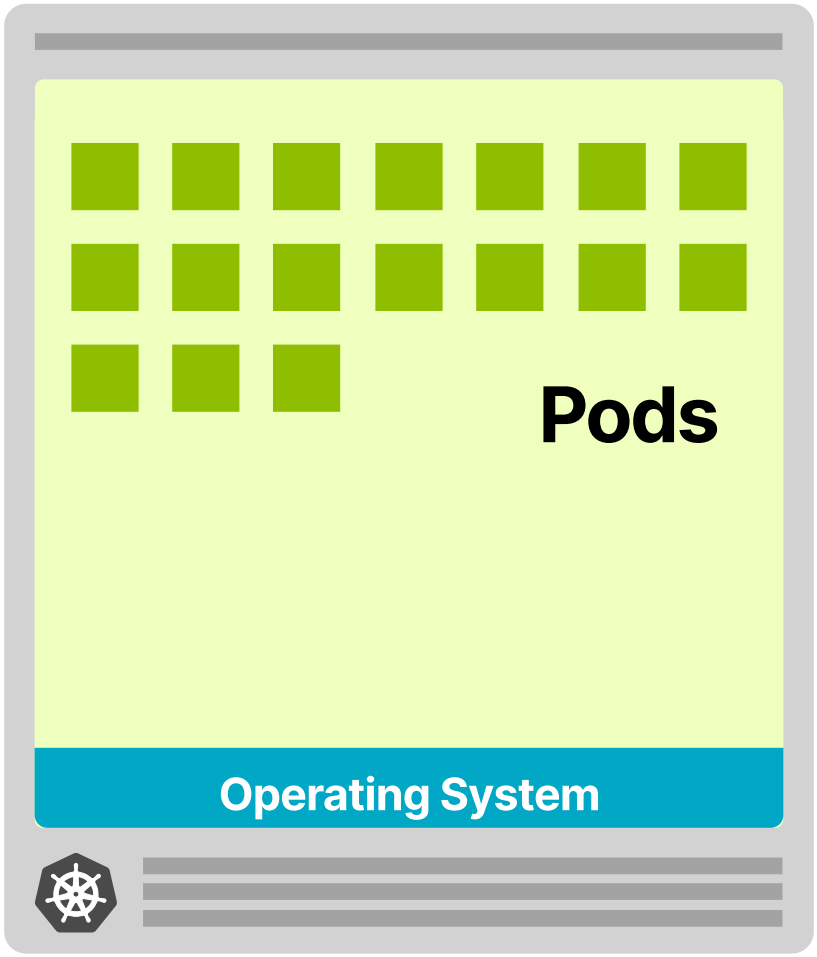



underutilized
appropriately utilized node
20%

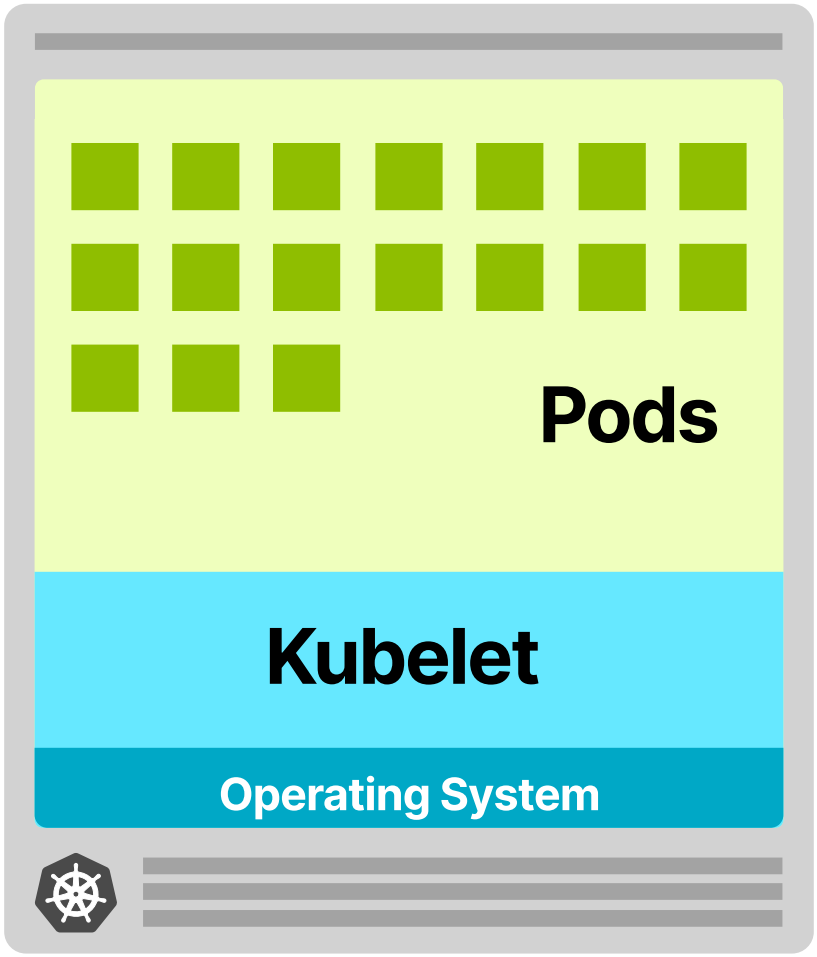

underutilized
appropriately utilized node
20%

Allocatable


Pods

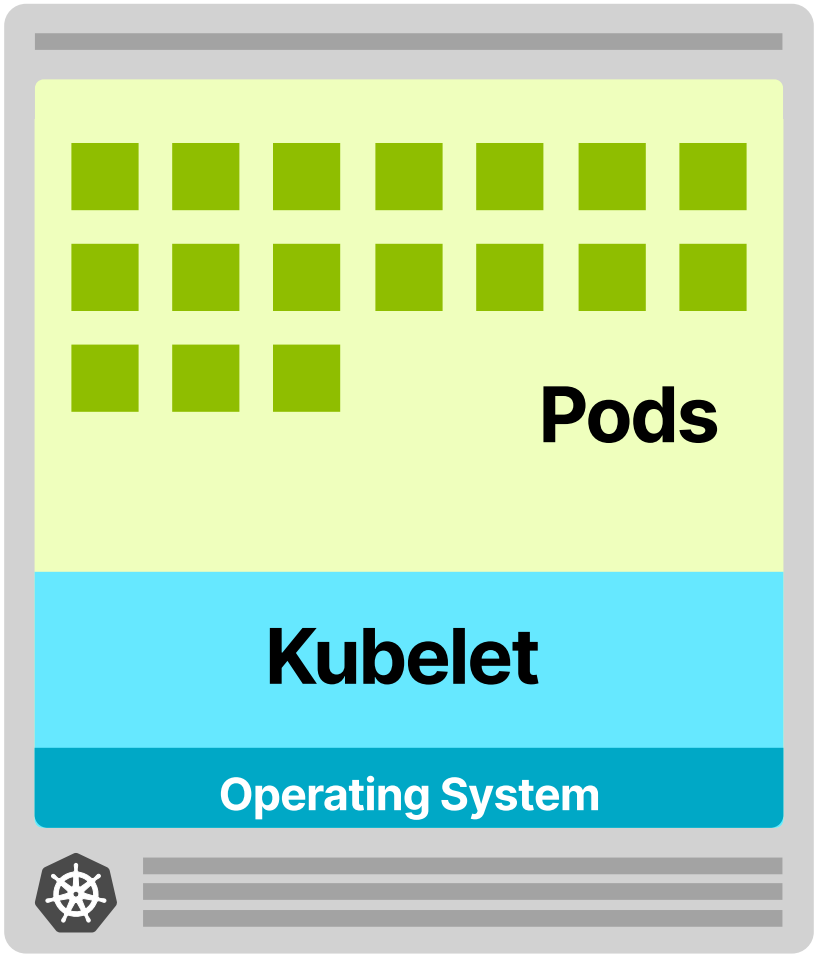


4 **Memory and CPU reserved to the OS**



3 Memory and CPU reserved to the kubelet

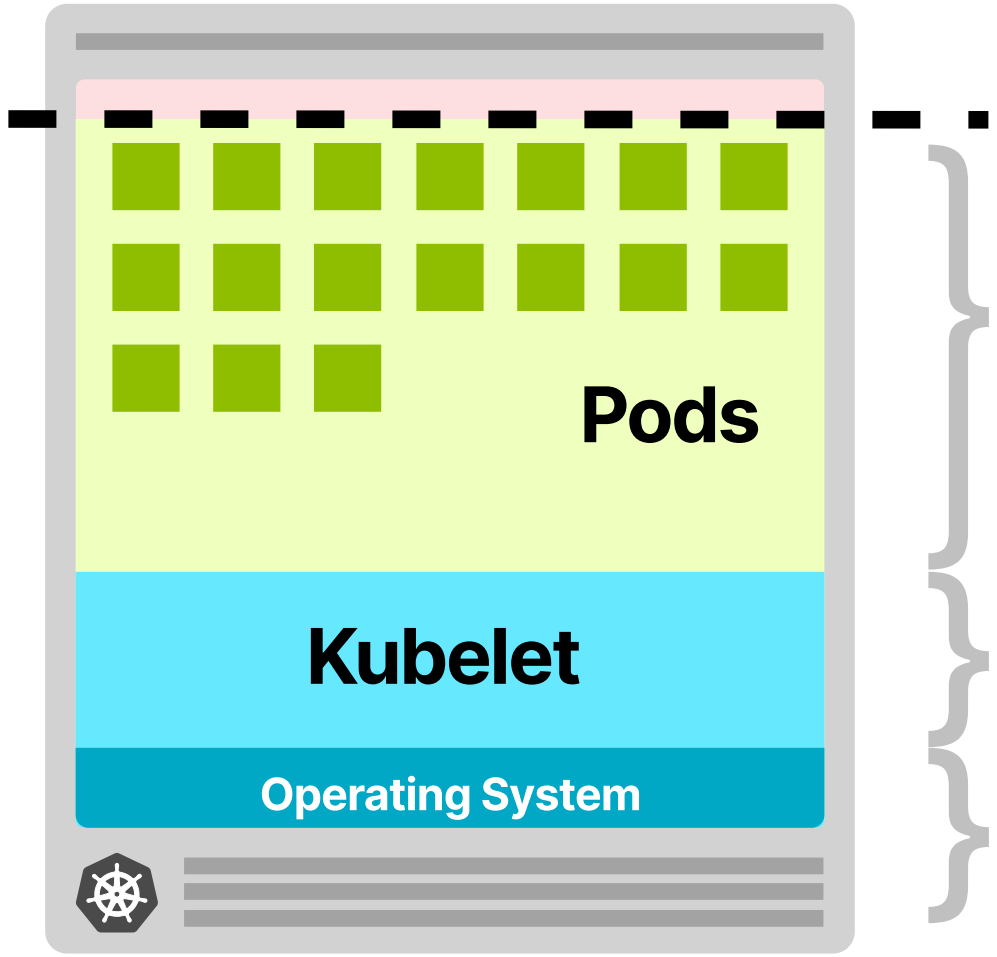
4 Memory and CPU reserved to the OS



2 Memory and CPU left to Pods

3 Memory and CPU reserved to the kubelet

4 Memory and CPU reserved to the OS



1. Eviction threshold
2. Memory and CPU left to Pods
3. Memory and CPU reserved to the kubelet
4. Memory and CPU reserved to the OS

