How To Blow Up a Kubernetes Cluster

Resource management for application developers

Felix Hoffmann

Hello 👋

My name is Felix

Software engineer @iteratec

@felixmhoffmann

Types of compute resources

- CPU
- Memory
- Ephemeral storage
- PID limiting
- …

Types of compute resources

- CPU
- Memory
- Ephemeral storage
- PID limiting
- …

Requests and limits

- Request: Amount of memory/CPU that is guaranteed for you container
- Limit: Amount of memory/CPU that your container cannot exceed

Resource units

- 1 CPU unit = 1 core (physical or virtual)
  - Fractions (0.5)
  - Millicpu (100m)
  - 1000m = 1 CPU unit
- 1 memory unit = 1 byte
  - E, P, T, G, M, k
  - Ei, Pi, Ti, Gi, Mi, Ki

Requests and 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"
```

What happens when a Pod exceeds its memory limit?

- Out of memory kill (OOMKill)
- Memory is an incompressible resource

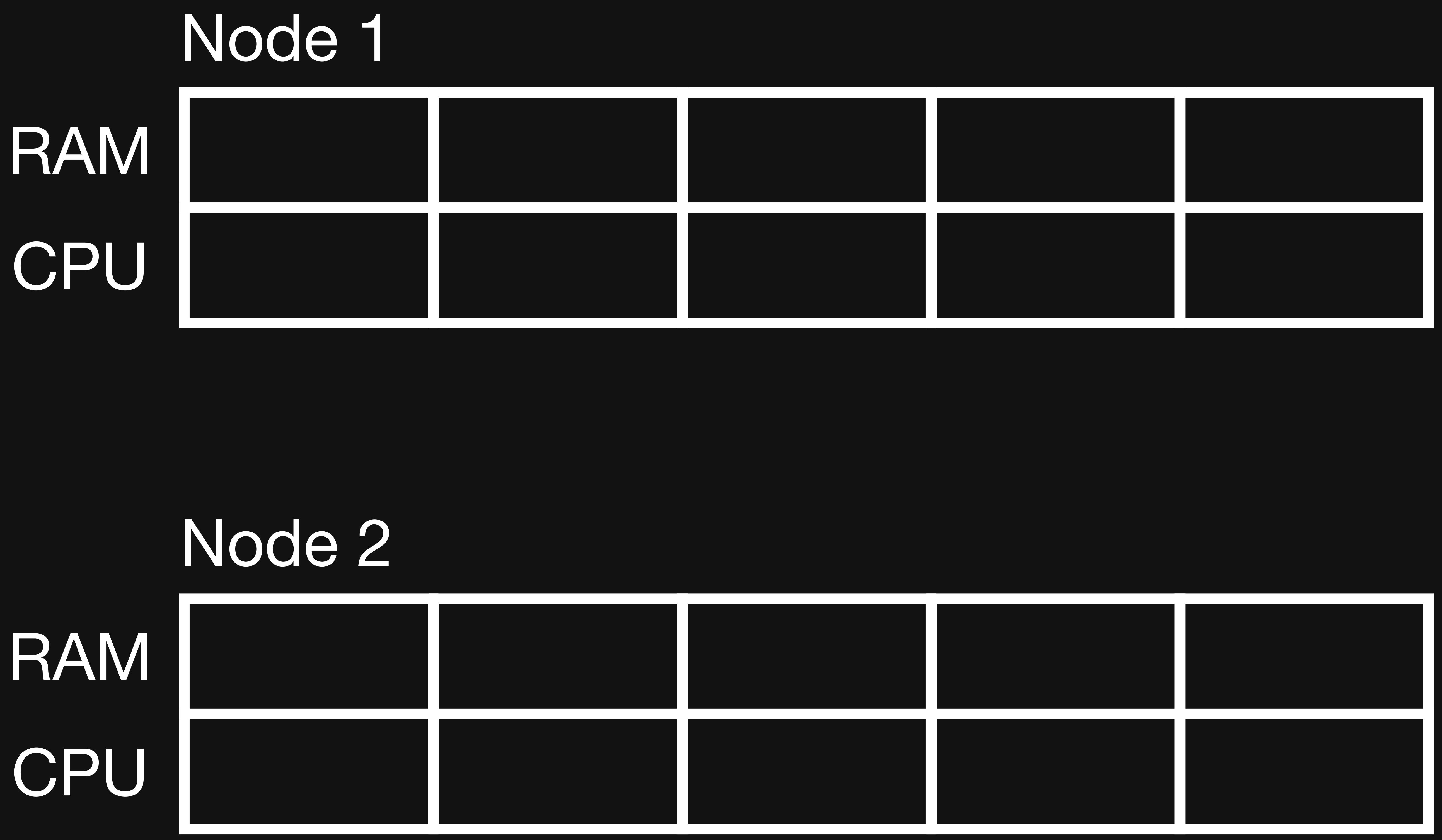
What happens when a Pod exceeds its memory limit?

- Out of memory kill (OOMKill)
- Memory is an incompressible resource

What happens when a Pod exceeds its CPU limit?

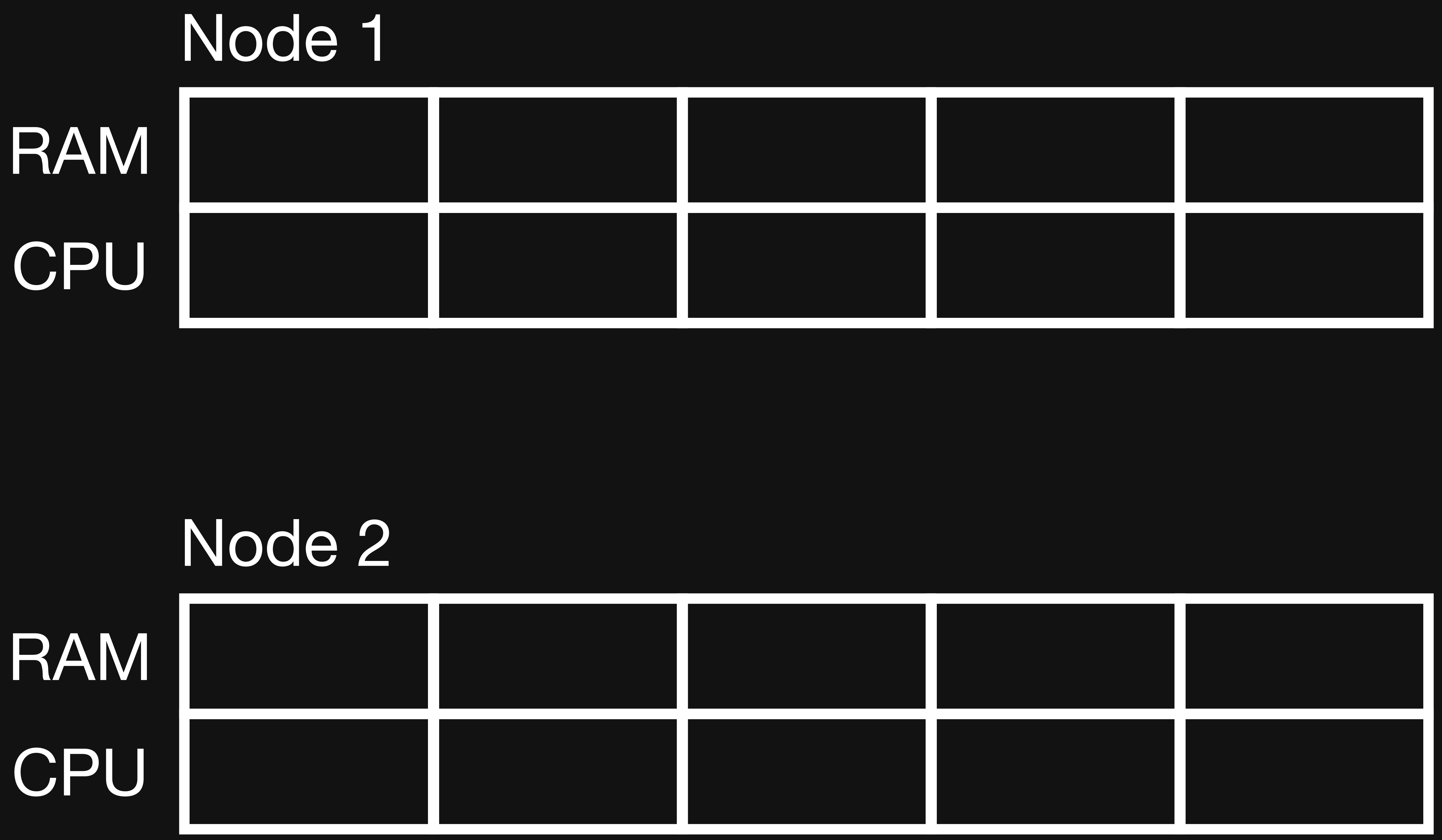
- Throttling
- No termination
- CPU is a compressible resource

How Pods are scheduled



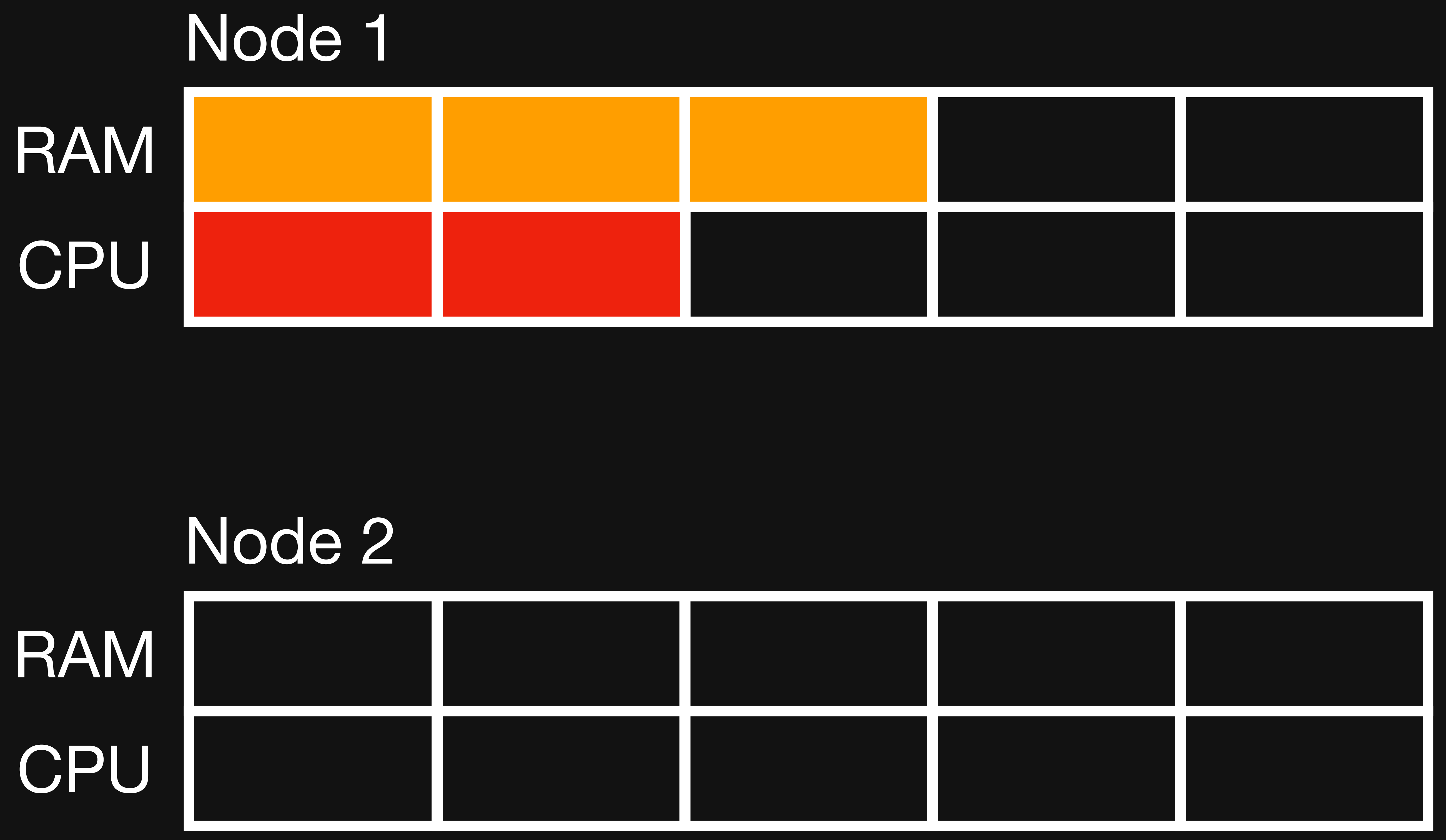
How Pods are scheduled

```
resources:
  requests:
    memory: "3Gi"
    cpu: 2
  limits:
    memory: "5Gi"
    cpu: 2
```

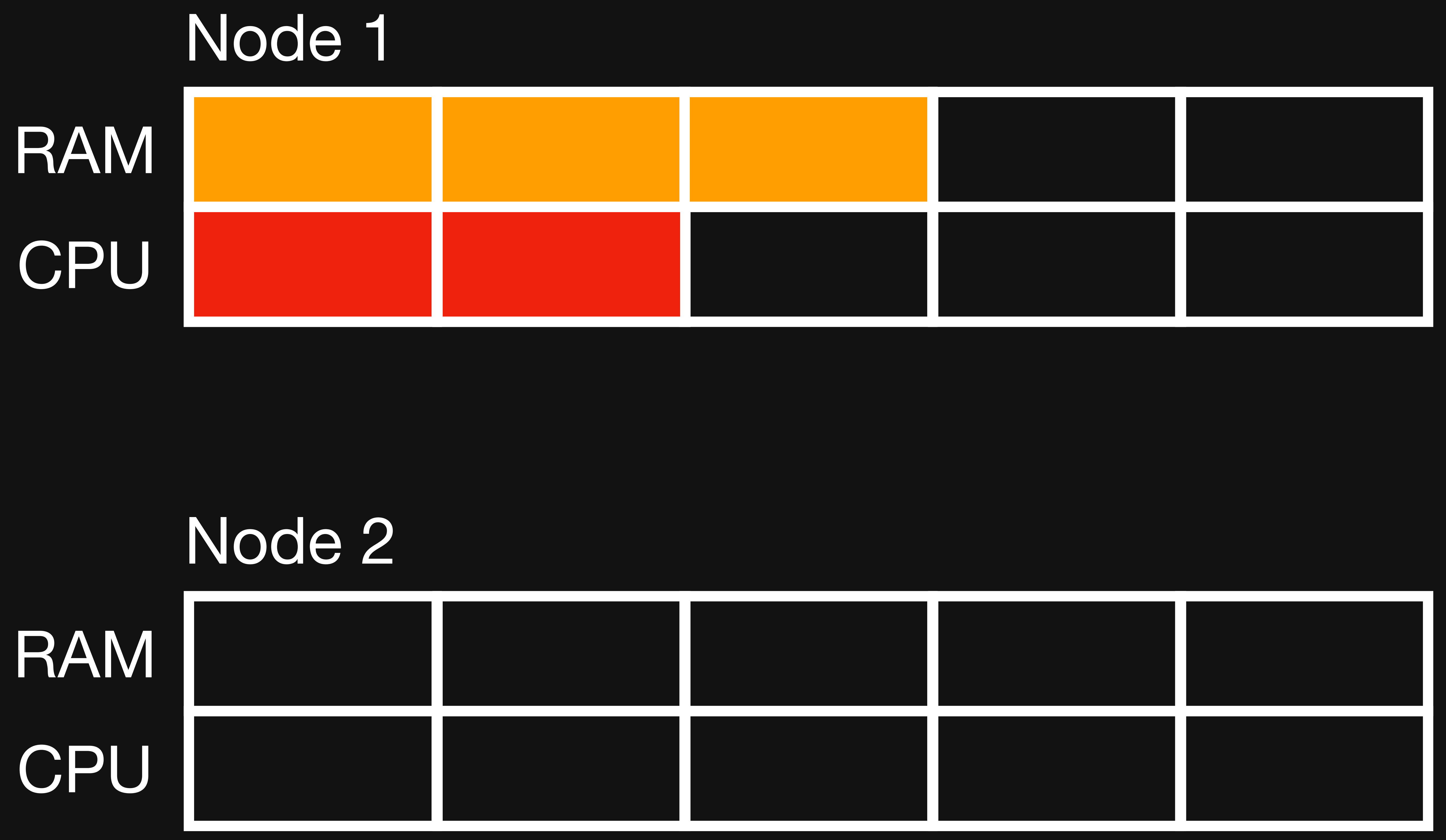


How Pods are scheduled