HOW TO (RIGHT) SIZE YOUR KUBERNETES CLUSTER FOR EFFICIENCY

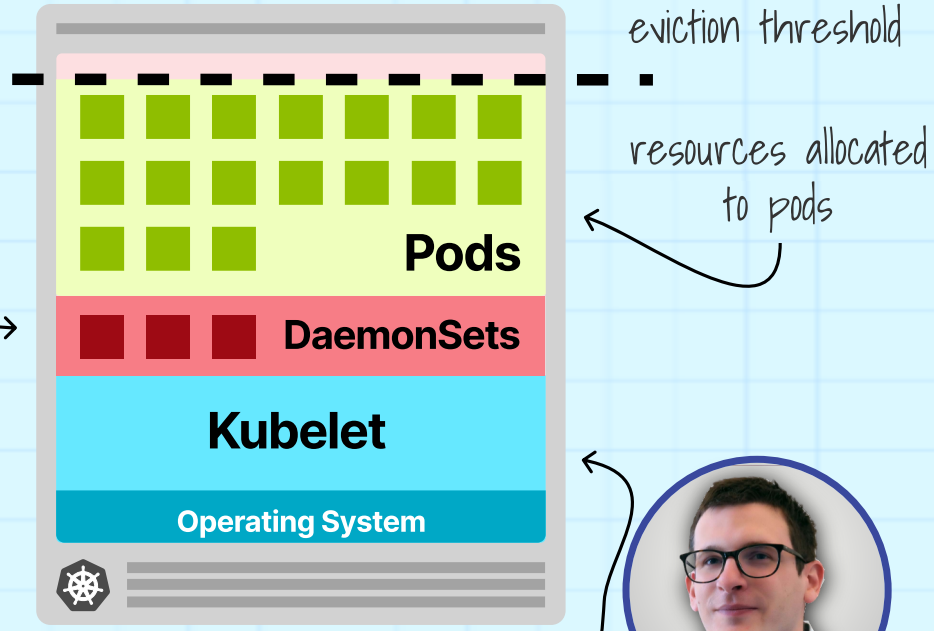
14th of Sep

8am PT | 5pm CET

LIVE STREAMING

REGISTER HERE

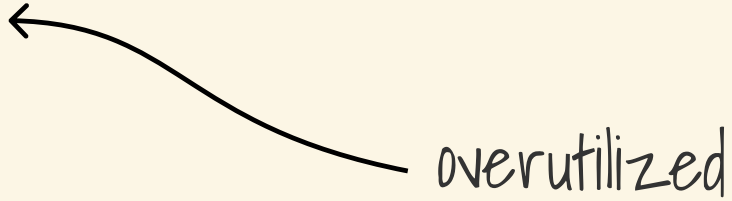
bit.ly/k8s-optimize-1



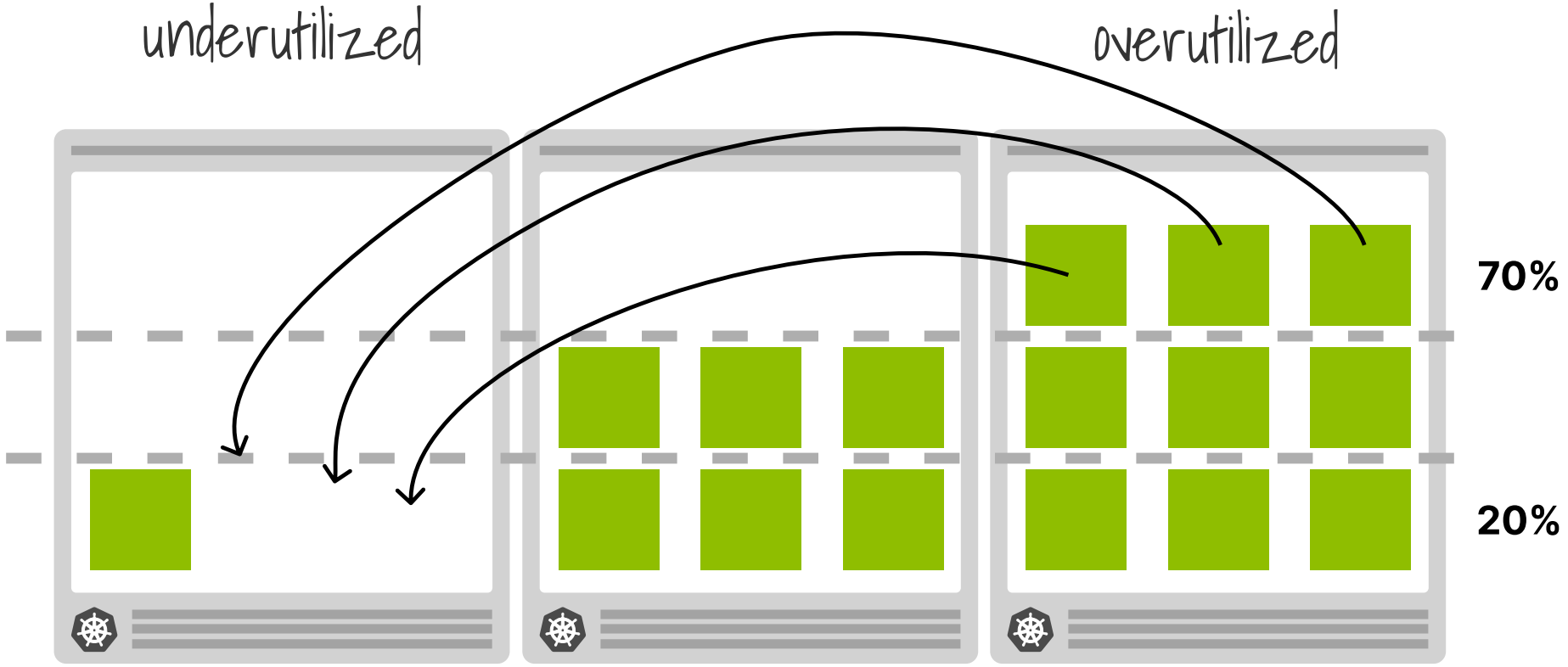
Daniele Polencic

Low utilization policy

```yaml
apiVersion: "descheduler/v1alpha2"
kind: "DeschedulerPolicy"
profiles:
  - name: ProfileName
    pluginConfig:
    - name: "LowNodeUtilization"
      args:
        thresholds:
          "memory": 20
        targetThresholds:
          "memory": 70
    plugins:
      balance:
        enabled:
          - "LowNodeUtilization"
```