```
resources:
  requests:
    memory: "3Gi"
    cpu: 2
  limits:
    memory: "5Gi"
    cpu: 2
```

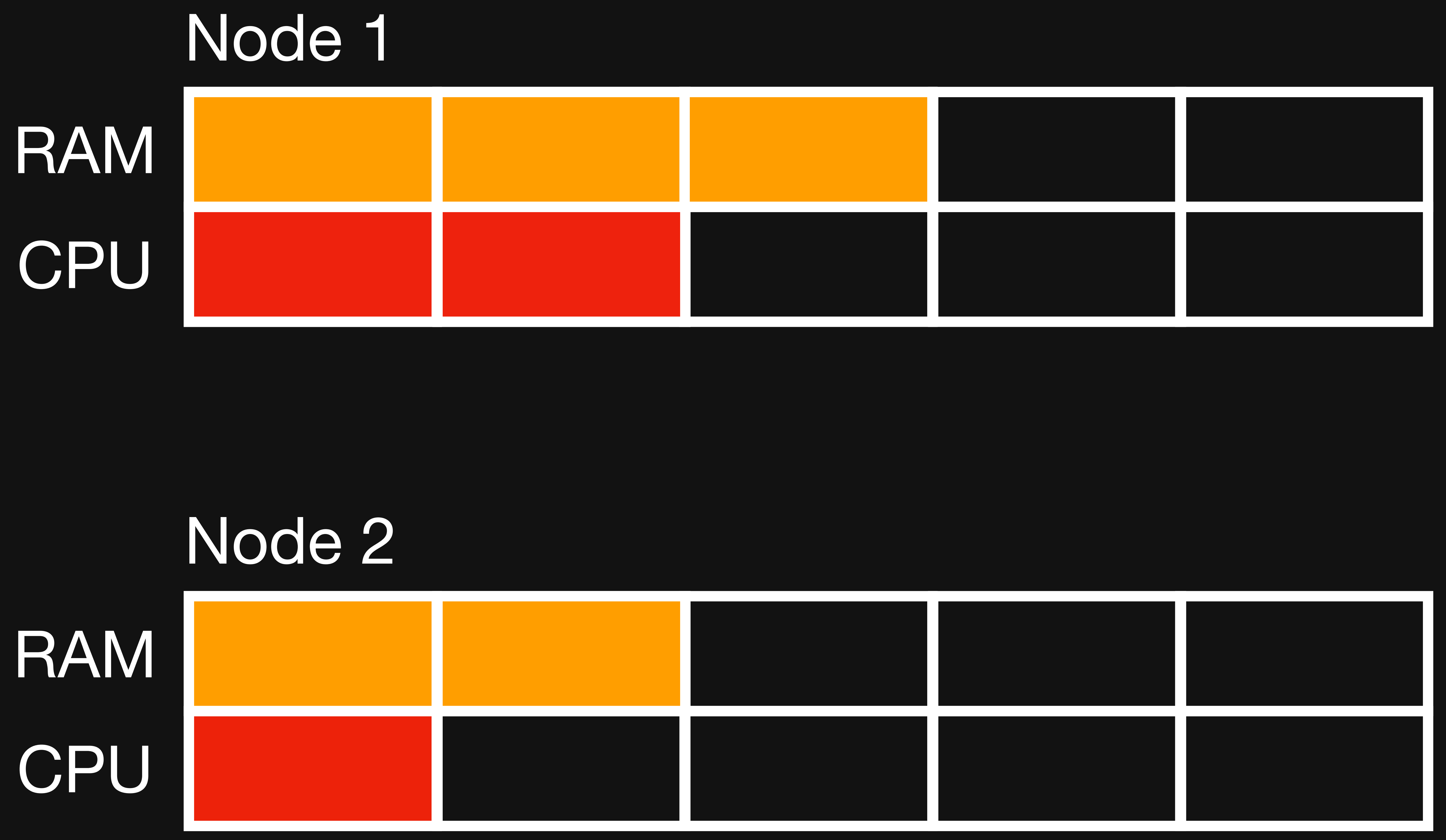


How Pods are scheduled



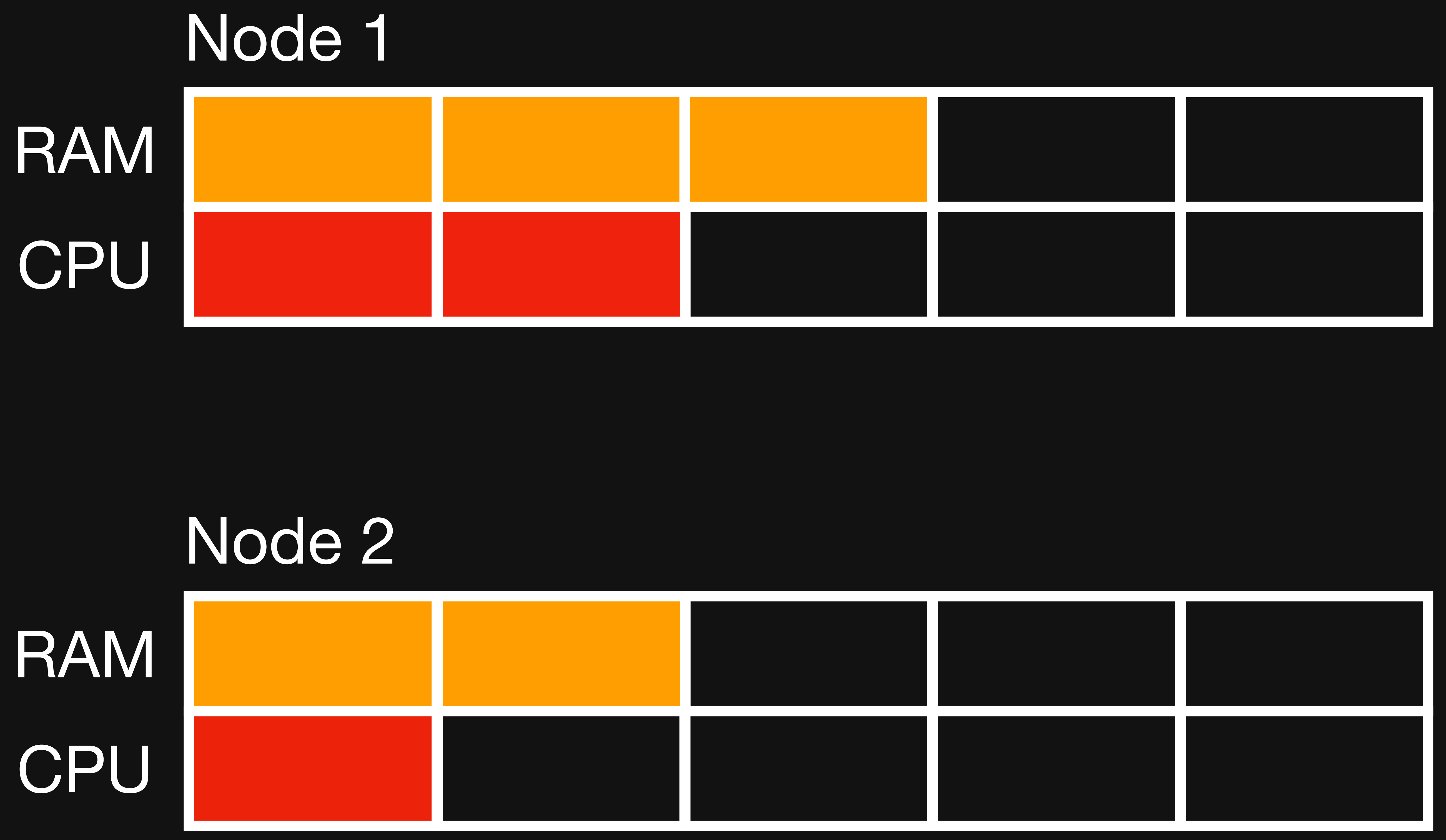
```yaml
resources:
  requests:
    memory: "2Gi"
    cpu: 1
  limits:
    memory: "4Gi"
    cpu: 1
```

How Pods are scheduled



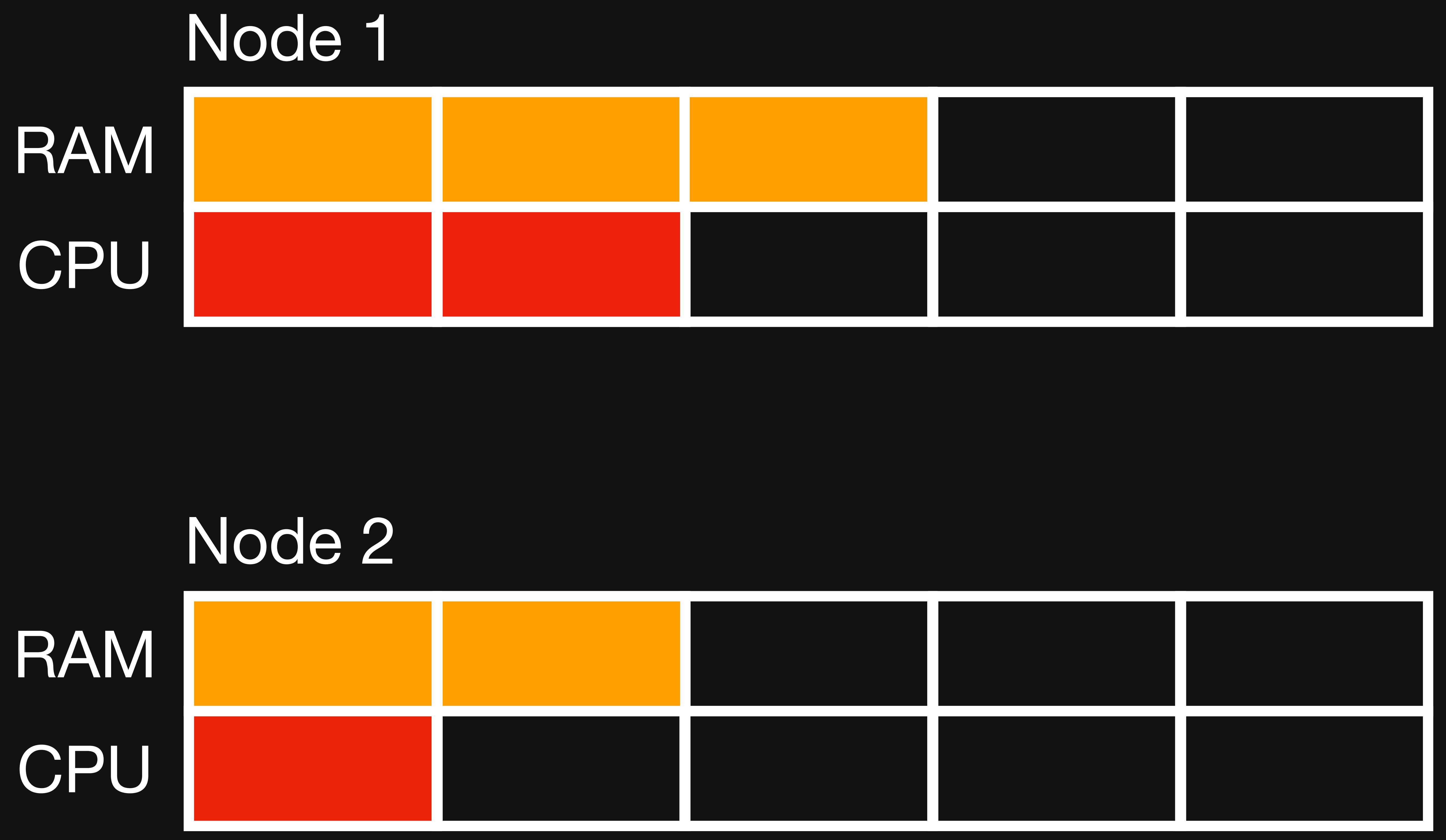
```
resources:
  requests:
    memory: "2Gi"
    cpu: 1
  limits:
    memory: "4Gi"
    cpu: 1
```

How Pods are scheduled



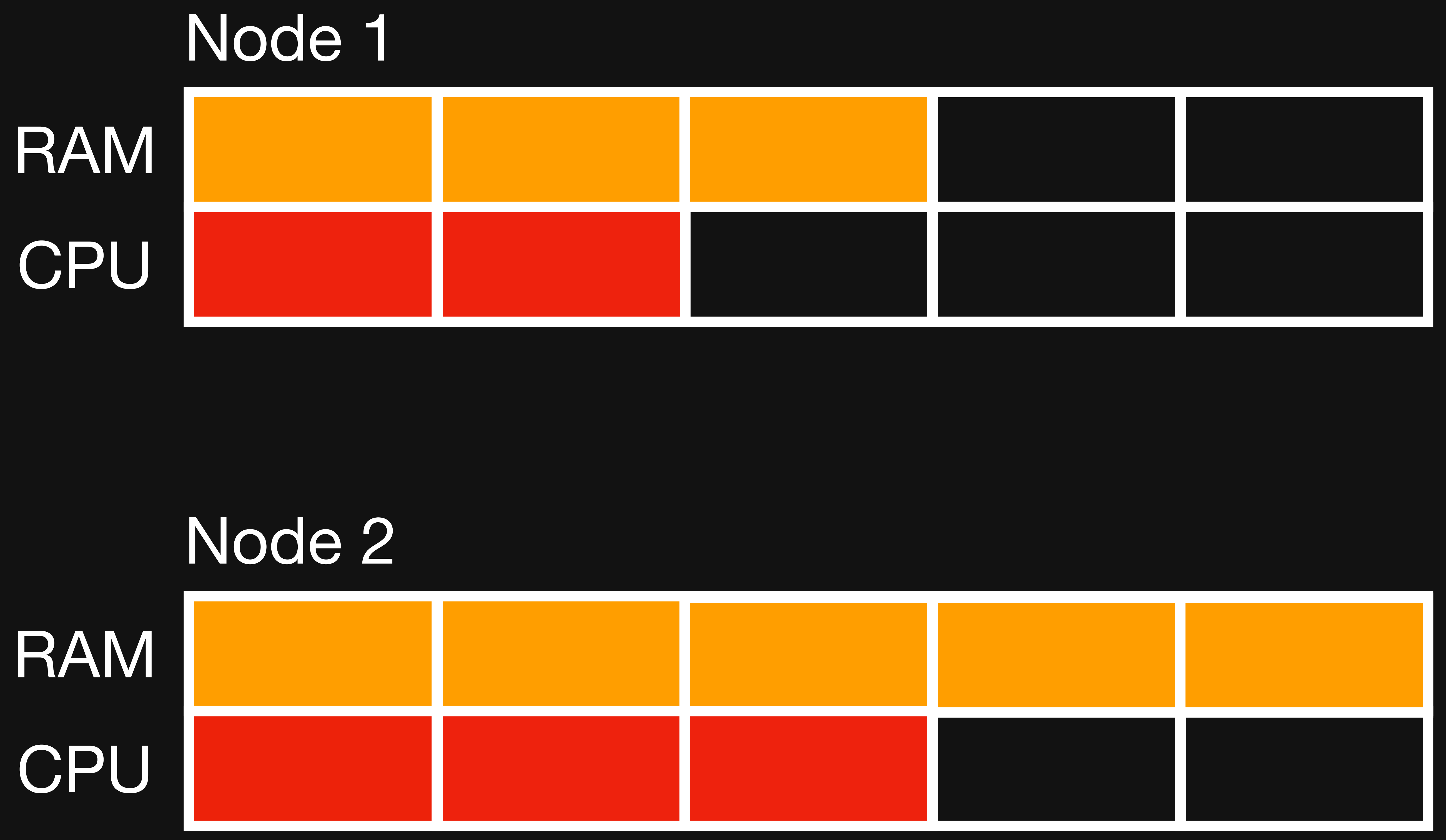
```yaml
resources:
  requests:
    memory: "3Gi"
    cpu: 2
  limits:
    memory: "3Gi"
    cpu: 2
```

How Pods are scheduled



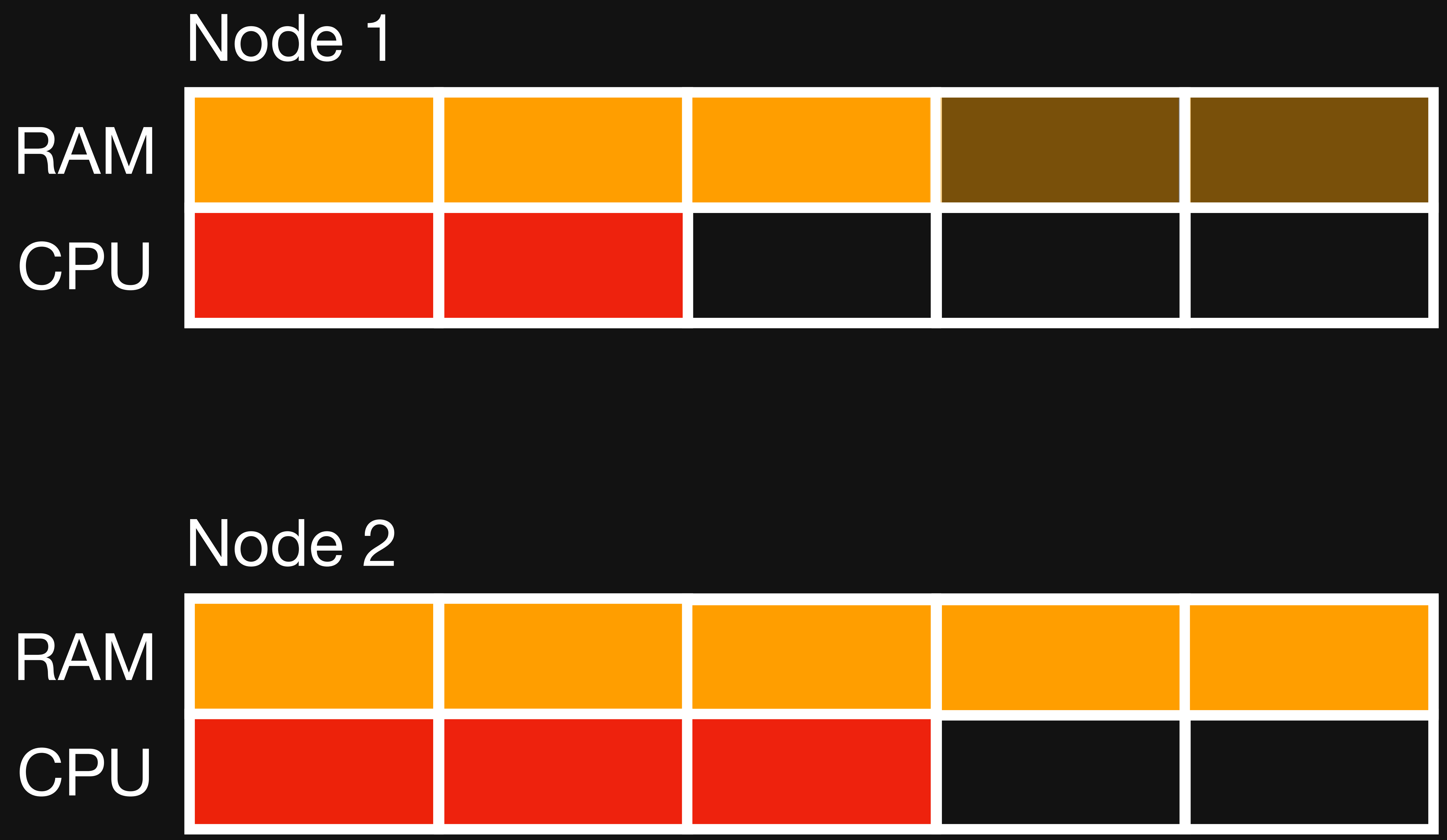
```yaml
resources:
  requests:
    memory: "3Gi"
    cpu: 2
  limits:
    memory: "3Gi"
    cpu: 2
```

How Pods are scheduled



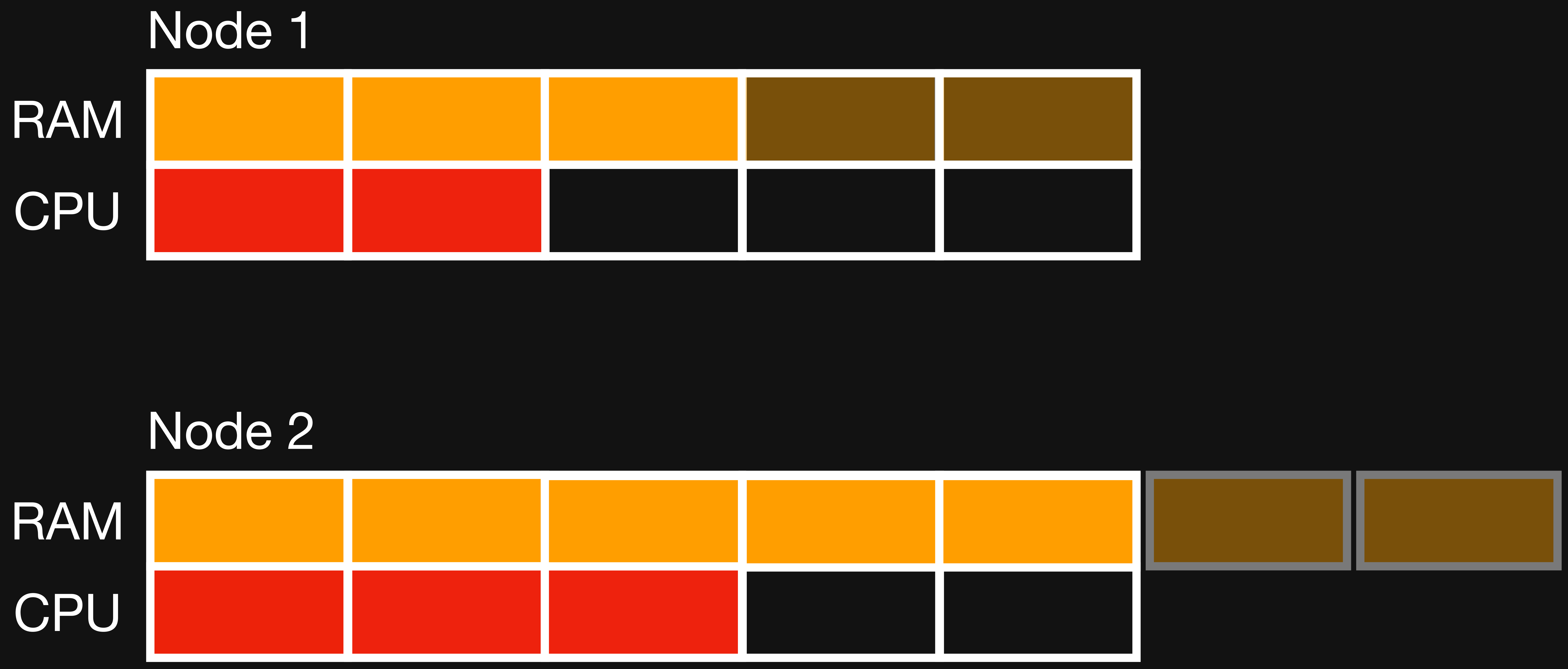
```
resources:
  requests:
    memory: "3Gi"
    cpu: 2
  limits:
    memory: "3Gi"
    cpu: 2
```

How Pods are scheduled



Total limits:
5 Gi RAM
2 CPU

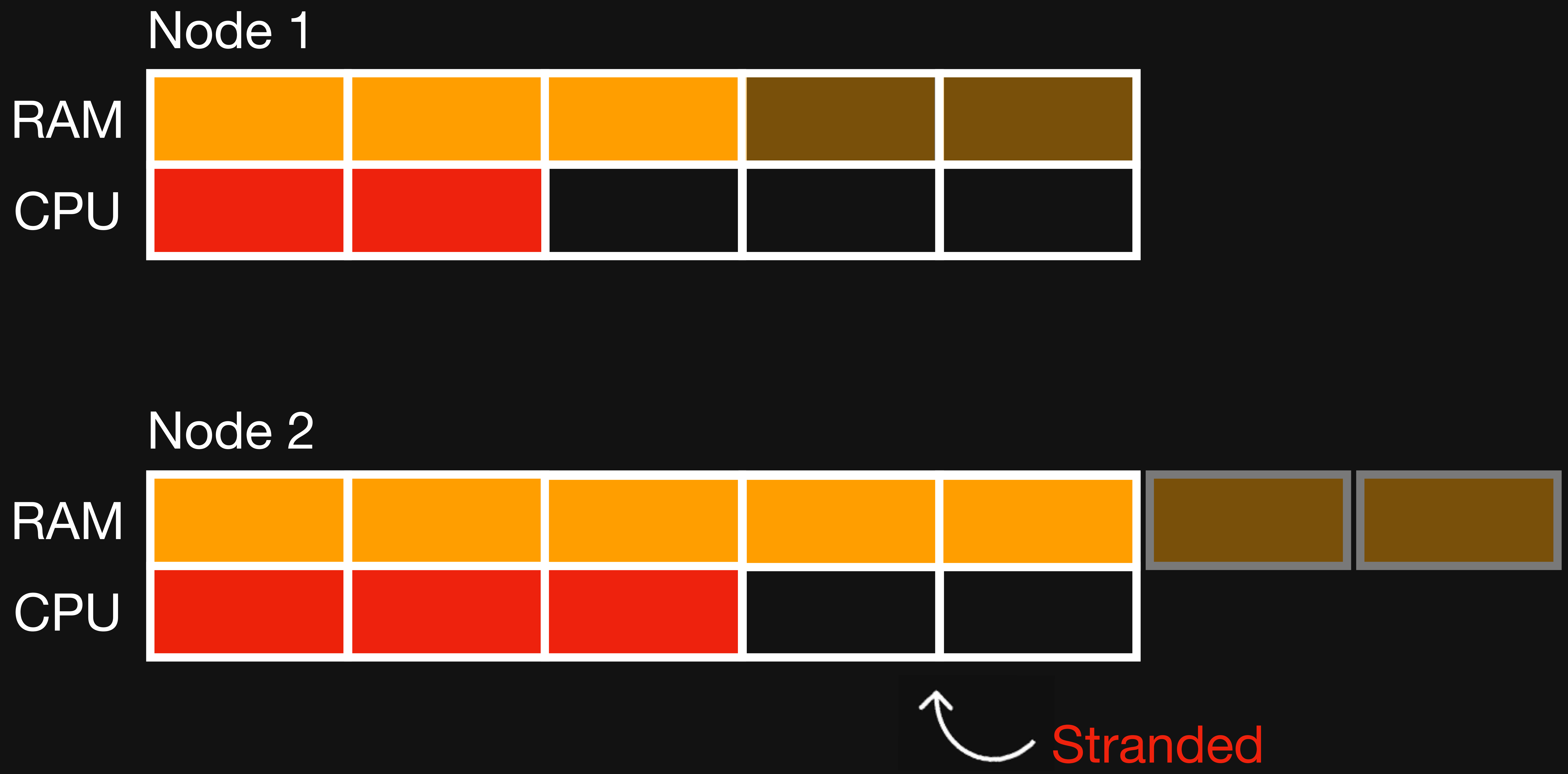
How Pods are scheduled



Total limits:
5 Gi RAM
2 CPU

Total limits:
7 Gi RAM
3 CPU

How Pods are scheduled



Total limits:
5 Gi RAM
2 CPU

Total limits:
7 Gi RAM
3 CPU

What happens when a Node runs out of memory?

- Kubernetes terminates pods that exceed their memory requests
- Limits don’t matter

How To Blow Up a Kubernetes Cluster

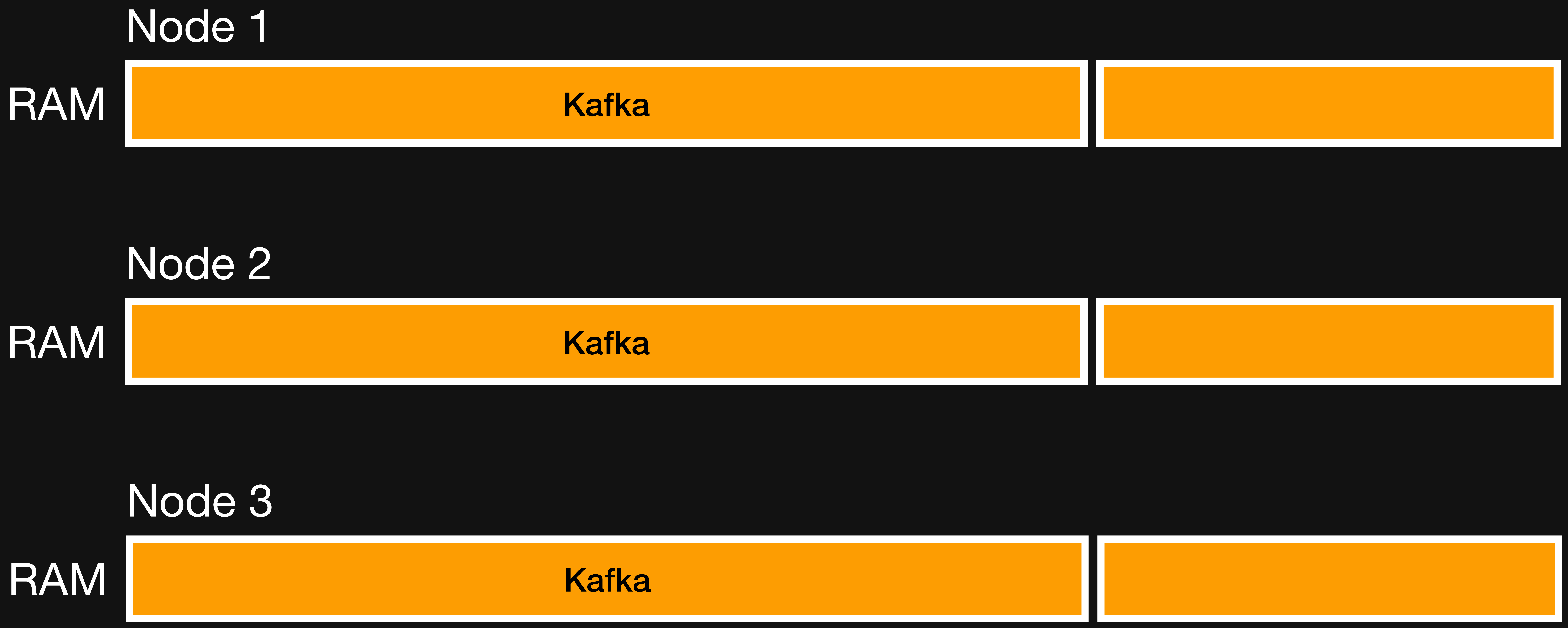
- Ingredients:
  - a couple of microservices
  - Kafka
  - barely enough memory

About Kafka

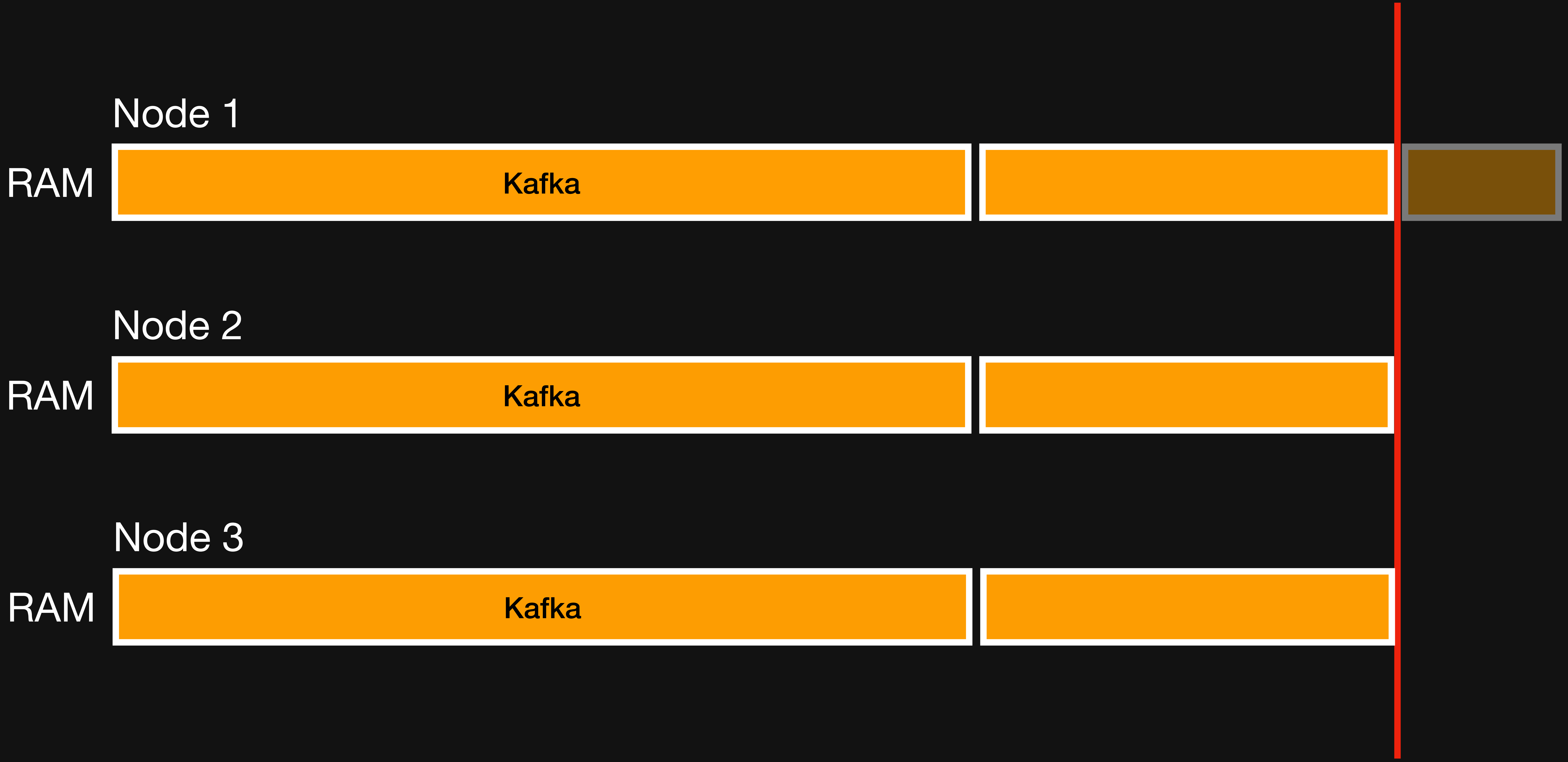
- Distributed event streaming
- Keeps state in memory

- Our Kafka Pods where using ~2.8GiB memory
  - request: 3 GiB
  - limit: 8 GiB

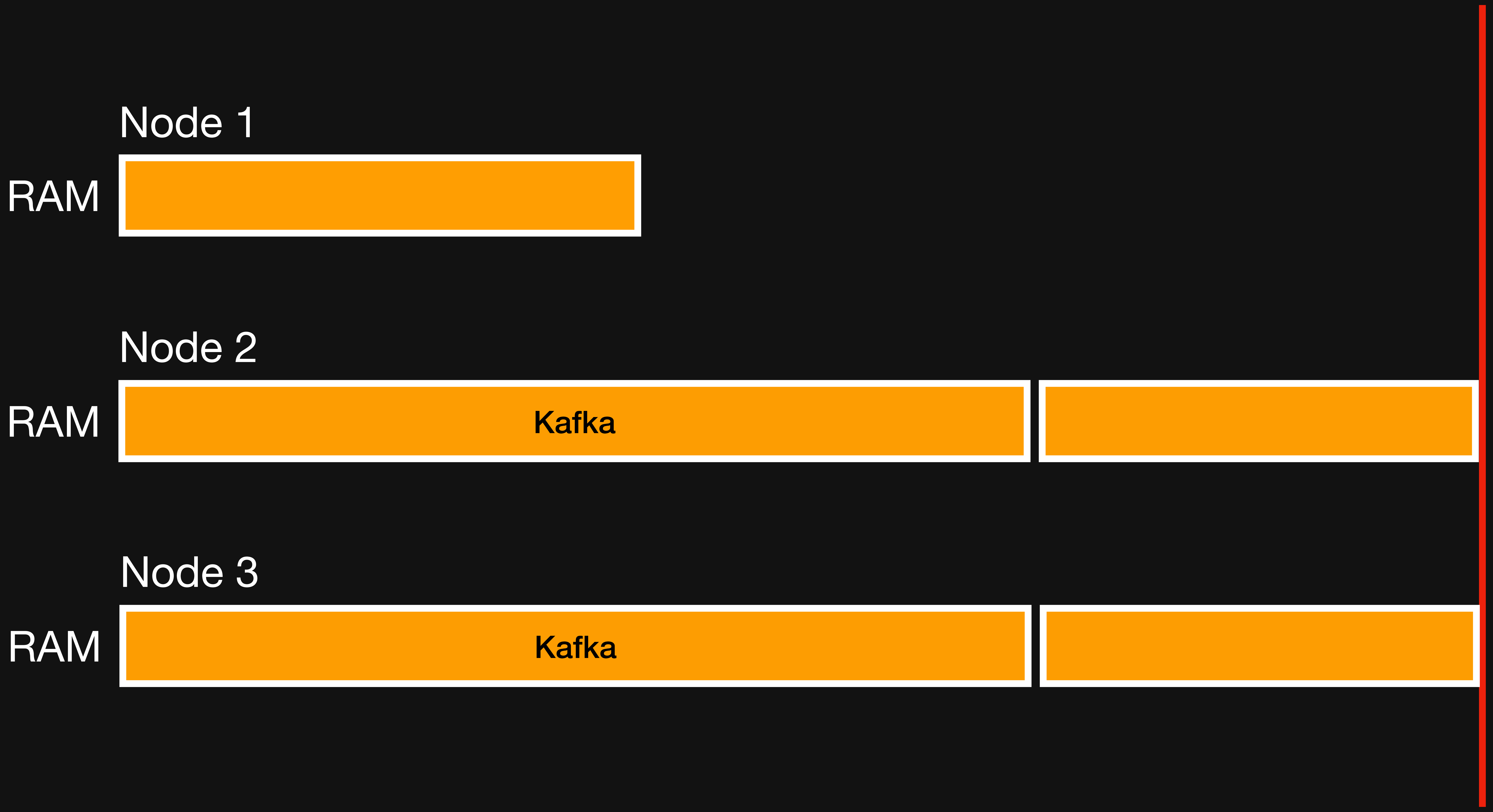
The Incident



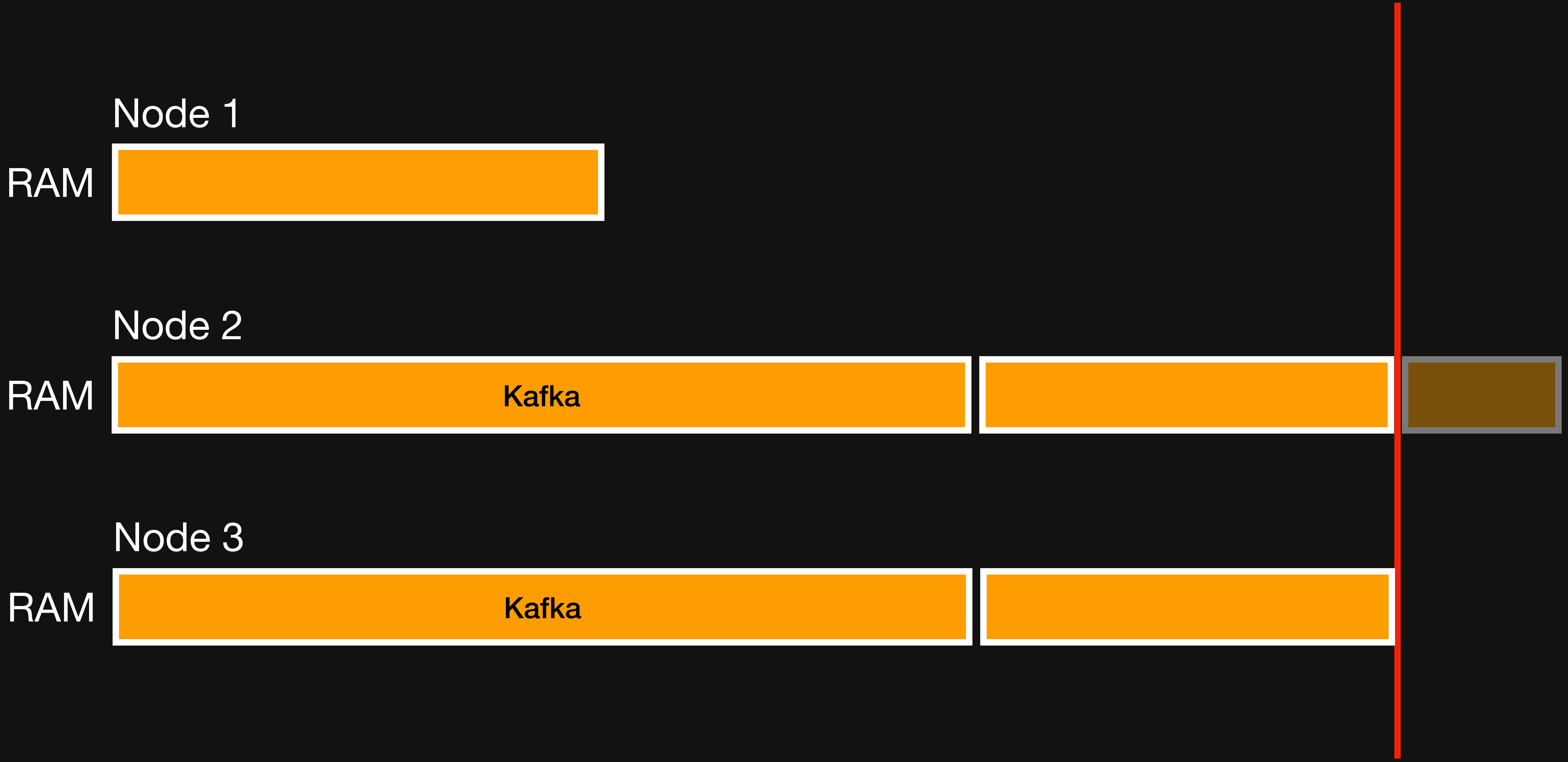
The Incident



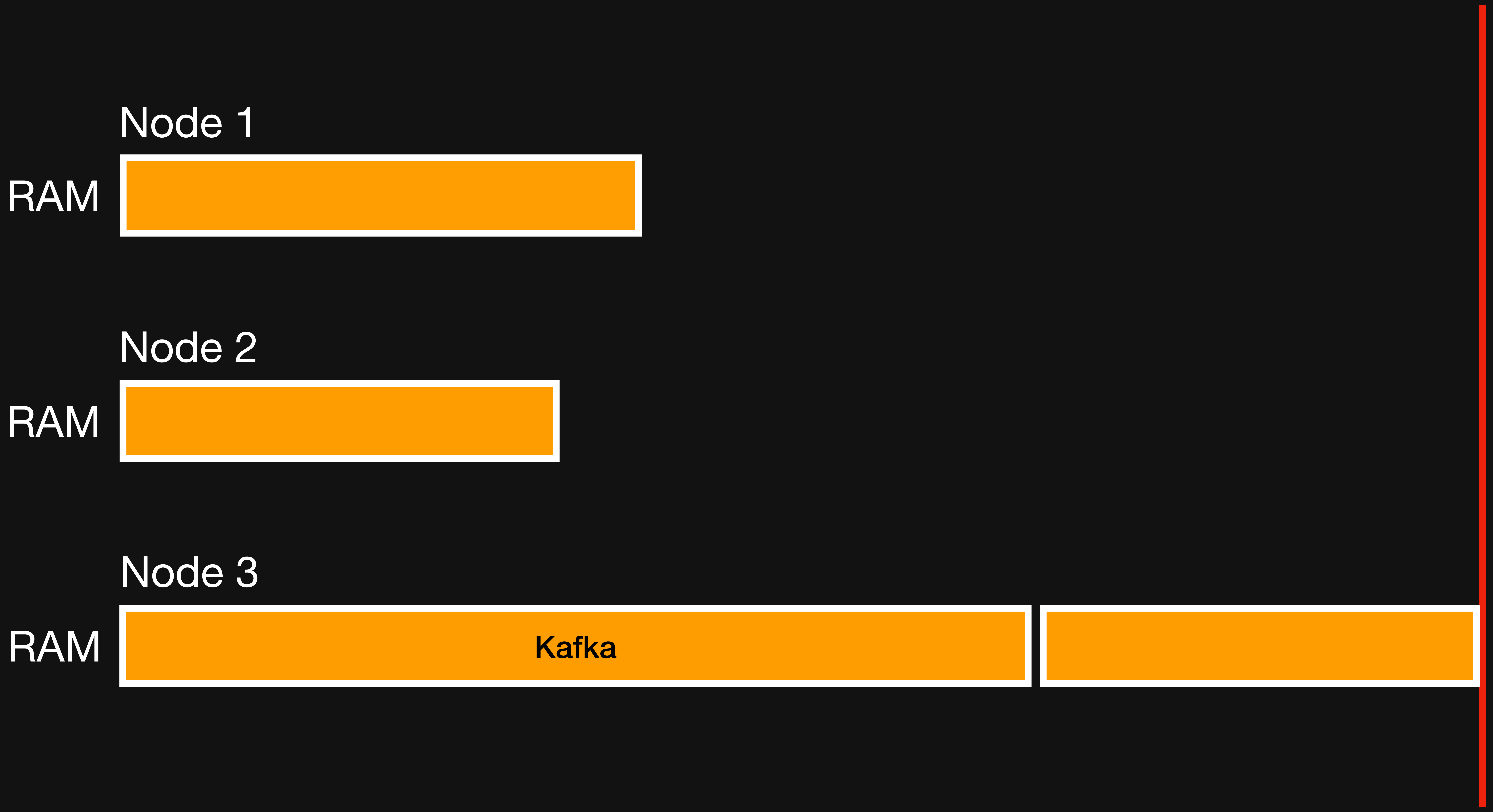
The Incident



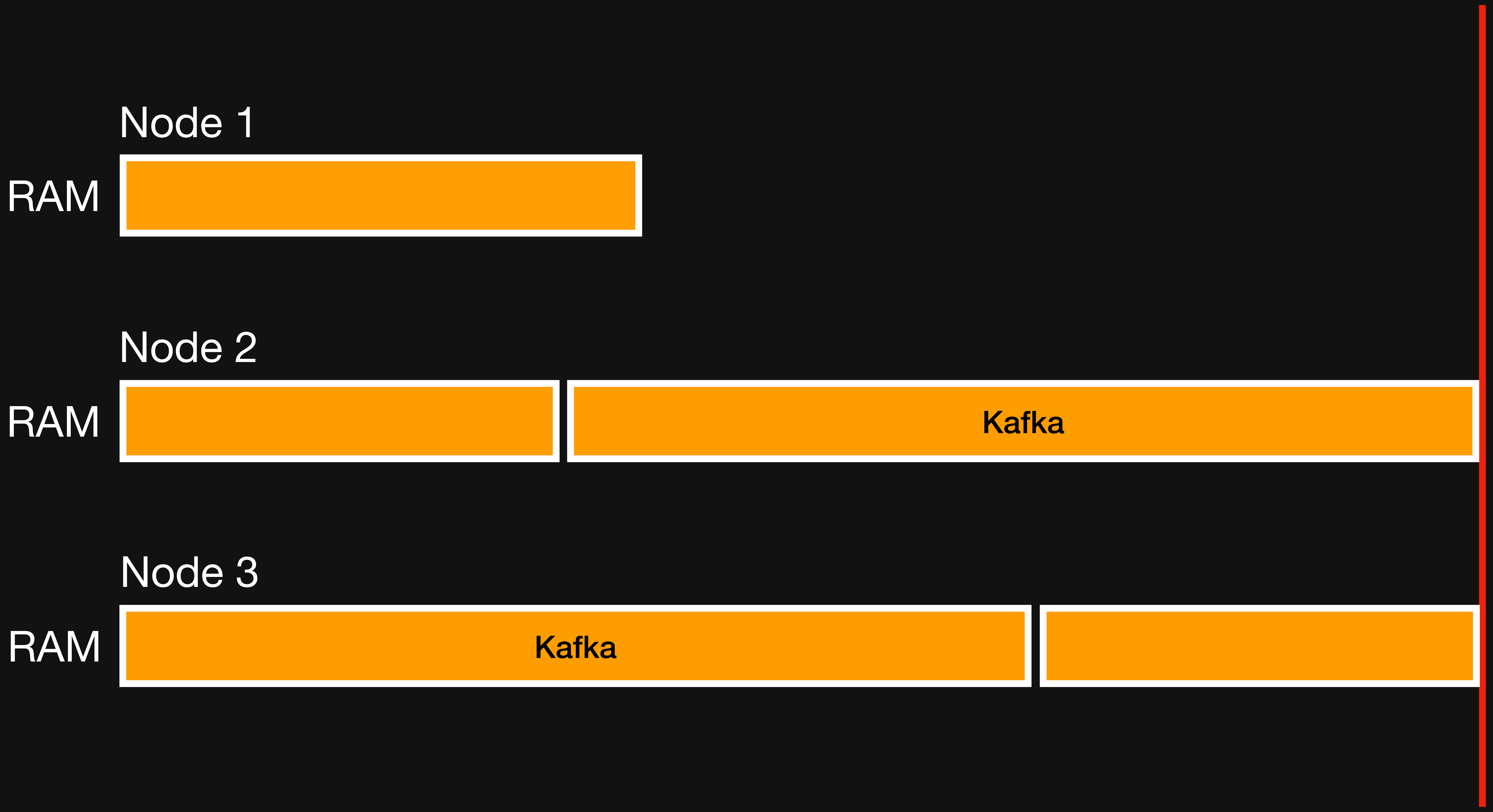
The Incident



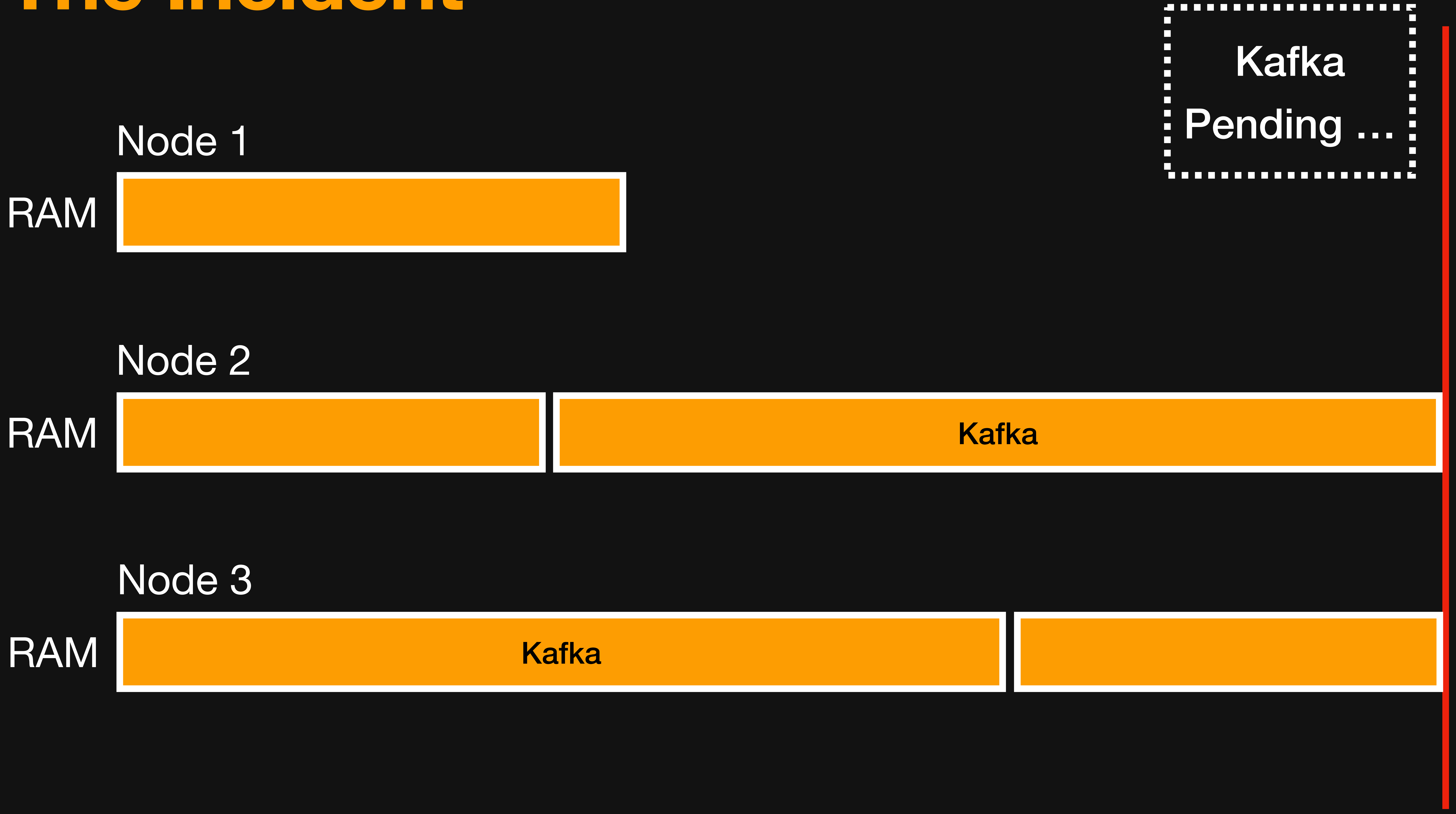
The Incident



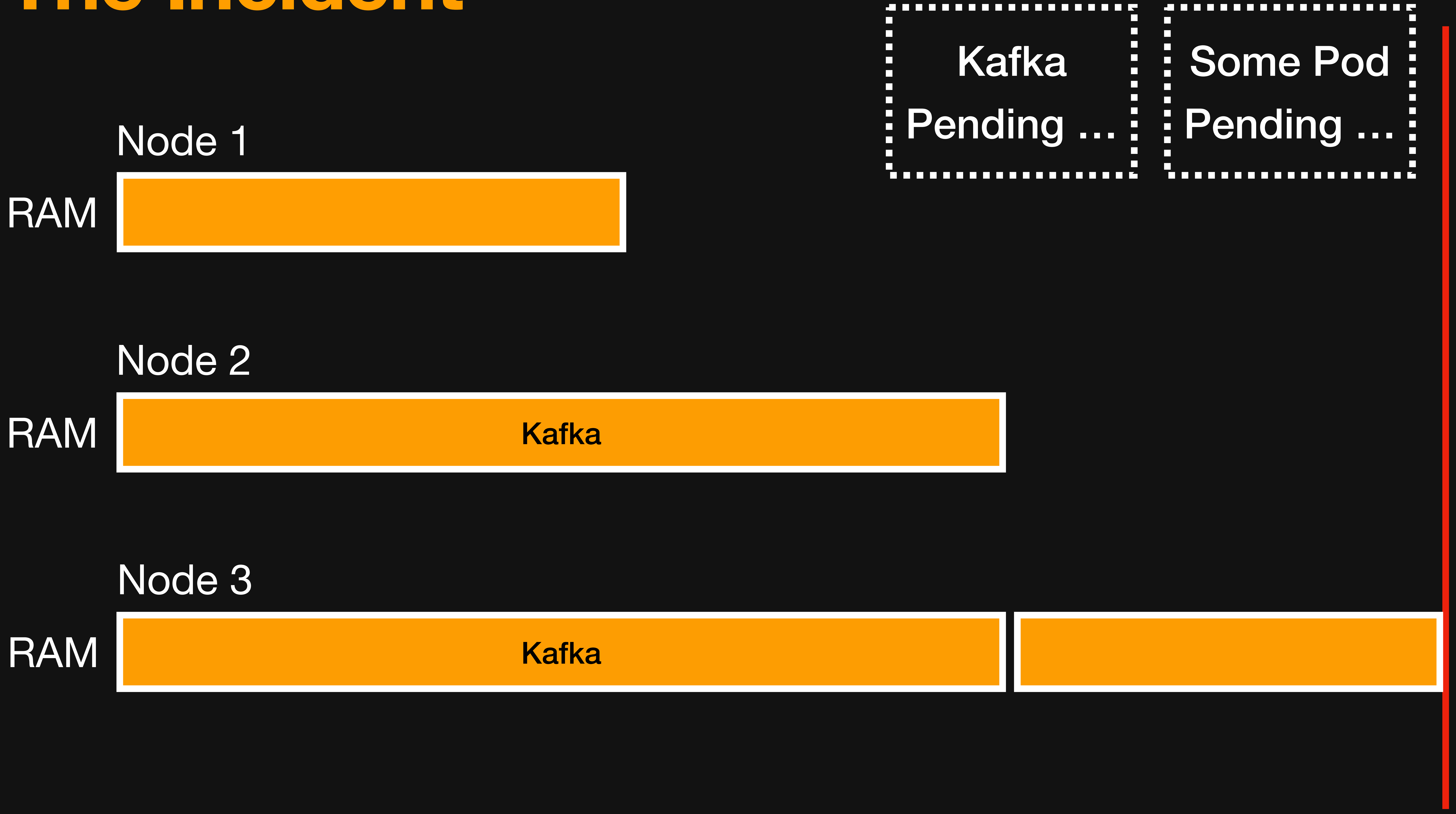
The Incident



The Incident



The Incident



Lessons learned

- Memory request == memory limit
- Do not overcommit on memory
- Clusters need room to operate
- Memory is an incompressible resources

What about CPU?

- CPU is a compressible resource
- Resource management is different to memory

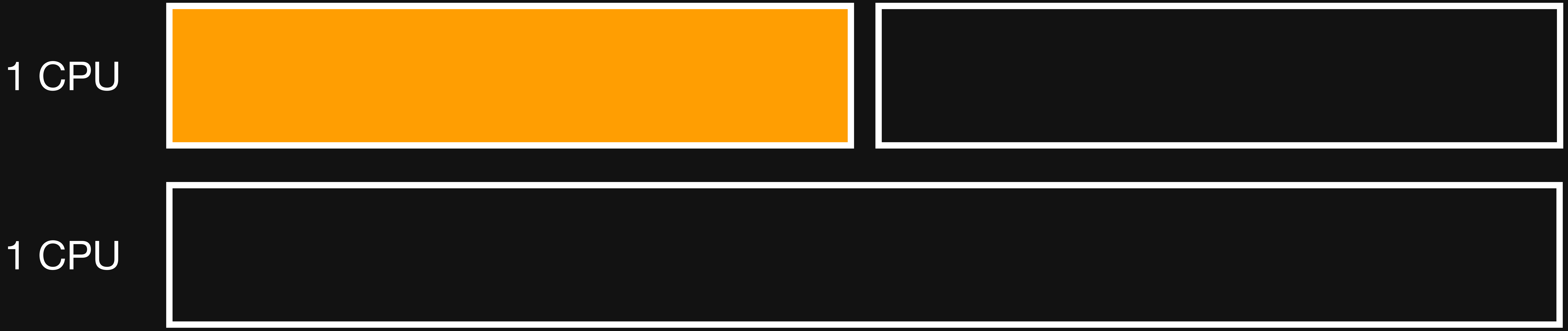
About containers without CPU limits:

“The Container has no upper bound on the CPU resources it can use. The Container could use all of the CPU resources available on the Node where it is running.”

Kubernetes Documentation

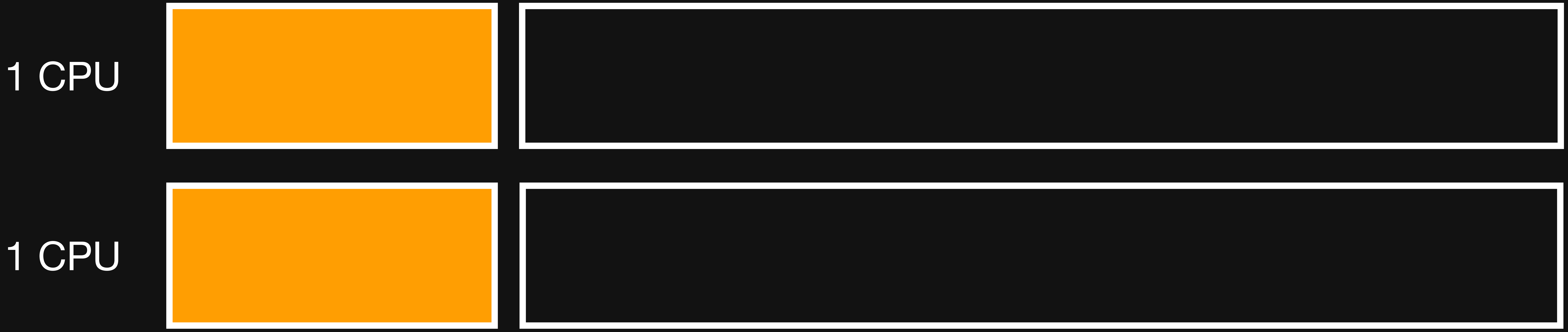
Best/worst case with CPU limit