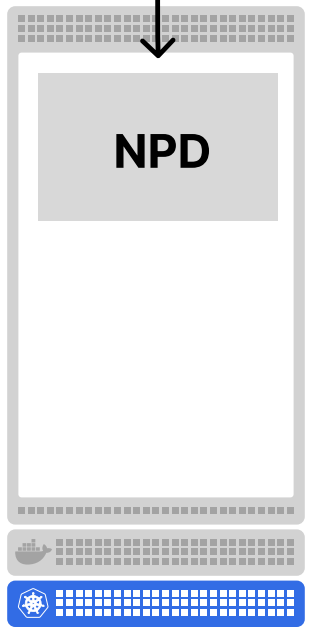

underutilized
overutilized
70%
20%


underutilized
overutilized
70%
20%

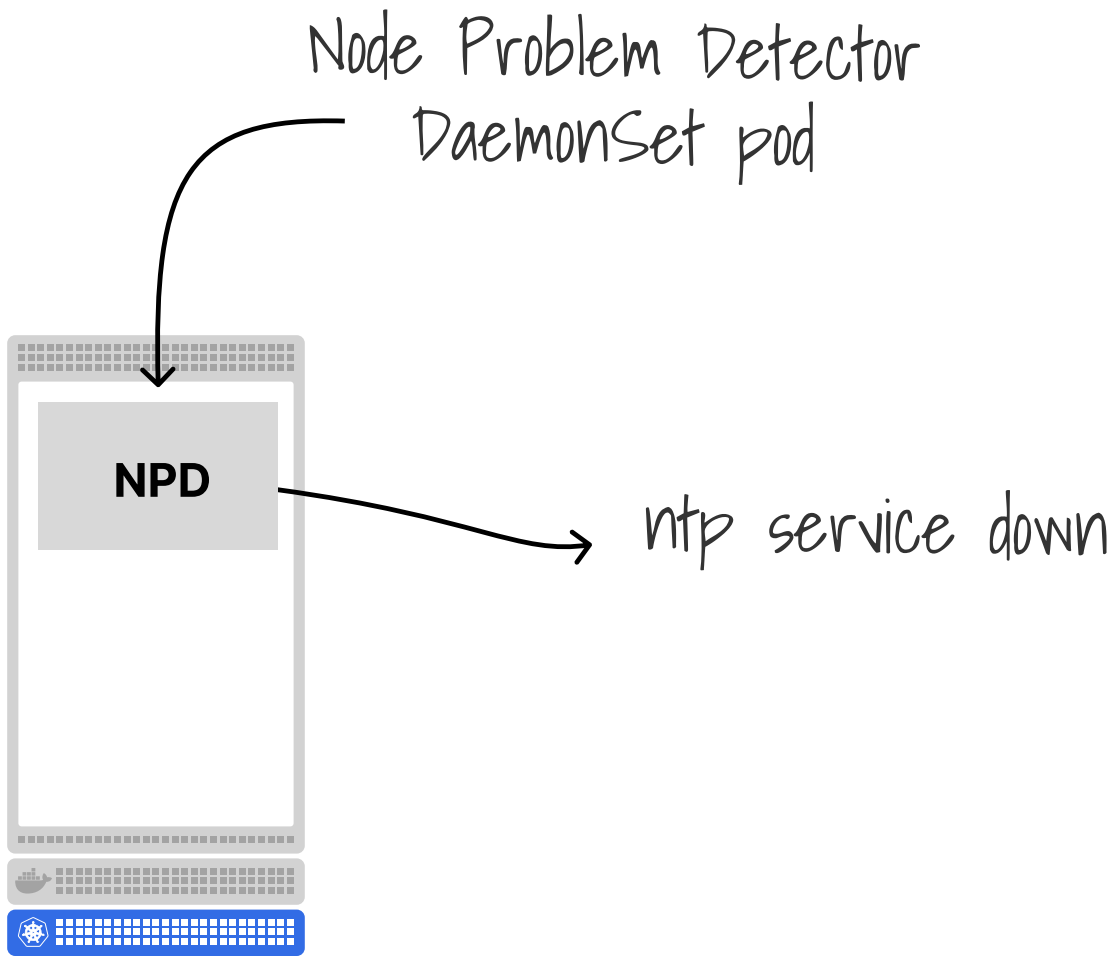
Demo

Node Problem Detector

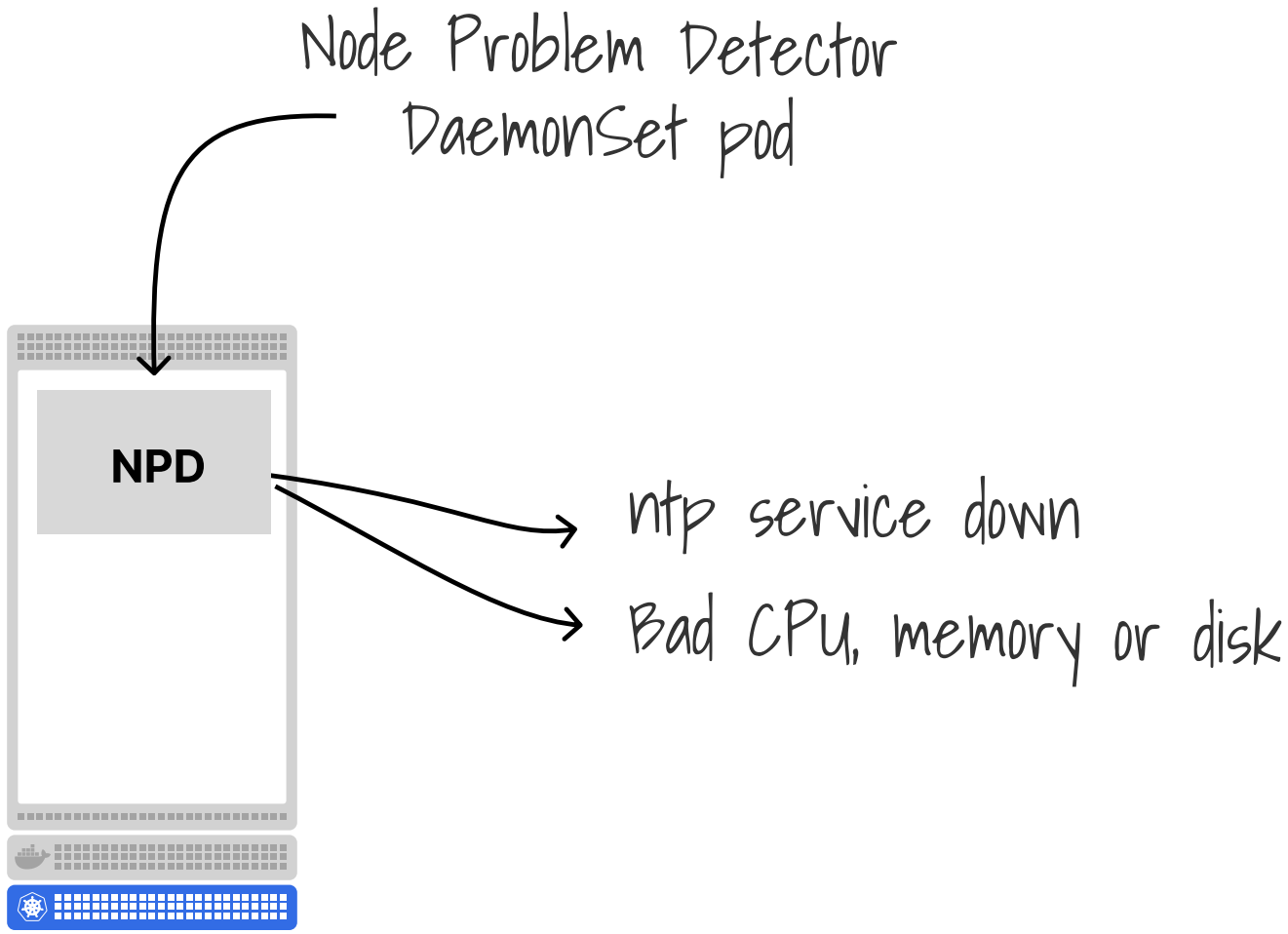
Kubernetes control plane



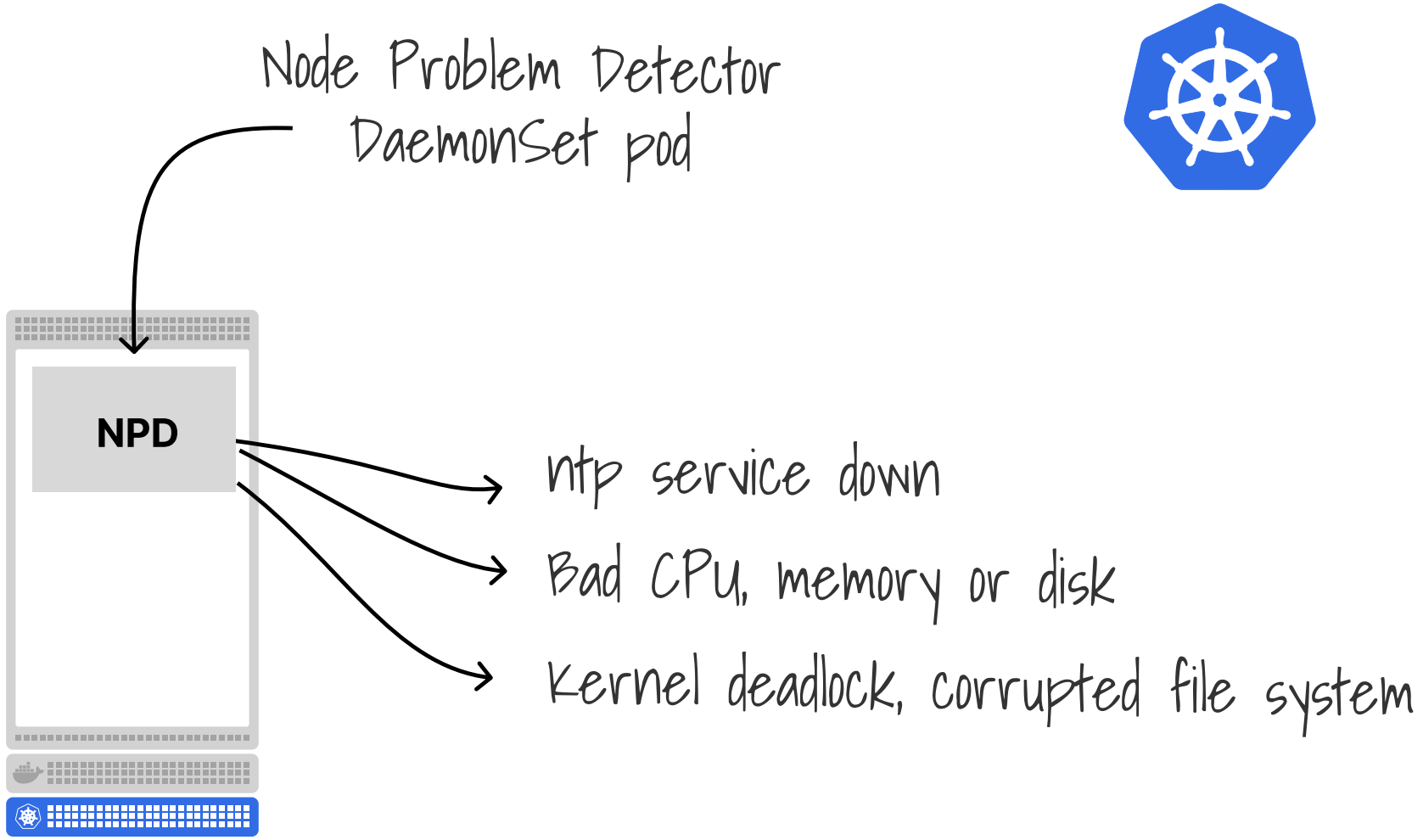
Kubernetes control plane



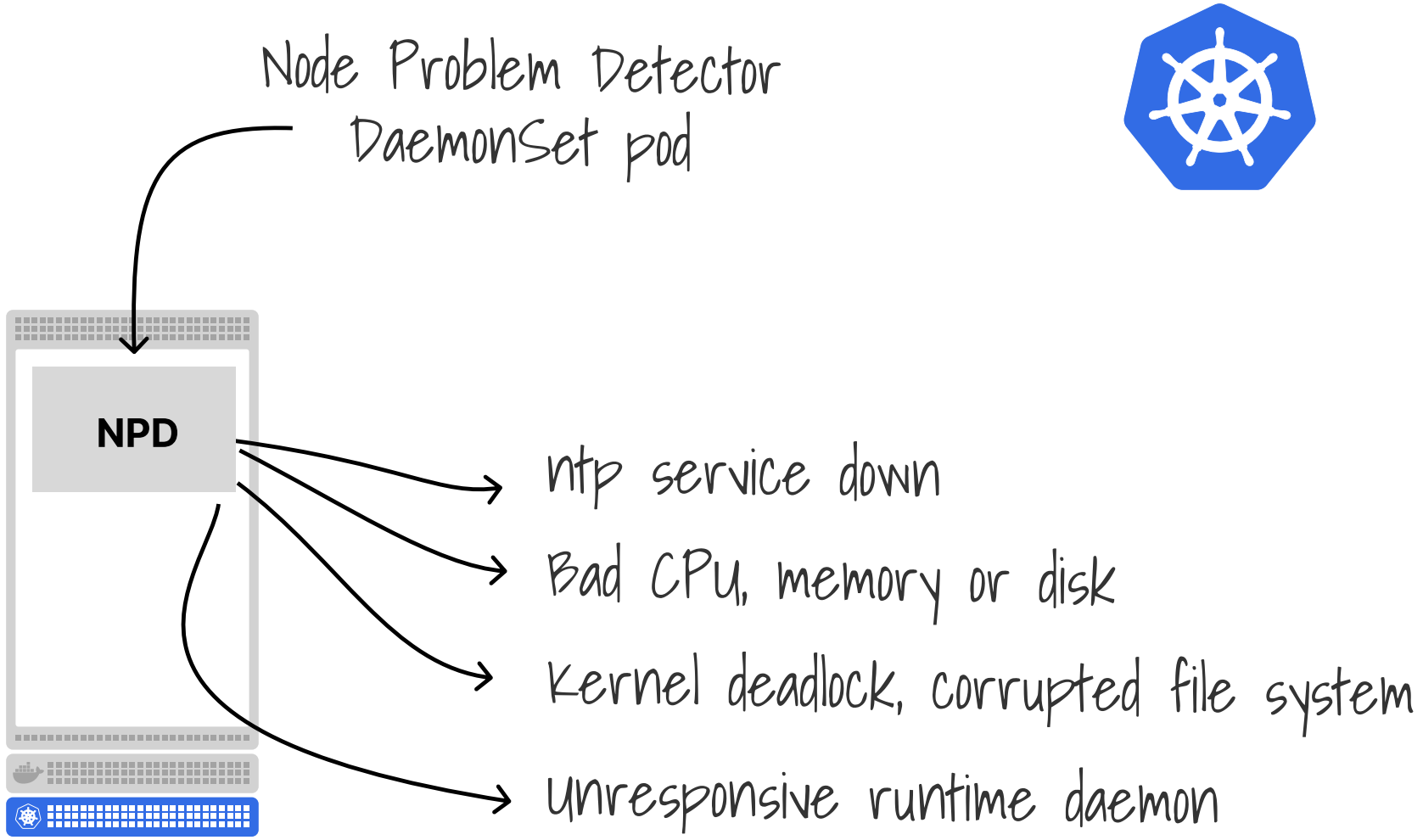
Kubernetes control plane



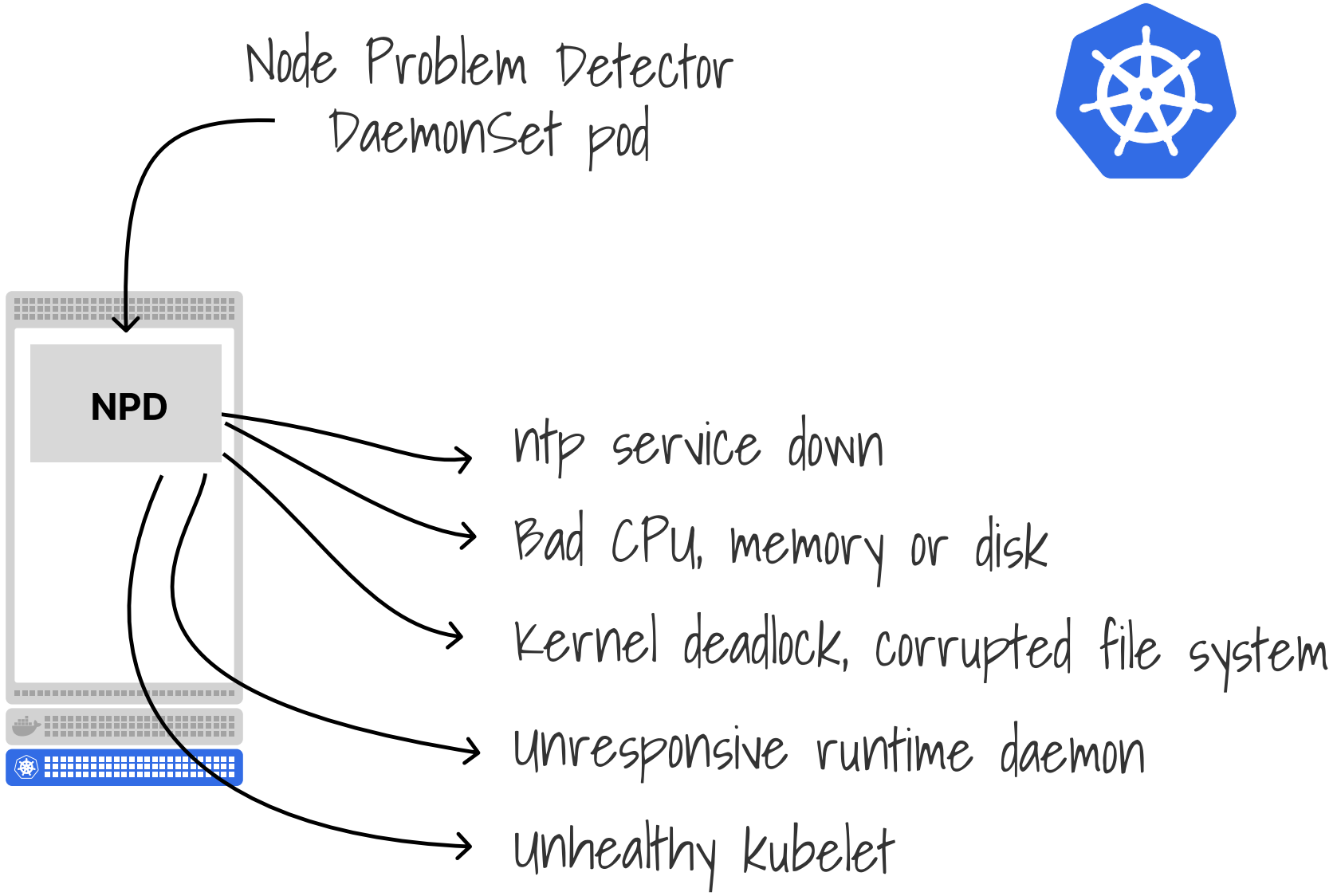
Kubernetes control plane

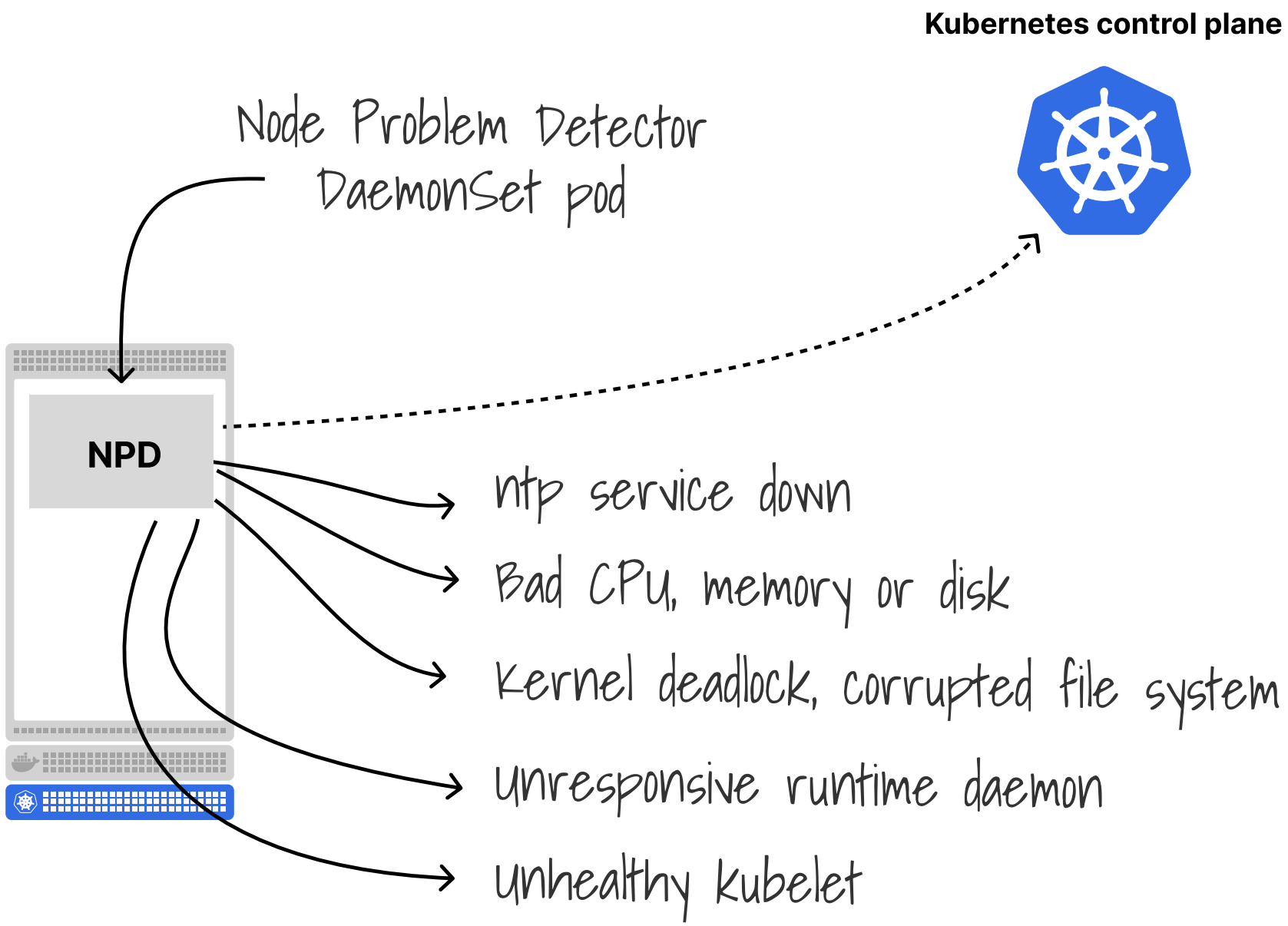


Kubernetes control plane

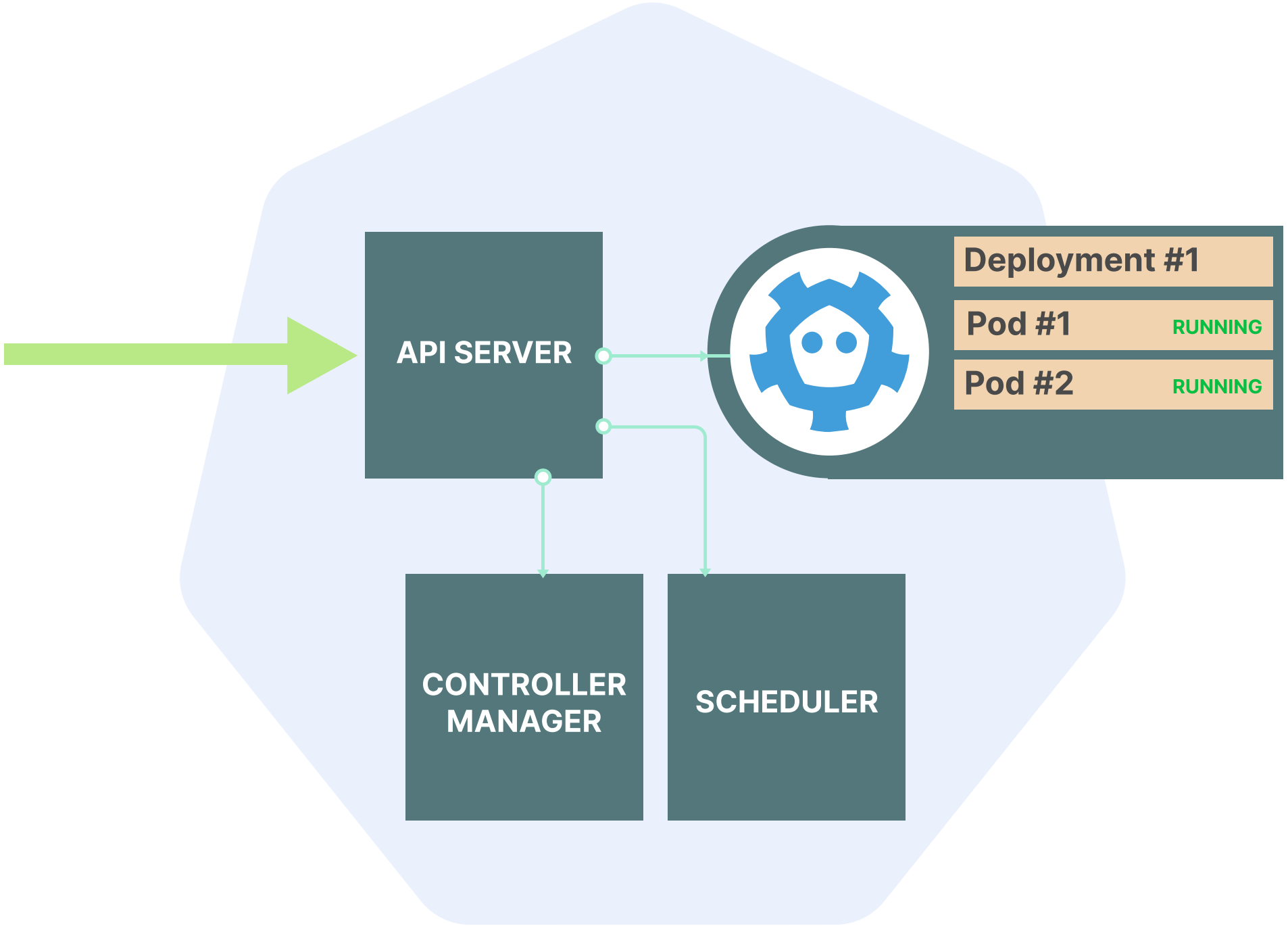


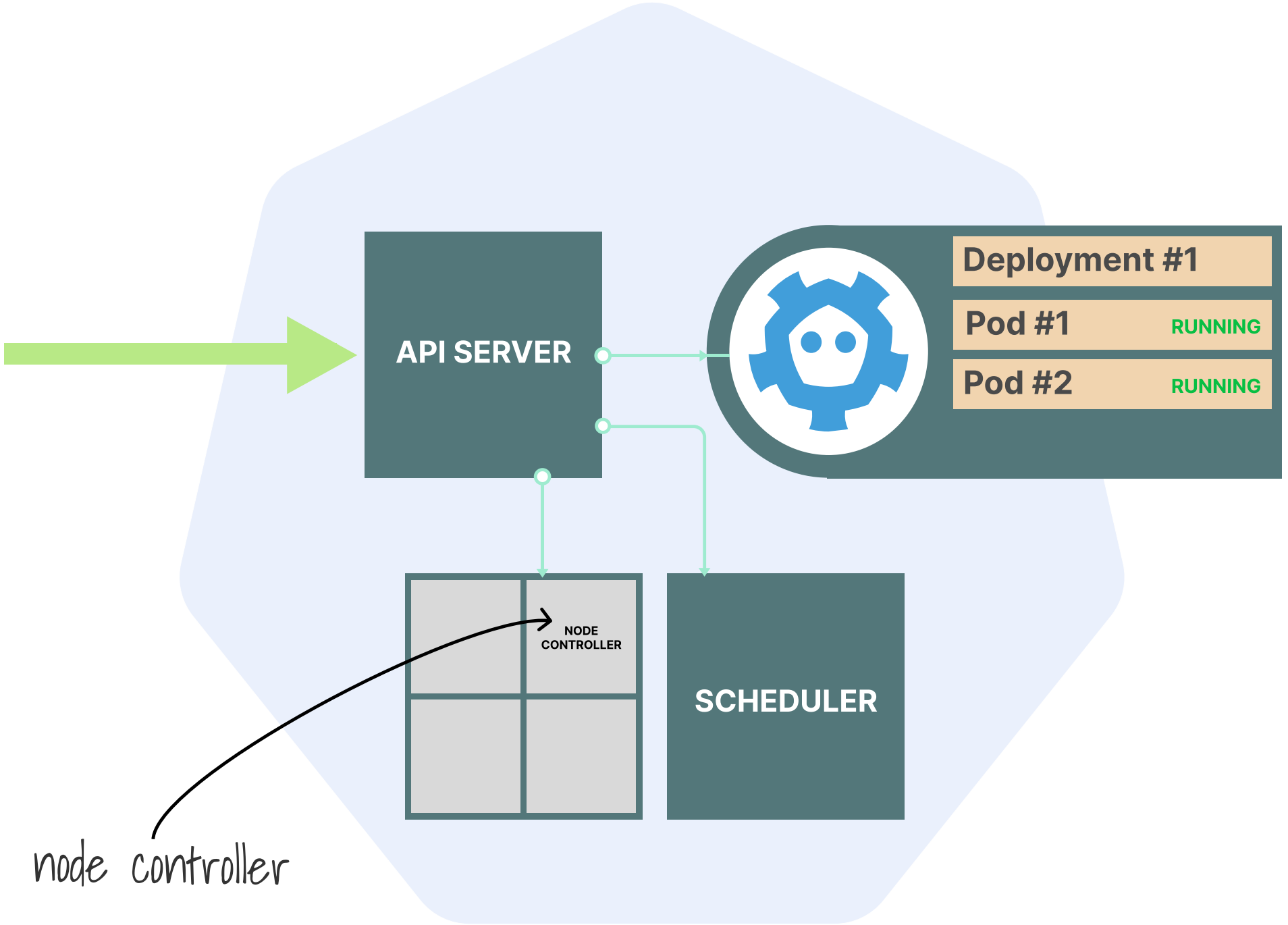
Kubernetes control plane




Kubernetes control plane
Node Problem Detector
DaemonSet pod
NPD
ntp service down
Bad CPU, memory or disk
Kernel deadlock, corrupted file system
Unresponsive runtime daemon
Unhealthy kubelet

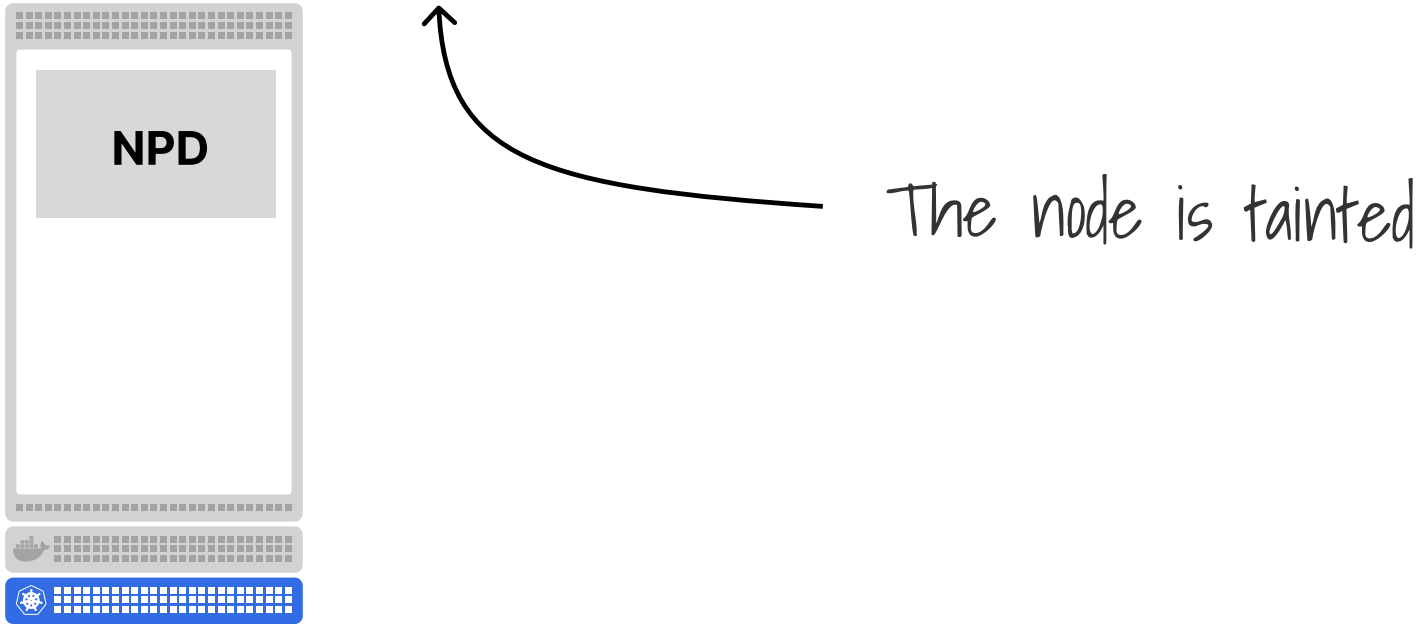
Node controller


API SERVER
Deployment #1
Pod #1
RUNNING
Pod #2
RUNNING
CONTROLLER MANAGER
SCHEDULER


API SERVER
Deployment #1
Pod #1
RUNNING
Pod #2
RUNNING
NODE CONTROLLER
SCHEDULER
node controller