```
resources:
  requests:
    cpu: "500m"
  limits:
    cpu: "500m"
```



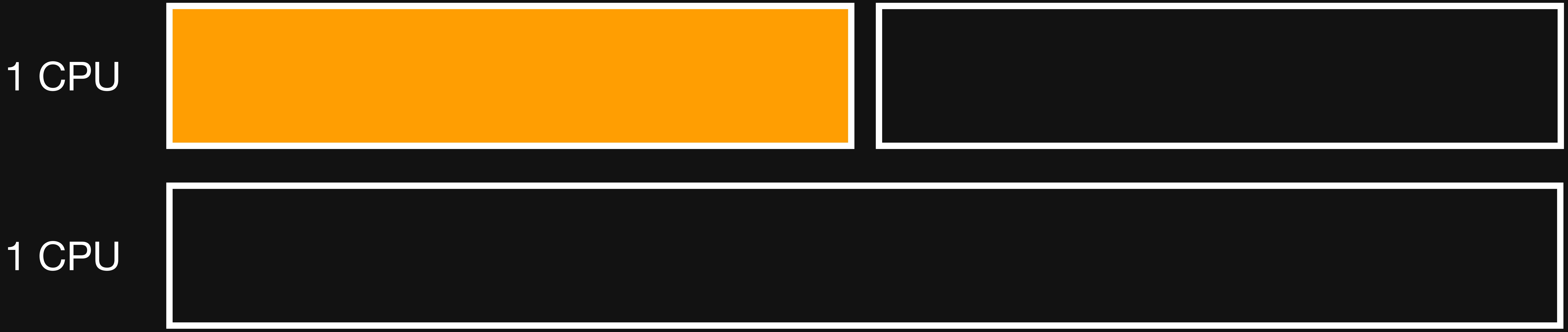
Best/worst case with CPU limit

```
resources:
  requests:
    cpu: "500m"
  limits:
    cpu: "500m"
```



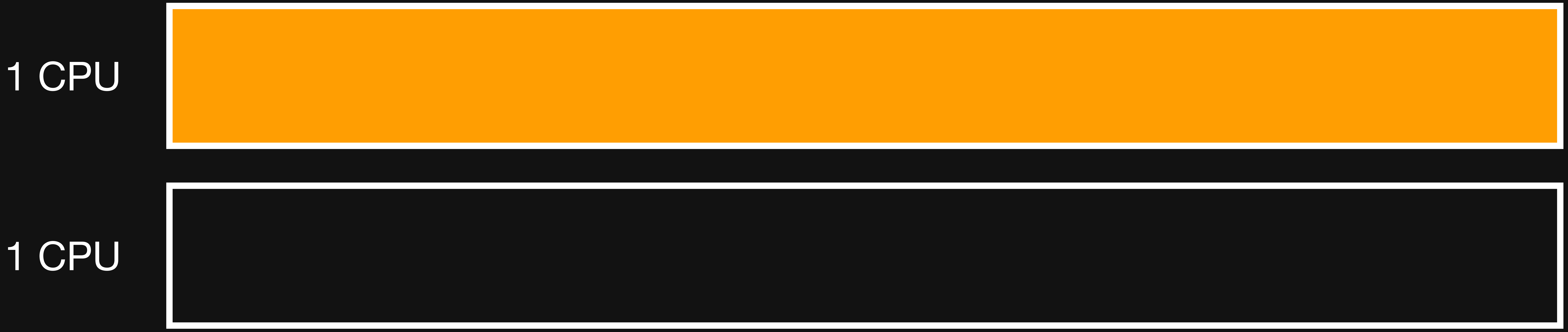
Worst case without CPU limit