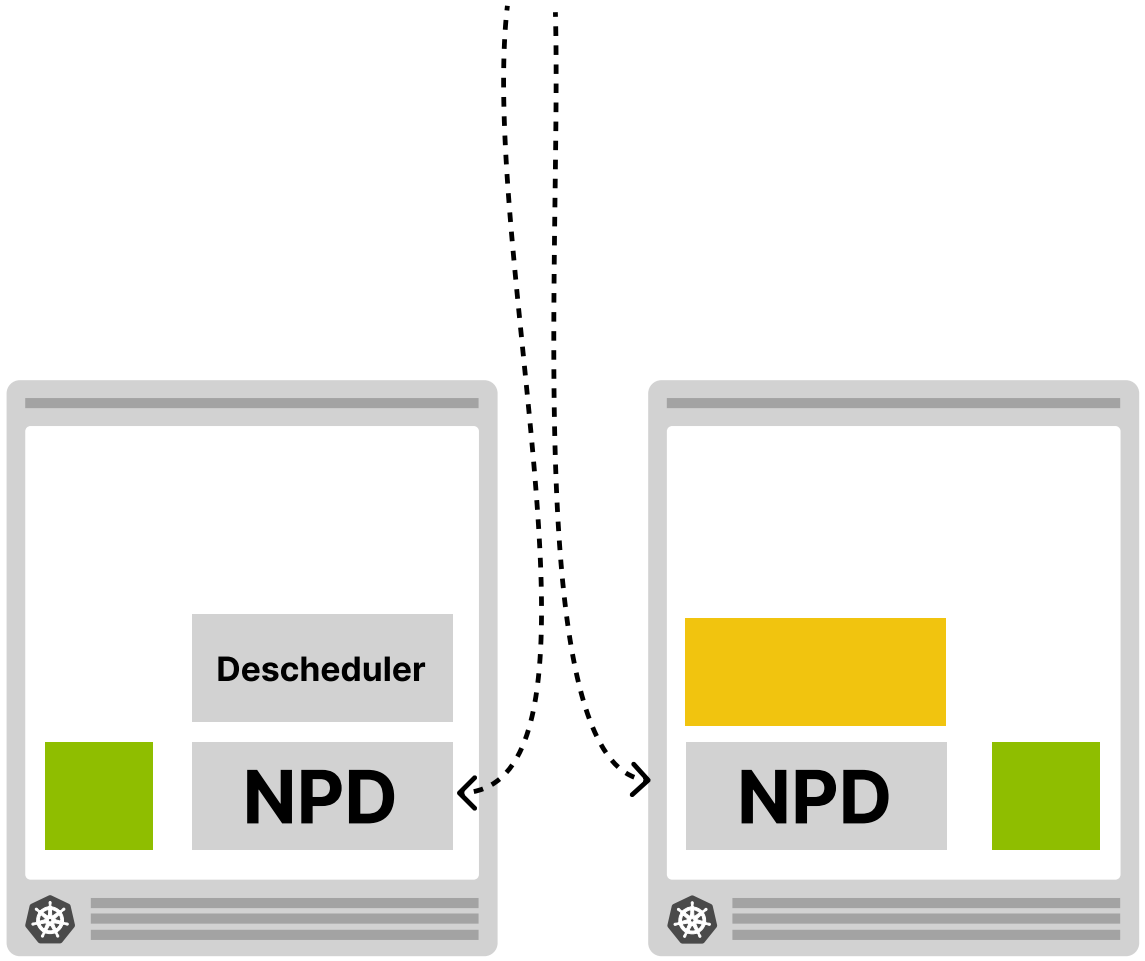
Kubernetes control plane

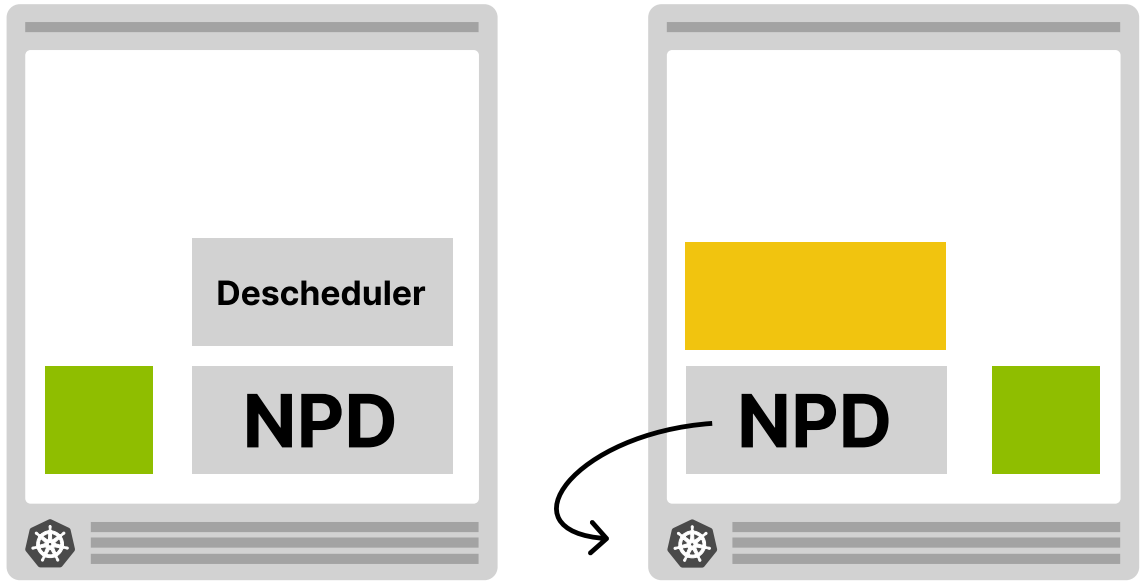
node.kubernetes.io/unreachable



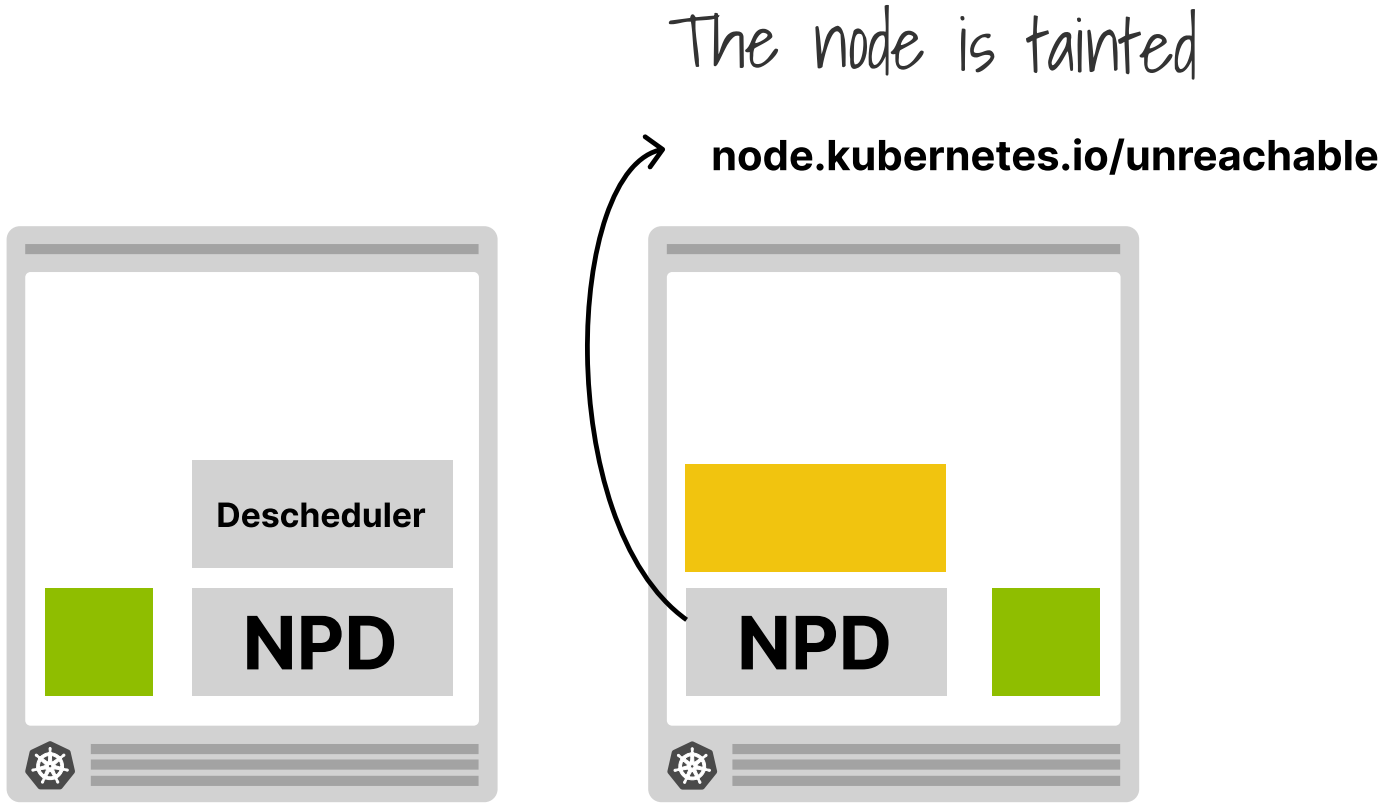
Node Problem Detector + Descheduler + Cluster Autoscaler

The Node Problem Detector is deployed as a DaemonSet





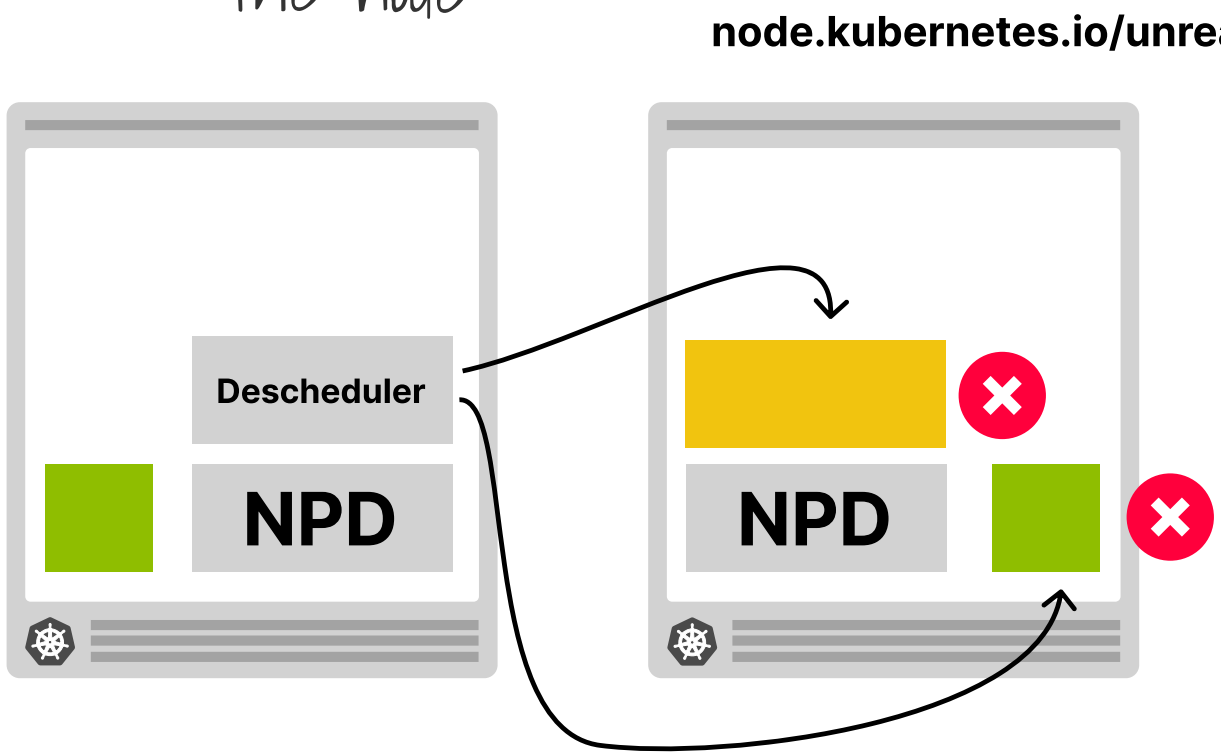
The Node Problem detector pod detects that the node is unreachable

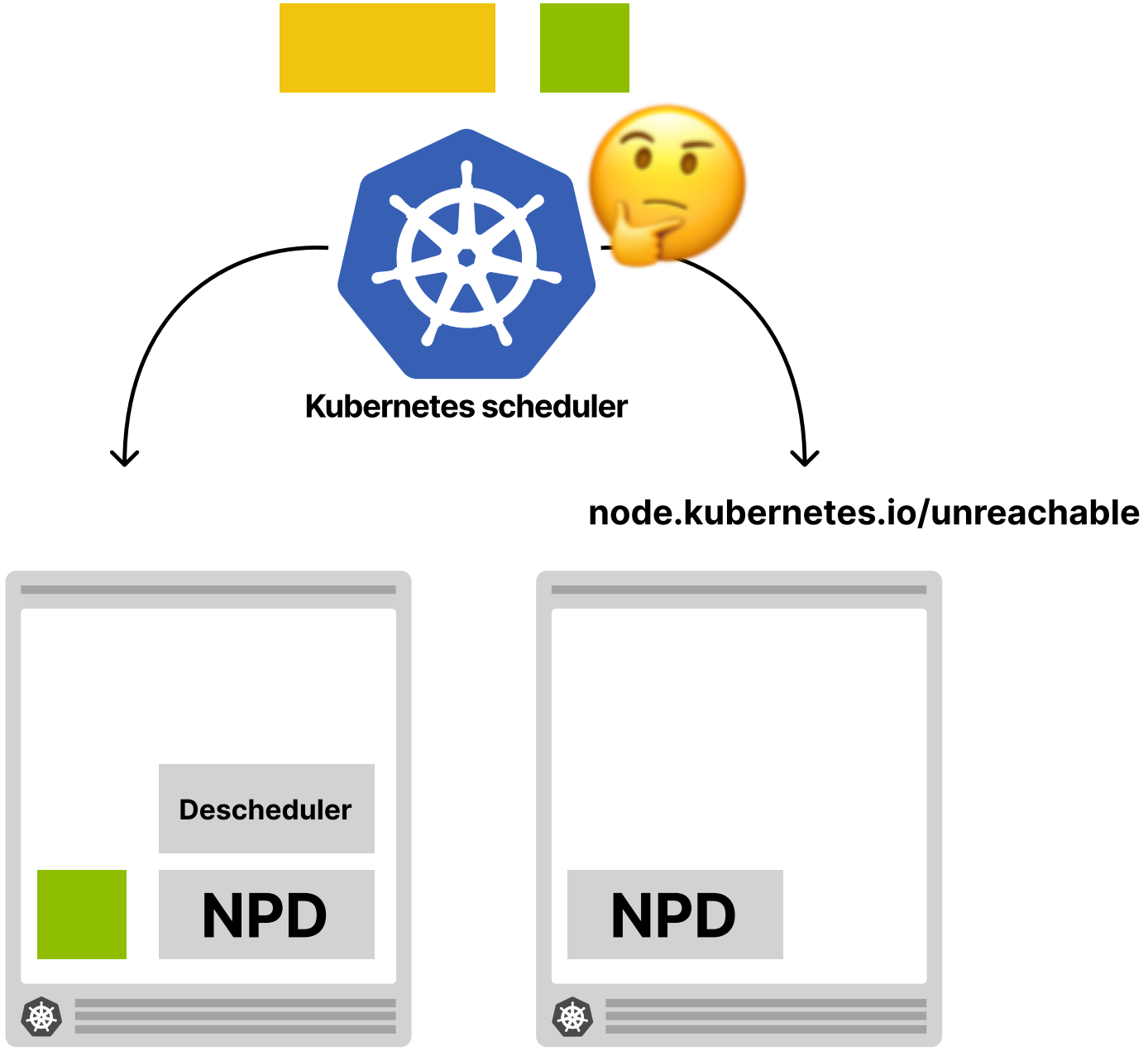

The node is tainted
node.kubernetes.io/unreachable
Descheduler
NPD
NPD

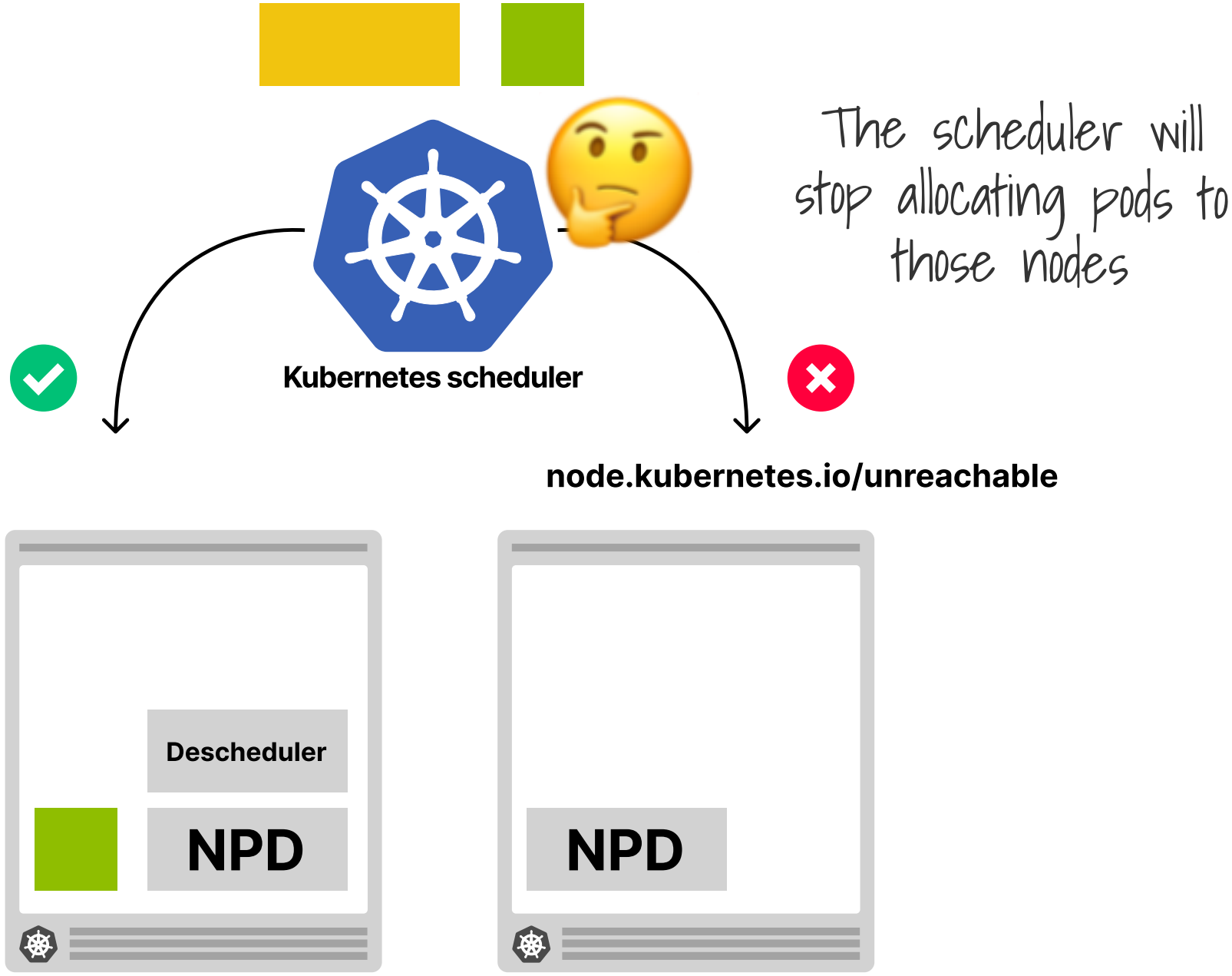
Taints violation policy

```yaml
apiVersion: "descheduler/v1alpha2"
kind: "DeschedulerPolicy"
profiles:
  - name: ProfileName
    pluginConfig:
    - name: "RemovePodsViolatingNodeTaints"
    plugins:
      deschedule:
        enabled:
          - "RemovePodsViolatingNodeTaints"
```

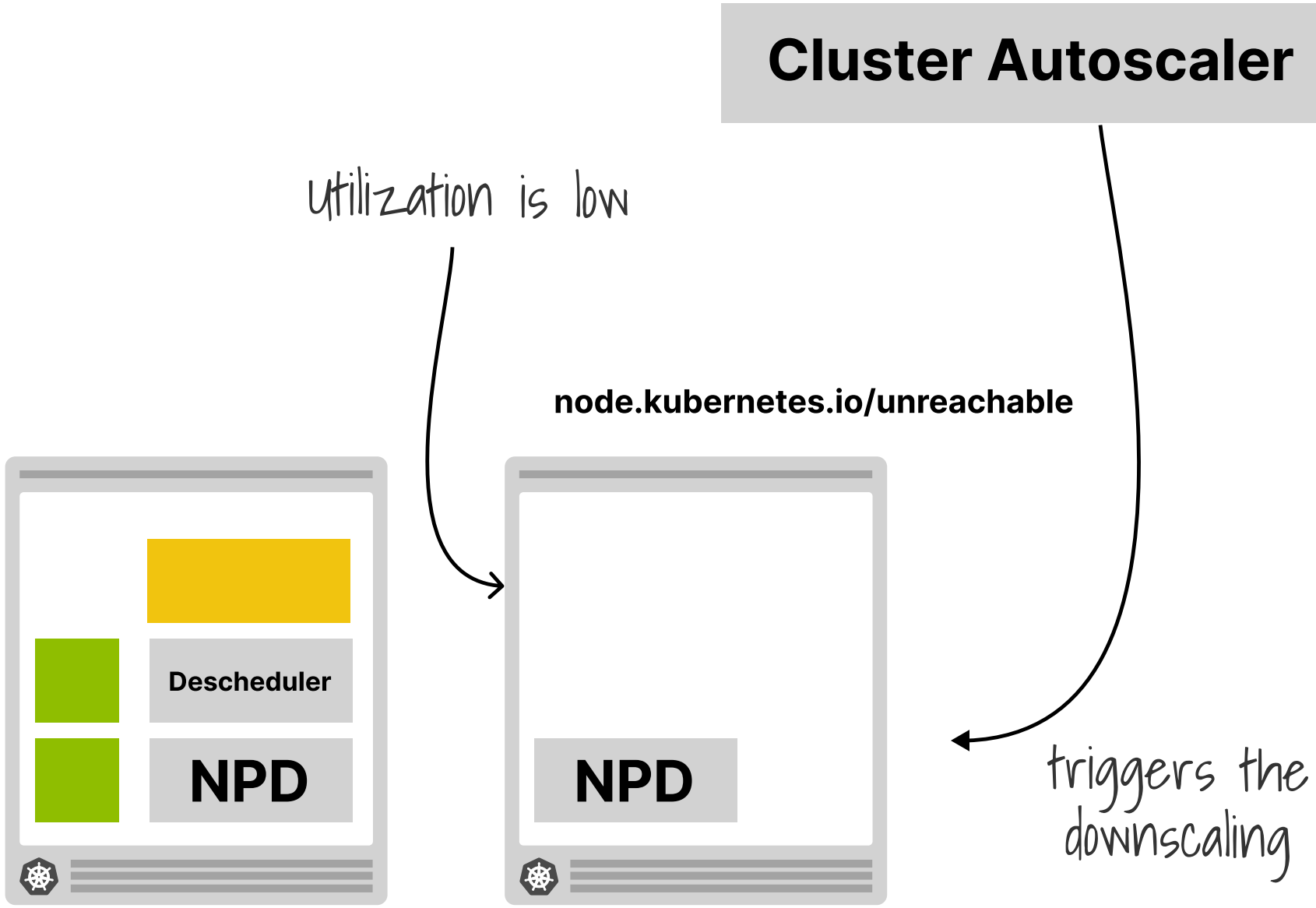
The descheduler evicts all pods from the node




Kubernetes scheduler
node.kubernetes.io/unreachable
Descheduler
NPD
NPD


The scheduler will stop allocating pods to those nodes
Kubernetes scheduler
node.kubernetes.io/unreachable
Descheduler
NPD
NPD

Cluster Autoscaler


Cluster Autoscaler
utilization is low
node.kubernetes.io/unreachable
Descheduler
NPD
NPD
triggers the downscaling

Takeaways

Recap

Recap

1. Kuberneres (not) rebalancing pods
2. Descheduler
3. Descheduler policies
4. Metrics pipeline
5. Node Problem Detector

Recap

1. Kuberneres (not) rebalancing pods
2. **Descheduler**
3. Descheduler policies
4. Metrics pipeline
5. Node Problem Detector

Recap

1. Kuberneres (not) rebalancing pods
2. Descheduler
3. **Descheduler policies**
4. Metrics pipeline
5. Node Problem Detector

Recap

1. Kuberneres (not) rebalancing pods
2. Descheduler
3. Descheduler policies
4. **Metrics pipeline**
5. Node Problem Detector

Recap

1. Kuberneres (not) rebalancing pods
2. Descheduler
3. Descheduler policies
4. Metrics pipeline
5. **Node Problem Detector**

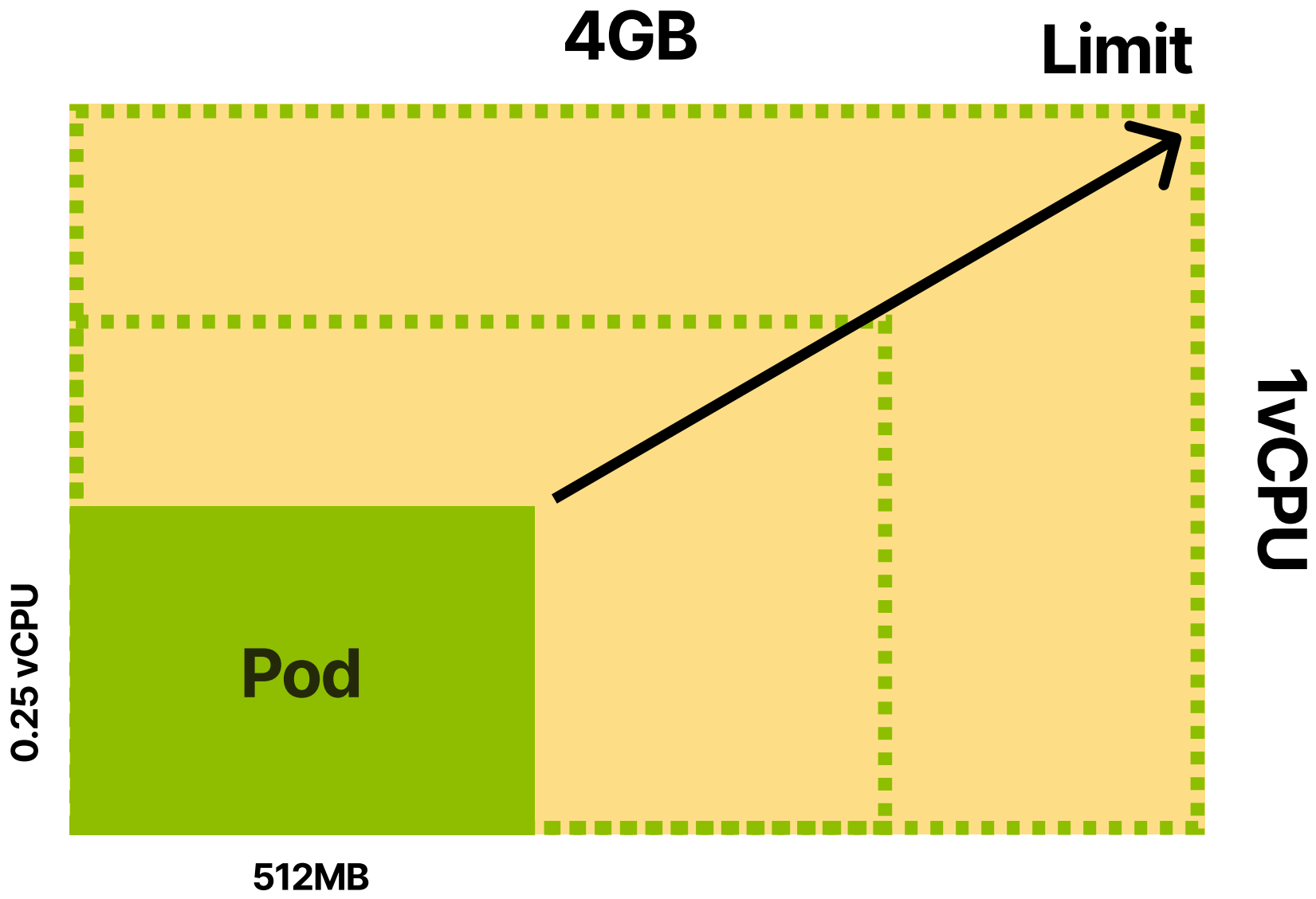
Thank you!

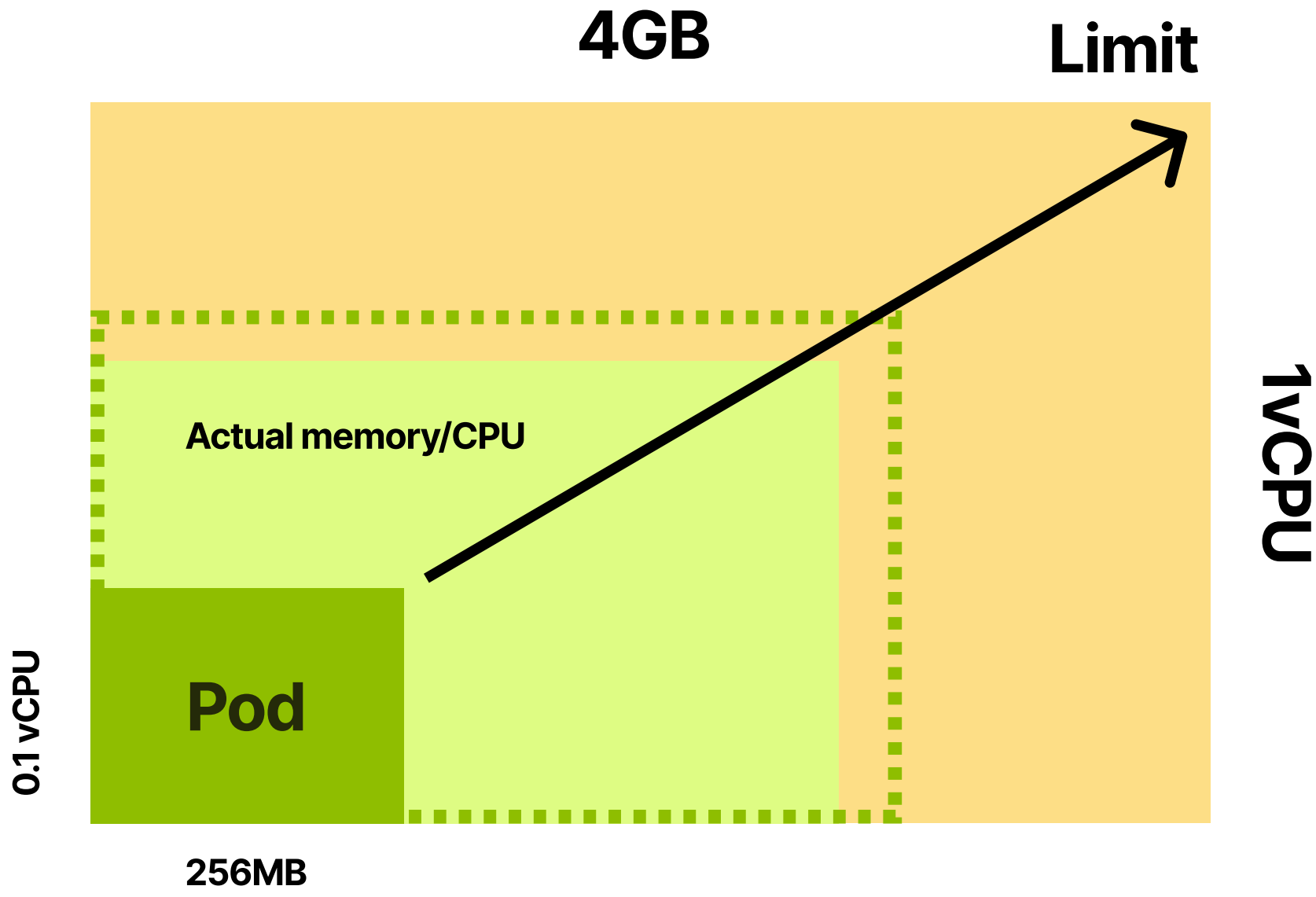
Chris Nesbitt-Smith
cns.me

Chris Nesbitt-Smith

cns.me

Requests & limits


4GB
Limit
1vCPU
0.25 vCPU
Pod
512MB


4GB
Limit
1vCPU
Actual memory/CPU
Pod
0.1 vCPU
256MB

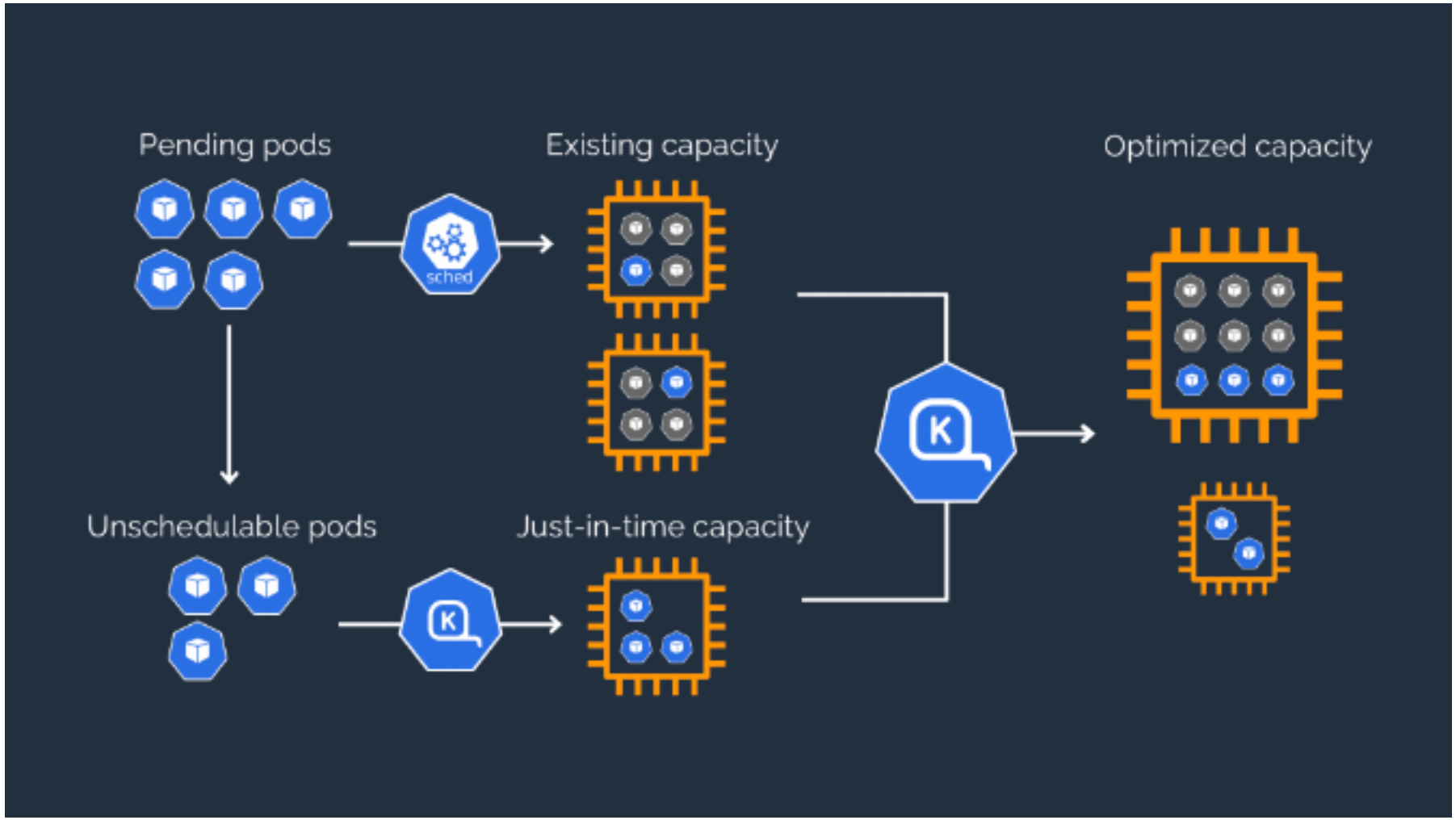
Requests and limits

1. Requests → scheduler
2. Limits → kubelet
3. QoS

Scheduler hints

1. nodeSelector
2. Node affinity
3. Pod affinity/anti-affinity
4. Taints and tolerations
5. Topology constraints
6. Scheduler profiles