```
resources:
  requests:
    cpu: "500m"
```



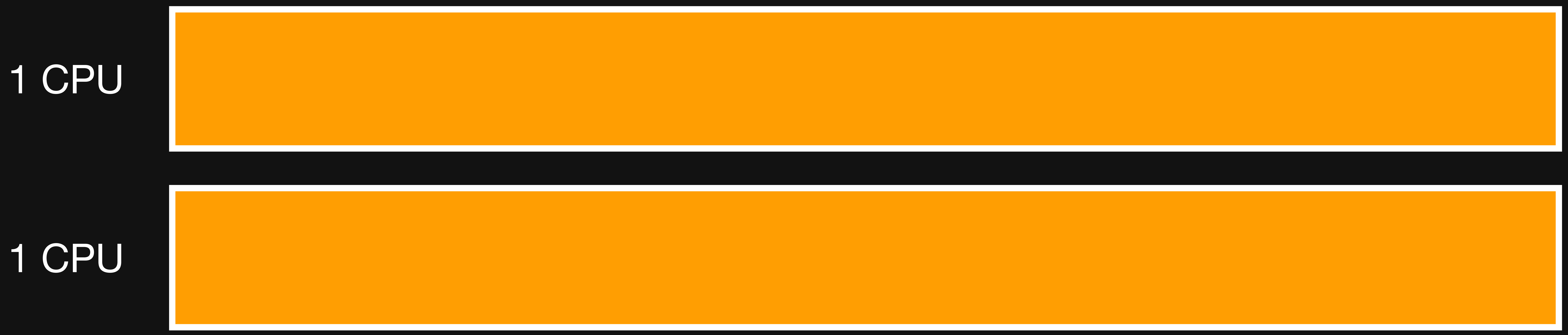
Best case without CPU limit

```
resources:
  requests:
    cpu: "500m"
```



Best case without CPU limit

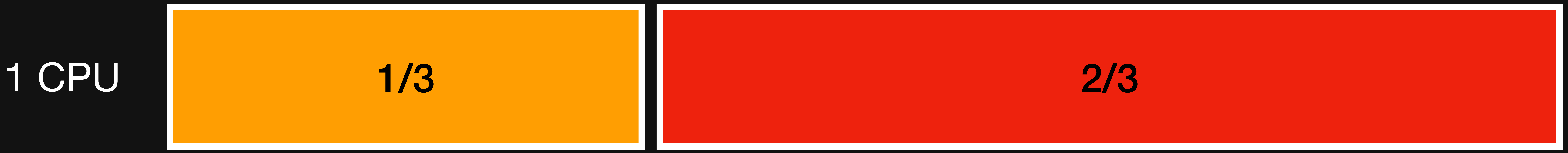
```yaml
resources:
  requests:
    cpu: "500m"
```



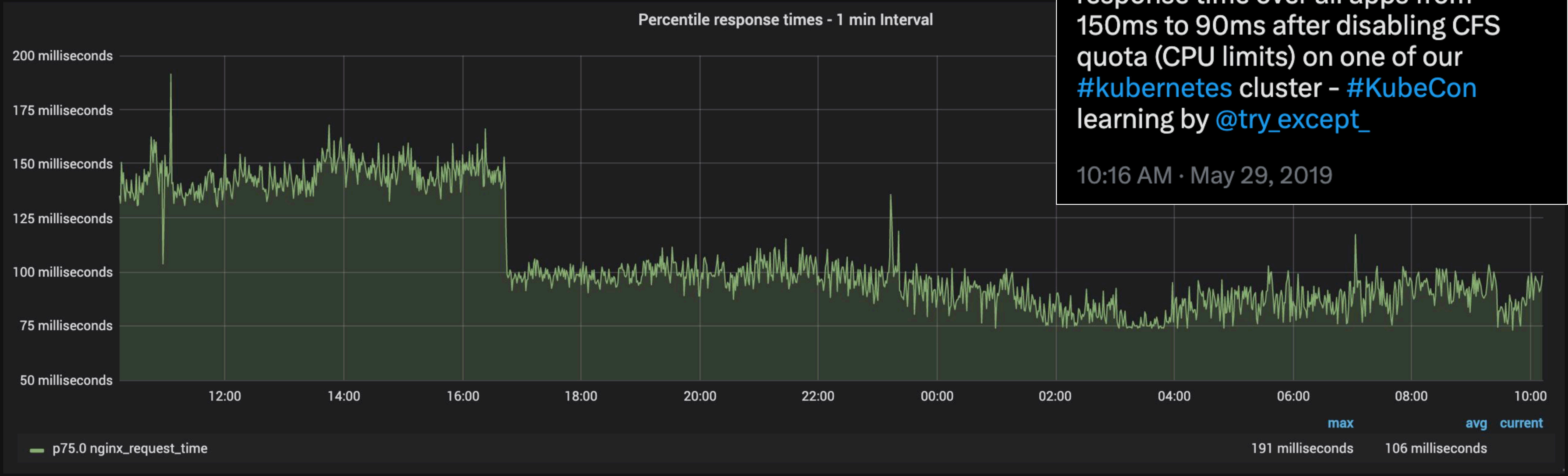
Requests determine CPU shares

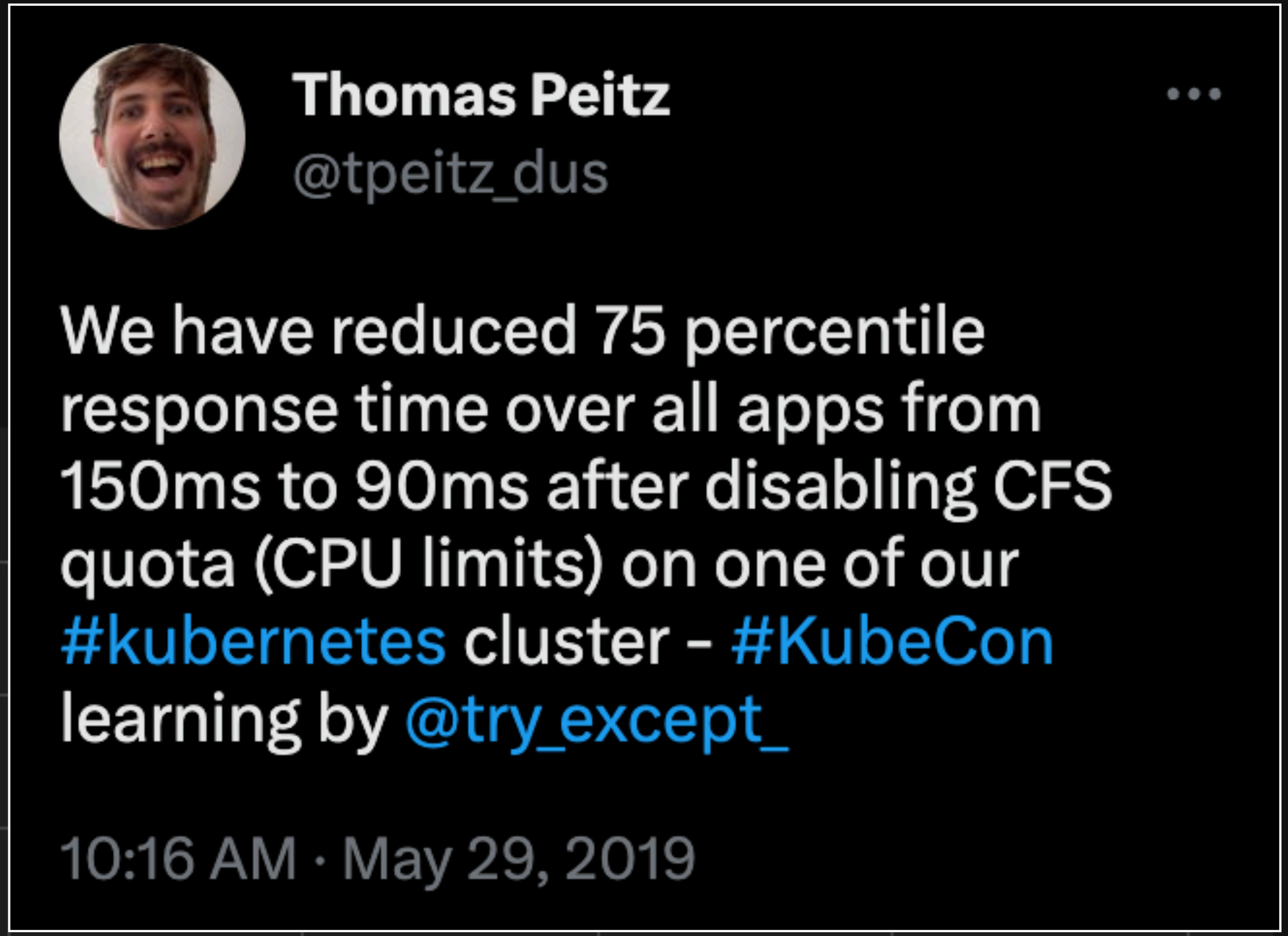
```yaml
resources:
  requests:
    cpu: "250m"
```