Karpenter

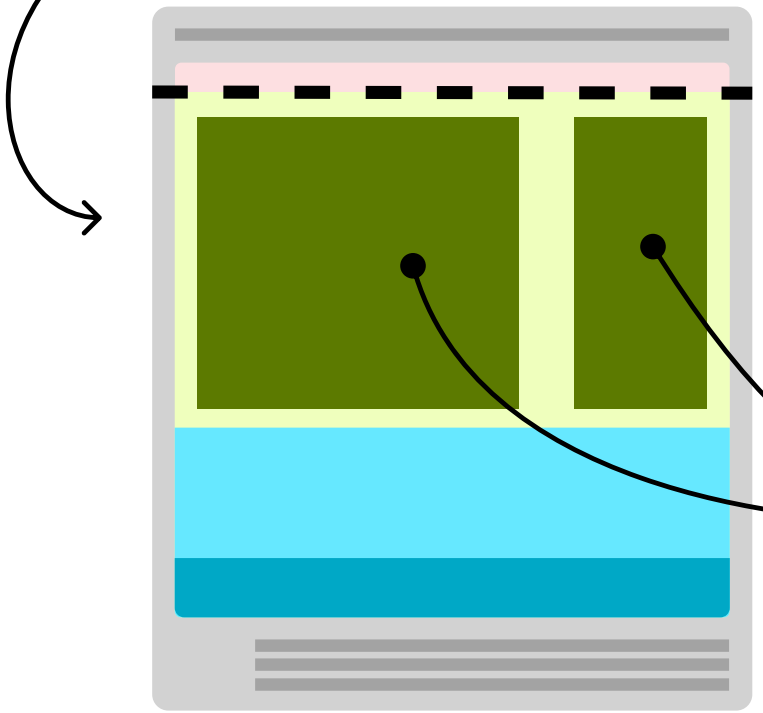

Pending pods
sched
Existing capacity
Optimized capacity
Unschedulable pods
Just-in-time capacity