```yaml
resources:
  requests:
    cpu: "500m"
```



Faster response times

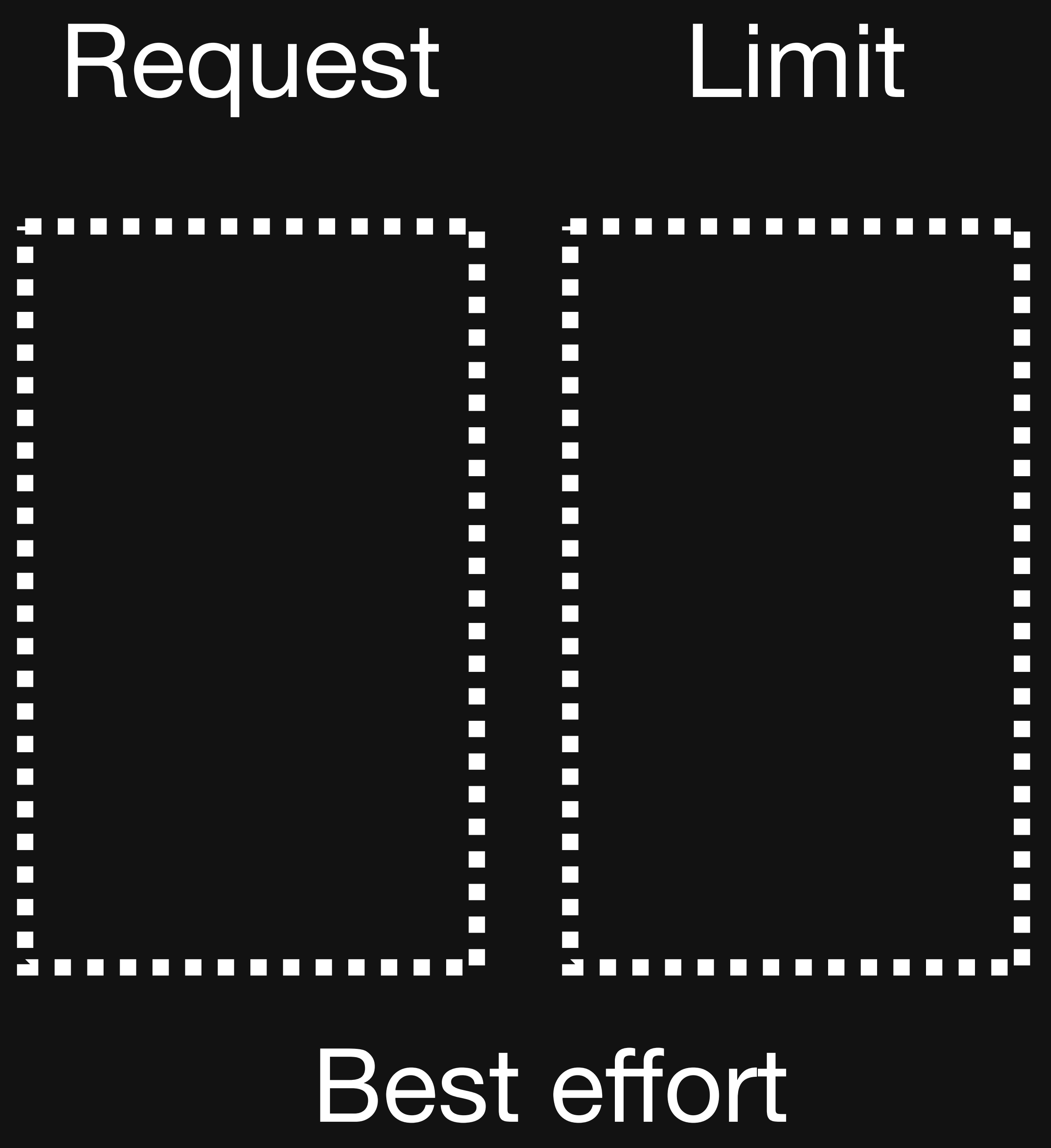



Thomas Peitz
@tpeitz_dus

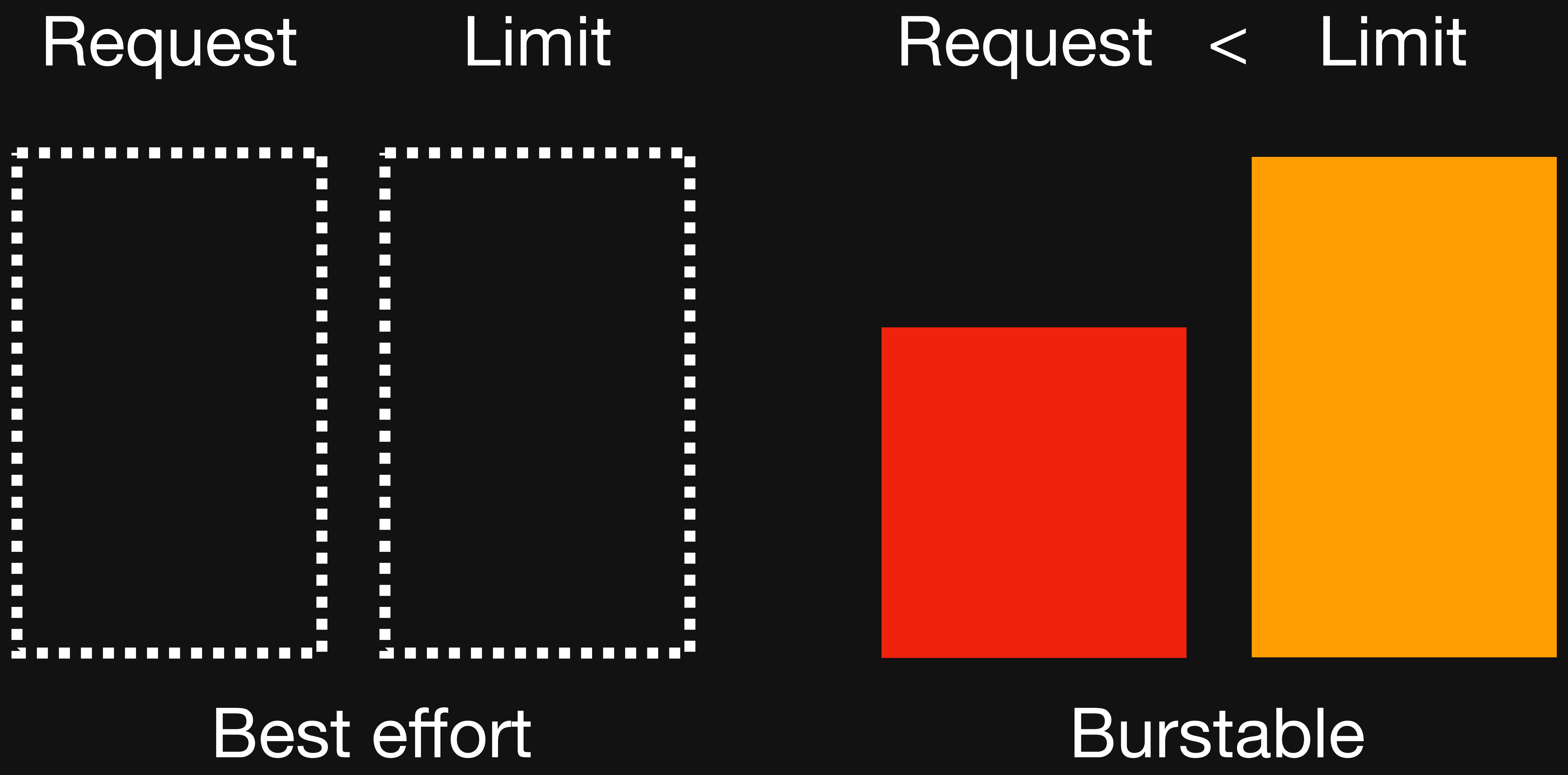
We have reduced 75 percentile response time over all apps from 150ms to 90ms after disabling CFS quota (CPU limits) on one of our #kubernetes cluster - #KubeCon learning by @try_except_

10:16 AM · May 29, 2019

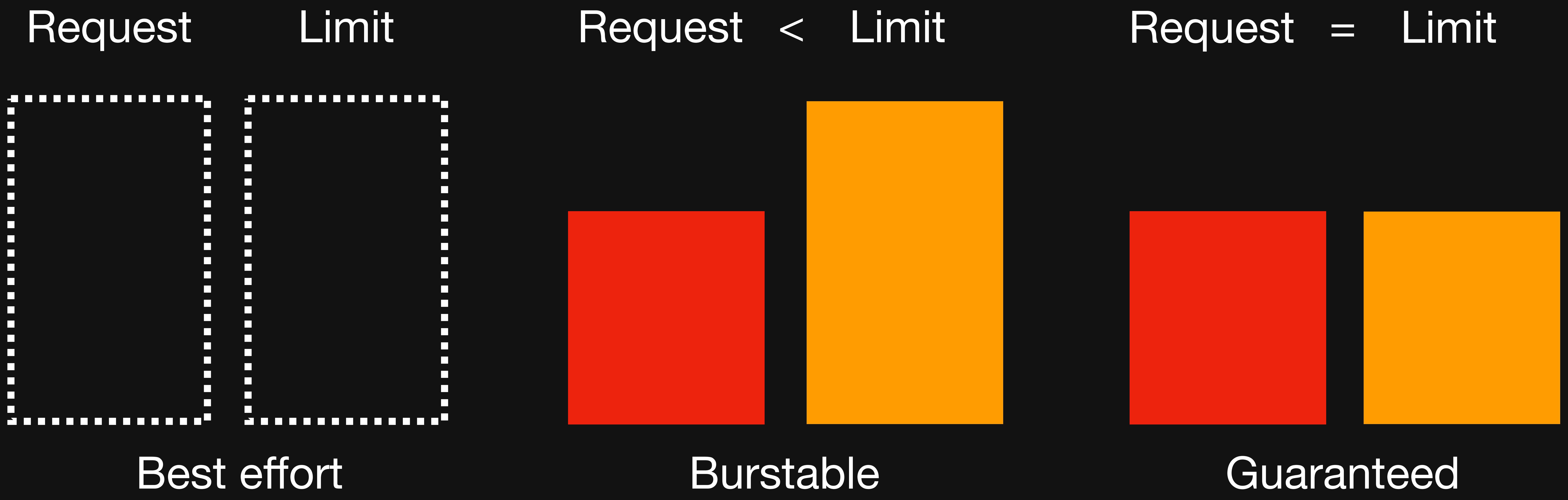
Quality of Service



Quality of Service



Quality of Service



Lessons learned

- Do not set CPU limits
- Always set CPU requests

Two more pitfalls

- Know your resources: Each Node has a `.status.allocatable` field
- Watch out for namespace limits

Watch out for namespace limits

```yaml
apiVersion: v1
kind: LimitRange
metadata:
  name: cpu-resource-constraint
spec:
  limits:
    - default:
        cpu: 500m
      type: Container
```

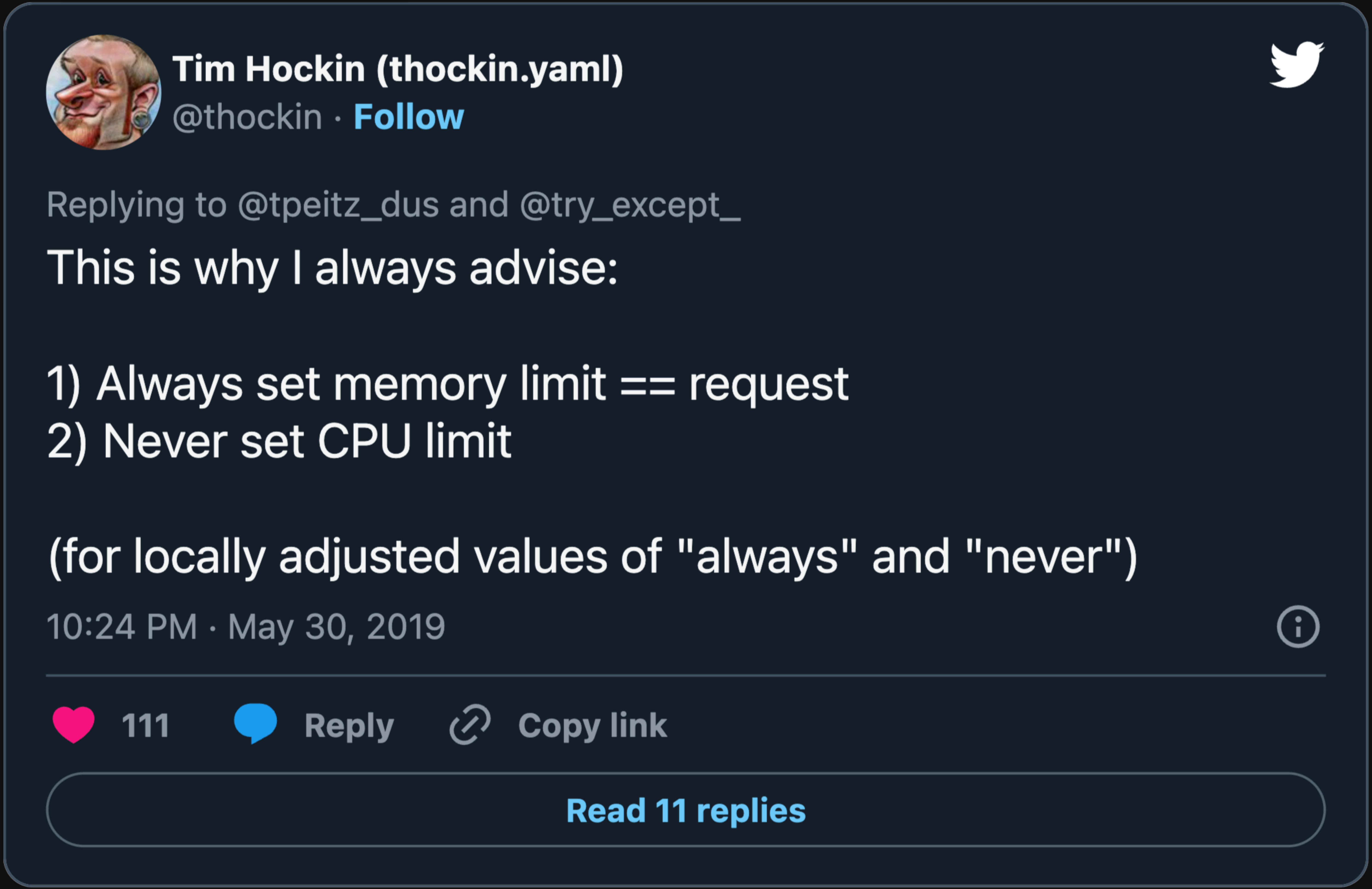
```yaml
apiVersion: v1
kind: Pod
metadata:
  name: example-conflict-with-limitrange-cpu
spec:
  containers:
    - name: demo
      image: registry.k8s.io/pause:2.0
      resources:
        requests:
          cpu: 700m
```

Summary

- Clusters need room to operate
- Memory request == memory limit
- No CPU limit

Exceptions

- Set CPU limits when you prefer consistent workloads over performant workloads
- Overcommit on memory when you want your workloads to be as cheap as possible and don’t care about termination


Tim Hockin (thockin.yaml)
@thockin · Follow

Replying to @tpeitz_dus and @try_except_

This is why I always advise:

1) Always set memory limit == request
2) Never set CPU limit

(for locally adjusted values of "always" and "never")

10:24 PM · May 30, 2019

111 Reply Copy link

Read 11 replies

Thank you

@felixmhoffmann