Guaranteed, burstable, best-effort

4GB **Limit == request**

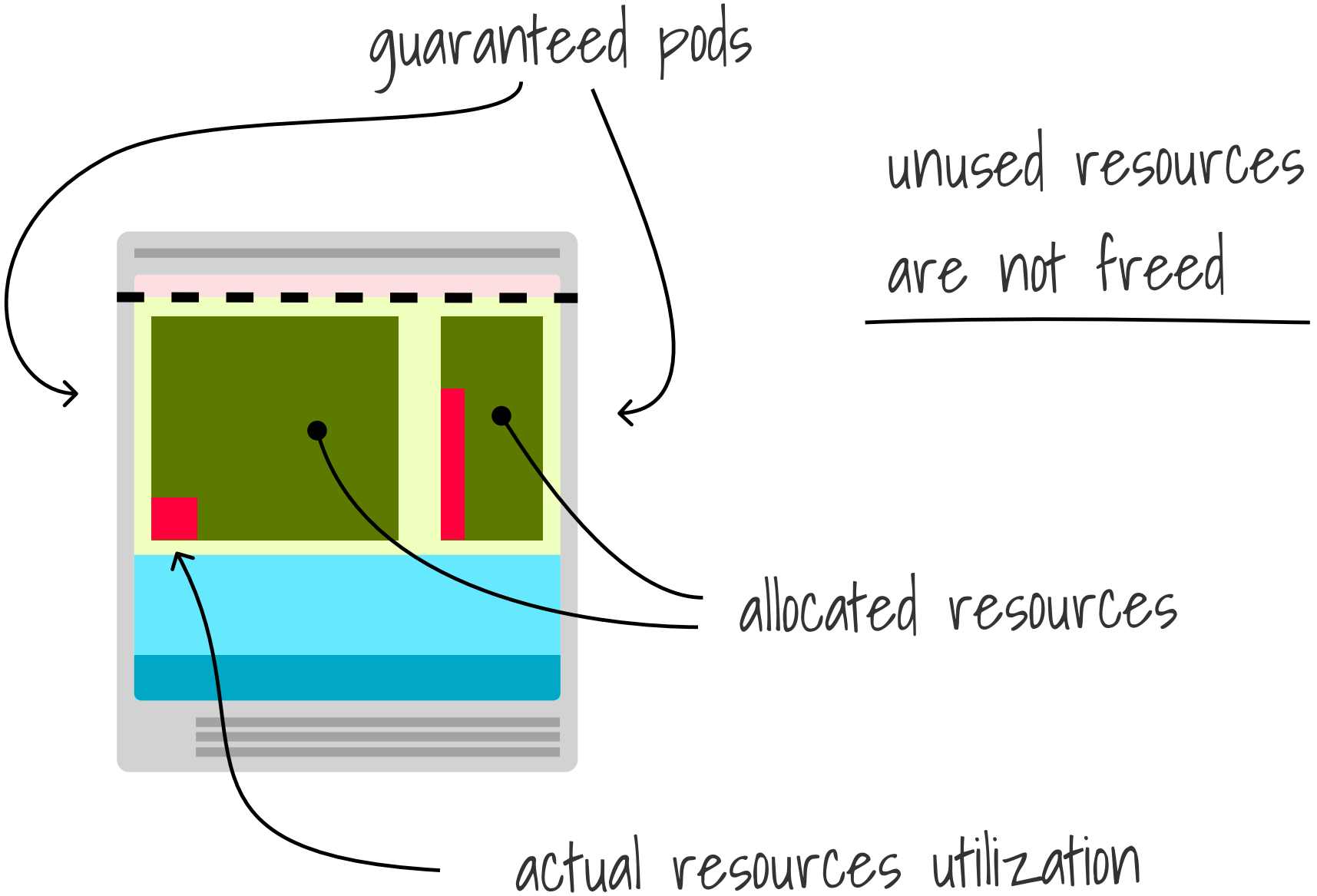
Guaranteed QoS class

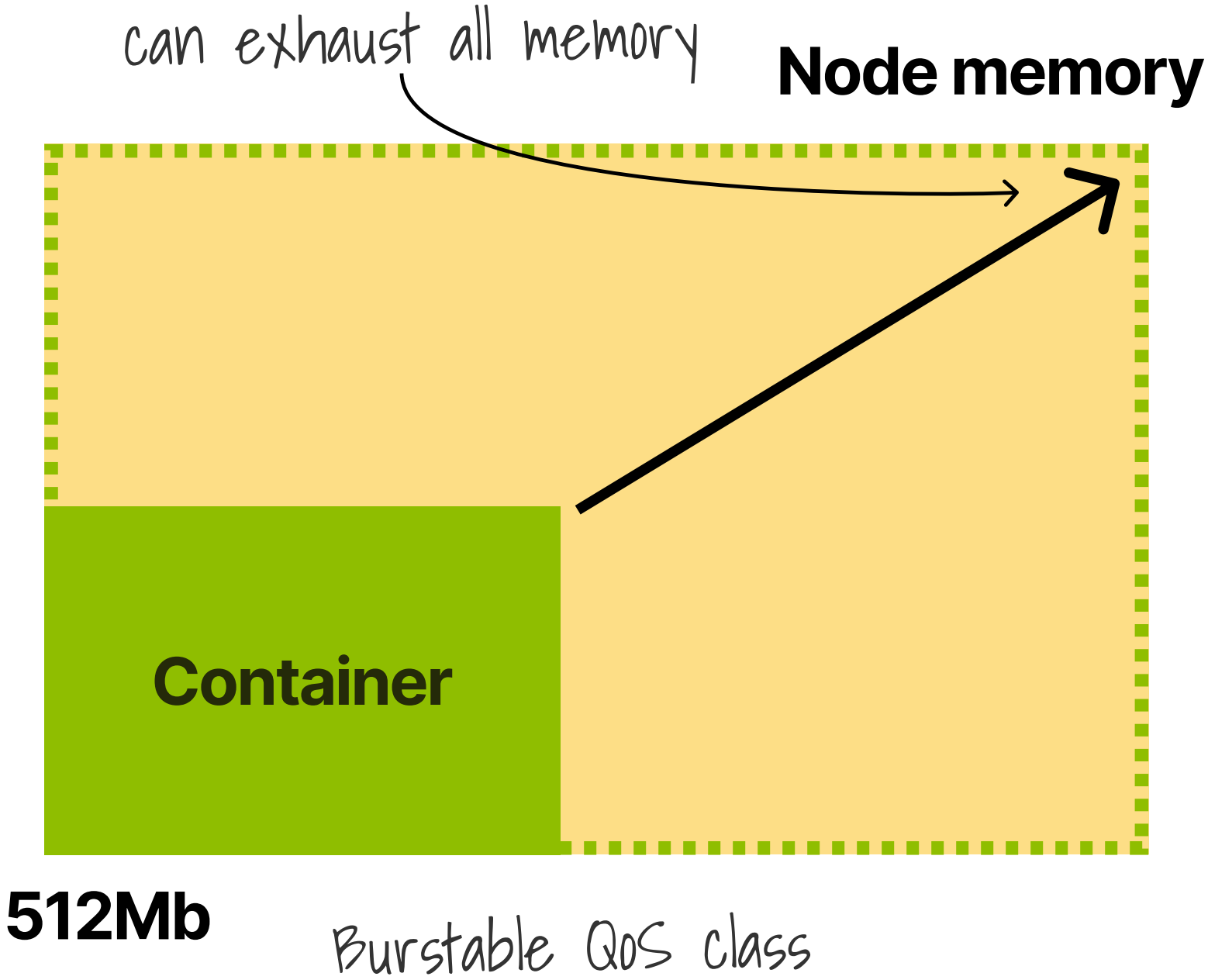
guaranteed pods



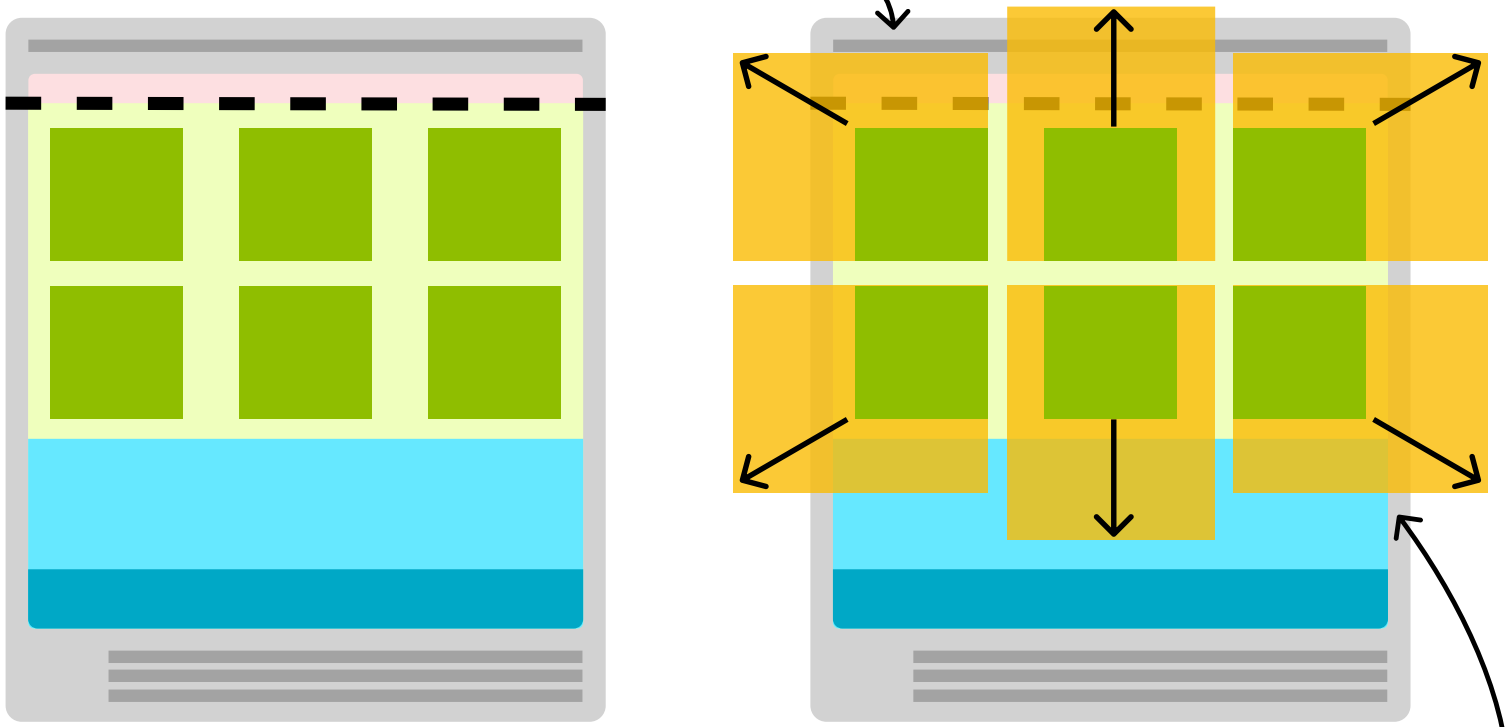
unused resources are not freed

allocated resources

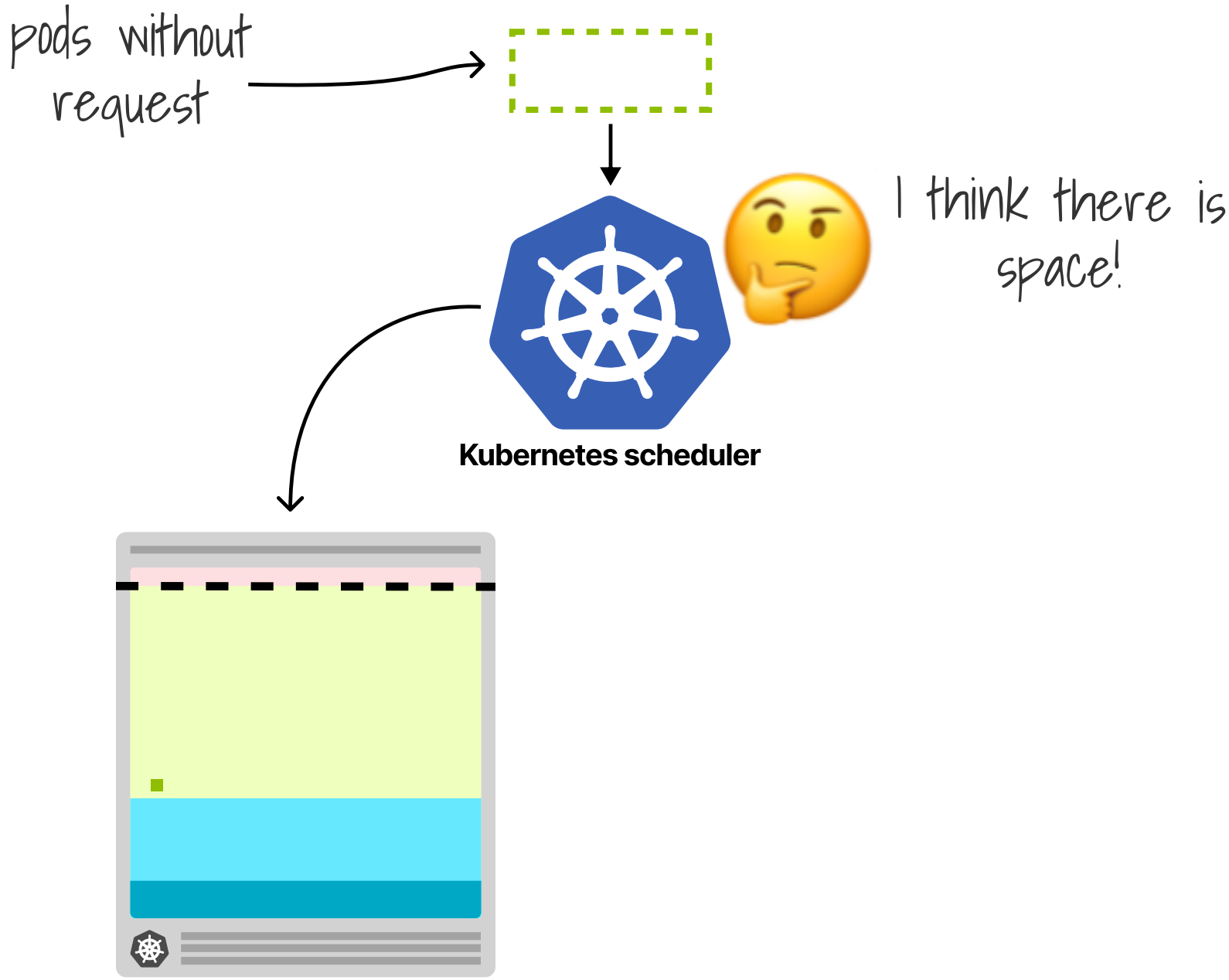

guaranteed pods
unused resources
are not freed
allocated resources
actual resources utilization

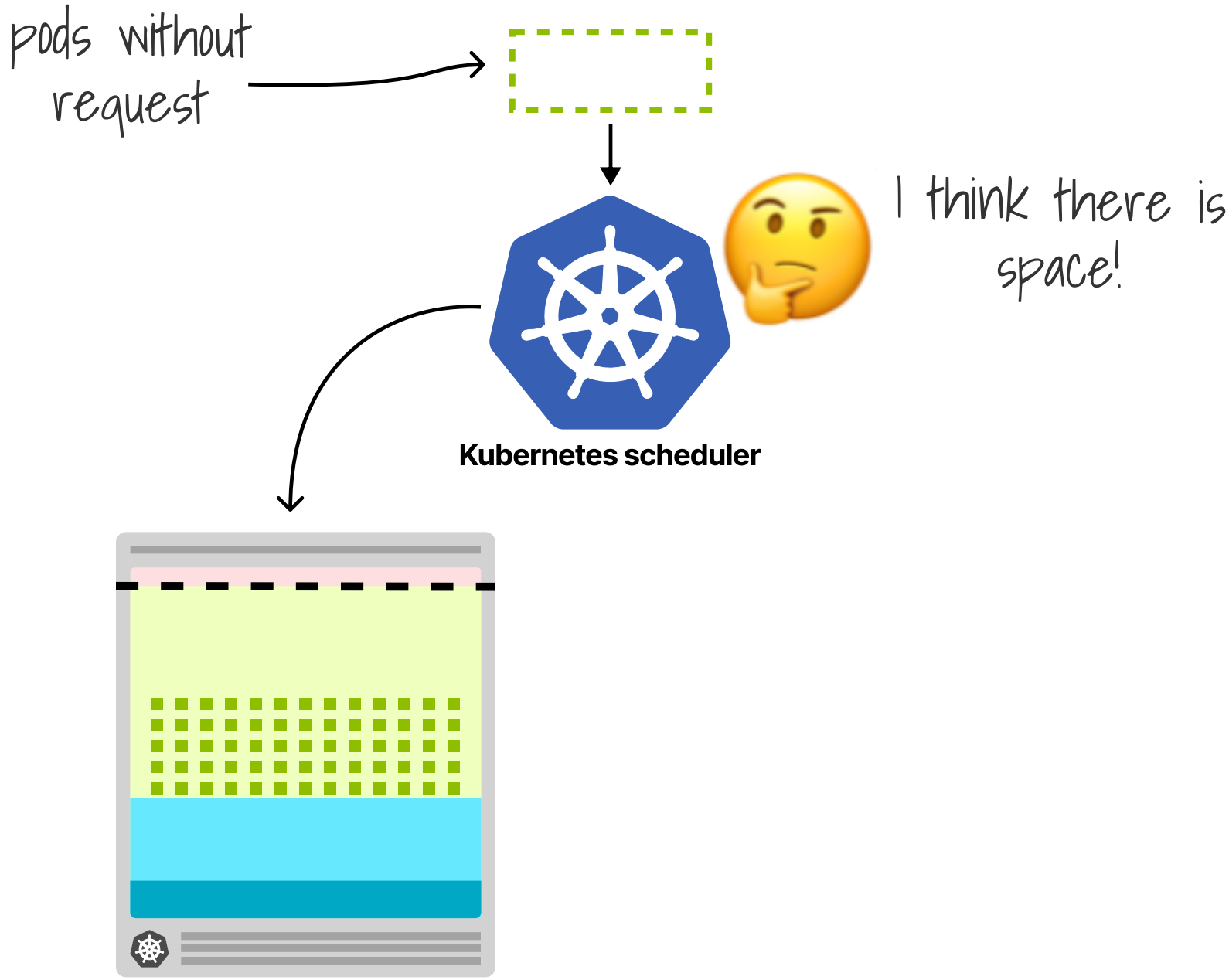

can exhaust all memory
Node memory
Container
512Mb
Burstable QoS class

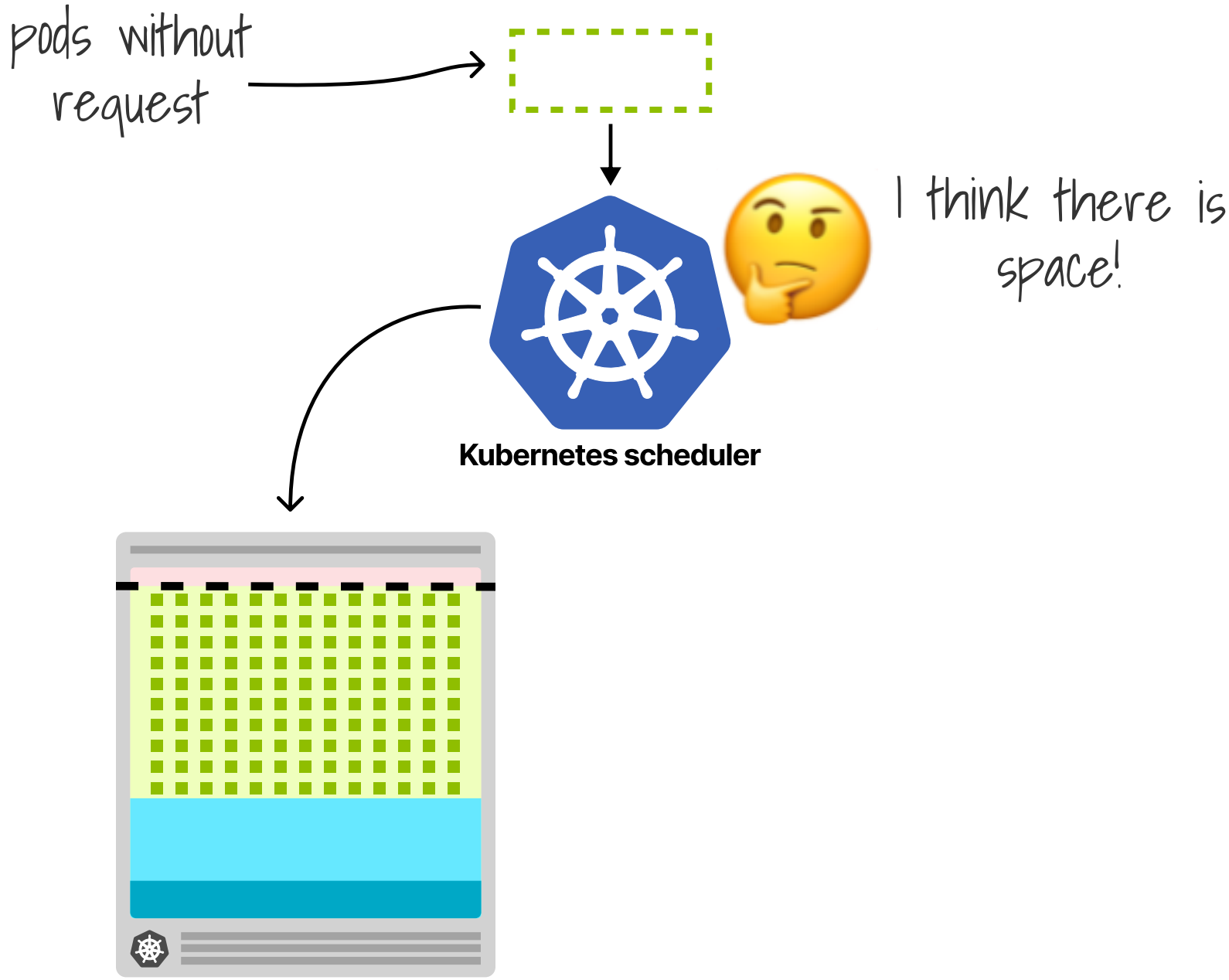
burstable pods can expand and use more resources

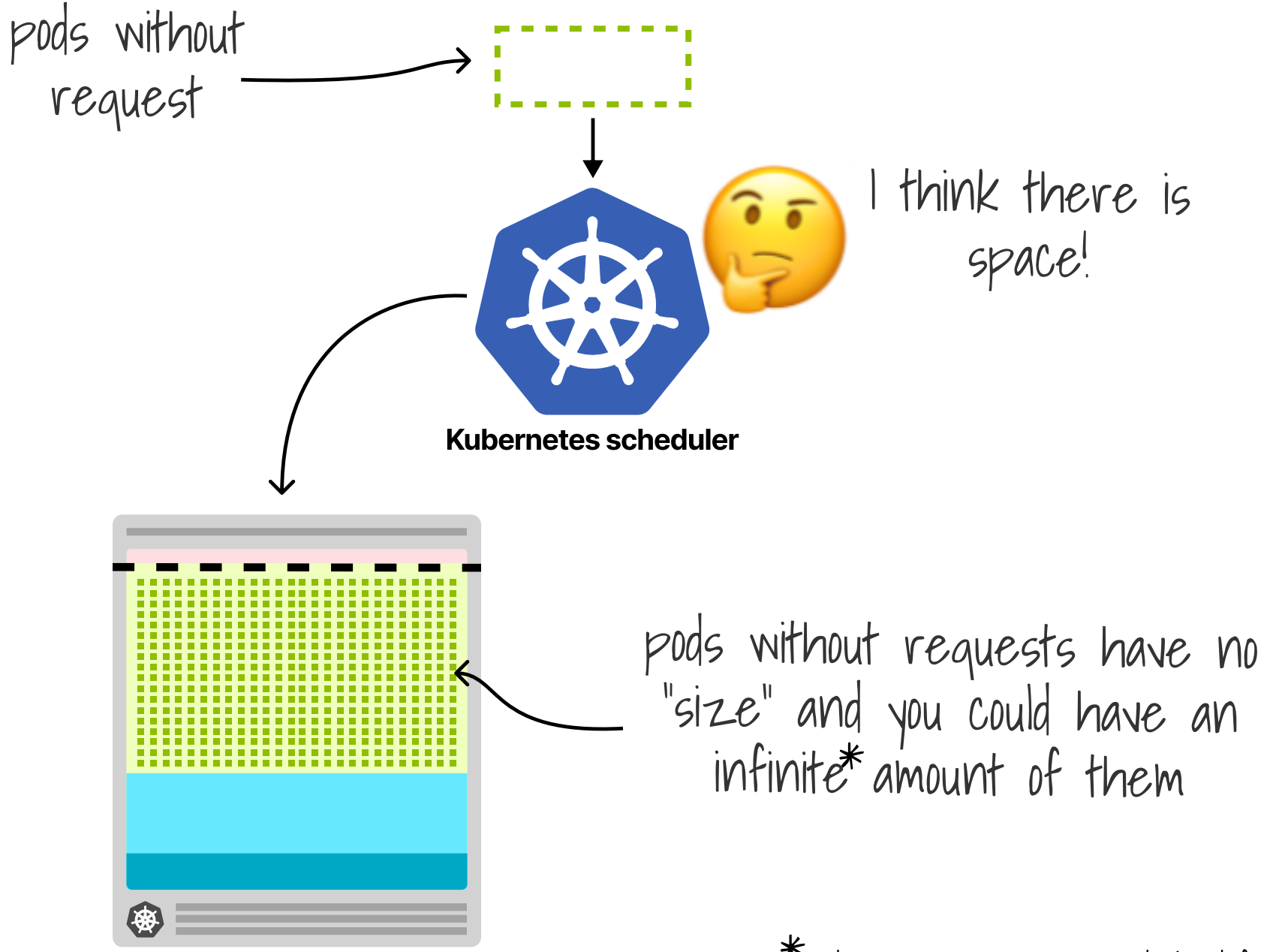


you could also oversubscribe the node


pods without request
I think there is space!
Kubernetes scheduler


pods without request
I think there is space!
Kubernetes scheduler


pods without request
I think there is space!
Kubernetes scheduler


pods without request
I think there is space!
Kubernetes scheduler
pods without requests have no "size" and you could have an infinite* amount of them
*nodes can host up to 250 pods by default